



EPworX 12.0

Systemadministration

12.0.42203

PDTS GmbH
Moeringgasse 20
A-1150 Wien
Handelsgericht Wien, FN 73236 a, ATU 15125308
<https://www.frequentis.com>
Support: WFM-support@frequentis.com

Alle Rechte vorbehalten. Kein Teil dieses Dokuments darf in irgendeiner Form oder in irgendeinem Verfahren, egal ob elektronisch oder mechanisch, und zu welchem Zweck auch immer, ohne schriftliche Genehmigung der Firma PDTS GmbH vervielfältigt, verarbeitet oder verbreitet werden.

Firmen- und Produktnamen, die in diesem Dokument erwähnt werden, können warenzeichen-, marken- oder patentrechtlichem Schutz unterliegen.

Einleitung

Die vorliegende Dokumentation beschreibt das Workflow-Management-System (WFMS) *EPworX*. Die Ziel-Lesergruppe sind Systemadministratoren, die für die Online-Wartung des Systems sowie für die Verwaltung workflowspezifischer Informationen (z.B. Anlegen neuer Organisationseinheiten, Definition von Prozessen etc.) verantwortlich sind.

Das Handbuch für den Systemadministrator besteht aus folgenden Kapiteln:

- **Systemarchitektur:** Hier wird die Architektur des Systems – das Zusammenwirken der einzelnen Komponenten – beschrieben.
- **Das HTML-Interface:** Die Administration des Systems erfolgt weitgehend über ein HTML-Interface. Die allgemeine Struktur dieses Interfaces wird in diesem Kapitel beschrieben.
- **Ids, Namen und Internationalisierung:** Hier wird beschrieben welche Attribute einer Objektklasse in *EPworX* als Id verwendet werden, und wie eine Id aussieht. Weiters finden Sie hier Informationen über die Internationalisierung von Stammdatenobjekten und Objektklassen.
- **Definition der Organisationsstruktur:** Ein Teil der Stammdaten, die für die Ausführung eines Workflows benötigt werden, besteht aus der Aufbauorganisation Ihres Unternehmens: Organisationseinheiten, OE-Bäume, Personen, Rollen dieser Personen, Rechte und Berechtigungslisten von Personen bzw. Rollen. Die folgenden Objektklassen werden dazu in *EPworX* verwaltet:
 - **Server:** Eine *EPworX* Installation kann aus mehreren Servern bestehen.
 - **Rollen:** dienen zur Beschreibung von Funktionen und Aufgabengebieten von Personen.
 - **Rechte:** werden an Rollen bzw. an Benutzer vergeben. Dadurch kann man den Benutzerkreis für Systemfunktionen einschränken.
 - **Berechtigungslisten:** Mehrere Rechte können in Berechtigungslisten zusammengefasst, und dann gemeinsam an eine Rolle oder einen Benutzer vergeben werden.
 - **Benutzer:** sind die im System registrierten Personen.
 - **Organisationseinheiten** dienen zur Beschreibung der Struktur Ihrer Organisation. Im weiteren Verlauf dieses Dokumentes wird Organisationseinheit mit *OE* oder *Org.-Einheit* abgekürzt.

- **Organisationsklassen:** dienen der Typisierung von Organisationseinheiten.
- **OE-Bäume:** Die Organisationseinheiten können Hierarchien bilden, d.h. eine OE kann einer anderen untergeordnet sein und umgekehrt. Die Hierarchie von Organisationseinheiten wird durch das Einhängen der entsprechenden OEs in einen OE-Baum definiert. Dabei kann eine OE mehrere untergeordnete Organisationseinheiten besitzen, eine untergeordnete OE (in einem Baum) aber nur eine übergeordnete OE besitzen. Eine OE kann in mehrere OE-Bäume eingehängt werden (auf diese Weise können abteilungsübergreifende Organisationseinheiten abgebildet werden).
- **Berechtigungssystem für EPworX :** Dieses Kapitel beschreibt das Berechtigungssystem von EPworX mit dessen Hilfe Zugriffsrechte von Benutzern auf Objekte definiert werden können.
- **Workflowelemente:** Aufbauend auf die Definition der o.g. Stammdaten können Prozesse (Workflows) definiert werden. Die einzelnen Elemente werden in den folgenden Kapiteln beschrieben:
 - **Applikationen:** Logisch zusammengehörende Prozesse können zu sog. Applikationen zusammengefasst werden, z.B. *Einkauf, Forschung, Entwicklung, Verkauf, Personal, Kreditabteilung, usw.*
 - **Tasks** sind die elementaren Bestandteile (Schritte) in Prozessen.
 - **Funktionen** sind Programmstücke, die automatisierte Teilaufgaben eines Tasks darstellen.
 - **Formulare** enthalten in strukturierter Form die Daten der Prozesse.
 - **Prozesse** beschreiben den Ablauf eines Geschäftsprozesses als Ganzes.
- **Prozessdefinition:** In diesem Kapitel wird die Definition von Prozessen erläutert, dafür gibt es in EPworX drei Möglichkeiten:
 - **Workflow-Definition-Language (WDL):** eine Script-Sprache zur Prozessdefinition,
 - **XWDL:** Die XML-Version von WDL. Eine Beschreibung von XWDL finden Sie im [Programmierhandbuch](#) im Kapitel **XWDL**.
 - **grafische Prozessdefinition** mit dem Prozesseditor von EPworX .
- **Die Suche in EPworX :** Dieses Kapitel verweist auf die entsprechenden Handbücher, in denen die Suche von EPworX beschrieben ist.
- **Administrationsfunktionen:** Weitere Administrationsfunktionen, wie Import/Export, Reorganisation, SSL etc. werden in diesem Kapitel beschrieben.
- **Konfiguration:** In diesem Kapitel wird die Konfiguration des EPworX –Servers beschrieben.
- **Dashboard:** Hier wird beschrieben wie das Dashboard von EPworX verwendet werden kann.

Geschlechtsneutrale Differenzierung: Aus Gründen der einfacheren Lesbarkeit wird auf die geschlechtsneutrale Differenzierung, z.B. Benutzer/innen, verzichtet. Entsprechende Begriffe gelten im Sinne der Gleichbehandlung grundsätzlich für beide Geschlechter.

Inhaltsverzeichnis

1	Systemarchitektur	7
1.1	Das World Wide Web (WWW)	7
1.2	Die Systemkomponenten	8
2	Das HTML–Interface	9
2.1	Tabellendarstellung	11
2.1.1	Spaltenauswahl, Sortierung und Filter	12
2.1.2	Standard-Funktionen	13
2.2	Objekt-Detailansicht	14
2.2.1	Registerkarte: Allgemein	15
2.2.2	Registerkarte: Historie	17
2.2.3	Registerkarte: Zugriff	17
2.2.4	Registerkarte: Referenzen	18
2.2.5	Weitere Funktionen	19
3	Ids, Namen und Internationalisierung	22
3.1	Ids und Namen	22
3.2	Internationalisierung der Stammdatenobjekte und Objektklassen	22
4	Definition der Organisationsstruktur	24
4.1	Rollen	24
4.1.1	Registerkarte: Allgemein	24
4.1.2	Registerkarte: Berechtigungen	26
4.1.3	Registerkarte: Benutzer	26
4.1.4	Systemrollen	27
4.2	Rechte	27
4.2.1	Registerkarte: Allgemein	28
4.2.2	Registerkarte: Berechtigungen	28
4.2.3	Standardrechte	29
4.3	Benutzer	30
4.3.1	Registerkarte: Allgemein	31
4.3.2	Registerkarte: Rollen	33
4.3.3	Registerkarte: Vertretungen	34
4.3.4	Registerkarte: Rollenvertretungen	35
4.3.5	Registerkarte: Berechtigungen	36
4.3.6	Registerkarte: Alle Berechtigungen	36
4.3.7	Registerkarte: Einstellungen	36

4.3.8	Registerkarte: Alle Einstellungen	36
4.3.9	Toolbarfunktion <i>GUI-Konfigurationen</i>	37
4.3.10	Berechtigungen überprüfen	37
4.3.11	Abgelaufene Passwörter	37
4.3.12	Datenschutz-Funktionen	38
4.4	Organisationseinheiten	38
4.4.1	Registerkarte: Allgemein	39
4.4.2	Registerkarte: Übergeordnete Org.-Einheiten	41
4.4.3	Registerkarte: Rollen	41
4.5	OE-Baum	41
4.5.1	Registerkarte: Allgemein	42
4.5.2	Registerkarte: OE-Baum	42
4.5.3	Funktion <i>OE-Bäume zusammenführen</i>	44
4.6	Organisationsklassen	45
4.6.1	Registerkarte: Allgemein	45
5	Berechtigungssystem	47
5.1	Allgemeines	47
5.1.1	Berechtigungsliste	47
5.1.2	Standardberechtigungen	48
5.1.3	Objektklassen	48
5.1.4	Formulare und Berechtigung	49
5.2	Verwendung von Berechtigungen	49
5.2.1	Berechtigungsdialog	50
5.2.2	Berechtigungen in der Benutzerverwaltung	52
5.2.3	Berechtigungen in der Rollenverwaltung	53
5.2.4	Berechtigungen beim Objekt – Zugriff	53
5.2.5	Verwaltung von Berechtigungslisten	53
5.2.6	Rechte um Berechtigungen zu bearbeiten	54
5.2.7	Sonderfall: Berechtigungen für Rollenzuordnungen	54
5.3	Standardeinstellungen	54
5.4	Wofür werden welche Rechte benötigt?	55
5.5	Beispiel	55
5.6	Berechtigung und Vertretung	55
6	Workflowelemente	59
6.1	Applikationen	60
6.1.1	Registerkarte: Allgemein	61
6.1.2	Registerkarte: Eigenschaften	62
6.1.3	Registerkarte: Eigenschaften XML	63
6.1.4	Report	64
6.2	Tasks	64
6.2.1	Registerkarte: Allgemein	64
6.2.2	Registerkarte: Funktionen	68
6.2.3	Ergänzen von Formularen	69
6.3	Funktionen	70
6.3.1	Registerkarte: Allgemein	70

6.3.2	Standardfunktionen	73
6.4	Formulare	77
6.4.1	Formulareditor	77
6.4.2	Anzeigen	86
6.4.3	View erzeugen	86
6.4.4	Bearbeiten	87
6.4.5	Report	98
6.5	Prozesse	98
6.5.1	Neuen Prozess erstellen	99
6.5.2	Im Prozesseditor bearbeiten	99
6.5.3	WDL / XWDL laden	99
6.5.4	Registerkarte: Allgemein	100
6.5.5	Registerkarte: Source	103
6.5.6	Registerkarte: Graph	103
6.5.7	Registerkarte: Komponenten	103
6.5.8	Registerkarte: Formular-Sichtbarkeiten	104
6.5.9	Registerkarte: Eskalationen	107
6.5.10	Registerkarte: Funktionen	110
6.5.11	Registerkarte: Ordneinstellungen	110
6.5.12	Registerkarte: Dokumenten-Berechtigungen	110
6.5.13	Registerkarte: Entscheidungsunterstützung	112
6.5.14	Report	118
6.5.15	Meilensteine	118
6.5.16	Plantypen	119
6.6	Funktionsgruppen	120
6.7	GUI-Konfiguration	121
6.7.1	Registerkarte: Allgemein	121
6.7.2	Registerkarte: Datei	140
6.7.3	Registerkarte: Zuordnungen	140
6.7.4	Anpassbare Aktionen	141
6.8	Style-Konfigurationen	148
6.8.1	Registerkarte: Allgemein	148
6.8.2	Registerkarte: Datei	148
6.9	Ressourcen Editor	149
6.9.1	Toolbarfunktionen	150
6.9.2	CSV-Datei Konvertierung	152
6.10	Wertelisten	152
6.11	Timer	153
6.11.1	Registerkarte: Allgemein	153
6.11.2	Cron-Pattern	156
6.11.3	Übersicht über Standard-Timer	157
6.12	Mailboxen	163
6.12.1	Registerkarte: Allgemein	163
6.12.2	Registerkarte: Aktion	165
6.12.3	Registerkarte: Mailbox anzeigen	166
6.13	Web-Dienste	166
6.13.1	Webservice Clients	166

6.13.2	Webservice Server	168
6.14	Nachrichten-Vorlagen	169
6.14.1	Registerkarte: Allgemein	171
6.14.2	Übersicht über Ereignisse und Versandarten	174
6.15	Testfälle	174
6.15.1	Toolbar	176
6.15.2	Testschritte	177
6.15.3	Prozesshistorie und Prozessdetails	179
7	Prozessdefinition	181
7.1	WDL	181
7.1.1	Lexikalische Konventionen	182
7.1.2	Prozesskopf	183
7.1.3	Deklarationsteil	184
7.1.4	Einfache Anweisungen	185
7.1.5	Kontrollstrukturen	190
7.1.6	Eventmechanismus	197
7.1.7	Web-Dienste	198
7.2	Grafische Prozessdefinition	201
7.2.1	Aufbau des Prozesseditors	201
7.2.2	Arbeiten mit dem Prozesseditor	201
7.2.3	Die Funktionen der Menüleiste	208
7.2.4	Prozesseigenschaften	213
7.2.5	Tasks	214
7.2.6	Eskalationen	217
7.2.7	Prozesspläne	219
7.2.8	Die Funktionsleiste	223
7.2.9	Die allgemeinen Eigenschaften eines Knotens	227
7.2.10	Eigenschaften einer Aktivität definieren	227
7.2.11	Bedingungen für Ifs, Choice, Schleifen	230
7.2.12	Methoden eines Systemschrittes bestimmen	231
7.2.13	Die Eigenschaften eines Batchschrittes bestimmen	232
7.2.14	Die Eigenschaften eines Subprozesses bestimmen	233
7.2.15	Eigenschaften eines Parallel For definieren	234
7.2.16	Eigenschaften einer UND-/ODER-Parallelität oder eines <i>Parallel for</i> Endknoten bestimmen	234
7.2.17	Eigenschaften eines Ereignisses definieren	235
7.2.18	Eigenschaften eines GOTO-Knoten definieren	237
7.2.19	Eigenschaften eines Web-Dienste-Knoten definieren	238
7.2.20	Bedingungseditor	240
8	Die Suche in EPworX	243
8.1	Prozesssuche	243
8.2	Dokumentensuche	243
8.3	Report Designer	243
8.4	Reports	243

9	Administrationsfunktionen	244
9.1	Server	244
9.1.1	Servermonitor	244
9.1.2	Serversteuerung	245
9.1.3	Log-Dateien	250
9.1.4	Datenbankverbindungen	251
9.1.5	Objekthistorie	251
9.1.6	Benutzer-Sessions	252
9.1.7	Ereignisse	253
9.1.8	Ausstehende Änderungen	254
9.1.9	Event Registrierungen	254
9.1.10	Batch-Jobs	254
9.1.11	Warteschritte	255
9.1.12	Klassenpfad	255
9.1.13	Zertifikatsmanagement	256
9.1.14	Query-Tool	260
9.2	Import/Export	262
9.2.1	Import/Export im XML Format	262
9.2.2	Prozesse archivieren	267
9.2.3	Applikation exportieren	267
9.2.4	Applikation installieren/aktualisieren	269
9.2.5	Dateiimport	270
9.3	Cluster	273
9.3.1	Clustermonitor	273
9.3.2	Server	273
9.4	DMS	274
9.4.1	Volltextsuche	274
9.4.2	Schlagworte	274
9.4.3	Suche in Papierkörben	274
9.5	Reorganisation	275
9.5.1	Rollenzuordnungen verändern	276
9.5.2	Prozessinstanzen analysieren	276
9.5.3	OE-Historie	277
9.6	Kommunikation	277
9.6.1	Mail-Queue	277
9.6.2	Mail Journal	278
9.6.3	Autorisierer	279
9.6.4	LDAP	279
9.6.5	Externe Datenbanken	283
9.6.6	Web Dienste	284
9.7	Entscheidungsunterstützung	284
9.7.1	Klassifikator-Bewertungen	285
9.7.2	PMML-Administration	285
10	Konfiguration	287

11 Dashboard	288
11.1 Neu	288
11.2 Öffnen	289
11.3 Speichern	289
11.3.1 Löschen	290
12 Administration via Kommandozeile	291
12.1 Architektur und Anwendung	291
12.2 Kommandos	292
12.2.1 Client Kommandos	292
12.2.2 Server Kommandos	293
12.3 Beispiele	293
12.3.1 Einen Konfigurationsparameter setzen	293
12.3.2 Serverneustart	294
12.3.3 Rolle einem Benutzer hinzufügen bzw. entziehen	294
12.3.4 Zeitintervall eines Timers setzen	294
12.3.5 Arbeitskorb Handling	294
12.3.6 Session Handling	295
13 Eingeschränkte Administration	296
13.1 Benutzer	296
13.2 Organisationseinheiten	297

1. Systemarchitektur

Das Workflowsystem *EPworX* ist vollständig webbasiert. Deshalb werden zuerst kurz die wesentlichen Konzepte dieser Technologie genannt und anschließend die Architektur des Systems beschrieben.

1.1 Das World Wide Web (WWW)

Das WWW beruht auf drei Grundkonzepten:

- die einheitliche Adressierung von Informationen im Internet mit dem Uniform Resource Locator (URL),
- der Darstellung von Information mit der Sprache HTML und
- der Übertragung über das HTTP Protokoll.

HTML (Hyper Text Markup Language) bietet die Möglichkeit, unterschiedliche Medientypen, wie Texte, Grafiken, Ton und Video, in einem Dokument zu vereinen. Mittels so genannter Hyperlinks kann mit Mausklick von einem Dokument zu einem anderen verzweigt werden - komplex strukturierte Informationen können so übersichtlich präsentiert werden. Wichtig für die Interaktion mit dem WWW ist die Möglichkeit, ausfüllbare Formulare zu repräsentieren. Sie erlauben dem Benutzer, Daten an den Server zu schicken. Eine HTML-Erweiterung erlaubt auch das Laden von Dateien auf den Server.

HTTP (Hyper Text Transfer Protocol) ist ein einfaches Übertragungsprotokoll von Dokumenten über das Internet. Der Client (Browser) fordert ein Dokument bei einem Server an, indem eine TCP/IP Verbindung zu diesem Server aufgebaut wird und der String *GET 'Dokument-URL'* übertragen wird. Der Server antwortet mit einigen Zeilen Statusinformation und dem Inhalt des Dokuments. Dann schließt er die Verbindung. Die Dokument-URL kann auch ein Programmaufruf am Server sein, zurückgeschickt wird dann die Ausgabe des Programms. Große Bedeutung erlangte auch das Prinzip von mobilem Code. Dabei werden in HTML Dokumenten kleine Programme, so genannte Applets, eingebettet, die dann am Client ausgeführt werden und somit die Funktionalität des Clients erhöhen. Als Programmiersprache wird dabei vor allem Java eingesetzt; die meisten Browser erlauben die Ausführung von vorkompiliertem Java-Code.

1.2 Die Systemkomponenten

Abbildung 1.1 zeigt den Aufbau des Workflow-Systems *EPworX* mit seinen Hauptkomponenten:

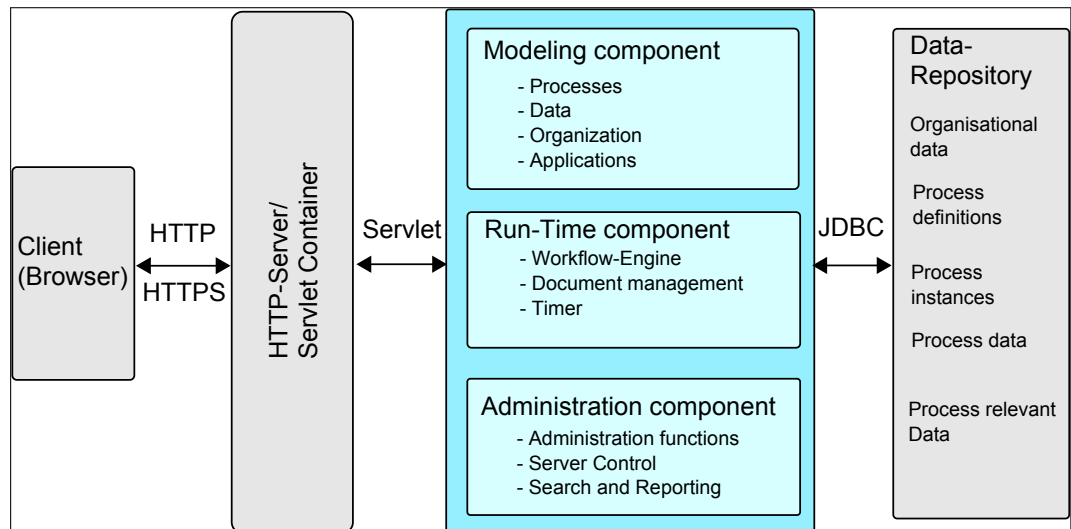


Abbildung 1.1: **EPworX Systemarchitektur**

Das System basiert vollständig auf dem World Wide Web. Jegliche Kommunikation sowohl des normalen Benutzers als auch des Systemadministrators mit dem System kann über einen Web-Browser durchgeführt werden.

Der HTTP-Server ist dabei die Schnittstelle des Workflowsystems nach außen. Anfragen von einem Client werden in Methodenaufrufe übersetzt und an das Workflow-Management-system weitergeleitet. Dieses besteht im wesentlichen aus zwei Komponenten: Der Engine und dem HTML Interface.

Die Workflow-Engine selbst ist eine Sammlung von Methoden, die die eigentliche Funktionalität des Workflow-Systems, wie z.B. Verändern von Formularen, Beenden einer Aktivität, usw. bereitstellt.

Das HTML-Interface interpretiert die Anfrage vom Browser, führt einen API-Call auf die Workflow-Engine durch und übersetzt das Ergebnis wieder in eine am Browser darstellbare HTML-Seite.

Das Datenbankmanagementsystem (DBMS) speichert alle Stammdaten und für die Prozessausführung relevanten Informationen. Dies sind die Prozessdefinitionen, die Stati der laufenden Prozesse, sowie alle Benutzerdaten, die in Formularen abgelegt werden.

Weitere Details zur Systemarchitektur entnehmen Sie dem Handbuch [Applikationsprogrammierung mit EPworX](#).

2. Das HTML–Interface

Sie gelangen in die *EPworX* -Systemadministration in dem Sie im Menü *EPworX* den Menüpunkt *Administration* auswählen. Dabei kann es abhängig von den Einstellungen des Servers vorkommen, dass Sie sich neu einloggen müssen (wenn Host und Port gleich sind), um die Administration zu sehen. Die nötigen Einstellungen dazu finden Sie im [Installationshandbuch](#) im Kapitel [HTTP server](#). Diese Funktion steht Ihnen nur zur Verfügung, wenn Sie das Recht *admin* besitzen.

Wenn Sie sich als Standard-Systemadministrator (Benutzer: *sysadm*) anmelden, gelangen Sie direkt in die *EPworX* -Systemadministration.

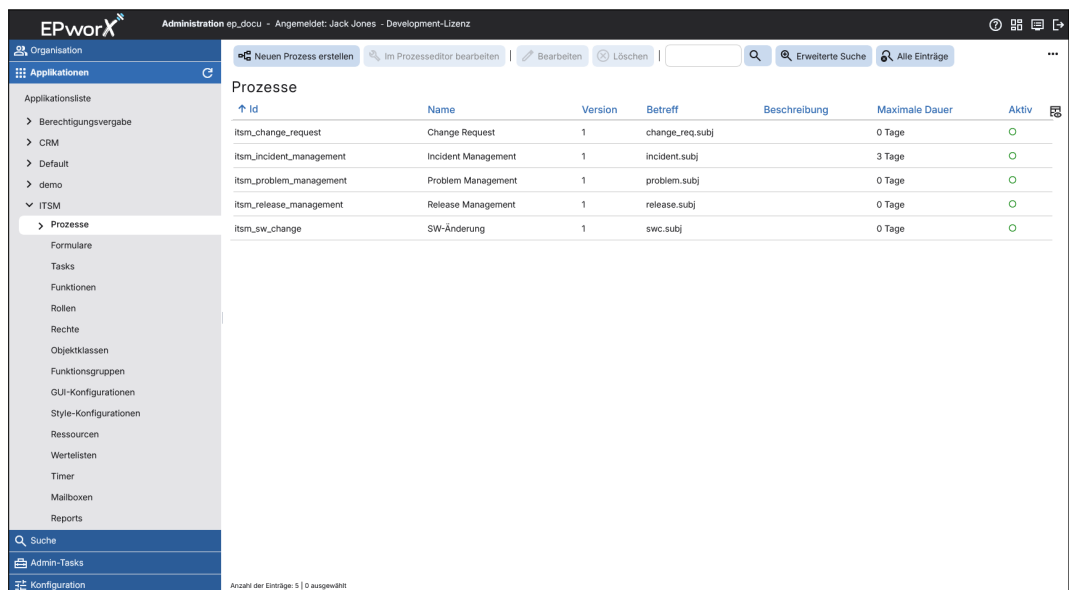


Abbildung 2.1: Die Systemadministration

Das HTML-Interface (siehe Abbildung 2.1) gliedert sich in die folgenden Bereiche:

1. **Information:** Im oberen Bereich der *EPworX* -Systemadministration sehen Sie Informationen zum angemeldeten Benutzer und dem aktuell laufenden Server. Die vier Funktionen *Hilfe*, *Dashboard*, *Arbeitskorb* und *Abmelden*, rechts außen, stehen Ihnen immer zur Verfügung.



- **Hilfe:** Wenn Sie diese Funktion aktivieren, öffnet sich in einem neuen Fenster die Hilfe zu der gerade aktiven Seite. Zur gleichen Hilfeseite gelangen



Sie, wenn Sie im aktiven Bereich die Taste F1 drücken.

- **Dashboard:** Nach dem Aktivieren dieser Funktion wird im Bearbeitungsbe-
reich Ihr Dashboard angezeigt (siehe Kapitel [Dashboard](#)).
- **Arbeitskorb:** Durch das Anklicken dieser Funktion gelangen Sie in die *EP-
worX* -Arbeitskorbkomponente.
- **Abmelden:** Wenn Sie diese Funktion aktivieren, melden Sie sich von *EP-
worX* ab.
- **Hinweis:** Wenn dieses Symbol erscheint, wurde eine Änderung am *EPworX*
-System vorgenommen. Sie erhalten durch Anklicken dieses Symbols nähe-
re Informationen darüber, ob Sie entweder den Server neu starten oder die
Cachestrukturen aktualisieren müssen.
- **Werkzeugleiste:** Direkt unter dem Informationsbereich befindet sich die Werk-
zeugleiste, die Ihnen zu den jeweils im Bearbeitungsbereich angezeigten
Informationen verschiedene Funktionen zur Verfügung stellt.

2. **Navigation:** Dieser Bereich enthält eine Auflistung der zur Verfügung stehen-
den Administrationsfunktionen. Diese Funktionen sind in verschiedene Bereiche -
Organisation, Applikationen, Suche, Admin-Tasks und *Konfiguration* - gegliedert.
Sie öffnen einen Bereich, indem Sie die jeweilige Schaltfläche aktivieren.

- **Organisation:** In diesem Bereich stehen Ihnen Links für folgende Verwal-
tungsaufgaben zur Verfügung: *Benutzer, Organisationseinheiten, Organisa-
tionsklassen, OE-Bäume* und *Berechtigungslisten*.
- **Applikationen:** Hier werden in einzelnen Unterordnern die jeweiligen *EP-
worX* -Applikationen aufgelistet. Pro Applikation werden folgende Bereiche
zur Verwaltung angeboten: *Prozesse, Formulare, Tasks, Funktionen, Rollen,
Rechte, Objektklassen, Funktionsgruppen, GUI-Konfigurationen, Ressour-
cen, Wertelisten, Timer, Mailboxen, Reports, Webservice Clients, Webser-
vice Server, Nachrichten-Vorlagen* und *Testfälle*. Die Übersicht über alle Ap-
plikationen wird als oberster Link in diesem Bereich dargestellt.
- **Suche:** In *EPworX* können Sie auf unterschiedliche Arten nach Prozess-
instanzen bzw. DMS-Objekten suchen. Folgende Möglichkeiten werden in
diesem Bereich angeboten: *Prozesssuche, Dokumentensuche, Report De-
signer, Reports*.
- **Admin-Tasks:** Dieser Bereich enthält alle Administrationsfunktionen. Eine
genaue Auflistung und Beschreibung der einzelnen Links finden Sie in einem
späteren Kapitel in diesem Handbuch.
- **Konfiguration:** Alle Funktionen zur Konfiguration Ihrer Installation finden
Sie in diesem Bereich. Eine genaue Beschreibung dieser Funktionen finden
Sie im [Installationshandbuch](#).

3. **Bearbeitung:** Der Hauptbereich der *EPworX* -Systemadministration steht Ihnen
zur Bearbeitung der ausgewählten Funktion zur Verfügung. Hier sehen Sie un-
terschiedliche Masken bzw. Tabellen, deren Inhalt Sie mit den Funktionen der
Werkzeugleiste bearbeiten können. Nach dem Öffnen der Systemadministration
wird in diesem Bereich Ihr Dashboard angezeigt.

2.1 Tabellendarstellung

Stammdaten werden in *EPworX* zunächst in einer Tabelle aufgelistet. Diese Tabelle enthält in den Zeilen die einzelnen Objekte des ausgewählten Bereichs und in den Spalten verschiedene Eigenschaften des Objektes.

Detailinformationen zum Objekt und weitere Funktionen zum Objekt (Historie, Zugriff etc.) werden Ihnen in einem eigenen Fenster angezeigt (siehe Kapitel [Objekt-Detailansicht](#)). Sie öffnen dieses Fenster, indem Sie auf einen Eintrag einen Doppelklick ausführen oder eine Zeile zunächst mit einem einfachen Klick selektieren und danach die Menüfunktion *Bearbeiten* aktivieren.

Bei der Anzeige der Tabelle wird, falls die Anzahl der gefundenen Einträge eine Grenze übersteigt, nachgefragt, ob wirklich alle Einträge dieser Tabelle angezeigt werden sollen. Diese Obergrenze ist in der Systemkonfiguration einstellbar (siehe [Installationshandbuch](#)).

Id	Name	Typ	Beschreibung	Aktiv
all	Alle	global		○
dept	Organisationseinheit	lokal		○
home	Home	lokal		○
owner	Eigentümer	global		○
process_agent	Prozessakteur	global		○
sys	Sys	global		○
useradmin	Benutzer-Administrator	lokal		○

Abbildung 2.2: Beispiel für Tabellendarstellung (Rollen)

In der Tabellendarstellung werden die folgenden Zeilenformatierungen verwendet:

- **Zuletzt bearbeitet:** Jener Eintrag, den Sie zuletzt bearbeitet haben, wird farblich hinterlegt dargestellt.
- **Inaktive Einträge:** Objekte, die inaktiv sind, werden in grauer Schrift und *kur-siv* dargestellt. Beim Objekt **Formulare** werden zusätzlich zu den inaktiven Formularen, auch Einträge als inaktiv markiert, bei denen die Formularklasse nicht geladen werden kann.
- **Selektierte Einträge:** Einträge, die Sie aktuell selektiert haben, werden farblich hinterlegt.



2.1.1 Spaltenauswahl, Sortierung und Filter

Sie können die Anzahl der Spalten beliebig ändern. Dazu steht Ihnen ganz rechts außen im Tabellenkopf die Spaltenauswahl zur Verfügung. Klicken Sie auf die Funktion und es öffnet sich direkt bei dieser Spalte ein Dialog, in dem alle auswählbaren Spalten aufgelistet werden.

Jene Spalten, die bereits angezeigt werden, werden mit einem Häkchen dargestellt. Um eine zusätzliche Spalte anzuzeigen, wählen Sie aus dieser Liste einen Namen (ohne Häkchen) aus. Die Tabelle wird nun aktualisiert und die zusätzliche Spalte angezeigt. Um eine bestimmte Spalte auszublenden, wählen Sie in der Spaltenauswahl diese Spalte (mit Häkchen) aus.

Durch Aktivieren eines einzelnen Spaltenkopfs ändern Sie die Sortierreihenfolge der Tabelle. Nach welcher Spalte und in welche Richtung sortiert wird, wird durch einen Pfeil links neben dem Spaltennamen dargestellt.

Der Link *Filter* soll Ihnen bei vielen Einträgen in Ihrer Tabelle helfen, die Übersicht nicht zu verlieren. Der Filter stellt eine Art Auswahlkriterium dar. Damit können Sie gewisse Einträge der Tabelle ausblenden.

Durch einen Klick in der entsprechenden Spaltenüberschrift der Tabelle wird ein kontextabhängiges Filtermenü mit folgenden Einträgen angezeigt.

- Aufsteigend sortieren: Die Einträge der Tabelle werden nach der aktuellen Spalte aufsteigend sortiert. Dies wird in der entsprechenden Spalte durch einen Pfeil nach oben angezeigt.
- Absteigend sortieren: Die Einträge der Tabelle werden nach der aktuellen Spalte absteigend sortiert. Dies wird in der entsprechenden Spalte durch einen Pfeil nach unten angezeigt.
- Alle Einträge: Die Anwendung des Spaltenfilters der aktuellen Spalte wird aufgehoben.
- Benutzerdefiniert: Durch das Auswählen dieses Menüpunktes wird eine HTML–Seite angezeigt, in der Sie einen Wert eintragen können. Bestätigen Sie in dieser Seite Ihre Eingaben mit der Schaltfläche *Ok*, so wird die aktuelle Tabelle nach dem eingegebenem Wert gefiltert.
- Alle Spaltenwerte werden hier aufgelistet: Wenn Sie einen oder mehrere dieser Werte auswählen, wird die aktuelle Tabelle nach ihrer Auswahl gefiltert.

Sie haben auf die erste Art einen Filter auf eine oder mehrere Spalten definiert. Wenn Sie diese Filterkombination abspeichern möchten, klicken Sie auf den Link *Filter* in der Überschrift des Arbeitsbereiches. Es wird das Filter–Menü mit folgenden Einträgen angezeigt:

- Filter speichern: Dadurch speichern Sie die gerade aktuelle Kombination der einzelnen Spaltenfilter unter einem von Ihnen zuvor festgelegtem Namen mit einer entsprechenden Beschreibung.
- Filter bearbeiten: Dadurch können Sie alle Filtern bearbeiten oder löschen.

- Eine Liste aller gespeicherten Filter: Wenn Sie einen dieser Werte auswählen, wird die aktuelle Tabelle nach diesem Filter gefiltert. Die Liste kann auch Filter enthalten, die nicht von Ihnen, sondern von anderen Benutzer definiert wurden.
- Alle Einträge: Die Anwendung des abgespeicherten Filters wird aufgehoben.



Hinweis: Achtung! Das Löschen eines Filters kann nicht mehr rückgängig gemacht werden.

2.1.2 Standard-Funktionen

Folgende Funktionen finden Sie bei den meisten Tabellen der Systemadministration.



- **Neu:** Wenn Sie diese Funktion aktivieren, öffnet sich eine leere Detailansicht. Nach Eingabe aller relevanten Daten wird ein neues Objekt angelegt.



- **Bearbeiten:** Diese Funktion öffnet die Detailansicht zum ausgewählten Objekt. Sie können diese Funktion auf ein markiertes Objekt anwenden. Der Doppelklick auf einen Tabelleneintrag führt ebenfalls diese Funktion aus.



- **Löschen:** Mit Hilfe dieser Funktion können Sie die ausgewählten Objekte löschen. Sie können die Funktion auf ein oder mehrere Objekte gleichzeitig anwenden.



- **Ansehen:** Diese Funktion öffnet die Objekt-Detailansicht readonly, d.h. Sie können die Objektdetails nicht verändern. Ausnahme: Bei den Formularen öffnet sich die Formularvorschau und bei Prozessen eine Übersicht der Prozessdefinition nachdem Sie diese Funktion aktiviert haben.



- **Kurzsuche:** Hier können Sie (in das links vor der Funktion platzierte Textfeld) die Id oder den Namen oder Teile von einem der beiden Attribute als Suchkriterium eintragen. *EPworX* sucht nach allen Einträgen des ausgewählten Bereichs, bei denen die Id oder der Name entweder ganz mit dem eingegebenen Wert übereinstimmen oder damit beginnen. Nach dem eingegebenen Suchkriterium wird gesucht, nachdem Sie die Funktion aktivieren oder im Textfeld die Eingabetaste aktivieren. Die Tabelle wird aktualisiert und zeigt nur jene Einträge an, die dem Suchkriterium entsprechen.



- **Erweiterte Suche:** Durch das Aktivieren dieser Funktion wird ein neues Fenster mit einer leeren Detailansicht des aktuellen Bereichs angezeigt. Diese Seite bietet alle Attribute des jeweiligen Objekts als Suchkriterium an. Nachdem Sie die gewünschten Suchkriterien eingegeben und die Suche gestartet haben, wird die Tabelle wieder aktualisiert.



- **Alle Einträge:** Wenn Sie diese Funktionen aktivieren, werden alle eingegebenen Suchkriterien (Kurzsuche oder erweiterte Suche) ignoriert und alle Objekte in der Tabelle dargestellt.



- **Alle Einträge auswählen:** Mit dieser Funktion können Sie alle in der Tabelle enthaltenen Objekte gleichzeitig markieren oder die Markierung für alle Objekte aufheben. Bitte beachten Sie, dass, wenn ein oder mehrere Einträge bereits ausgewählt sind, beim Klicken auf den Button die Markierung für alle Einträge in der Tabelle aufgehoben wird. Ein erneuter Klick markiert anschließend alle Einträge erneut.
- **Aktualisieren:** Nachdem Sie diese Funktion aktiviert haben, wird der Bearbeitungsbereich aktualisiert.
- **Kopieren:** Mit dieser Funktion können Sie das ausgewählte Objekt inkl. seiner Einstellungen kopieren, z.B. bei Berechtigungslisten wird das Objekt kopiert und alle Rechte, die bei der Berechtigungsliste eingetragen wurden.

2.2 Objekt-Detailansicht

Nach dem Aktivieren der Funktion *Bearbeiten* oder *Neu* bzw. einem Doppelklick auf ein Objekt, öffnet sich ein neues Fenster mit der Detailansicht des ausgewählten Objekts. Diese Detailansicht enthält verschiedene Registerkarten, die unterschiedliche Informationen bzw. Funktionen zum Objekt beinhalten (siehe Abbildung 2.3).

The screenshot shows a web interface for editing a role. The 'Allgemein' tab is active, displaying various fields for the role's configuration. The 'Id' and 'Name' fields are both set to 'home'. The 'Applikation' is set to 'Default' and the 'Typ' is set to 'lokal'. There is a description field, a reference role dropdown, and an 'Aktiv' checkbox which is checked. At the bottom, there are buttons for 'Löschen', 'Speichern und Schließen', 'Speichern', and 'Abbrechen', along with a section for scheduling changes.

Abbildung 2.3: Beispiel für Objekt-Detailansicht (Rollen)

Unabhängig von der gerade aktiven Registerkarte stehen Ihnen immer die Schaltflächen **Speichern**, **Speichern und Schließen** und **Abbrechen** bzw. **Schließen** zur Verfügung. Durch das Aktivieren der Schaltfläche **Abbrechen** wird die Detailansicht des Objekts ohne Änderungen zu speichern geschlossen. Hat eine Detailansicht mehrere Reiter (Tabbed Window) und auf diesen Reitern können keine Änderungen durchgeführt werden, dann lautet die Schaltfläche **Schließen**. Wenn Sie die Schaltfläche **Speichern und Schließen** aktivieren und Änderungen vorgenommen haben, werden diese zunächst gespeichert und danach die Detailansicht geschlossen. In der Regel wird diese Schaltfläche grün eingefärbt sobald Änderungen gemacht wurden und deutet darauf hin, dass dies die Standard-Aktion ist. Änderungen können Sie nur in Registerkarten mit Masken vornehmen (z.B. *Allgemein*). Hier steht Ihnen zusätzlich die Schaltfläche **Speichern** zur Verfügung.

Neben Masken können auch Tabellen in einzelnen Registerkarten dargestellt sein. Sie können diese Tabellen, wie in Kapitel [Tabellendarstellung](#) beschrieben, bearbeiten.

Felder, die Sie unbedingt ausfüllen müssen, um ein erfolgreiches Speichern durchführen zu können (Mussfelder), werden in *EPworX* **fett** dargestellt. Wenn Sie eines oder mehrere dieser Mussfelder inkorrekt ausgefüllt haben, werden diese nach dem Aktivieren der Schaltfläche *Speichern und Schließen*, *Speichern* oder einem *Registerkartenwechsel* rot markiert. Eine Speicherung wird erst erfolgreich ausgeführt, wenn alle Mussfelder korrekt befüllt sind.

2.2.1 Registerkarte: Allgemein

Die erste Registerkarte *Allgemein* enthält immer die eigentlichen Objekteigenschaften. Nach dem Sie Änderungen in der Maske durchgeführt haben, können Sie diese auf verschiedene Weise speichern:

- **Speichern und Schließen:** Aktivieren Sie diese Schaltfläche, wenn Sie die Änderungen abspeichern und anschließend die Detailansicht schließen wollen. In der Regel wird diese Schaltfläche grün eingefärbt sobald Änderungen gemacht wurden und deutet darauf hin, dass dies die Standard-Aktion ist.
- **Speichern:** Wenn nach dem Speichern weiterhin die Detailansicht geöffnet bleiben soll, aktivieren Sie diese Schaltfläche.
- **Registerkartenwechsel:** Wenn Sie nach dem Speichern auf eine andere Registerkarte der Detailansicht wechseln wollen, aktivieren Sie einfach die gewünschte Karte. Sie werden (bei Änderungen) gefragt, ob Sie die Registerkarte wirklich wechseln und gleichzeitig Ihre Änderungen speichern oder in der aktuellen Ansicht verbleiben wollen. Beenden Sie diesen Dialog mit *Speichern und Schließen*, so werden Ihre Änderungen gespeichert und die Ansicht wechselt zur ausgewählten Karte. Beenden Sie diesen Dialog mit *Abbrechen*, werden Ihre Änderungen NICHT gespeichert und die Ansicht bleibt in der aktuellen Karte.

In der ersten Karte ist außerdem die Schaltfläche **Löschen** aktiv. Mit dieser Funktion können Sie das aktuelle Objekt löschen. Danach wird die Detailansicht geschlossen. Diese Funktion entspricht der Standard-Funktion *Löschen* der Werkzeugleiste.

Änderungen zu späterem Zeitpunkt

Bei manchen Stammdaten– bzw. Workflow–Objekten besteht die Möglichkeit, die durchgeführten Änderungen erst zu einem bestimmten Zeitpunkt wirksam werden zu lassen. Dazu gibt es in der jeweiligen Änderungsmaske das Textfeld **Änderungen durchführen am** (siehe Abbildung 2.3).

Tragen Sie in dieses Textfeld jenes Datum (und Uhrzeit) ein, an dem die vorgenommenen Änderungen wirksam werden sollen und speichern Sie Ihre Eingabe ab. Bis zu diesem Datum enthält das entsprechende Objekt jene Werte in den Attributfeldern, die vor den Änderungen darin enthalten waren. Wird hier kein Datum eingetragen, so werden die Änderungen sofort übernommen. Bei der Eingabe des Datums unterstützt Sie der Kalender bzw. die Uhrzeit-Auswahl, die sich nach dem Aktivieren des entsprechenden Icons öffnet.



Sie können für ein Objekt immer nur eine Änderung zu einem späteren Zeitpunkt eintragen. Ob eine ausstehende Änderung eingetragen ist, erkennen Sie in der Maske am befüllten Textfeld **Objektänderung am** und an der nebenbei platzierten Funktion **Spätere Objektänderung anzeigen**. Wenn Sie diese Funktion aktivieren, öffnet sich ein neues Fenster, welches die ausstehende Objektänderung darstellt (siehe Abbildung 2.4). Wenn Sie hier die Schaltfläche **Änderungen verwerfen** aktivieren, wird die ausstehende Objektänderung verworfen.

Objekt aktivieren / deaktivieren

Bei manchen Stammdaten– bzw. Workflow–Objekten besteht die Möglichkeit diese zu aktivieren oder zu deaktivieren. Dazu gibt es in der jeweiligen Änderungsmaske die Checkbox *Aktiv*.

Solange diese Checkbox beim entsprechenden Objekt nicht angeklickt ist, kann es von den Benutzern des Workflow–Systems nicht verwendet werden und scheint auch bei den Auswahllisten im Arbeitskorb (z.B. bei Bearbeiter ändern) nicht auf. Nur wenn diese Checkbox angeklickt wird, ist das Objekt für alle Workflowbenutzer verwendbar.

Deaktivierte Objekte werden in der Tabellendarstellung in grauer und kursiver Schrift dargestellt.

Internationalisierung

Bei applikationsabhängigen Objekten kann der Name in die zur Verfügung stehenden Sprachen übersetzt werden.

Neben dem Feld *Name* wird bei solchen Objekten der für die aktuelle Sprache internationalisierte Name als Link angezeigt. Wenn Sie auf diesen Link klicken, wird die Internationalisierung für alle zur Verfügung stehenden Sprachen angezeigt. Durch das Aktivieren der Schaltfläche *Schließen* wird diese Anzeige wieder geschlossen. Wie Sie die Internationalisierung verändern können, ist in Kapitel [Internationalisierung der Stammdatenobjekte und Objektklassen](#) genauer beschrieben.

2. Das HTML-Interface

Durchführungszeitpunkt: * 05-09-2030 05:15

Eingefügt am: 09-07-2026 14:56

Eingefügt von: Jack Jones

Änderung verwerfen Abbrechen

Id: * walter

Nachname: * Heisenberg

Vorname: Walter

Titel: Dr Amtstitel: Namenszusatz:

Geschlecht: männlich

Beschreibung:

E-Mail: walter@pdts.at

Telefon-Nr.:

Administrationskonto: ☒

Sprache: * Deutsch/Österreich

Aktiv: ☒

Sortierattribut:

Passwort: ●●●●●●●●

Datum der letzten Passwort Änderung: 29-03-2023 18:15

Passwort Policy: ☒ Passwort läuft nie ab. ☐ Muss Passwort beim nächsten Login ändern
☐ Kann Passwort nicht ändern ☐ E-Mail Warnung über Ablauf des Passworts gesendet

WH

Abbildung 2.4: Änderungen zu späteren Zeitpunkt (Detail)

2.2.2 Registerkarte: Historie

Wenn Sie in der Detailansicht eines Objekts auf die Registerkarte *Historie* wechseln, wird eine Tabelle angezeigt, die alle am Objekt durchgeführten Änderungen auflistet (siehe Abbildung 2.5). In dieser Tabelle ist ersichtlich, wer, wann, was mit dem Objekt gemacht hat. Sie können jede einzelne dieser Änderungen im Detail ansehen (Funktion **Anzeigen**). In der Detailansicht können Sie Vergleiche mit vorherigen Änderungen machen. Die gleiche Funktionalität wird z.B. auch bei der Prozesshistorie angeboten (siehe dazu [Benutzerhandbuch](#), Abschnitt [Funktionen der Prozesshistorie](#)).

2.2.3 Registerkarte: Zugriff

Die Registerkarte **Zugriff** der Objekt-Detailansicht zeigt Ihnen, wer, welchen Zugriff direkt oder über eine Berechtigungsliste auf das aktuelle Objekt hat (siehe Abbildung 2.6).

2. Das HTML-Interface

Allgemein	Datei	Zuordnungen	Historie	Zugriff	Referenzen
Änderungsmodus		Akteur	↓ Änderungszeitpunkt		
✎ update		Jones Jack jack	10-07-2026 10:53		
+ insert		Jones Jack jack	10-07-2026 10:45		

Abbildung 2.5: **Registerkarte: Historie**

Sie können diesen Zugriff beliebig ändern (siehe Kapitel [Berechtigungssystem](#)).

Allgemein	Rollen	Vertretungen	Rollenvertretungen	Berechtigungen	Alle Berechtigungen	Einstellungen	Alle Einstellungen	Historie	Zugriff							
↑ Akteur		Gilt nur, wenn Rolle in Org.-Einheit		Recht	Anwenden in Org.-Einheit											
Jones Jack jack				Objekte löschen												
Anzahl der Einträge: 1 0 ausgewählt																
Zugriff über Berechtigungsliste																
Berechtigungsliste:		MyAcilList														
Zugeordnete Rechte:		↑ Akteur	Gilt nur, wenn Rolle in Org.-Einheit		Recht	Anwenden in Org.-Einheit										
		Alle			Objekte bearbeiten											
		Heisenberg Walter Dr walter			Objekt ausführen											
Anzahl der Einträge: 2 0 ausgewählt																
Löschen				Speichern und Schließen		Speichern	Abbrechen									

Abbildung 2.6: **Registerkarte: Zugriff**

2.2.4 Registerkarte: Referenzen

Wenn Sie die Registerkarte **Referenzen** ausgewählt haben, wird Ihnen eine Übersicht über alle Objekte gezeigt, welche auf das aktuelle Objekt referenzieren (siehe Abbildung 2.7). Die Objekte werden hierbei hierarchisch dargestellt. Folgende Funktionen stehen zur Verfügung:

- *Alles ausklappen*: Mit dieser Schaltfläche können Sie alle Objekte und deren un-



tergeordnete Objekte ausklappen.



- *Alles einklappen:* Mit dieser Schaltfläche können Sie alle Objekte und deren untergeordnete Objekte einklappen.



- *Einzelnes Objekt ein- oder ausklappen:* Mit dem Pfeilsymbol links neben dem Objektnamen können Sie die zugehörigen untergeordneten Objekte ein- oder ausklappen. Ein nach rechts zeigender Pfeil zeigt an, dass die untergeordneten Objekte derzeit verborgen sind, während ein nach unten zeigender Pfeil anzeigt, dass die untergeordneten Objekte sichtbar sind.

2.2.5 Weitere Funktionen

In diesem Abschnitt werden jene Funktionen beschrieben, die sich in den Masken ständig wiederholen und im jeweiligen Kontext dieselbe Funktion besitzen.



- **Auswählen:** Durch das Aktivieren der Funktion *Auswählen* neben dem zugehörigen Feld öffnet sich ein HTML-Auswahldialog, mit dem Sie ein Objekt für dieses Feld auswählen können.

Beispiel: Auswahl eines Benutzers, einer Organisationseinheit, etc.



- **Entfernen:** Durch das Aktivieren der Funktion *Entfernen* neben dem zugehörigen Feld wird der Eintrag in diesem Feld entfernt. Diese Funktion steht Ihnen immer zusammen mit der Funktion *Auswählen* zur Verfügung.

Seit *EPworX* 8.0 werden neben bzw. anstatt den Funktionen *Auswählen* und *Entfernen* Dropdown Listen (DOJO) verwendet. Diese können durch Anklicken dieses Symbols aufgeklappt werden. Anschließend können Sie das gewünschte Objekt auswählen bzw. einen Wert eingeben und danach suchen. Sind sehr viele Elemente vorhanden, werden initial nur soviel Einträge angezeigt, wie unter *Konfiguration/Lokalisierung/Tabelleneinträge pro Seite* definiert wurde. Weitere Einträge (durchblättern) erhalten Sie durch die entsprechenden Navigationsfunktionen.



- **Kalender:** Wenn Sie diese Funktion aktivieren, öffnet sich ein Kalender, der Sie dabei unterstützt ein Datum einzufügen. Eine detaillierte Beschreibung zum Kalender finden Sie im [Benutzerhandbuch](#) im Abschnitt [Kalender und Uhrzeit-Auswahl](#).



- **Uhrzeit-Auswahl:** Mit Hilfe dieser Funktion können Sie bei einem Feld, wo eine Uhrzeit angegeben werden kann (alternativ auch händisch), aus einer Liste eine Uhrzeit auswählen. Eine detaillierte Beschreibung zu dieser Funktion finden Sie im [Benutzerhandbuch](#) im Abschnitt [Kalender und Uhrzeitauswahl](#).

AllgemeinBerechtigungenBenutzerHistorieZugriffReferenzen

↕ Alles ausklappen

⌵ Alles einklappen

Referenzen auf Home

▼ Prozesse (6)

Bestellung (1)

Dienstreise (2)

Eingangsrechnungsprüfung (1)

Krankheit (1)

Personalauswahl (1)

Urlaub (1)

▼ Prozessinstanzen (18)

Prozess 13

Prozess 17

Prozess 20

Prozess 21

Prozess 25

Prozess 26

Prozess 27

Prozess 31

Prozess 32

Prozess 4

Prozess 54

Prozess 55

Prozess 57

Prozess 58

Prozess 60

Prozess 78

Prozess 95

Prozess 96

▼ Rollenzuordnungen (6)

Home → Dr Walter Heisenberg

Home → Fritz Huber

Home → Hans Steiner

Home → Jack Jones

Home → Rudolf Berger

Home → Stefan Schmied

Löschen

Speichern und Schließen

Speichern

Schließen

Abbildung 2.7: Registerkarte: Referenzen (von Rollen)

- **Klassenpfad-Checker:** Mit dieser Funktionen können Sie die im zugehörigen Textfeld eingegebene URL überprüfen. Hierbei wird überprüft, ob die Klasse vor-



handen ist. Weiters wird auch auf das Vorhandensein der Methode und ihrer korrekten Signatur überprüft. Nur in gewissen Fällen wird überprüft, ob die Klasse das geforderte Interface implementiert (z.B. Logger-Klasse muss das Interface *com.groiss.log.ILogger* implementieren). Das Symbol für den Klassenpfad-Checker wird grün dargestellt, wenn die angegebene URL im Klassenpfad gefunden werden kann. Ansonsten wird dieser rot angezeigt.

3. Ids, Namen und Internationalisierung

3.1 Ids und Namen

Alle *EPworX* Stammdatenobjekte werden durch ein oder mehrere Attribute eindeutig gekennzeichnet (siehe Tabelle 3.1, Schlüsselattribute).

Darüberhinaus gibt es bei manchen Objekten weitere Bedingungen, um eine eindeutige Identifizierbarkeit von Objekten sicherzustellen (siehe Tabelle 3.1, Eindeutig).

Die Id-Attribute müssen bei einigen Objekten bestimmte Voraussetzungen erfüllen, was die Verwendung von Sonderzeichen betrifft, hier gibt es vier Gruppen:

- Id-Syntax: beginnen mit Buchstabe oder \$, danach können Buchstabe, Ziffer, \, /, _ und \$ vorkommen.
- User-Id: es dürfen die folgenden Sonderzeichen nicht vorkommen: Komma, !, Tabulator, Leerzeichen und Zeilenwechsel
- SQL-Syntax: beginnt mit Buchstabe oder _, danach können Buchstabe, Ziffer und _ vorkommen (es sind keine Umlaute erlaubt).
- Filename-Syntax: es dürfen die folgenden Sonderzeichen nicht vorkommen: /, \, :, *, ?, ; <, >, |, Tabulator

In der Tabelle 3.1 ist pro Objekt in Spalte Syntax angegeben, welche Vorschriften gelten (jeweils für das erste Schlüsselattribut).

Eine weitere Besonderheit ist, dass sich Benutzer und Rollen den Scope teilen, d.h. es kann innerhalb eines *EPworX* –Servers keinen Benutzer und keine Rolle geben, deren Ids übereinstimmen.

3.2 Internationalisierung der Stammdatenobjekte und Objektklassen

In *EPworX* können folgende Stammdaten internationalisiert werden:

- Applikationen

3. Ids, Namen und Internationalisierung

Objektklasse	Schlüsselattribute	Eindeutig	Syntax
Benutzer	Id		User-Id
Organisationseinheit	Id		Id-Syntax
Organisationsklasse	Id	Name	Id-Syntax
OE-Baum	Id	Name	Id-Syntax
Berechtigungsliste	Name		Id-Syntax
Applikation	Id	Name	Id-Syntax
Prozessdefinition	Id+Version	Name+Version	Id-Syntax
Formulartyp	Id+Version		SQL-Syntax
Task	Id+Version		Id-Syntax
Funktion	Id	Name	Id-Syntax
Rolle	Id	Name	Id-Syntax
Recht	Id	Name	Id-Syntax
Objektklasse	Name	Klassenname	
Funktionsgruppe	Id	Name	Id-Syntax
GUI-Konfiguration	Id	Name	Filename-Syntax
Style-Konfiguration	Id	Name	Filename-Syntax
Werteliste	Id		Id-Syntax
Timer	Id		
Mailbox	Id		
Report	Id		Id-Syntax
Webservice-Client	Id		
Webservice-Server	Id		
Nachrichten-Vorlage	Id		
Server	Id	Name	Id-Syntax
Autorisierer	Id		
LDAP-Server	Id		
Externe Datenbank	Id		

Tabelle 3.1: **Objektklassen und ihre Identifier**

- Tasks
- Funktionen
- Rollen
- Rechte

Die Stammdaten von eigenen Applikationen können internationalisiert werden, indem man entsprechende *java.lang.ResourceBundles* implementiert, und diese in das entsprechende Applikationsverzeichnis stellt. Nähere Informationen dazu finden Sie im [Programmierhandbuch](#). Dort finden Sie auch die Information, wie man die Stammdaten der Default-Applikation internationalisieren kann.

4. Definition der Organisationsstruktur

4.1 Rollen

Rollen dienen dazu, sinnvolle Berechtigungsprofile zusammenzufassen. Beispiele für Rollen wären *Sekretär*, *Manager* oder *Brandschutzbeauftragter*. In der Rollenliste stehen Ihnen die in Kapitel [Tabellendarstellung](#) beschriebenen Funktionen zur Verfügung.

Die Objekt-Detailansicht von Rollen gliedert sich in die folgenden Bereiche:

- Allgemein
- Berechtigungen
- Benutzer
- Historie
- Zugriff
- Referenzen

4.1.1 Registerkarte: Allgemein

Informationen der Registerkarte *Allgemein* des Stammdatenobjekts *Rolle* (Mussfelder sind fett dargestellt):

- **Id:** Eindeutige Kurzbezeichnung der Rolle. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#)
- **Name:** Eindeutiger Name der Rolle. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#). Der I18n-Link neben diesem Feld ermöglicht direkt die Anpassung der Ressource (falls in Applikationsdetailmaske - Registerkarte *Eigenschaften* festgelegt). Die Änderungen werden nach Aktivieren der Schaltfläche *Speichern* in der Ressource Datei (Strings) dieser Applikation gespeichert (siehe Abschnitt [Ressourcen Editor](#)).
- **Typ:** In *EPworX* werden drei Arten von Rollen unterschieden:

The screenshot shows a web form for defining roles. The 'Allgemein' tab is active. The form fields are as follows:

- Id:** * home
- Name:** * home (with a link 'I18n: Home' next to it)
- Applikation:** * Default (dropdown menu)
- Typ:** * lokal (dropdown menu)
- Beschreibung:** (empty text area)
- Referenzrolle:** (empty dropdown menu)
- Aktiv:** ☒

Below the form fields, there is a section for 'Änderung durchführen am:' with two date/time pickers. At the bottom, there are four buttons: 'Löschen', 'Speichern und Schließen', 'Speichern', and 'Abbrechen'.

Abbildung 4.1: **Objekt-Detailansicht (Allgemein): Rollen**

- *lokal*: Eine lokale Rolle ist eine Rolle, die genau einer Organisationseinheit zugeordnet wird, d.h. ihr Gültigkeitsbereich beschränkt sich genau auf diese Organisationseinheit.
- *global*: Eine globale Rolle ist keiner Organisationseinheit zugeordnet, und daher in jeder Organisationseinheit verfügbar.
- *hierarchisch*: Eine hierarchische Rolle wird, wie eine lokale Rolle, einer bestimmten Organisationseinheit zugeordnet. Ihr Gültigkeitsbereich erstreckt sich über sämtliche untergeordnete Organisationseinheiten.

Beispiel: Die Rolle *X* ist eine hierarchische Rolle und wird der Organisationseinheit *A* zugeordnet. Die Organisationseinheit *A* hat zwei untergeordnete Organisationseinheiten *Sub1* und *Sub2*. Die Rolle *X* ist auch in den Organisationseinheiten *Sub1* und *Sub2* bekannt.

- Beschreibung: Freier Text.
- Referenzrolle: Durch den Einsatz von Referenzrollen ist es möglich, mehreren Benutzern die selbe Rolle zuzuordnen und dennoch für jeden dieser Benutzer die Berechtigungen, die mit dieser Rolle verbunden sind, individuell festzulegen. Mehrstufige Referenzrollen sind nicht möglich, d.h. für eine Rolle, die als Referenzrolle verwendet wird, kann keine Referenzrolle eingetragen werden. Wie eine

Referenzrolle angewendet werden kann, wird anhand des folgenden Beispiels beschrieben:

1. Beispiel: In einem Unternehmen gibt es mehrere Sekretäre. Manche von ihnen haben mehrere Rechte, andere wiederum haben weniger Rechte. In der Prozessdefinition wird als Akteur, für eine Aufgabe die von einem Sekretär erledigt werden soll, die Rolle *sek* eingetragen.

Ein Sekretär *V* mit vielen Rechten bekommt die Rolle *sek_viele_rechte* und ein Sekretär *W* mit weniger Rechten bekommt die Rolle *sek_wenige_rechte* zugeordnet. Bei der Rolle *sek_viele_rechte* UND bei der Rolle *sek_wenige_rechte* wird als Referenzrolle die Rolle *sek* eingetragen.

Wird nun der Prozess abgearbeitet, landet der Task im Rollenarbeitskorb beider Sekretäre (da beiden die Rolle *sek* als Referenzrolle zugeordnet wurde).

Welche Aktionen der Sekretär *V* auf den Task und die damit verbundenen Formulare anwenden kann, wird jedoch durch die Rolle *sek_viele_rechte* festgelegt, und welche Aktionen der Sekretär *W* auf den Task und die damit verbundenen Formulare anwenden kann, wird durch die Rolle *sek_wenige_rechte* festgelegt.

2. Beispiel: Im Unternehmen gibt es Sekretäre und einen Abteilungsleiter. In der Prozessdefinition *P* wird als Akteur die Rolle *abt_sek* eingetragen. Diese Rolle wird bei der Rolle *abt* und *sek* als Referenzrolle ausgewählt. Die Benutzer haben weiterhin nur entweder die Rolle *abt* bzw. *sek*, jedoch können Sekretäre und Abteilungsleiter den Prozess *P* starten und haben somit die gleichen Rechte im 1. Schritt.

- **Applikation:** Innerhalb dieser Applikation ist die Rolle bekannt.
- Aktiv: siehe Kapitel [Objekt aktivieren / deaktivieren](#).
- Änderung durchführen am: siehe Kapitel [Änderungen zu späterem Zeitpunkt](#).

In dieser Ansicht stehen Ihnen die Funktionen *I18n*, *Aktivieren* und *Änderungen zu späteren Zeitpunkt* zur Verfügung.



Hinweis: Wenn Sie eine Rolle **löschen**, werden auch die dieser Rolle zugeordneten Berechtigungen gelöscht.

4.1.2 Registerkarte: Berechtigungen

In der Registerkarte **Berechtigungen** können Sie beliebig viele Rechte dieser Rolle zuordnen. Benutzer, die dieser Rolle zugeordnet sind, haben wiederum die hier festgelegten Berechtigungen. Eine detaillierte Beschreibung zum Berechtigungsdialog und zum Tabellen-Filter *Berechtigungen bei Akteur anzeigen* finden Sie in Kapitel [Berechtigungssystem](#).

4.1.3 Registerkarte: Benutzer

In der Registerkarte **Benutzer** sehen Sie eine Auflistung aller Benutzer, die dieser Rolle zugeordnet sind. Sie können sich die Details dieser Zuordnung ansehen, eine neue Zuordnung erstellen oder bestehende editieren.

4.1.4 Systemrollen

In *EPworX* sind folgende Systemrollen immer vorhanden:

- **Alle (all)**: Eine nützliche Rolle, die allen Benutzer zugeordnet werden kann. Damit können allen Benutzern zugängliche Funktionen mit dieser Rolle verbunden werden (z.B. hat ein Prozess, den jeder starten darf, als Akteur des ersten Tasks die Rolle *all*).
- **Sys (sys)**: Die Rolle *sys* wird an Administratoren vergeben. Sie ermöglicht das Durchführen sämtlicher Systemadministrationsfunktionen, wie Stammdaten ändern, Prozesse abbrechen, u.ä.
- **home (home)**: Die lokale Rolle *home* verbindet einen Benutzer mit seiner *home*-Organisationseinheit. Ein Benutzer kann maximal in einer Organisationseinheit eine *home*-Rolle haben.
- **OE (dept)**: Die lokale Rolle *dept* wird als *Eingangskorb* der Organisation verwendet. Wenn man einen Workflow an eine Organisationseinheit schicken will, ohne den Empfänger zu kennen, kann man die Rolle *dept* angeben.
- **Benutzer-Administrator (useradmin)**: Die lokale Rolle *useradmin* wird für die eingeschränkte Administration verwendet. Nähere Informationen dazu finden Sie im Abschnitt [Eingeschränkte Administration](#).

4.2 Rechte

Rechte werden verwendet, um bestimmte Funktionen nur einem eingeschränkten Benutzerkreis zur Verfügung zu stellen. Die Zuordnung zu Benutzern erfolgt entweder direkt oder über Rollen. Eine ausführliche Beschreibung des Berechtigungssystems finden Sie in Kapitel [Berechtigungssystem](#). In der Rechteliste stehen Ihnen die in Kapitel [Tabellendarstellung](#) beschriebenen Funktionen zur Verfügung.

Die Objekt-Detailansicht von Rechten gliedert sich in die folgenden Bereiche:

- Allgemein
- Berechtigungen
- Historie
- Zugriff
- Referenzen

4.2.1 Registerkarte: Allgemein

The screenshot shows a web interface for editing an object. At the top, there are five tabs: 'Allgemein' (selected), 'Berechtigungen', 'Historie', 'Zugriff', and 'Referenzen'. Below the tabs is a form with the following fields:

- Id: ***: A text input field containing 'itsm_startProblem'.
- Name: ***: A text input field containing 'startproblem'.
- Applikation: ***: A dropdown menu with 'ITSM' selected.
- Beschreibung:**: A large, empty text area.

At the bottom of the form, there are four buttons: 'Löschen', 'Speichern und Schließen', 'Speichern', and 'Abbrechen'.

Abbildung 4.2: Objekt-Detailansicht (Allgemein): Rechte

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Id:** Eindeutige Kurzbezeichnung des Rechtes. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).
- **Name:** Eindeutiger Name des Rechtes. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#). Der I18n-Link neben diesem Feld ermöglicht direkt die Anpassung der Ressource (falls in Applikationsdetailmaske - Registerkarte *Eigenschaften* festgelegt). Die Änderungen werden nach Aktivieren der Schaltfläche *Speichern* in der Ressource Datei (Strings) dieser Applikation gespeichert (siehe Abschnitt [Ressourcen Editor](#)).
- **Applikation:** Innerhalb dieser Applikation ist das Recht bekannt.
- **Beschreibung:** Freier Text.

In dieser Ansicht steht Ihnen die Funktion *I18n* zur Verfügung.

4.2.2 Registerkarte: Berechtigungen

In der Registerkarte *Berechtigungen* sehen Sie eine Auflistung aller Benutzer und Rollen, die das gerade ausgewählte Recht besitzen. Sie können sich die Details dieser

Berechtigungszuordnung ansehen. Ist dieses Recht auf ein bestimmtes Objekt beschränkt, so wird dieses in der Spalte *Betroffenes Objekt* angezeigt.

4.2.3 Standardrechte

In *EPworX* sind standardmäßig folgende Rechte vorhanden:

- **Administration (admin):** Ermöglicht den Zugang zur Administration.
- **Akteur ändern (set_agent):** Erforderlich, um den Akteur von Prozessinstanzen zu bearbeiten. Mit diesem Recht werden auch alle Berechtigungen erteilt, die durch das Recht *Prozessinstanzen ansehen* definiert sind.
- **Berechtigungen bearbeiten (edit-acl):** Bearbeiten von Berechtigungen
- **Filter freigeben (share_filter):** Um anderen Benutzern die Verwendung des Objekts zu gestatten (z.B. Arbeitskorb-Filter)
- **Freigabe (share):** Um anderen Benutzern die Verwendung des Objekts zu gestatten (z.B. Arbeitskorb-Filter)
- **Kalendertermine ansehen (viewCal):** Um auch Kalendertermine zu sehen, die nicht die eigenen sind.
- **Kalendertermine bearbeiten (editCal):** Berechtigung zum Bearbeiten von Kalendereinträgen.
- **Kalendertermine erzeugen (insertCal):** Um Kalendertermine zu erzeugen.
- **Konfiguration ansehen (view_conf):** Konfiguration ansehen.
- **Konfiguration bearbeiten (conf):** Konfiguration bearbeiten.
- **Named User (named_user):** Um Benutzer als *Named User* zu qualifizieren und damit das Anmelden unabhängig von der Anzahl der aktuell angemeldeten Benutzer immer zu erlauben.
- **Objekt ausführen (execute):** Ausführen einer Funktion.
- **Objekte ansehen (view):** Anzeigen eines Objekts.
- **Objekte bearbeiten (edit):** Bearbeiten eines Objekts.
- **Objekte erzeugen (create):** Erzeugen eines Objekts.
- **Objekte löschen (delete):** Löschen eines Objekts.
- **OE-Verweisordner verwalten (admin_dept_reffolder):** Ein Benutzer mit dieser Berechtigung kann folgendes:
 - Erstellen einer Wurzel für den Verweisordner-Baum einer OE
 - Erstellen von allen anderen Verweisordner für eine OE
 - Verweisordner dieser OE bearbeiten

- Verweisordner dieser OE löschen
 - Hinzufügen bzw. Entfernen von Inhalten zu bzw. aus Verweisordnern einer OE
 - Hinzufügen bzw. Entfernen von Unterordnern zu bzw. von Verweisordnern einer OE
 - Bearbeiten der Berechtigungen eines Verweisordners
- **Ordner freigeben Anwender (grant_share_user):** Erlaubt die Vergabe einer Berechtigung auf einen Verweisordner für eine globale Rolle als Akteur.
- **Ordner freigeben OE (grant_share_dept):** Erlaubt die Vergabe einer Berechtigung auf einen Verweisordner für eine lokale oder hierarchische Rolle als Akteur.
- **Ordner freigeben System (grant_share_system):** Erlaubt die Vergabe einer Berechtigung auf einen Verweisordner für einen Benutzer als Akteur.
- **Plan bearbeiten (edit_plan):** Bearbeiten eines Prozessplans.
- **Prozesshistorie ansehen (view_history):** Nur ansehen der Prozesshistorie.
- **Prozessinstanzen ansehen (view_procinst):** Ansehen von Prozesshistorie, Liste der Dokumente und Notizen sowie alle Prozessformulare und -versionen.
- **Prozessinstanzen bearbeiten (proc_inst):** Erforderlich, um Prozessinstanzen abubrechen oder den Akteur von Prozessinstanzen zu bearbeiten. Mit diesem Recht werden alle Berechtigungen erteilt, die durch das Recht *Prozessinstanzen ansehen* definiert sind. Das Recht wird entweder für eine Organisationseinheit oder eine Prozessdefinition vergeben. Im ersten Fall wirkt es auf alle Instanzen in der Organisationseinheit, im zweiten Fall auf alle Instanzen der Prozessdefinition.
- **Schritt abbrechen (abort_step):** Um den Task einer Prozessinstanz abubrechen.
- **Statistik (stat):** Erstellung von Reports mit dem *Report Designer*.
- **Suchbar (searchable):** Um in Formularen suchen zu können bzw. Liste der gespeicherten Reports anzuzeigen
- **Suche (find):** Berechtigung einen Verweisordner zu finden.
- **Vertreter erfassen (grant_subst):** Berechtigung zur Definition von persönlichen und Rollenvertretungen.

4.3 Benutzer

Alle Personen, die mit *EPworX* arbeiten möchten, müssen als Benutzer eingetragen werden. In der Benutzerliste stehen Ihnen die unter [Tabellendarstellung](#) beschriebenen Funktionen zur Verfügung. In der erweiterten Suche können Sie durch verschiedene Suchattribute die Anzahl der angezeigten Benutzer in der Benutzerliste beeinflussen.

Wenn Sie zum Beispiel als Suchattribut eine Organisationseinheit angeben, werden nur jene Benutzer aufgelistet, die eine Rolle in dieser OE besitzen.

Die Objekt-Detailansicht von Benutzern gliedert sich in die folgenden Bereiche:

- Allgemein
- Rollen
- Vertretungen
- Rollenvertretungen
- Berechtigungen
- Alle Berechtigungen
- Einstellungen
- Alle Einstellungen
- Historie
- Zugriff

4.3.1 Registerkarte: Allgemein

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Id:** Die eindeutige Kurzbezeichnung des Benutzers. Die Id des Benutzers muss beim Einloggen in das System im Feld *Benutzer-Id* eingegeben werden. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).
- **Nachname:** Der Nachname des Benutzers.
- Vorname: Der Vorname des Benutzers.
- Titel: Der akademische oder sonstige Titel.
- Amtstitel: Der Amtstitel, z.B. Hofrat.
- Namenszusatz: Ein beliebiger Text, der dem Namen nachgestellt wird, z.B. Sen.
- Geschlecht: Auswahl zwischen männlich, weiblich und divers.
- Beschreibung: Freier Text.
- E-Mail: Die E-Mail Adresse des Benutzers.
- Telefon-Nr: Die Telefonnummer des Benutzers.
- **Sprache:** Hier ist die Sprache auswählbar, in der das Benutzerinterface erscheint. Es werden jene Sprachen zur Auswahl angeboten, in die *EPworX* übersetzt wurde.

4. Definition der Organisationsstruktur

The screenshot shows a web-based user management interface. The 'Allgemein' (General) tab is selected, displaying a form for user details. The form is organized into sections: personal information, administrative settings, and password management. A circular profile picture placeholder with the initials 'WH' is located on the right side of the form. At the bottom, there are four buttons: 'Löschen' (Delete), 'Speichern und Schließen' (Save and Close), 'Speichern' (Save), and 'Abbrechen' (Cancel).

Field	Value
Id: *	walter
Nachname: *	Heisenberg
Vorname:	Walter
Titel:	Dr
Amtstitel:	
Namenszusatz:	
Geschlecht:	männlich
Beschreibung:	
E-Mail:	
Telefon-Nr.:	
Administrationskonto:	☒
Sprache: *	Deutsch/Österreich
Aktiv:	☒
Sortierattribut:	
Passwort:	••••••••
Datum der letzten Passwort Änderung:	29-03-2023 18:15
Passwort Policy:	☒ Passwort läuft nie ab. ☐ Muss Passwort beim nächsten Login ändern ☐ Kann Passwort nicht ändern ☐ E-Mail Warnung über Ablauf des Passworts gesendet
Änderung durchführen am:	📅 🕒

Abbildung 4.3: Objekt-Detailansicht (Allgemein): Benutzer

- Aktiv: Kapitel [Objekt aktivieren / deaktivieren](#).
- Sortierattribut: Hier kann ein freier Text eingetragen werden. Dieser Text kann bei der Implementierung von eigenen Applikationen für eine attributunabhängige Sortierung (z.B. zuerst alle Ärzte, dann alle dipl.Krankenschwestern, dann alle Krankenpfleger, ... in der Benutzerliste auflisten) verwendet werden.
- Profilbild: Hier können Sie für den aktuellen Benutzer ein Profilbild hochladen, das im Smartclient beim Benutzerprofil angezeigt wird. Standardmäßig wird hier ein Platzhalter für das Profilbild mit den Initialen des Benutzers angezeigt.
- Passwort: Das Passwort des Benutzers. Das Passwort muss beim Einloggen in das System im Feld *Passwort* eingegeben werden.
- Datum der letzten Passwortänderung: Datum, wann das Passwort das letzte mal geändert wurde.
- Passwort-Policy:
 1. Passwort läuft nie ab: Das Passwort dieses Benutzers muss nie geändert werden.

2. Muss Passwort beim nächsten Login ändern: Der Benutzer wird beim nächsten Login aufgefordert sein Passwort zu ändern.
3. Kann Passwort nicht ändern: Der Benutzer sollte sein Passwort nicht ändern können.
4. E-Mail Warnung über Ablauf des Passworts gesendet: Zeigt, ob ein E-Mail durch den Timer *PasswordExpiration* an die Benutzer E-Mail Adresse gesendet wurde. Diese Checkbox wird automatisch zurückgesetzt, falls das Passwort geändert wurde.

- Änderung durchführen am: siehe Kapitel [Änderungen zu späterem Zeitpunkt](#).

In dieser Ansicht stehen Ihnen die Funktionen *Aktivieren* und *Änderungen zu späteren Zeitpunkt* zur Verfügung.



Hinweis: Wird ein Benutzer gelöscht, so werden auch die ihm zugeordneten Rollen, Rechte und Vertreter gelöscht.

4.3.2 Registerkarte: Rollen

In der Registerkarte *Rollen* sehen Sie alle Rollen, die dem Benutzer zugeordnet sind. Sie können in dieser Ansicht die Rollen des Benutzers bearbeiten. Es stehen Ihnen hierzu die Funktionen in der Werkzeugliste zur Verfügung. Wenn Sie eine neue Rollenzuordnung einfügen oder eine bestehende ändern, öffnet sich ein neuer Objektdialog.

Über diesen Dialog legen Sie außerdem die Rollenvertreter des Benutzers fest. Aktivieren Sie hierfür die Registerkarte *Vertretungen* im Objektdialog der Rollenzuordnung. Hier sehen Sie alle Vertretungen für die ausgewählte Rollenzuordnung. Sie können in dieser Ansicht die Vertretungen bearbeiten. Es stehen Ihnen hierzu die Funktionen in der Werkzeugliste zur Verfügung. Wenn Sie eine neue Vertretung einfügen oder eine bestehende ändern, öffnet sich ein weiterer Objektdialog.

Rollenzuordnungen bearbeiten

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Benutzer:** Der Benutzer, dem die Rolle zugeordnet werden soll - standardmäßig jener Benutzer, in dessen Detailmaske man sich befindet.
- **Rolle:** Die Rolle, die dem aktuellen Benutzer zugeordnet werden soll.
- **Organisationseinheit:** Der Benutzer hat die Rolle in dieser Organisationseinheit. Wird dem Benutzer eine hierarchische Rolle zugeordnet, so hat er diese Rolle auch in allen, der hier ausgewählten Organisationseinheit, untergeordneten Organisationseinheiten (siehe [Rollen](#)).
Wenn man dem Benutzer eine globale Rolle zuordnet, so kann dieses Feld leer gelassen werden. In diesem Fall hat der Benutzer die Rolle in allen Organisationseinheiten.

- Aktiv: siehe Kapitel [Objekt aktivieren / deaktivieren](#).
- Änderungen durchführen am: siehe Kapitel [Änderungen zu späterem Zeitpunkt](#)

Stellvertreter für einen Benutzer innerhalb einer bestimmten Rolle definieren

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt);

- **Vertreter:** Der Benutzer, der die aktuelle Rolle vertreten soll.
- Von: Ab diesem Datum beginnt die Vertretung. Wenn Sie hier kein Datum eingeben, beginnt die Vertretung sofort. Durch das Anklicken des Kalender-Icons neben dem Feld wird ein Kalender angezeigt, der Sie bei der Eingabe des Datums unterstützt.
- Bis: Mit diesem Datum endet die Vertretung. Wenn Sie hier kein Datum eingeben, endet diese Vertretung erst, wenn sie explizit aufgehoben wird. Durch das Anklicken des Kalender-Icons neben dem Feld wird ein Kalender angezeigt, der Sie bei der Eingabe des Datums unterstützt.

Hinweis: Durch den Timer *CurrentSubstitutes* wird bei Angabe eines Von-/Bis-Datums die Vertretung aktiviert bzw. wieder deaktiviert (siehe Abschnitt [Übersicht über Standard-Timer](#)).

4.3.3 Registerkarte: Vertretungen

In dieser Ansicht sehen Sie die *Persönlichen Vertreter* des Benutzers, d.h. alle Benutzer, die den ausgewählten Benutzer vertreten und die *Vertretungen* des Benutzers, d.h. alle Benutzer, die vom ausgewählten Benutzer vertreten werden.

Sie können in dieser Ansicht die *Persönlichen Vertreter* des Benutzers bearbeiten. Es stehen Ihnen hierzu die Funktionen in der Werkzeugliste zur Verfügung. Wenn Sie einen neuen Vertreter einfügen oder einen bestehenden ändern, öffnet sich ein neuer Objektdialog.

Persönliche Vertreter für einen Benutzer definieren

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Vertreter:** Der Benutzer, der den aktuellen Benutzer vertreten soll.
- Vertretung inkl. Rollen: Ist diese Checkbox angekreuzt, so wird der aktuelle Benutzer auch in all seinen Rollen vertreten.
- Von und Bis: Wie bei Stellvertreter für einen Benutzer in Kapitel [Stellvertreter für einen Benutzer innerhalb einer bestimmten Rolle definieren](#). Durch den Timer *CurrentSubstitutes* wird bei Angabe eines Von-/Bis-Datums die Vertretung aktiviert bzw. wieder deaktiviert (siehe Abschnitt [Übersicht über Standard-Timer](#)).
- Änderungen durchführen am: siehe Kapitel [Änderungen zu späterem Zeitpunkt](#)

4.3.4 Registerkarte: Rollenvertretungen

In dieser Ansicht erscheint eine HTML-Seite, in der Ihre Rollenvertretungen angezeigt werden.

Diese HTML-Seite setzt sich aus 2 Bereichen zusammen:

- Im ersten Bereich *Benutzer, die meine Rollen vertreten* sind alle Benutzer aufgelistet, von denen Sie in einer Rolle vertreten werden. Wird eine Aufgabe an eine Rolle weitergeleitet, die Sie innehaben und in der Sie von einem Benutzer vertreten werden (und zwar inkl.Rollen), so scheinen diese Aufgaben im Rollenarbeitskorb Ihres Vertreters auf.
- Im zweiten Bereich *Benutzer, deren Rollen ich vertrete* sind alle Benutzer aufgelistet, die von Ihnen in einer Rolle vertreten werden. Wird eine Aufgabe an eine Rolle weitergeleitet, die einem Benutzer zugeordnet ist, den Sie vertreten (und zwar inkl.Rollen), so scheinen diese Aufgaben in Ihrem Rollenarbeitskorb auf.

Die Tabelle *Benutzer, die meine Rollen vertreten* enthält die folgenden Informationen:

- **Aktiv:** Zeigt an, ob die Rolle aktiv (= grüner Punkt) oder inaktiv (= roter Punkt) ist.
- **Rolle:** Die Rolle, in der Sie von dem Benutzer vertreten werden.
- **Organisationseinheit:** In dieser Organisationseinheit besitzen Sie die Rolle.
- **Benutzer:** Der Benutzer, der Sie in einer bestimmten Rolle vertritt.
- **Von:** In dieser Spalte ist eingetragen, ab welchem Zeitpunkt die Rollenvertretung wirksam ist.
- **Bis:** In dieser Spalte ist eingetragen, bis zu welchem Zeitpunkt die Rollenvertretung wirksam ist.

Die Tabelle *Benutzer, deren Rolle ich vertrete* enthält die folgenden Informationen:

- **Aktiv:** Zeigt an, ob die Rolle aktiv (= grüner Punkt) oder inaktiv (= roter Punkt) ist.
- **Rolle:** In dieser Spalte ist die Bezeichnung für die Ihnen (über einen Vertreter) zugeordnete Rolle zu finden.
- **Organisationseinheit:** In dieser Spalte ist ersichtlich, in welcher Organisationseinheit Sie die entsprechende Rolle vertreten.
- **Benutzer:** Benutzer, den Sie in einer bestimmten Rolle vertreten.
- **Von:** In dieser Spalte ist eingetragen, ab welchem Zeitpunkt die Rollenvertretung wirksam ist.
- **Bis:** In dieser Spalte ist eingetragen, bis zu welchem Zeitpunkt die Rollenvertretung wirksam ist.

4.3.5 Registerkarte: Berechtigungen

In dieser Registerkarte können Sie die Berechtigungen des aktuellen Benutzers bearbeiten. Es stehen Ihnen hierzu die Funktionen der Werkzeugleiste zur Verfügung. Informationen zum Einfügen einer neuen bzw. Bearbeiten einer bestehenden Berechtigung und zum Tabellen-Filter *Berechtigungen bei Akteur anzeigen* sind im Kapitel [Berechtigungssystem](#) beschrieben.

4.3.6 Registerkarte: Alle Berechtigungen

Diese Ansicht zeigt Ihnen alle Berechtigungen an, die der Benutzer hat. Berechtigungen können Benutzern über Rollen oder direkt zugeordnet werden. Eine allgemeine Einführung in das Berechtigungssystem finden Sie in Kapitel [Berechtigungssystem](#). Sie können diese Tabelle mit den Funktionen in der Werkzeugleiste bearbeiten.

Weiters können Sie die Berechtigungen des Benutzers zu einem bestimmten Zeitpunkt ansehen. Geben Sie dazu ein beliebiges Datum (und Uhrzeit) ein und starten die Suche.

4.3.7 Registerkarte: Einstellungen

Hier ist es dem Systemadministrator möglich die lokalen Einstellungen eines Benutzers zu bearbeiten. Um welche Einstellungen es sich dabei handelt entnehmen Sie dem [Benutzerhandbuch](#). Folgende Einstellungen sind nur über Administration einstellbar:

- **Tabelleneinträge pro Seite:** Dieser Parameter wird nur in der alten Oberfläche verwendet! Um die Übersicht in Ihrem Arbeitskorb zu verbessern, ist es möglich hier einzustellen, wie viele Einträge maximal im Arbeitskorb aufgelistet werden, auch wenn mehr als diese vorhanden sind.
Existieren mehrere Einträge, so können diese durch das Navigieren mit den Links `/ < 1 2 3 > > /` erreicht werden. In diesem Beispiel gibt es drei Seiten, die mit Arbeitskorbeinträgen befüllt sind. Durch das Anklicken des Links *1* wird zur ersten Seite navigiert, durch das Anklicken des Links *2* wird zur zweiten Seite navigiert, usw.
- **Startseite:** Sie können hier eine URL eintragen, die als Startseite angezeigt wird, wenn Sie sich bei *EPworX* anmelden.

Hinweis: Ist keine Zeitzone ausgewählt, wird standardmäßig die Zeitzone des Betriebssystems des Clients angewendet.

4.3.8 Registerkarte: Alle Einstellungen

Hier ist es dem Systemadministrator möglich die lokalen Einstellungen eines Benutzers anzusehen und zu entfernen.

4.3.9 Toolbarfunktion GUI-Konfigurationen

Mit Hilfe dieser Funktionen können Sie für einen Benutzer alle zugeordneten GUI-Konfigurationen anzeigen lassen. Die Zuordnungen erfolgt über den gleichnamigen Reitern bei einem GUI-Konfigurationsobjekt (siehe Abschnitt [Registerkarte: Zuordnungen](#)). Diese zugeordneten GUI-Konfigurationen werden als Auswahl in der Arbeitskorb-Benutzeroberfläche zwischen dem Hilfe-Symbol und der Logout-Funktion angezeigt.

4.3.10 Berechtigungen überprüfen

Diese HTML-Seite ermöglicht es Ihnen festzustellen, ob für einen Benutzer ein bestimmtes Recht vergeben wurde.

Wählen Sie zunächst einen **Benutzer** aus, dessen Berechtigungen Sie überprüfen wollen. Danach wählen Sie das **Recht** aus, das Sie überprüfen wollen. Sie können optional das Recht auf die *Objektklasse* (und *Objekt*), *Formularklasse* oder auf ein *Dokument* noch weiter einschränken. Nachdem Sie die Schaltfläche **Überprüfen** aktiviert haben, wird die Überprüfung gestartet. Das Ergebnis der Überprüfung wird Ihnen im unteren Bereich der Seite angezeigt.

Berechtigungen überprüfen

Benutzer: *

Heisenberg Walter Dr walter

Recht: *

Objekte ansehen

Anwenden auf

☒ Objektklasse:

Objekt / Berechtigungsliste:

☐ Formularklasse:

☐ Dokument:

☐ Sonstiges:

Überprüfen

Berechtigungsüberprüfung für dummyObject ergibt: ● Erlaubt

Berechtigung erhalten über

Akteur	Recht	Objektklasse	Objekt	Zugriff
Sys	Objekte bearbeiten			● Erlaubt
Sys	Objekte ansehen			● Erlaubt
Externer Benutzer	Objekte ansehen			● Erlaubt

Abbildung 4.4: Berechtigungen überprüfen

4.3.11 Abgelaufene Passwörter

Ist in der Passwortpolicy in der Systemkonfiguration (siehe [Installationshandbuch](#)) eine maximale Gültigkeitsdauer für Passwörter definiert, liefert Ihnen diese Funktion eine Liste mit allen Benutzern, deren Passwort bereits die maximale Gültigkeitsdauer überschritten hat.

4. Definition der Organisationsstruktur

Abgelaufene Passwörter	
3 Benutzer mit abgelaufenem Passwort gefunden.	
Id	Name
berger	Rudolf Berger
huber	Fritz Huber
schmied	Stefan Schmied

Abbildung 4.5: **Abgelaufene Passwörter**

4.3.12 Datenschutz-Funktionen

Mit Hilfe dieser Maske können Sie für einen ausgewählten Benutzer folgende Aktionen durchführen:

- Durch das Aktivieren der Schaltfläche *Referenzen* alle Vorkommnisse anzeigen lassen, wo der Benutzer referenziert wird. Das Ergebnis der Abfrage können Sie mit Hilfe der Toolbarfunktion *Druckansicht* und der Browser-Druckfunktionalität ausdrucken.
- Den ausgewählten Benutzer anonymisieren, d.h.
 - seine ID wird auf den String *nn+OID* umgeändert,
 - alle anderen Felder des Benutzers zurückgesetzt (Nachname, Vorname, Telefonnummer, etc.),
 - der Benutzer wird inaktiv gesetzt,
 - sein persönlicher DMS-Ordner wird auf die anonymisierte Benutzer-ID gesetzt und
 - alle Historie-Einträge (Logeinträge), wo der Benutzer involviert war, werden gelöscht.

Hinweis: Bei einer projektspezifische Verwendung von Benutzerdaten (z.B. Benutzerdaten werden in einem eigenen Formularfeld eines Prozessformulars gespeichert) müssen die jeweiligen Applikationsentwickler dafür sorgen, dass die Benutzerdaten anonymisiert werden! Zu diesem Zwecke bietet das Interface `com.groiss.wf.ApplicationAdapter` folgende Methode an: `public void onAnonymize(User u, String newId)`

4.4 Organisationseinheiten

Um die Aufbauorganisation Ihres Unternehmens verwalten zu können, bietet *EPworX* zwei Konstrukte an: Organisationseinheiten und OE-Bäume. Organisationseinheiten oder kurz *OE*s entsprechen den Abteilungen in Unternehmen. OE-Bäume bilden das Organigramm des Unternehmens ab und werden in Kapitel [OE-Baum](#) beschrieben.

In der Liste der Organisationseinheiten stehen Ihnen die unter [Tabellendarstellung](#) beschriebenen Funktionen zur Verfügung.

4. Definition der Organisationsstruktur

Die Objekt-Detailansicht von Organisationseinheiten gliedert sich in die folgenden Bereiche:

- Allgemein
- Übergeordnete Org.-Einheiten
- Rollen
- Historie
- Zugriff
- Referenzen

4.4.1 Registerkarte: Allgemein

Allgemein Übergeordnete Org.-Einheiten Rollen Historie Zugriff Referenzen

Id: *

pdts

Name: *

PDTS

Beschreibung:

PDTS is a trusted software and service company within the Frequentis Group, delivering reliable digital solutions for communication, process digitalization, and critical infrastructure environments.

E-Mail:

office@pdts.at

Telefon-Nr.:

+43 (0) 664 60830

Adresse:

Strutzmanstraße 10/4
9020 Klagenfurt
Austria

Externe Org.-Einheit:

☐

Unselbständig:

☐

EPworX installiert:

☒

Organisationsklasse:

Aktiv:

☒

Nachfolger Org.-Einheit:

Sortierattribut:

Änderung durchführen am:





Löschen

Speichern und Schließen

Speichern

Abbrechen

Abbildung 4.6: **Objekt-Detailansicht: Organisationseinheiten**

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

4. Definition der Organisationsstruktur

- **Id:** Eindeutige Kurzbezeichnung der OE. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).
- **Name:** Eindeutiger Name der OE. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).
- **Beschreibung:** Freier Text.
- **E-Mail:** Die E-Mail-Adresse der OE.
- **Tel-Nr:** Die Telefonnummer der OE.
- **Adresse:** Die Anschrift der OE.
- **Externe Org.-Einheit:** Ist diese Checkbox angekreuzt, so handelt es sich bei der aktuellen OE um eine externe OE, d.h. um eine Organisationseinheit außerhalb Ihres Unternehmens. Dadurch kann während der Abarbeitung eines Prozesses kein Task an diese OE oder eine(n) Rolle (Benutzer) in dieser OE weitergeleitet werden.
- **Unselbständig:** Dieses Attribut dient der OE-Untergliederung. Es wird verwendet, um OEs zu kennzeichnen, die zu einer anderen OE gehören, aber keine untergeordneten OEs sind (Bsp.: Stabsstellen in einer Organisation, etc.). In der Hierarchie (OE-Baum) werden diese unselbständigen OEs jedoch trotzdem der OE, der sie zugeordnet sind, untergeordnet.
Interpretiert wird dieses Attribut nur innerhalb des Berechtigungssystems (siehe Kapitel [Berechtigungssystem](#)) von *EPworX*. Es wird dort dafür verwendet, den OE-Scope von Berechtigungen einzuschränken.
- **EPworX installiert:** Ist diese Checkbox angekreuzt, so ist in der aktuellen OE *EPworX* installiert. Sonst ist in der OE *EPworX* nicht installiert, d.h. es kann während der Abarbeitung eines Prozesses kein Task an diese OE oder eine(n) Rolle(Benutzer) in dieser OE weitergeleitet werden.
- **Organisationsklasse:** Die Organisationsklasse der OE. Weitere Informationen zu Organisationsklassen finden Sie im Kapitel [Organisationsklassen](#).
- **Aktiv:** siehe Kapitel [Objekt aktivieren / deaktivieren](#).
- **Nachfolger Org.Einheit:** Im Rahmen von Umstrukturierungen in Ihrem Unternehmen kann es vorkommen, dass Organisationseinheiten durch andere Organisationseinheiten ersetzt werden. Hier kann festgehalten werden, welche OE die aktuelle OE ersetzt hat.
- **Sortierattribut:** Hier kann ein freier Text eingetragen werden. Dieser Text kann bei der Implementierung von eigenen Applikationen für eine attributsunabhängige Sortierung verwendet werden.
- **Änderungen durchführen am:** siehe Kapitel [Änderungen zu späterem Zeitpunkt](#)

In dieser Ansicht stehen Ihnen die Funktionen *Aktivieren* und *Änderungen zu späterem Zeitpunkt* zur Verfügung.

4.4.2 Registerkarte: Übergeordnete Org.-Einheiten

Eine Organisationseinheit kann in mehrere OE-Bäume eingehängt werden. Daher kann eine OE, die normalerweise nur zu einer übergeordneten OE gehören kann, mehrere übergeordnete OEs besitzen (pro OE-Baum genau eine). In dieser Registerkarte sehen Sie, welche übergeordneten OE's die aktuelle Organisationseinheit hat.

4.4.3 Registerkarte: Rollen

In dieser Registerkarte werden alle Rollenzuordnungen für die aktuelle OE aufgelistet.

In der Tabelle *Direkte Rollen* wird angezeigt, welche Rolle welchem Benutzer in der aktuellen OE direkt zugeordnet ist.

In der Tabelle *Von übergeordneten OEs geerbte Rollen* wird angezeigt, welche hierarchische Rolle, in welcher Applikation, welchem Benutzer durch welche übergeordnete OE in die aktuelle OE vererbt wurde. Genauere Informationen zu hierarchischen Rollen und deren Gültigkeitsbereich finden Sie in Kapitel [Rollen](#).

Hinweis: Wenn entweder die Rollenzuordnung oder eines der referenzierten Objekte (Benutzer, Rolle, Organisationseinheit) inaktiv ist, wird die Zeile kursiv und grau angezeigt.

4.5 OE-Baum

OE-Bäume werden in *EPworX* verwendet um das Organigramm eines Unternehmens abzubilden. Durch das Einhängen von Organisationseinheiten in einen OE-Baum wird die Hierarchie der einzelnen OEs abgebildet. Dabei kann eine OE in einem OE-Baum mehrere untergeordnete OEs, aber nur eine übergeordnete OE, besitzen. Eine OE kann auch in mehrere OE-Bäume eingehängt werden.

Es ist möglich mehrere OE-Bäume zu definieren und sie dann Applikationen zuzuordnen. Ein OE-Baum kann in mehreren Applikationen verwendet werden, eine Applikation verwendet aber immer genau einen OE-Baum.

Default-OE-Baum: Nach der Installation enthält das System genau einen OE-Baum mit dem Namen *default*. Dieser OE-Baum ist der Applikation *default* zugeordnet.

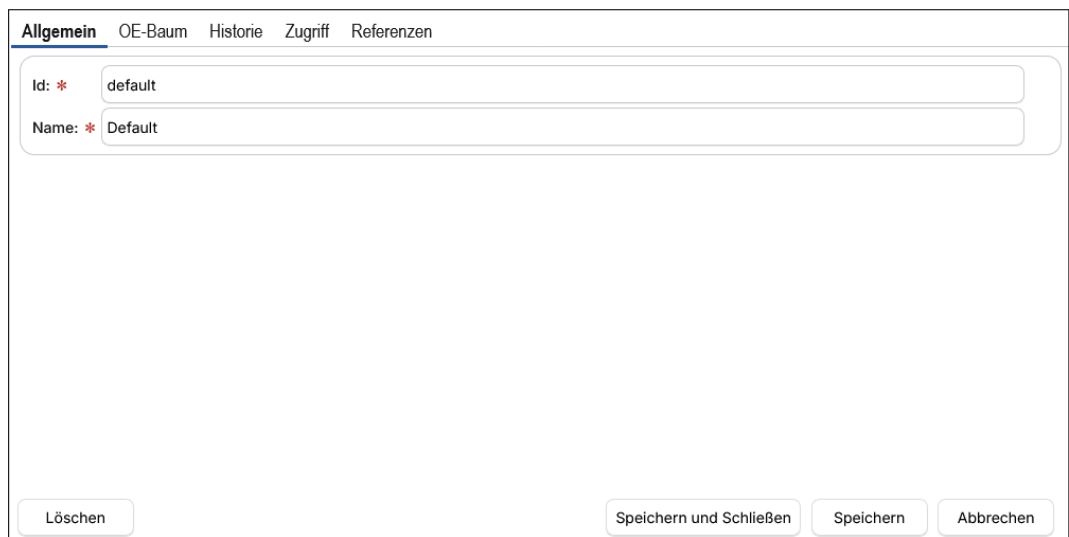
Die Objekt-Detailansicht von OE-Bäumen gliedert sich in die folgenden Bereiche:

- Allgemein
- OE-Baum
- Historie
- Zugriff
- Referenzen

4.5.1 Registerkarte: Allgemein

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Id:** Eindeutige Kurzbezeichnung des OE–Baums. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).
- **Name:** Eindeutiger Name des OE–Baums. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).



The screenshot shows a web interface for editing an OE-Baum object. The 'Allgemein' tab is active, showing fields for 'Id' and 'Name'. Both fields are marked as required with an asterisk. The 'Id' field contains the text 'default' and the 'Name' field contains 'Default'. Below the form, there are four buttons: 'Löschen', 'Speichern und Schließen', 'Speichern', and 'Abbrechen'.

Abbildung 4.7: Objekt-Detailansicht: OE-Baum

4.5.2 Registerkarte: OE-Baum

In der Registerkarte *OE-Baum* sehen Sie den Aufbau des OE-Baums. Dabei werden alle OEs, die dem OE-Baum angehören mit dem Namen dargestellt.

Anmerkung: Bei der folgenden Beschreibung der Funktionen und der Navigation im Baum wird jene Organisationseinheit, durch deren Aktivierung das PopUp–Fenster geöffnet wurde, als *aktuelle OE* bezeichnet.

Funktionen im Baum

Alle Funktionen zum Bearbeiten eines OE-Baums werden Ihnen in einem separaten Toolbar angeboten. Funktionen, die keine vorherige Selektion im Baum benötigen, sind sofort aktiv. Alle sonstigen Funktionen werden erst durch eine Selektion bzw. Vor-Aktion aktiv.

- **Bestehende Org.-Einheit hinzufügen:** Durch das Anklicken der Schaltfläche *Bestehende Org.-Einheit hinzufügen* wird der HTML-Dialog zur Auswahl einer OE

4. Definition der Organisationsstruktur

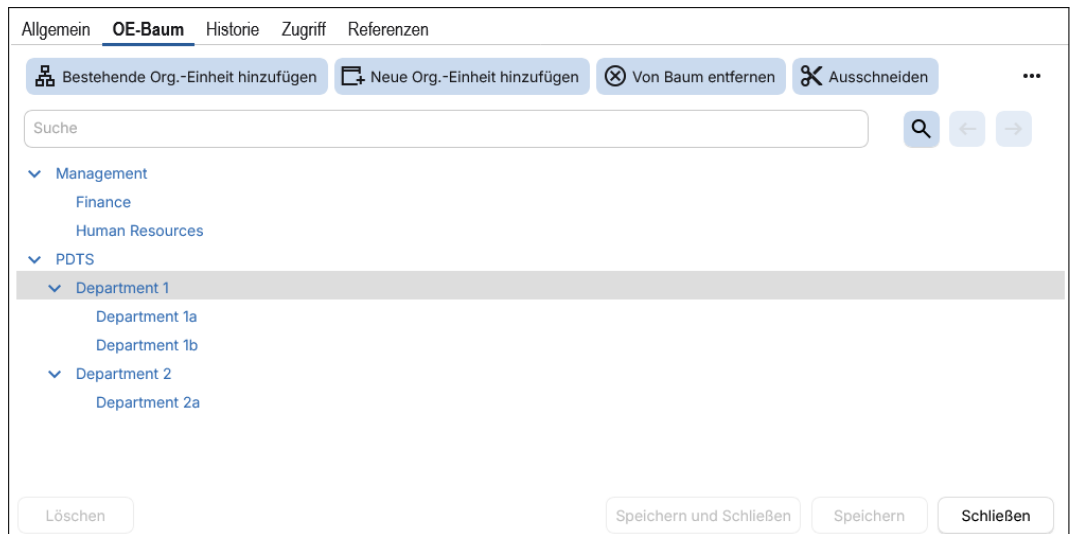


Abbildung 4.8: OE-Baum

angezeigt. Falls zuvor keine OE im OE-Baum selektiert wurde, wird die OE an oberster Stelle eingefügt. Wurde eine OE selektiert, dann wird die zu hinzufügende OE unter der aktuellen OE im OE-Baum eingefügt.

- **Neue Org.-Einheit hinzufügen:** Durch das Anklicken der Schaltfläche *Neue Org.-Einheit hinzufügen* kann in einem Schritt zunächst eine neue OE angelegt und anschließend unter die aktuell selektierte OE im OE-Baum eingefügt werden bzw. wenn zuvor keine OE selektiert war, wird die neue OE an oberster Stelle im OE-Baum eingefügt.
- **Von Baum entfernen:** Durch das Anklicken des Links *Von Baum entfernen* wird die ausgewählte OE aus dem OE-Baum entfernt.
- **Ausschneiden und Einfügen:** Mit Hilfe dieser Funktionen lässt sich eine OE innerhalb des OE-Baums an eine andere Stelle positionieren. Dazu
 - wählen Sie zuerst eine OE aus,
 - aktivieren die Toolbarfunktion *Ausschneiden*,
 - wählen eine OE im OE-Baum aus wo die ausgeschnittene OE eingefügt werden soll und
 - aktivieren die Toolbarfunktion *Einfügen*.

Alternativ können Sie auch das Drag & Drop Feature nutzen, indem Sie einen OE selektieren und mit gedrückter Maustaste an eine gewünschte Stelle im OE-Baum einfügen.

- **Details:** Durch das Anklicken des Links *Detail* wird die Objekt-Detailansicht der ausgewählten OE geöffnet.

- **Teilbaum expandieren:** Durch das Anklicken des Links *Teilbaum expandieren* wird der gesamte Teilbaum (inkl. seiner Subteilbäume), der sich unter der aktuellen OE befindet, expandiert, d.h. alle untergeordneten OEs werden sichtbar.



Sie können über die Suchfunktion nach einer beliebigen OE suchen. Diese Funktion verhält sich wie die **Kurzsuche** in der Tabellendarstellung (siehe Kapitel [Tabellendarstellung](#)). Haben Sie mit der Funktion *Suche* nach einer oder mehreren OEs gesucht, wird die zuerst gefundene OE selektiert. Mit Hilfe der *Voriges*- und *Nächstes*-Schaltflächen kann durch alle gefundenen OEs navigiert werden.



Hinweis: Wird der OE-Baum in der Zwischenzeit verändert (z.B. durch Verschieben von OEs in der Hierarchie), so werden die Schaltflächen *Voriges* und *Nächstes* inaktiv und eine neue Suche muss gestartet werden.

Im OE-Baum werden noch folgende Symbole verwendet:



- **Plus-Zeichen:** Diese OE hat eine oder mehrere untergeordnete OEs, die jedoch noch nicht angezeigt werden. Wenn Sie auf dieses Plus-Zeichen klicken, wird der OE-Baum an dieser Stelle expandiert, und die direkt untergeordneten OEs werden angezeigt. Gleichzeitig wird aus dem Plus-Zeichen ein Minus-Zeichen.



- **Minus-Zeichen:** Dieses Zeichen zeigt Ihnen, dass der Teilbaum bereits expandiert dargestellt wird, d.h. Sie sehen die untergeordneten OEs einer bestimmten OE. Wenn Sie auf das Minus-Zeichen klicken, werden alle untergeordneten OEs verborgen. Gleichzeitig wird dadurch auch das Minus-Zeichen vor dieser OE in ein Plus-Zeichen umgewandelt.



4.5.3 Funktion *OE-Bäume zusammenführen*

Mit der Toolbarfunktion *OE-Bäume zusammenführen* ist es möglich einen OE-Baum zu einem zweiten OE-Baum hinzuzufügen. Dies kann unter Umständen nötig sein, wenn es zwei *EPworX* Installationen gibt, deren OE-Bäume abgeglichen werden sollen. In der ersten Installation (A) wird der OE-Baum gewartet und an die zweite Installation (B) über XML Export übertragen. In dieser Installation B gibt es allerdings zusätzliche OEs mit Hierarchien, die erhalten bleiben sollen.

In Installation B wird nun ein eigener OE-Baum für die nur in dieser Installation vorhandenen OEs und Beziehungen erstellt. Nach Synchronisation des OE-Baums mit der Installation A (durch XML Import) werden mit dieser Funktion die nur in Installation B vorhandenen Relationen wieder in den OE-Baum hinzugefügt.

Beispiel:

OE-Baum default (auf A):

Abt. A

 Abt. A1

 Abt. A2

A1 und A2 sind Sub-OEs von A.

In der Installation B soll der Baum folgendermaßen aussehen:

```
Abt. A
  Abt. A1
  Abt. A2
    Abt. X
Abt. Y
  Abt. Z
```

Das heißt, X unter A2 und Z unter Y in einem eigenen Teilbaum.

Um die Funktion "OE-Bäume zusammenführen" nutzen zu können, wird die private Struktur auf B in einem eigenen OE-Baum definiert:

```
Abt. A2
  Abt. X
Abt. Y
  Abt. Z
```

Ein Zusammenführen dieser Struktur in den default OE-Baum ergibt die gewünschte Struktur.

4.6 Organisationsklassen

Organisationsklassen dienen der Klassifizierung von Organisationseinheiten. Diese Information wird in der Logik von *EPworX* nicht verwendet, ist allerdings hilfreich für die Strukturierung großer Organisationen. In der Liste der Organisationsklassen stehen Ihnen die in Kapitel [Tabellendarstellung](#) beschriebenen Funktionen zur Verfügung.

Die Objekt-Detailansicht von Organisationsklassen gliedert sich in die folgenden Bereiche:

- Allgemein
- Historie
- Zugriff
- Referenzen

4.6.1 Registerkarte: Allgemein

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Name:** Eindeutiger Name der Organisationsklasse. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).
- **Id:** Eindeutige Kurzbezeichnung der Organisationsklasse. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).

Allgemein Historie Zugriff Referenzen

Id: *

it

Name: *

IT

Beschreibung:

OU's in IT

Löschen

Speichern und Schließen

Speichern

Abbrechen

Abbildung 4.9: **Objekt-Detailansicht: Organisationsklasse**

- Beschreibung: Freier Text.

5. Berechtigungssystem

5.1 Allgemeines

Das *EPworX* Berechtigungssystem erlaubt eine flexible Definition der Zugriffsrechte von Benutzern auf Objekte.

In *EPworX* werden zur Umsetzung des Berechtigungssystems die folgenden Begriffe und Objekte verwendet:

- **Rechte:** siehe Kapitel [Rechte](#)
- **Berechtigungsliste:** Berechtigungslisten können Sie in *EPworX* verwenden, um mehrere Rechte zusammenzufassen. Sie können diese Berechtigungslisten mehreren Objekten, die das gleiche Berechtigungsprofil haben sollen, zuordnen.
- **Objektklasse:** Objektklassen definieren applikationsabhängige Klassen, die vom Berechtigungssystem verwaltet werden können. Dieses Verhalten können Sie auch bei Formularen definieren.
- **Berechtigung:** Eine Berechtigung legt fest, wer mit welchem Recht auf welches Objekt zugreifen kann. Ein Objekt kann keine, eine oder mehrere Berechtigungen haben. (Kapitel [Berechtigungsdialo](#)g)
- **Standardberechtigungen:** Standardberechtigungen werden bei Objektklassen und Formularen definiert. Wird ein neues Objekt erzeugt, werden die in der zugehörigen Objektklasse bzw. Formular definierten Standardberechtigungen dem neuen Objekt zugeordnet.

5.1.1 Berechtigungsliste

Berechtigungslisten sind Aggregationen von Berechtigungen. Die Objekte, die dem Berechtigungssystem unterliegen, haben ein Attribut (acl), das auf eine Berechtigungsliste verweist. Die für ein Objekt relevanten Berechtigungen sind

- Berechtigungen, die direkt auf das Objekt verweisen
- Berechtigungen, die auf die Berechtigungsliste verweisen, die im Objekt verwendet wird.

5.1.2 Standardberechtigungen

Standardberechtigungen spezifizieren, welche Berechtigungsliste beim Anlegen eines Objektes an das neue Objekt angehängt wird. Standardberechtigungen werden mit einer Objektklasse, einer Berechtigungsliste und optional einer Organisationseinheit definiert. Die Angabe der Organisationseinheit hat den Zweck, Standardberechtigungen OE-abhängig zu definieren. Je nachdem, welcher OE ein Objekt zugeordnet ist, wird die entsprechende Standardberechtigung zugeordnet.

5.1.3 Objektklassen

Objektklassen definieren applikationsabhängige Klassen, die vom Berechtigungssystem für die einzelnen Objekte in *EPworX* verwaltet werden können.

Die Objekt-Detailansicht von Objektklassen gliedert sich in die folgenden Bereiche:

- Allgemein
- Rechte
- Standardberechtigungen
- Fremdschlüssel
- Historie
- Zugriff
- Referenzen

Registerkarte: Allgemein

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Name:** Der Name der Objektklasse. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).
- **Klasse:** Die Java-Klasse, durch die die Objektklasse implementiert ist.
- **Applikation:** Gibt an, zu welcher Applikation die Objektklasse gehört.
- Berechtigungen bei Akteur anzeigen: Ist diese Checkbox aktiviert, werden alle Berechtigungen für diese Objektklasse bei der Registerkarte *Berechtigungen* eines Akteurs (Benutzer/Rolle) angezeigt. Ist diese Checkbox nicht aktiv, wird in der Registerkarte *Berechtigungen* ein Tabellen-Filter gesetzt, der die Berechtigungen definiert für diese Objektklasse auf ein bestimmtes Objekt ausblendet bzw. nicht auflistet. Berechtigungen, die allgemein auf die Objektklasse (also ohne Objektreferenz) definiert wurden, werden auch trotz gesetzten Filter angezeigt. Standardmäßig ist diese Checkbox bei allen Objektklassen von *EPworX* aktiviert und sollte nur inaktiv gesetzt werden, wenn es Performance-Probleme beim Anzeigen der Berechtigungen gibt.

Registerkarte: Rechte

Zu jeder Objektklasse wird definiert, welche Rechte für die Objekte dieser Objektklasse verwendet werden können. Die Zuordnung von Rechten zu Objektklassen ermöglicht eine Eingrenzung der Rechte, die für die jeweilige Objektklasse beim Festlegen von Berechtigungen verwendet werden können.

Registerkarte: Standardberechtigungen

In dieser Registerkarte erfolgt die Zuordnung von Standardberechtigungen zu einer Objektklasse. Es kann eine OE-unabhängige und je eine OE-abhängige Standardberechtigung vergeben werden.

Registerkarte: Fremdschlüssel

In dieser Registerkarte sehen Sie, ob Referenzen in anderen Tabellen vorhanden sind. Werden zum Beispiel Spalten der Benutzertabelle in einer anderen Tabelle referenziert, werden diese Spalten in dieser Registerkarte gelistet. Gibt es eine Referenz, kann das Benutzerobjekt über die *EPworX* Administration nicht gelöscht werden.

5.1.4 Formulare und Berechtigung

Formular-Objekte haben wie Objektklassen die Registerkarten *Rechte* und *Standardberechtigungen*. Hier können Sie für einzelne Formulare das gleiche Verhalten, wie für Objektklassen definieren.

5.2 Verwendung von Berechtigungen

In der Stammdatenverwaltung können Berechtigungen von vier Seiten aus bearbeitet werden:

- in der Benutzerverwaltung
- in der Rollenverwaltung
- beim Objekt (Zugriff)
- in der Verwaltung von Berechtigungslisten

Egal, von welcher Seite aus, Sie Berechtigungen definieren, es wird immer der gleiche HTML-Dialog zur Bearbeitung von Berechtigungen verwendet (siehe Abbildung 5.1). Je nachdem von welcher Seite Sie kommen, sind bestimmte Felder befüllt bzw. deaktiviert. Dies soll Sie beim Umgang mit Berechtigungen in *EPworX* unterstützen.

Allgemein Historie

Recht: * Objekte löschen

Zugriff: * Erlaubt

Akteur

☒ Benutzer: Jones Jack jack

☐ Rolle:

Gilt nur, wenn Rolle in Org.-Einheit:

Vertretung nicht möglich: ☐

Anwenden auf

☒ Objektklasse: Benutzer

☐ Formularklasse:

Objekt / Berechtigungsliste: Jones Jack jack

Gültigkeitsbereich: Objekt

Anwenden in Org.-Einheit:

Zeitraum: [] bis []

Löschen Speichern und Schließen Speichern Abbrechen

Abbildung 5.1: **Objekt-Detailansicht: Berechtigungen**

5.2.1 Berechtigungsdialog

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Recht:** Wählen Sie hier aus, welches Recht Sie dem Benutzer zuordnen wollen. Diese Liste enthält alle Standardrechte von *EPworX* und alle Rechte, die Sie zusätzlich in Ihren Applikationen angelegt haben. Wenn Sie ein Recht aus der Liste auswählen, werden die aktiven Felder *Anwenden auf*, *Gültigkeitsbereich*, *Anwenden in OE* oder *Akteur* zurückgesetzt.
- **Zugriff:** Legen Sie hier fest, ob es sich um eine positive (erlaubte) oder negative (nicht erlaubte) Berechtigung handelt.
- **Akteur:** Hier legen Sie fest, für wen diese Berechtigung gilt: Benutzer, Rolle oder Rolle in Organisationseinheit.
 - Wenn Sie von der Benutzerverwaltung aus Berechtigungen bearbeiten, ist der Akteur immer mit dem ausgewählten Benutzer befüllt. Sie können diesen

nicht ändern.

- Wenn Sie aus der Rollenverwaltung Berechtigungen bearbeiten, ist der Akteur immer mit der ausgewählten Rolle befüllt. Auch diese können Sie nicht bearbeiten. Sie können allerdings bei *lokalen* bzw. *hierarchischen* Rollen das Feld *Gilt nur, wenn Rolle in OE* bearbeiten.
 - Wenn Sie vom Objekt aus Berechtigungen (Zugriff) bearbeiten, müssen Sie den Akteur festlegen.
 - Wenn Sie die Rechte einer Berechtigungsliste festlegen, müssen Sie den Akteur ebenfalls festlegen.
 - Mittels der Checkbox *Vertretung nicht möglich* können Sie definieren, dass dieses Recht bei einem Vertreter nicht angewendet werden soll.
- Anwenden auf: Dieser Bereich legt fest, auf welches Objekt, auf welche Objektklasse bzw. welches Formular Sie eine Berechtigung definieren möchten. Ein **Objekt** können Sie nur auswählen, wenn Sie zuvor eine **Objektklasse** definiert haben. Welche Objektklasse Sie zur Auswahl haben, hängt davon ab, welches Recht Sie ausgewählt haben. Wenn Sie die Berechtigung auf ein **Formular** anwenden wollen, stehen Ihnen hier wiederum nur in Abhängigkeit zum ausgewählten Recht bestimmte Formulare zur Auswahl.
 - Gültigkeitsbereich: Wählen Sie hier den Gültigkeitsbereich der ausgewählten Objektklasse, des Objekts bzw. des Formulars aus. Je nach Auswahl im Bereich *Anwenden auf* stehen Ihnen unterschiedliche Auswahlmöglichkeiten zur Verfügung.
 - **alle Objekte:** Wenn Sie keine Auswahl im Feld *Anwenden auf* getroffen haben (keine Objektklasse, kein Objekt und kein Formular) ausgewählt haben, gilt die Berechtigung für *alle Objekte*.
 - **Objektklasse:** Wenn Sie ein Formular oder eine Objektklasse, aber kein Objekt ausgewählt haben, gilt die Berechtigungen für alle Objekte dieser Objektklasse.
 - **Rollen-OE:** Wenn der Akteur eine Rolle ist (und die Objektklasse nicht *Organisationseinheiten*), gilt die Berechtigung für Objekte der Organisationseinheit, in der der jeweilige Benutzer die bei Akteur angegebene Rolle hat bzw. weitere, abhängig vom OE-Wirkungsbereich.
 - **Objekt:** Wenn Sie eine Objektklasse und ein Objekt ausgewählt haben, die Objektklasse aber nicht *Berechtigungsliste* ist, gilt die Berechtigungen für genau dieses Objekt.
 - **Objekte der Berechtigungsliste:** Wenn Sie die Objektklasse *Berechtigungsliste* und ein Objekt ausgewählt haben, gilt die Berechtigung für alle Objekte der ausgewählten Berechtigungsliste.
 - **Objekte der OE:** Wenn Sie die Objektklasse *Organisationseinheiten* und ein Objekt ausgewählt haben, gilt das Recht für Objekte dieser OE und eventuell weitere, abhängig vom OE-Wirkungsbereich.

- Anwenden in OE (OE-Wirkungsbereich): Der OE-Wirkungsbereich ist notwendig, wenn die Berechtigung OE-bezogen vergeben werden soll. Die Organisationseinheit, in der dem Benutzer die Berechtigung zugewiesen wird, kann mit Hilfe des OE-Wirkungsbereichs abgeändert oder erweitert werden. Sie können hier eine Auswahl vornehmen, wenn der Gültigkeitsbereich *Rollen-OE* ist und Sie kein Objekt ausgewählt haben oder wenn der Gültigkeitsbereich *Objekte der OE* ist und eine beliebige Objektklasse, mit Ausnahme von *Organisationseinheit*, ausgewählt ist. Es stehen Ihnen folgende Möglichkeiten zur Auswahl:
 - **lokal:** Die Berechtigung gilt für die angegebene Organisationseinheit.
 - **hierarchisch:** Die Berechtigung gilt für die angegebene und alle Sub-OEs.
 - **unselbständig-hierarchisch:** Die Berechtigung gilt für die angegebene und alle unselbständigen Sub-OEs.
 - **selbständig:** Die Berechtigung gilt für die nächst übergeordnete selbständige OE.
 - **selbständig und unselbständig-hierarchisch:** Die Berechtigung gilt für nächst übergeordnete selbständige OE und alle unselbständigen Sub-OEs dieser OE.
 - **übergeordnete OE:** Die Berechtigung gilt für die übergeordnete OE.
- Zeitraum: Hier können Sie festlegen, in welchem Zeitraum die Berechtigung ihre Gültigkeit hat.

OE-Wirkungsbereich

Die unterschiedlichen OE-Wirkungsbereiche werden anhand des folgenden Beispiels verdeutlicht:

Beispiel: Abbildung 5.2 zeigt einen OE-Baum mit selbständigen und unselbständigen Organisationseinheiten. Die dunklen Kreise (OE4, OE6 und OE7) stellen die unselbständigen OEs dar. Eine Berechtigungsvergabe für die Organisationseinheit OE2 in den unterschiedlichen Wirkungsbereichen umfasst die folgenden OEs:

- lokal: OE2
- hierarchisch: OE2, OE3, OE4, OE5, OE6, OE7, OE8, OE9
- unselbständig-hierarchisch: OE2, OE4, OE6, OE7
- selbständige OE: OE2
- selbständige und unselbständige Sub-OEs: OE2, OE4, OE6, OE7
- übergeordnete OE: OE1

5.2.2 Berechtigungen in der Benutzerverwaltung

Bei jedem Benutzer-Objekt gibt es die Registerkarte *Berechtigungen*, die eine Tabelle aller direkt vergebenen Berechtigungen darstellt. Diese Tabelle kann mit den Funktionen der Werkzeugliste bearbeitet werden. Als Akteur ist der aktuelle Benutzer eingetragen. Sie können diesen nicht ändern.

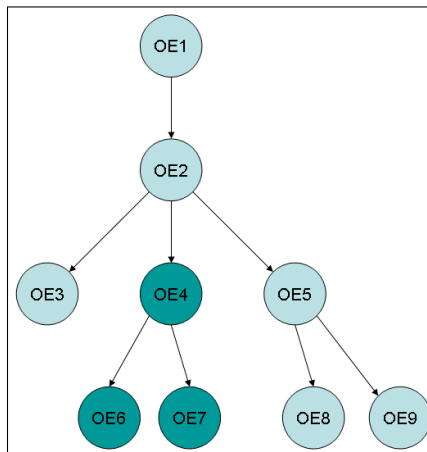


Abbildung 5.2: OE-Baum mit selbständigen und unselbständigen OEs

5.2.3 Berechtigungen in der Rollenverwaltung

Bei jedem Rollen-Objekt gibt es die Registerkarte *Berechtigungen*, die eine Tabelle aller an die Rolle geknüpften Berechtigungen darstellt. Diese Tabelle kann mit den Funktionen der Werkzeugliste bearbeitet werden. Als *Akteur* ist die aktuelle Rolle eingetragen. Sie können die Rolle nicht verändern. Allerdings können Sie das Feld **Gilt nur, wenn Rolle in OE** bearbeiten, wenn es sich bei der aktuellen Rolle um eine *lokale* oder *hierarchische* Rolle handelt.

5.2.4 Berechtigungen beim Objekt – Zugriff

Bei allen Objekten, die der Kontrolle des Berechtigungssystems unterliegen, gibt es die Registerkarte *Zugriff*. Hier tragen Sie die Berechtigungen des Objekts ein, d.h. wer mit welchem Recht auf das Objekt zugreifen kann. Sie können den Zugriff auf ein Objekt über direkte Berechtigungen (oberer Teil der Registerkarte) oder über eine Berechtigungsliste (unterer Teil der Registerkarte) definieren.

5.2.5 Verwaltung von Berechtigungslisten

Um Rechte gruppieren und wiederverwenden zu können, können Sie Berechtigungslisten anlegen. Im Bereich *Organisation* finden Sie den Link *Berechtigungslisten*. Dieser öffnet die Tabellendarstellung, die alle bereits definierten Berechtigungslisten anzeigt. Über die Funktionen der Werkzeugleiste können Sie die Einträge bearbeiten. Eine Berechtigungsliste wird nur mit ihrem Namen (Kapitel [Ids und Namen](#)) beschrieben.

In der Registerkarte *Rechte* können Sie beliebig viele Rechte zur jeweiligen Berechtigungsliste hinzufügen bzw. diese bearbeiten. Dieser Dialog entspricht dem Berechtigungsdialog in der Benutzerverwaltung (Kapitel [Berechtigungsdialog](#)). Allerdings können Sie hier die Felder im Bereich *Anwenden auf* nicht bearbeiten. Auf welches Objekt die jeweilige Berechtigungsliste angewendet wird, hängt davon ab, an welches Objekt

Sie die Berechtigungsliste binden. Um Ihnen diesen Umstand zu signalisieren, wird der Gültigkeitsbereich **Objekte dieser Berechtigungsliste** verwendet.

Um überhaupt Berechtigungen einer Berechtigungsliste zuordnen zu können, brauchen Sie allerdings das Recht *Berechtigungen bearbeiten (edit-acl)* auf die Berechtigungsliste.

5.2.6 Rechte um Berechtigungen zu bearbeiten

Benutzer dürfen Berechtigungen bearbeiten, wenn Sie das Recht *Berechtigungen bearbeiten (edit-acl)* für alle Objekte besitzen, sofern der Eintrag auf kein bestimmtes Objekt verweist. Verweist der Eintrag auf ein bestimmtes Objekt, muss der Akteur das Recht *Berechtigungen bearbeiten (edit-acl)* für das betreffende Objekt oder die Objektklasse besitzen. Außerdem ist es notwendig, das Recht *Objekt ausführen (execute)* für das Recht, welches in der Berechtigung auftritt, zu besitzen. Damit ist es möglich, die Vergabe von Berechtigungen auf bestimmte Rechte einzuschränken.

5.2.7 Sonderfall: Berechtigungen für Rollenzuordnungen

Die Vergabe von Rollenzuordnungen stellt einen Sonderfall dar, da durch eine Zuordnung von Rollen ein Benutzer sich selbst Rechte geben kann. Für die Manipulation von Rollenzuordnungen ist es daher notwendig folgende Rechte zu besitzen:

- die Berechtigung den Benutzer zu bearbeiten – *Objekt bearbeiten (edit)*
- die Berechtigung Rollenzuordnungen zu vergeben – *Objekt ausführen (execute)* auf die Rolle
- die Berechtigung Rollenzuordnungen zu vergeben – *Objekt ausführen (execute)* auf die Organisationseinheit

5.3 Standardeinstellungen

Die in *EPworX* zur Verfügung stehenden Rechte sind in Kapitel [Rechte](#) beschrieben.

Der Rolle **sys** ist das Recht *Objekte bearbeiten (edit)*, *Objekt ausführen (execute)*, *Berechtigungen bearbeiten (edit-acl)*, *Objekte erzeugen (create)* und *Administration (admin)* für alle Objekte zugeordnet.

Der Benutzer **sysadm** hat die Rolle *sys*, ihm ist außerdem das Recht *Konfiguration (conf)* direkt zugeordnet.

Alle Stammdatenmanipulationen können daher von einem Benutzer mit der Rolle *sys* durchgeführt werden. Änderungen in der Konfiguration, sowie das Ausführen der Funktionen *Server abschalten* und *Logfile ansehen* darf nur der Benutzer *sysadm* oder ein Benutzer, dem das Recht *Konfiguration (conf)* zugeordnet ist.

5.4 Wofür werden welche Rechte benötigt?

Die Tabellen 5.1 und 5.2 geben Ihnen einen Überblick darüber, welche Rechte Sie für welche Aktionen benötigen.

5.5 Beispiel

Anhand des folgenden Beispiels soll die Verwendung des Berechtigungssystems in *EPworX* gezeigt werden.

Aufgabenstellung: Der Benutzer *Franz Maier* soll die Berechtigung erhalten, Benutzer der Abteilung *Service* zu administrieren, d.h. die Eigenschaften eines Benutzers inkl. dessen Rollenzuordnungen zu bearbeiten. Die Benutzer werden zentral angelegt.

Lösung: Um die Benutzer der OE *Service* administrieren zu können, wird ihm das Recht *Objekte bearbeiten (edit)* für die Objektklasse *Organisationseinheit* und dem Objekt *Service* gegeben. Um in die Administration von *EPworX* zu gelangen, wird ihm das Recht *Administration (admin)* zugeordnet.

Um die Rollenzuordnungen administrieren zu können, wird eine Rolle definiert, die alle Rollenzuordnungen (mit Ausnahme der Rolle *sys*) für Benutzer bearbeiten darf. Eine lokale Rolle mit dem Namen *Rollenzuordnungen bearbeiten* wird angelegt. Dieser Rolle werden die Berechtigungen *Objekt ausführen (execute)* auf die Objektklasse *Rolle* erlaubt und die Berechtigung *Objekt ausführen (execute)* auf die Rolle *sys* NICHT erlaubt (siehe Abb. 5.3).

5.6 Berechtigung und Vertretung

Das Verhalten von Berechtigungen bei Vertretungen ist noch eine Betrachtung wert. Die Implementierung folgt den folgenden Regeln:

1. Bei Übernahme einer Vertretung verliert der Benutzer keine eigenen Rechte.
2. Bei Übernahme einer Vertretung von einer anderen Person hat der Benutzer nachher nie mehr Rechte als beide Personen in Summe haben.

Die Auswertungsstrategie der Berechtigungen geht folgendermaßen vor sich:

Frage: Auf welche Objekte hat ein Benutzer ein bestimmtes Recht?

- Schritt 1: Ermittle die Menge der Objekte ohne Berücksichtigung der Vertretungen.
- Schritt 2: Für alle Vertretenen: Berechne die Menge der Objekte mittels aller positiven Rechte der durch die Vertretung übernommenen Rollen. Subtrahiere davon alle Objekte, für die negative Rechte beim Vertretenen existieren, egal ob dieses Recht zu einer Rolle gehört, die vertreten wird oder nicht. Addiere die resultierende Menge von Objekten zur Ergebnismenge.

5. Berechtigungssystem

Allgemein Rollen Vertretungen Rollenvertretungen Berechtigungen Alle Berechtigungen Einstellungen Alle Einstellungen Historie Zugriff				
☒ Aktuell ☐ Stand vom: <input type="calendar"/> <input type="clock"/> <input type="search"/>				
Recht	Betroffenes Objekt	Gültigkeitsbereich	Anwenden in Org.-Einheit	Zugriff
Persönliche Berechtigungen				
Objekt ausführen	Berechtigungsliste: MyAcLList	Objekte dieser Berechtigungsliste		●
Berechtigungen bearbeiten	Berechtigungsliste: Dokument/Ordner	Objekt		●
Objekte löschen	Berechtigungsliste: Dokument/Ordner	Objekt		●
Berechtigungen über Rolle Abteilungsleitung in Organisationseinheit Finance				
Objekte erzeugen		Alle Objekte		●
Objekt ausführen	Funktionen	Rollen-OE	Hierarchisch	●
Berechtigungen über Rolle Abteilungsleitung in Organisationseinheit Human Resources				
Objekte erzeugen		Alle Objekte		●
Objekt ausführen	Funktionen	Rollen-OE	Hierarchisch	●
Berechtigungen über Rolle Abteilungsleitung in Organisationseinheit IT				
Objekte erzeugen		Alle Objekte		●
Objekt ausführen	Funktionen	Rollen-OE	Hierarchisch	●
Berechtigungen über Rolle Abteilungsleitung in Organisationseinheit Management				
Objekte erzeugen		Alle Objekte		●
Objekt ausführen	Funktionen	Rollen-OE	Hierarchisch	●
Berechtigungen über Rolle Abteilungsleitung in Organisationseinheit PDTs				
Objekte erzeugen		Alle Objekte		●
Objekt ausführen	Funktionen	Rollen-OE	Hierarchisch	●
Löschen		Speichern und Schließen		Speichern Schließen

Abbildung 5.3: Beispiel: Berechtigungen

Bei Übernahme einer persönlichen Vertretung übernimmt der Benutzer alle persönlichen Rechte des Vertretenen, bei Übernahme einer Rollenvertretung alle mit der Rollenzuordnung verbundenen Rechte. Bei Konflikten, z.B. ein Recht wird einem Vertretenen explizit entzogen, dem Vertreter aber gewährt, gilt Regel 1.

5. Berechtigungssystem

Benutzer A will ...	benötigtes Recht (Name, Id)	Anwenden ...
ein Objekt anlegen	Objekte erzeugen (create)	auf alle Objekte oder die Objektklasse
ein Objekt ändern	Objekte bearbeiten (edit)	auf das Objekt, die Objektklasse, alle Objekte oder die OE (wenn das Objekt zu einer OE gehört).
eine OE ändern ein Objekt löschen	Objekte löschen (delete)	auf das Objekt, die Objektklasse, alle Objekte oder die OE (wenn das Objekt zu einer OE gehört).
eine OE ändern	OE bearbeiten (edit)	auf alle Objekte, die Objektklasse <i>Organisationseinheiten</i> oder eine bestimmte OE
in die Administration wechseln	Administration (admin)	auf alle Objekte
Konfigurationen ansehen, Server Details unter Admin-Tasks ansehen	Administration (admin) UND Konfiguration ansehen (view_conf)	auf alle Objekte
Konfigurationen ändern, Serversteuerung, Import/Export durchführen, Datenbank-Upgrade ausführen	Administration (admin) UND Konfiguration bearbeiten (conf)	auf alle Objekte
die Log-Datei ansehen	Administration (admin) UND Konfiguration ansehen (view_conf)	auf alle Objekte
eine Berechtigung eintragen, ändern oder löschen	Berechtigungen bearbeiten (edit_acl) UND Objekt ausführen (execute)	auf das Objekt, die Objektklasse oder alle Objekte (<i>Berechtigungen bearbeiten</i>); auf das Recht (<i>Objekt ausführen</i>)
eine Rollenzuordnung eintragen, ändern oder löschen	Objekt bearbeiten (edit) UND Objekt ausführen (execute)	auf Benutzer (Objekt bearbeiten); auf die Rolle und Organisationseinheit (<i>Objekt ausführen</i>)
eine Funktion ausführen	Objekt ausführen (execute)	auf alle Objekte, die Objektklasse <i>Applikation</i> oder eine bestimmte Applikation
einen Prozess abbrechen	Prozessinstanzen bearbeiten (proc_inst)	auf alle Objekte oder alle OEs oder die OE, in der der Prozess gestartet wurde (<i>Objekt ansehen</i>)
den Prozessakteur in der Prozesshistorie ändern	Prozessinstanzen bearbeiten (proc_inst) ODER Akteur ändern (set_agent)	auf alle Objekte oder alle OEs oder die OE, in der der Prozess gestartet wurde (<i>Objekt ansehen</i>)
die Prozesshistorie, Liste der Dokumente und Notizen, alle Prozessformulare und -versionen ansehen	Prozessinstanzen ansehen (view_procinstant)	auf einen bestimmten Prozess oder alle
Prozessdokument erstellen	Objekte erzeugen (create) UND A ist aktueller Schrittakeur ODER wenn A nicht Schrittakeur, dann hat A Recht <i>Prozessinstanzen ansehen</i> (view_procinstant) + Recht <i>Objekt bearbeiten</i> (edit) auf die Prozessinstanz	auf das Objekt
Prozessdokument bearbeiten	Objekte bearbeiten (edit) ODER A ist aktueller Schrittakeur	auf das Objekt
Prozesse archivieren	Administration (admin) UND Konfiguration bearbeiten (conf)	auf alle Objekte
Prozessinstanzen suchen	Objekte ansehen (view) UND Prozessinstanzen bearbeiten (proc_inst)	auf alle Objekte oder alle OEs oder die OE, in der der Prozess gestartet wurde (<i>Objekt ansehen</i>)
Reports mit Report Designer erstellen	Statistik (stat)	auf alle Objekte
abgespeicherte Reports ausführen	Objekt ausführen (execute) oder Statistik (stat)	auf alle Objekte, die Objektklasse <i>Reports</i> oder eine bestimmte Anfrage
Stammdatenobjekt(e) löschen	Objekte löschen (delete)	auf alle Objekte, die Objektklasse
standalone Formular löschen	Objekte löschen (delete)	auf die Formulkasse
Formularänderungen in Historie sehen	Prozessinstanz ansehen (view_procinstant) ODER A ist Akteur der Version	auf die OE, in der Prozessinstanz läuft oder die Prozessdefinition
beliebige Vertreter zuordnen können	Vertreter erfassen (grant_subst)	auf alle Objekte, die Objektklasse OE
Vertreter einer bestimmten OE zuordnen können, in der Vertreter home-Rolle besitzt	Vertreter erfassen (grant_subst)	auf die Objektklasse Organisationseinheit und eine bestimmte OE als Objekt
Kalendertermine mit Teilnehmer erzeugen	Kalendertermine erzeugen (insertCal)	auf alle Objekte ODER Einschränkung auf Benutzer/Organisationseinheiten/Ressourcen
Kalendertermine mit Teilnehmer bearbeiten/löschen	Kalendertermine bearbeiten (editCal)	auf alle Objekte ODER Einschränkung auf Benutzer/Organisationseinheiten/Ressourcen
Style Konfiguration (Theme) auswählen	Objekte ausführen	auf die Objektklasse Style-Konfigurationen
Style Konfiguration in der Administration ansehen	Objekte ansehen	auf die Objektklasse Style-Konfigurationen
Style Konfiguration in der Administration manipulieren	Objekte erzeugen /löschen /bearbeiten	auf die Objektklasse Style-Konfigurationen

Tabelle 5.1: Wofür braucht man welches Recht?

5. Berechtigungssystem

Benutzer A will ...	Benötigtes Recht (Name, Id)	Anwenden ...
ein Objekt erzeugen	Objekte erzeugen (create) UND Objekt bearbeiten (edit)	create auf die Objektklasse des Objekts und edit auf den Ordner, in dem das Objekt erstellt wird
ein Objekt ändern, Metadaten ändern oder ein Dokument ersetzen	Objekte bearbeiten (edit)	auf das Objekt
ein Objekt löschen, einen Folder mit Inhalt löschen	Objekte bearbeiten (delete)	auf das Objekt, auf den Ordner und wenn das Objekt ein Ordner ist auf alle Inhalte
ein Objekt ansehen	Objekte anzeigen (view)	auf das Objekt
ein Objekt verschieben	Objekte bearbeiten (edit)	auf den Quell-Folder und auf den Ziel-Folder
ein Objekt kopieren	Objekte bearbeiten (edit) UND Objekte anzeigen (view)	auf den Ziel-Folder (<i>Objekte bearbeiten</i>); auf das Objekt bzw. wenn das Objekt ein Ordner auf alle Inhalte (<i>Objekte anzeigen</i>)
ein Objekt umbenennen	Objekte bearbeiten (edit)	auf das Objekt
Berechtigungen auf das Objekt ändern (Zugriff)	Berechtigungen bearbeiten (edit_acl)	auf das Objekt
eine Version erzeugen	Objekte bearbeiten (edit)	auf das Objekt
eine Version ansehen	Objekte anzeigen (view)	auf das Objekt
eine Version löschen	Objekte bearbeiten (edit)	auf das Objekt
die Eigenschaften ansehen	kein Recht notwendig	
ein Formular löschen	Objekte löschen (delete), Objekte bearbeiten (edit)	<i>Objekte löschen</i> auf die Formularklasse, <i>Objekte bearbeiten</i> auf den Ordner, in dem sich das Formular befindet
ein Subformular löschen	Objekte löschen (delete), Objekte bearbeiten (edit)	<i>Objekte löschen</i> auf die Formularklasse, <i>Objekte bearbeiten</i> auf das Hauptformular

Tabelle 5.2: Wofür braucht man welches Recht im DMS?

6. Workflowelemente

Anmerkung: Im folgenden werden die Begriffe *Workflow* und *Prozess* synonym verwendet. Ein Prozess im Workflow-Management-System entspricht einem Geschäftsprozess in Ihrem Unternehmen (z.B.: wie läuft die Aufnahme einer Bestellung ab, wie werden Kreditanträge in einer Bank abgewickelt, wie wird ein Urlaubsantrag bearbeitet).

Im folgenden werden die Objektklassen vorgestellt, die für die Definition von Prozessen notwendig sind. Im wesentlichen beinhaltet die Definition neuer Prozesse Antworten auf folgende Frage:

WER macht WAS, WANN und WOMIT?

- **WER:** Wer ist für die Bearbeitung eines Workflows zuständig? Es muss für jede Aktivität eines Workflows definiert sein, wer für dessen Abarbeitung verantwortlich ist. Die Zuordnung von Akteuren zu Aktivitäten basiert in der Regel auf dem Rollenkonzept, d.h. einer Aktivität wird eine Rolle als Akteur zugeordnet, und jeder Benutzer, der diese Rolle besitzt, kommt als potenzieller Bearbeiter der Aktivität in Frage. Der Benutzer, der die entsprechende Aktivität als erster annimmt, wird dann zum tatsächlichen Akteur (Bearbeiter) für diese Aktivität. Ein Benutzer kann auch direkt als Bearbeiter festgelegt werden.
- **WAS:** Was (welche Arbeit) wird mit Hilfe des Workflows erledigt? Eine Aufgabe wird in Einzelaufgaben (Aktivitäten) zerlegt, die von verschiedenen Bearbeitern ausgeführt werden können.
- **WANN:** Nachdem festgelegt wurde, welche Aktivitäten es gibt und wer dafür zuständig ist, muss noch geklärt werden, in welcher Reihenfolge die einzelnen Aktivitäten vernünftigerweise bearbeitet werden sollen. In der Regel werden Aktivitäten sequentiell, also nacheinander, abgearbeitet. Es kann in bestimmten Situationen aber durchaus auch z.B. eine parallele Abarbeitung sinnvoll und vor allem zeitsparend sein.
- **WOMIT:** Für die Bearbeitung der Aktivitäten sind bestimmte Informationen erforderlich. Es muss also festgelegt werden, welche Aktivität welche Information benötigt und wie Aktivitäten ihrerseits Informationen für andere Aktivitäten zugänglich machen. Die Verwendung von Formularen mit strukturierten und nicht strukturierten Feldern ermöglicht eine sehr einfache Form des Informationsaustausches zwischen Aktivitäten.

Eine Prozessdefinition besteht aus folgenden Elementen:

- **Applikationen:** Logisch zusammengehörende Prozesse werden zu Applikationen zusammengefasst. Beispiele: Einkauf, Qualitätssicherung, administrative Aufgaben, Callcenter.
- **Tasks:** Die elementaren Bestandteile in Prozessen werden als Task (oder auch als Aktivität oder Schritt) bezeichnet. Beispiele: Bestellung aufnehmen, informieren, genehmigen, kopieren, Anruf entgegennehmen, testen. (Beantwortung der Frage *was?*)
- **Funktionen:** Programmstücke, die automatisierte Teilaufgaben eines Tasks darstellen, werden als Funktionen bezeichnet. Beispiele: Notiz anhängen, kopieren. (Beantwortung der Frage *was?*)
- **Prozesse:** Ein Prozess wird verwendet, um den Ablauf eines Geschäftsprozesses als Ganzes zu beschreiben. Beispiele: Kreditantrag, Urlaubsantrag, Bestellung. (Beantwortung der Frage *wann?*)
- **Formulare:** Um die Daten (Informationen) der Prozesse zu verwalten, und diese über die einzelnen Tasks hinweg auszutauschen, werden Formulare verwendet. (Beantwortung der Frage *womit?*)
- **Akteure:** Die in einem Prozess beteiligten Akteure werden über Rollen (und Organisationseinheiten) oder durch Benutzer direkt beschrieben. (Beantwortung der Frage *wer?*)

Die einzelnen Elemente einer Prozessdefinition werden im folgenden Abschnitt beschrieben. Rollen, Organisationseinheiten und Benutzer wurden bereits weiter oben in diesem Handbuch detailliert beschrieben (siehe Kapitel [Rollen](#), Kapitel [Benutzer](#) und Kapitel [Organisationseinheiten](#)).

6.1 Applikationen

Applikationen erlauben es, zusammengehörende Prozesse und deren Elemente (Formulare, Rollen etc.) zu gruppieren. Eine Übersicht über alle Elemente einer Applikation erhalten Sie, wenn Sie im Navigationsbaum auf eine Applikation klicken. In der Applikationsliste stehen Ihnen die in Kapitel [Tabellendarstellung](#) beschriebenen Funktionen zur Verfügung. Eine Übersicht über alle Elemente aller Applikationen können Sie sehen, wenn Sie im Navigationsbaum auf *Alle Applikationen* klicken.

Die Objekt-Detailansicht von Applikationen gliedert sich in die folgenden Bereiche:

- Allgemein
- Eigenschaften
- Eigenschaften XML
- Historie

- Zugriff
- Referenzen

The screenshot shows a dialog box titled 'Objekt-Detailansicht: Applikationen' with several tabs: 'Allgemein', 'Eigenschaften', 'Eigenschaften XML', 'Historie', 'Zugriff', and 'Referenzen'. The 'Allgemein' tab is selected. It contains the following fields:

- Id:** * itsm
- Name:** * ITSM I18n: ITSM
- OE-Baum:** * Default
- Beschreibung:** (empty text area)
- Applikationsklasse:** com.groiss.itsm.ITSMApplication
- Applikationsverzeichnis:** * appls\itsm
- Version:** 12.0.1934
- Startup Position:** (empty text field)

At the bottom, there are four buttons: 'Löschen', 'Speichern und Schließen', 'Speichern', and 'Abbrechen'.

Abbildung 6.1: **Objekt-Detailansicht: Applikationen**

6.1.1 Registerkarte: Allgemein

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Id:** Eindeutige Kurzbezeichnung der Applikation. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).
- **Name:** Eindeutiger Name der Applikation. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#). Der I18n-Link neben diesem Feld ermöglicht direkt die Anpassung der Ressource (falls in Registerkarte *Eigenschaften* festgelegt). Die Änderungen werden nach Aktivieren der Schaltfläche *Speichern* in der Ressource Datei (Strings) dieser Applikation gespeichert (siehe Abschnitt [Ressourcen Editor](#)).
- **OE-Baum:** Nach dieser OE-Hierarchie werden hierarchische Rollen und Rechte ausgewertet. Pro Applikation kann genau ein OE-Baum ausgewählt werden. Genauere Informationen zu OE-Bäumen finden Sie im Kapitel [OE-Baum](#).
- **Beschreibung:** Freier Text.
- **Applikationsklasse:** Hier kann eine Java-Klasse angegeben werden, die das Interface `com.groiss.wf.ApplicationAdapter` implementiert. Dieses Interface kann das Verhalten der Applikation beeinflussen. Für eine genaue Beschreibung siehe die [API-documentation](#) und das [Programmierhandbuch](#).

- **Applikationsverzeichnis:** Dieses Verzeichnis gibt an, wo die Applikation installiert ist. Wenn das Verzeichnis nicht existiert, wird das System den Benutzer auffordern, es zu erstellen. Dies umfasst das Applikationsverzeichnis und das *classes* Unterverzeichnis.
- **Version:** Hier können Sie eine Version für die Applikation definieren. Dies kann z.B. bei einem Upgrade hilfreich sein, um Applikationen aus älteren *EPworX* - Versionen zu unterscheiden.
- **Startup Position:** Die Applikationen werden aufgrund der hier eingegebenen Position in aufsteigender Reihenfolge geladen. Dies kann z.B. nötig sein, wenn eine Applikation A2 Abhängigkeiten zu einer Applikation A1 besitzt und daher sichergestellt werden muss, dass Applikation A1 vor A2 geladen wird.
- **Button *Upgrade*:** Dieser Button erscheint nur, wenn eine aktuellere Version der Applikation am Dateisystem gefunden wurde als die installierte. Durch Aktivieren dieses Buttons werden alle definierten Aktionen im Hintergrund ausgeführt. Für nähere Informationen darüber, sehen Sie bitte im API von *EPworX* nach (*ApplicationAdapter.getVersion()* und *ApplicationAdapter.upgrade()*).

6.1.2 Registerkarte: Eigenschaften

In dieser Registerkarte können Sie weitere Eigenschaften für die Applikation festlegen, die unter anderem in einer XML-Datei (*properties.xml*) gespeichert werden. Folgend wird kurz auf die einzelnen Elemente dieser Registerkarte eingegangen:

- **Ressource bundle:** Hier können Sie den Pfad zu einem Ressource-Bundle (*.xls bzw. *.properties-Files) angeben, dass von dieser Applikation zur Internationalisierung verwendet wird, z.B. *com.groiss.itsm.resource.Strings*. Falls noch kein Ressource-Bundle existiert, wird eines angelegt, wenn die Ressourcen Editor Toolbarfunktion *Neu* verwendet und anschließend gespeichert wird (siehe Kapitel [Ressourcen Editor](#)). Weitere Informationen über Ressource-Bundles können Sie im [Programmierhandbuch](#) unter dem Kapitel [Internationalization of Applications](#) finden.
- **Applikations-Parameter:** Hier können Sie Parameter festlegen, die für diese Applikation verwendet werden. Applikations-Parameter können in Gruppen eingeteilt werden. Diese Parameter werden in einer XML-Datei (*properties.xml*) im Applikationsklassenpfad gespeichert und sind daher auch in der *Konfiguration* von *EPworX* verfügbar (siehe Abschnitt [Konfiguration](#)). Klicken Sie auf das Icon *Konfiguration ansehen*, um den Inhalt der entsprechende Konfigurationsdatei (der Applikation) anzuzeigen. Gibt es mehrere Gruppen, werden die Gruppen in einer Baumstruktur in der Navigation angezeigt. Die Checkbox *Sichtbar* ist standardmäßig aktiviert. Wenn diese Checkbox deaktiviert ist, ist die Applikations-Parameter Gruppe in der *EPworX* Konfiguration nicht sichtbar.
- **Benutzer-Parameter:** In diesem Bereich können Sie Parameter für Benutzer festlegen, die diese Applikation verwenden. Diese Parameter werden ebenfalls in einer XML-Datei gespeichert und sind über die *Einstellungen* eines jeden Benutzers aufrufbar (siehe [Benutzerhandbuch](#)).

- Upgrade-Pfad: Hier können Sie den Upgrade-Pfad für die Applikation definieren, der für das Upgrade dieser jener dient. Der Pfad sollte dabei wie folgt definiert werden:
 - Im Reiter *Allgemein* die Version auf den neuen Wert setzen (z.B. von 1 auf 1.1).
 - Im Reiter *Eigenschaften* bei Upgrade-Pfad eine neue Zeile anlegen und die Ziel-Version auf den gleichen Wert wie im Feld *Version* am Reiter *Allgemein* setzen.
 - Falls die Checkbox *Groovy* aktiviert wird, muss im Feld *Methode/Script* ein Groovy-Script eingetragen werden, z.B.:

```
admin.importXML("<appl-id>/exports/<appl-id>.xml")
```

Ist die Checkbox nicht aktiviert, muss eine JAVA-Methode eingetragen werden, die genau einen Parameter des Typs *com.groiss.org.Application* definieren muss, wie z.B.:

```
com.acme.myappl.Upgrades.upgradeToVersion1Dot1(Application appl)
```

Diese Methode liefert einen *String* als Rückgabewert. In diesem *String* kann man dann z.B. festhalten, was das Upgrade gemacht hat. Das wird dann geloggt und im GUI angezeigt (wenn das Upgrade via GUI ausgeführt wird). Mit Hilfe des Klassen-Checkers kann geprüft werden, ob die eingegebene Methode/das eingegebene Skript korrekt ist. Bitte beachten Sie, dass der Upgrade-Pfad in jener Reihenfolge durchlaufen wird, wie er in der Datei *properties.xml* definiert ist, also von oben nach unten. Das bedeutet, dass der 1. Eintrag (oberste) die älteste Version darstellt und der letzte Eintrag (unterste) die aktuellste Version.

Der hier definierte Pfad ist im *properties.xml* der Applikation gespeichert und wird beim Exportieren der Applikation (siehe Abschnitt [Applikation exportieren](#)) im ZIP-Archiv mitverpackt. Die Applikation lässt sich somit leicht am Ziel-System aktualisieren (siehe [Installation and Configuration](#) Handbuch, Abschnitt [Upgrading your Application](#)).

Hinweis: Die Funktionen dieser Registerkarte stehen nur dann zur Verfügung, wenn für die Applikation ein *Applikationsverzeichnis* festgelegt wurde. Als *Applikations-* und *Benutzer-Parameter* können hierbei über diesen Dialog nicht alle HTML-Elemente (z.B. Textfelder, Radio-Buttons, mehrzeilige Standardwerte etc.) abgebildet werden. Nähere Informationen über solchen Elementen der Datei *properties.xml* finden Sie in diesem [Kapitel](#). Wir stellen einen Editor zum direkten Bearbeiten der XML-Datei in der Registerkarte *Eigenschaften XML* zur Verfügung (siehe unten).

6.1.3 Registerkarte: Eigenschaften XML

Nachdem Sie diesen Reiter aktiviert haben, öffnet sich der Editor zum direkten Bearbeiten des XML. Mit Hilfe der Tastatur-Taste *F6* kann das XML Schema geöffnet werden.

6.1.4 Report



Diese Funktion zeigt eine Übersicht über alle verwendeten Komponenten der ausgewählten Applikationen (siehe Abbildung 6.2). Die einzelnen Komponenten werden in Blöcke gegliedert und können auf- bzw. zugeklappt werden. Weiters besteht die Möglichkeit ein PDF zu generieren.

Ein Report kann auch nur für Prozesse (siehe Abschnitt [Report](#)) oder Formulare (siehe Abschnitt [Report](#)) erstellt werden.

6.2 Tasks

Tasks sind die einzelnen Schritte eines Prozesses. Tasks können in unterschiedlichen Prozessen verwendet werden. In der Taskliste stehen Ihnen die in Kapitel [Tabellendarstellung](#) beschriebenen Funktionen zur Verfügung.

Die Objekt-Detailansicht von Tasks gliedert sich in die folgenden Bereiche:

- Allgemein
- Funktionen
- Historie
- Zugriff
- Referenzen

6.2.1 Registerkarte: Allgemein

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Id:** Die eindeutige Kurzbezeichnung des Tasks. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).
- **Name:** Der Name des Tasks. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#). Der I18n-Link neben diesem Feld ermöglicht direkt die Anpassung der Ressource (falls in Applikationsdetailmaske - Registerkarte *Eigenschaften* festgelegt). Die Änderungen werden nach Aktivieren der Schaltfläche *Speichern* in der Ressource Datei (Strings) dieser Applikation gespeichert (siehe Abschnitt [Ressourcen Editor](#)).
- **Version:** Die eindeutige Versionsnummer des Tasks. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).

Urlaub 📄

▼ 1. Allgemein

Name	Urlaub										
Applikation	Personalprozesse										
Beschreibung	desc_proc_vacation										
Id (Version)	hr_vacation (1)										
Übersetzungen	Englisch: Vacation, Deutsch: Urlaub										
Formulare	Id: form_vacation, Name: Urlaub, Typ: Urlaub										
Start	Manuell: Alle										
Eskalation	<table style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="text-align: left; color: #0070C0;">Eskalationstyp</th> <th style="text-align: left; color: #0070C0;">Zeitabstand</th> <th style="text-align: left; color: #0070C0;">Aktion</th> <th style="text-align: left; color: #0070C0;">Aktionsdetails</th> <th style="text-align: left; color: #0070C0;">Beschreibung</th> </tr> </thead> <tbody> <tr> <td>Prozessfähigkeit</td> <td>1 Tag</td> <td>E-Mail senden</td> <td>Aktuelle Akteure</td> <td></td> </tr> </tbody> </table>	Eskalationstyp	Zeitabstand	Aktion	Aktionsdetails	Beschreibung	Prozessfähigkeit	1 Tag	E-Mail senden	Aktuelle Akteure	
Eskalationstyp	Zeitabstand	Aktion	Aktionsdetails	Beschreibung							
Prozessfähigkeit	1 Tag	E-Mail senden	Aktuelle Akteure								

▼ 2. WDL

```

process hr_vacation(form_vacation hr_vacation "form_vacation")
version 1;
name "proc_vacation";
description "desc_proc_vacation";
subject "{form_vacation.employee}, {form_vacation.vacfrom,date} - {form_vacation.vacto,date}";
application staffprocs;
begin
  <label_4>
    all vacation_request(form_vacation);
    manager vacation_approve(form_vacation);
    if xpath:"$form_form_vacation/approved = ""true"" then "@@approved@"
      andpar
        form_vacation.employee vacation_informapproved(form_vacation);
        |
        hr!home vacation_process(form_vacation);
      end;
    else
      form_vacation.employee vacation_informrefused(form_vacation);
    end;
  end
end

```

▼ 3. Graph

```

graph TD
    Start(( )) --> Task1[Alle, Anfordern (1)]
    Task1 --> Task2[Abteilungsleitung (2)]
  
```

Abbildung 6.2: Prozess Übersicht

- **Applikation:** Gibt an, zu welcher Applikation der Task gehört.
- **Beschreibung:** Freier Text. Dieser Text wird im Arbeitskorb auf der HTML-Seite

6. Workflowelemente

Allgemein Funktionen Historie Zugriff Referenzen

Id: *

businessstrip_approve

Name: *

approve

I18n: [Genehmigen](#)

Version: *

1

Applikation: *

Personalprozesse

Beschreibung:

HTML Text

B **I** **U**

desc_task_approve_businessstrip

Aktiv:

☒

Methoden:

Preprocessing:

com.groiss.hrproc.HRProcesses.setApprovedBy()

✓

Postcondition:

✓

Postcondition-Meldung:

Compensation:

com.groiss.hrproc.BusinessTrip.compansateApproval()

✓

Take:

com.groiss.wf.SystemAction.setFieldToAgent("form_businessstrip.approvedby")

✓

Untake:

✓

Maximale Dauer:

0

Tage

Kosten:

0

Erstakteur zur Laufzeit setzbar:

Alle Organisationseinheiten

Aufwand:

0

Weitere Akteure zur Laufzeit setzbar:

Alle Organisationseinheiten

Löschen

Speichern und Schließen

Speichern

Abbrechen

Abbildung 6.3: **Objekt-Detailansicht: Tasks**

Taskbeschreibung (durch Anklicken des Links in der Spalte *Task* eines Arbeitskorbeintrags) angezeigt. Sie können hier z.B. eine Kurzhilfe oder eine Anleitung für den Benutzer anführen. *EPworX* bietet hierbei die Möglichkeit, den Text in diesem Feld entsprechend zu formatieren wie es z.B. aus Microsoft Word bekannt ist, indem Sie den Reiter *HTML* auswählen. Im Reiter *Text* wird dann der erzeugte HTML-Code angezeigt.

- Aktiv: siehe Kapitel [Objekt aktivieren / deaktivieren](#)
- Preprocessing: Hier kann der Aufruf einer Java-Methode oder einer Sequenz von Java-Methoden (getrennt durch Strichpunkt) angegeben werden. Die Ausführung erfolgt, bevor der Task in den Arbeitskorb eines Benutzers gelangt. Weiters gibt es die Möglichkeit ein GROOVY-Skript einzutragen, indem Sie *groovy*: in das Feld schreiben und anschließend das Skript anfügen. Details dazu finden Sie im [Programmierhandbuch](#).
- Postcondition: Hier können Sie eine Bedingung angeben, die vom System überprüft wird, bevor ein Benutzer einen Task weiterleitet. Sollte die Bedingung nicht erfüllt sein, ist ein Weiterleiten nicht möglich.

Die Syntax der WDL Bedingungen wird in Kapitel [Bedingungen](#) erläutert.

Es ist auch möglich, eine Sequenz von Aufrufen von Java-Methoden anzugeben (durch Strichpunkt getrennt). In diesem Falle werden die Methoden der Reihe nach durchgeführt. Die Methoden müssen nicht unbedingt einen booleschen Wert zurückgeben. Das Ergebnis der Bedingung ist der letzte boolesche Wert, der von einer der Methode zurückgeliefert wurde. Hat keine der Methoden einen Rückgabewert, dann ist die Bedingung nicht wahr.

- Postcondition-Meldung: In dieses Feld können Sie einen Text eingeben, der im Falle einer negativen Auswertung der Postcondition ausgegeben wird.
- Compensation: Hier wird eine durch Beistrich getrennte Sequenz von Java-Methoden oder ein GROOVY-Skript angegeben, die/das beim Zurückgehen zu einem früheren Schritt ausgeführt wird. Diese Code kann verwendet werden, um die Auswirkungen der Bearbeitung des aktuellen Tasks rückgängig zu machen. So könnten z.B. die Auswirkungen der in den Feldern *Methodenaufruf*, *Take Hook* und *Untake Hook* definierten Java-Methoden rückgängig gemacht werden, sofern die entsprechenden Methoden bei der Bearbeitung des Tasks aufgerufen wurden.
- Take: Hier wird eine Java-Methodensequenz oder ein GROOVY-Skript angegeben, die/das beim Annehmen (vom Rollenarbeitskorb in den persönlichen Arbeitskorb) aufgerufen wird.
- Untake: Hier wird eine Java-Methodensequenz oder ein GROOVY-Skript angegeben, die/das beim Zurücklegen (vom Arbeitskorb in den Rollenarbeitskorb) aufgerufen wird.
- Maximale Dauer: Mit diesem Attribut wird die maximale Dauer des Tasks festgelegt (in Tagen, Stunden oder Minuten).
- Kosten: Diese Information wird von *EPworX* nicht verwendet. Sie steht nur für etwaige Auswertungen zur Verfügung.
- Erstakteur zur Laufzeit setzbar: Durch dieses Attribut wird festgelegt, ob der Erstakteur für diesen Task erst zur Laufzeit eingetragen werden kann. Ist dies der Fall, so muss in der Prozessdefinition bei diesem Task kein Akteur angegeben

werden. Weiters muss bei der Prozessdefinition darauf geachtet werden, dass dieser Task nicht als erster Task (Starttask) im Prozess verwendet wird, da der Prozess sonst nicht gestartet werden kann. Der Default-Wert für dieses Attribut ist *alle OEs*. Grundsätzlich stehen drei Werte zur Verfügung:

- *nicht*: Der Erstakteur ist nicht zur Laufzeit setzbar, d.h. der Akteur muss schon bei der Prozessdefinition angegeben werden. Wurde der Erstakteur nicht bei der Prozessdefinition angegeben, kann der Prozess zur Laufzeit nicht weitergeleitet werden, wenn dieser Task der Folgetask des aktuellen Tasks des Prozesses ist.
 - *innerhalb OE*: Der Erstakteur ist zur Laufzeit setzbar, wenn beim Weiterleiten kein OE-Wechsel erfolgt.
 - *alle OEs*: Der Erstakteur ist in jedem Fall zur Laufzeit setzbar, unabhängig davon, ob beim Weiterleiten ein OE-Wechsel erfolgt oder nicht.
- Aufwand: Hier können Sie den Aufwand in Minuten angeben. Diese Information wird von *EPworX* nicht verwendet. Sie steht nur für etwaige Auswertungen zur Verfügung.
 - Weitere Akteure zur Laufzeit setzbar: Durch dieses Attribut wird festgelegt, ob zur Laufzeit für diesen Task weitere Akteure angegeben werden können, d.h. ob für diesen Task die Funktion *Bearbeiter ändern* (siehe [Benutzerhandbuch](#)) zur Verfügung steht oder nicht. Der Default-Wert für dieses Attribut ist *alle OEs*. Grundsätzlich stehen drei Werte zur Verfügung:
 - *nicht*: Für diesen Task steht die Funktion *Bearbeiter ändern* nicht zur Verfügung.
 - *innerhalb OE*: Für diesen Task kann ein Akteur gesetzt werden, wenn beim Weiterleiten kein OE-Wechsel erfolgt.
 - *alle OEs*: Für diesen Task kann ein Akteur gesetzt werden, unabhängig davon, ob beim Weiterleiten ein OE-Wechsel erfolgt oder nicht.



Hinweis: Wird ein Task gelöscht, werden auch die vorhandenen Zuordnungen von Task-Funktionen gelöscht.

6.2.2 Registerkarte: Funktionen

In dieser Registerkarte können Sie einzelne Task-Funktionen diesem Task zuordnen. Allgemeine Informationen zu Funktionen finden Sie im Kapitel [Funktionen](#).

In der Liste **Verfügbare Funktionen** sehen Sie alle Task-Funktionen, die Sie zum Task hinzufügen können. In der Liste **Ausgewählte Funktionen** sehen Sie alle Funktionen, die dem Task bereits zugeordnet sind.

Um eine Funktion dem Task zuzuordnen, wählen Sie zunächst eine Funktion aus der Liste *Verfügbare Funktionen* aus und aktivieren anschließend die Schaltfläche >. Die



Abbildung 6.4: **Registerkarte: Funktionen**

Funktion scheint nun in der Liste *Ausgewählte Funktionen* auf. Gespeichert wird diese Änderung aber erst nach Aktivieren der Schaltfläche **Speichern und Schließen**, **Speichern** oder beim Wechseln auf eine andere Registerkarte.

Um vom aktuellen Task wieder eine Funktion zu entfernen, wählen Sie diese Funktion aus der Liste *Ausgewählte Funktionen* aus und aktivieren anschließend die Schaltfläche <. Die Funktion scheint nun in der Liste *Verfügbare Funktionen* auf.

6.2.3 Ergänzen von Formularen

Formulare können normalerweise nur von den aktuellen Akteuren im Rahmen der für den jeweiligen Prozessschritt definierten Feldsichtbarkeiten bearbeitet werden. Dies würde selbst bei einfachsten Korrekturen (Umsetzen eines Wertes in einem Read-Only Feld) ein Zurückgehen zum jeweiligen Akteur und Schritt erfordern.

Mit Hilfe des vordefinierten *Ergänzen Tasks* (engl. *Supplement-Task*) kann diese im Einzelfall umständliche Handhabung vereinfacht werden. Der Ergänzen Task wird der jeweiligen Prozessdefinition zugeordnet (Prozesseditor:

Prozess → *Tasks*

). Nun können diesem Task Formulare zugeordnet werden und Feldsichtbarkeiten definiert werden.

Benutzer mit dem Recht *Prozessinstanzen bearbeiten* oder *Akteur ändern* erhalten dann für solche Prozesse aus der Ansicht der Prozesshistorie die Möglichkeit, Änderungen an den Formularinhalten vorzunehmen.

Dazu erscheint in der Historie ein Icon *Ergänzen*, über das man eine Aktivität Ergänzen für die aktuelle Prozessinstanz in den eigenen Arbeitskorb legen kann. Dort ändert man die Formulare und leitet die Aktivität weiter. Die Änderungen sind über die Historie umfassend nachvollziehbar.

Wenn man selbst nicht aktueller Akteur der Prozessinstanz ist (aber eines der o.g. Rechte besitzt), dann kann man die Erstellung des Ergänzen Tasks und dessen Wei-

terleitung automatisch durch einfaches Ändern und Speichern der Formulare direkt aus der Historie auslösen.

6.3 Funktionen

Task-Funktionen (oder auch kurz Funktionen genannt) sind einer Applikation zugeordnete Java-Methoden. Abhängig vom Typ der Funktion kann/muss/darf diese allen Tasks, einem Task oder keinem Task zugeordnet werden. Durch das Anklicken des Task-Funktions-Icons (siehe [Benutzerhandbuch](#)) neben einem Task werden in einem Kontextmenü alle Funktionen aufgelistet, die dem Task entweder direkt zugeordnet wurden oder die vom Typ *zu allen Tasks* sind. In der Funktionsliste stehen Ihnen die in Kapitel [Tabellendarstellung](#) beschriebenen Funktionen zur Verfügung.

Die Objekt-Detailansicht von Task-Funktionen gliedert sich in die folgenden Bereiche:

- Allgemein
- Historie
- Zugriff
- Referenzen

6.3.1 Registerkarte: Allgemein

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Id:** Die eindeutige Kurzbezeichnung der Funktion. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).
- **Name:** Der Name der Funktion. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#). Der I18n-Link neben diesem Feld ermöglicht direkt die Anpassung der Ressource (falls in Applikationsdetailmaske - Registerkarte *Eigenschaften* festgelegt). Die Änderungen werden nach Aktivieren der Schaltfläche *Speichern* in der Ressource Datei (Strings) dieser Applikation gespeichert (siehe Abschnitt [Ressourcen Editor](#)).
- **Applikation:** Gibt an, zu welcher Applikation die Funktion gehört.
- **Anwenden auf:** Hier kann der Typ ausgewählt werden, ob die Task-Funktion eine globale oder lokale Funktion ist:
 - keinen Eintrag: Die Funktion kann als globale Funktion ausgeführt werden, d.h. es muss kein Eintrag z.B. im Arbeitskorb selektiert sein.
 - einen Eintrag: Die Funktion wird als lokale Funktion ausgeführt, d.h. es muss nur 1 Eintrag ausgewählt sein.
 - mehrere Einträge: Analog zu *einen Eintrag*, jedoch können mehrere Einträge ausgewählt werden.

6. Workflowelemente

Allgemein Historie Zugriff Referenzen

Id: *

set_duedate

Name: *

set_duedate

Applikation: *

Default

Anwenden auf:

einen Eintrag

Anzeigen:

☒ zu allen Tasks

☒ Arbeitskorb

☐ Historie

☐ Bei den Prozessdetails

☐ Rollen-Arbeitskorb

☐ In Funktionsliste

☐ Direkt bei Eintrag

Auf ungespeicherte Änderungen prüfen:

☐

Beschreibung:

HTMLText

B **I** **U**    

Methode:

✓

Zielfenster:

Sortierattribut:

Client-Aktion:

ep/widget/smartclient/wl/actions/SetDueDate

Mobile Client-Aktion:

Komponentenschlüssel:

actions/worklist/set_duedate

Parameter:

Funktionsgruppe:

Icon Name:

Vorschau:

Schnellzugriff:

Löschen

Speichern und Schließen

Speichern

Abbrechen

Abbildung 6.5: **Objekt-Detailansicht: Task-Funktionen**

- Anzeigen:
 - zu allen Tasks: Die Funktion ist automatisch bei allen Tasks dieser Applikation sichtbar. Ist die Funktion in der Applikation *default* gespeichert, dann gilt

sie für Tasks aller Applikationen (inkl. *default*).

- Arbeitskorb: Ist diese Checkbox aktiviert, scheint die Funktion (ihrem Typ entsprechend) im (Rollen-)Arbeitskorb über das Funktionsrädchen auf.
 - Rollen-Arbeitskorb: Ist diese Checkbox angekreuzt, scheint die Funktion auch bei jenen Tasks im Prozessformular auf, die sich im Rollenarbeitskorb befinden.
 - Historie: Ist diese Checkbox aktiviert, scheint die Funktion (ihrem Typ entsprechend) in der Prozesshistorie auf.
 - in Funktionsliste: Die Funktion kann keinem Task zugeordnet werden (z.B. Administrationsfunktion oder Suchfunktion). Diese Funktion scheint dann im Arbeitsbereich auf, wenn im Navigationsbaum der Link *Funktionen* angeklickt wird.
 - Bei den Prozessdetails: Ist diese Checkbox aktiviert, wird die Funktion im Toolbar angezeigt, wenn die Detailansicht von einem Arbeitskorbeintrag geöffnet wurde. Damit diese Option im Arbeitskorb angewendet wird, muss zusätzlich die Checkbox *Arbeitskorb* aktiviert sein. Diese Option ist nur im Smartclient anwendbar!
 - Direkt bei Eintrag: Mit dieser Checkbox geben Sie an, ob die Funktion direkt in der Zeile angezeigt wird. Dies ist nur möglich für Arbeitskörbe mit dem Attribut *showInlineDetailsAt* (definiert im XML). Nähere Informationen finden Sie im [Programmierhandbuch](#). Damit diese Option im Arbeitskorb angewendet wird, muss zusätzlich die Checkbox *Arbeitskorb* aktiviert sein. Diese Option ist nur im Smartclient anwendbar!
- Auf ungespeicherte Änderungen prüfen: Ist diese Checkbox aktiviert, wird eine Warnung bei nicht gespeicherten Änderungen angezeigt, bevor die Funktion ausgeführt wird.
 - Beschreibung: Freier Text. *EPworX* bietet hierbei die Möglichkeit, den Text in diesem Feld entsprechend zu formatieren wie es z.B. aus Microsoft Word bekannt ist, indem Sie den Reiter *HTML* auswählen. Im Reiter *Text* wird dann der erzeugte HTML-Code angezeigt.
 - Methode: Die Bezeichnung der Java-Methode, durch die die Funktion realisiert ist. Sie können bei der Methode auch Parameter angeben. Fügen Sie an den Methodennamen ein *?* und die Parameterliste an.
Beispiel: `com.groiss.DemoClass.demoMethod?param1=val1&param2=val2`

Weiters gibt es auch die Möglichkeit wie bei *Tasks* ein GROOVY-Skript einzutragen, indem Sie *groovy:* in das Feld schreiben und anschließend das Skript anfügen.

- Zielfenster: Es kann sein, dass die Funktion eine Ausgabe erzeugt. Der Inhalt dieses Feldes bestimmt wo diese Ausgabe angezeigt wird. Wird in dieses Feld nichts eingetragen, wird die Ausgabe im Arbeitskorb angezeigt. Das Feld kann folgende Einträge enthalten:

- *_top*: Die Ausgabe wird im gesamten aktuellen Browser-Fenster angezeigt.
 - *Leerer Eintrag*: Die Ausgabe wird im Hauptfenster angezeigt.
 - *Name eines Fensters*: Die Ausgabe wird in diesem Fenster angezeigt. Es können Attribute wie *width*, *height*, *scrollbars* etc. mit Beistrich getrennt an den Namen gehängt werden. Syntax wie bei JavaScript-Methode *window.open*;
- **Sortierattribut**: Geben Sie hier jenes Attribut ein, nach dem sortiert werden soll.
- **Client-Aktion**: Soll die Funktion in der neuen GUI funktionieren, müssen Sie hier ein DOJO-Widget angeben, das *ep/widget/smartclient/_Action* implementiert. Nähere Informationen dazu finden Sie im [Programmierhandbuch](#).
- **Mobile Client-Aktion**: Soll die Funktion im mobilen GUI funktionieren bzw. angezeigt werden, müssen Sie hier analog zur *Client-Aktion* ein DOJO-Widget angeben.
- **Komponentenschlüssel**: Definieren Sie hier den Komponentenschlüssel der Funktion. Dieser Komponentenschlüssel wird zum Identifizieren der GUI-Komponente verwendet.
- **Parameter**: Hier können Sie eine Parameterliste für die *Client-Aktion* oder *Mobile-Client-Aktion* definieren.
- **Funktionsgruppe**: Eine selbst definierten Funktionsgruppe kann hier ausgewählt werden.
- **Icon Name**: Sie können im Java-Client zu einer Funktion ein beliebiges Icon definieren. Diese Angabe kann ein absoluter oder ein relativer Pfad zum Klassenpfad sein.
- **Schnellzugriff**: Hier können Sie eine Tastenkombination angeben, die für diese Funktion angewendet wird. Eine Liste der Tasten ist [hier](#) zu finden.

Beispiel für einen Schnellzugriff: CTRL+SHIFT+D

Wird diese Tastenkombination im entsprechenden Kontext gedrückt, so wird die Funktion ausgeführt. Mit entsprechendem Kontext ist gemeint, dass die Funktion über den Shortcut erreichbar sein muss wie z.B. die Arbeitskorbtabelle wird angezeigt und die Funktion ist im Toolbar aktiv oder auch die Funktion scheint in der Liste der globalen Funktionen auf und muss daher zuerst das Funktionsmenü öffnen.

Damit ein Benutzer eine Task-Funktion ausführen darf, muss er das Recht *Objekt ausführen (execute)* für die jeweilige Funktion haben.

6.3.2 Standardfunktionen

In *EPworX* werden vordefinierte Funktionen, die zur Erweiterung der Arbeitskorbfunktionalität dienen, zur Verfügung gestellt:

- **Parfor Zweige hinzufügen** (addParforSteps): Anwenden auf: *einen Eintrag*, Anzeigen: *allen Tasks zuordnen, Arbeitskorb, In Funktionsliste*. Diese Funktion sieht nach, ob bei einem laufenden Parfor-Schritt weitere Subformulare hinzugefügt wurden und startet für diese neuen Parfor-Zweige. Diese Standardeinstellung kann mit folgendem Parameter im Feld *Parameter* verändert werden:

```
{step:"parfor_step_id"}
```

- **(Un)gelesen setzen** (toggle_seen): Anwenden auf: *einen Eintrag*, Anzeigen: *allen Tasks zuordnen, Arbeitskorb, Zu nicht angenommenen Tasks*. Diese Funktion setzt den aktuellen Eintrag auf gelesen bzw. ungelesen.
- **In die Zwischenablage** (into_clipboard)
Anwenden auf: *einen Eintrag*, Anzeigen: *allen Tasks zuordnen, Arbeitskorb*. Diese Funktion legt den aktuellen Eintrag in der Zwischenablage ab.
- **PDF** (mail_pdf): Anwenden auf: *mehrere Einträge*. Diese Funktion erstellt ein PDF-Dokument aus einer oder mehreren E-Mails.
- **Aus Zwischenablage einfügen** (from_clipboard): Anwenden auf: *einen Eintrag*, Anzeigen: *allen Tasks zuordnen, Arbeitskorb*. Mit dieser Funktion können Sie den Eintrag aus der Zwischenablage einfügen.
- **Kopie an...** (make_copy): Anwenden auf: *einen Eintrag*, Anzeigen: *allen Tasks zuordnen, Arbeitskorb*. Diese Funktion erstellt eine Kopie des aktuellen Eintrags.
- **Prozessnotiz** (attach_note): Anwenden auf: *einen Eintrag*, Anzeigen: *allen Tasks zuordnen, Arbeitskorb*. Mit dieser Funktion können Sie eine Notiz zum aktuellen Eintrag (Prozess) hinzufügen.
- **Prozessnotiz** (note_global): Anwenden auf: *keinen Eintrag*, Anzeigen: *Arbeitskorb, in Funktionsliste*. Mit dieser Funktion können Sie eine Notiz nach Auswahl eines Prozesses an diesen hängen.
- **Fälligkeit setzen** (set_duedate): Anwenden auf: *einen Eintrag*, Anzeigen: *allen Tasks zuordnen, Arbeitskorb*. Sie können mit dieser Funktion das Fälligkeitsdatum des Tasks und/oder des gesamten Prozesses setzen.
- **Prozessrelationen definieren** (addRelation): Anwenden auf: *einen Eintrag*, Anzeigen: *Arbeitskorb*. Weitere Informationen zum Konfigurieren der Relationen zwischen Prozessen finden Sie im Handbuch [Installationshandbuch](#), Abschnitt [Search/Process relations](#).
- **Priorität setzen** (set_priority): Anwenden auf: *einen Eintrag*, Anzeigen: *allen Tasks zuordnen, Arbeitskorb*. Mit Hilfe dieser Funktion kann die Priorität der Prozessinstanz verändert werden.

- **Prozesskopie erstellen** (`copy_process`): Mit dieser Funktion kann eine Kopie der aktuellen Prozessinstanz erstellt werden, sofern die Funktion in diesem Kontext aufgerufen wird. Standardmäßig werden alle Formulardaten und alle DMS-Objekte im Reiter *Dokumente* kopiert und die Prozessinstanz-Kopie wird immer vom ersten Prozessschritt gestartet. Weiters wird eine Systemnotiz (welcher Prozess der Ausgangsprozess ist) beim Start-Schritt eines Prozesses erzeugt. Diese Standardeinstellungen können mit folgenden Parametern im Feld *Parameter* verändert werden:
 - Formulardaten: Sie können entweder gewisse Formulardaten des Ausgangsprozesses inkludieren (Allow list [include]) oder explizit ausschließen (Disallow list [exclude]). Dazu geben Sie entweder `include=<params>` oder `exclude=<params>` an. Die Parameter (<params>) sind eine durch Beistrich getrennte Liste von Werten im folgenden Format:
 - * `<form_id>.<formfield_name>`: Der Inhalt des definierten Formularfelds dieses (Haupt-)Formulars. Die Formular-Id ist jene, die bei der Prozessdefinition im Reiter *Formulare* vergeben wurde (siehe Abschnitt [Prozesseigenschaften](#)).
 - * `<form_id>.subform_<subformid>`: Alle Subformular-Einträge (inkl. der Felder) dieses (Haupt-)Formulars.
 - DMS-Objekte: Mit dem Parameter `documents=true|false` können Sie definieren, ob die DMS-Objekte kopiert oder ignoriert werden.
 - Systemnotiz: Der Parameter `duplicationNote=true|false` gibt an, ob eine Systemnotiz beim kopierten Prozess erzeugt werden soll.

Zum Beispiel man kann folgendes ins Feld *Parameter* eintragen:

```
{include:"form0.field1,form0.field3,form0.subform_1,form1.field1"  
,documents=false,duplicationNote=false}
```

- **Prozessvorlagen** (`process_templates`): Diese Funktion kann nur als Formular-Toolbar-Funktion definiert werden und übernimmt die Formular-Daten aus der Prozessvorlage in das bestehende Prozessformular. Dabei gibt es folgende Einschränkungen:
 - Funktioniert nur mit XForms
 - Funktioniert nur im Kontext des Prozessformular-Toolbars - dazu fügen Sie die Funktion mit dem entsprechenden `<script id="toolbarfunctions">` Block ins Formular ein
 - Unterstützt nur Subformulare, die Inline eingebunden sind.
 - Es gibt keine Unterstützung für Sub-Subformulare

Standardmäßig werden alle Formulardaten aus dem ausgewählten Ausgangsprozess in das Prozessformular des aktuellen Prozesses übernommen. Diese Standardeinstellungen können mit Parametern am Ende der definierten *Client-Aktion*

verändert werden. Sie können entweder gewisse Formulardaten des Ausgangsprozesses inkludieren (Allow list [include]) oder explizit ausschließen (Disallow list [exclude]). Dazu geben Sie entweder *include=<params>* oder *exclude=<params>* an. Die Parameter (<params>) sind eine durch Beistrich getrennte Liste von Werten im folgenden Format:

- <form_id>.<formfield_name>: Der Inhalt des definierten Formularfelds dieses (Haupt-)Formulars. Die Formular-Id ist jene, die bei der Prozessdefinition im Reiter *Formulare* vergeben wurde (siehe Abschnitt [Prozesseigenschaften](#)).
- <form_id>.subform_<subformid>: Alle Subformular-Einträge (inkl. der Felder) dieses (Haupt-)Formulars.

Zum Beispiel man kann folgendes ins Feld *Client-Aktion* eintragen:

```
{include:"form0.field1,form0.field3,form0.subform_1"}
```



Hinweis: Bitte beachten Sie bei Verwendung des Prozess-Start-Knoten FORM (siehe [Programmierhandbuch](#), Abschnitt [Start process \(<processStart>\)](#)), dass die Option *Anwenden auf* auf *keinen Eintrag* gestellt ist, damit die Funktion im Toolbar ausgeführt werden kann!

- **Verfolge den Prozess** (follow_process): Anwenden auf: *einen Eintrag*, Anzeigen: *allen Tasks zuordnen, Arbeitskorb, Rollen-Arbeitskorb, Historie*. Mit Hilfe dieser Funktion kann der Benutzer für einen Prozess oder dessen Tasks definieren, dass beim Beenden des Prozesses bzw. Tasks eine Benachrichtigung per E-Mail versendet wird. Die entsprechende Nachrichten-Vorlage dazu ist in der Applikation *default* unter der ID *processTracker* definiert.
- **PDF** (history_pdf): Anwenden auf: *einen Eintrag* Diese Funktion ermöglicht es Benutzern, eine tabellarische Darstellung der Prozesshistorie als PDF-Dokument zu erstellen.
- **Report abonnieren** (subscribe_report): Diese Funktion ist nur im Reporting verwendbar. Der Benutzer muss eine *ausführen* Berechtigung auf die Funktion haben um diese im Reporting zu sehen. Mittels diese Funktion ist es möglich die Abonnements auf Reports zu erstellen. Mehr Informationen über Reports Abonnements finden Sie im [Reporting-Handbuch](#).
- **Zu Favoriten hinzufügen**: Anwenden auf: *einen Eintrag*. Diese Funktion ist nur im Reporting verwendbar. Der Benutzer muss eine *ausführen* Berechtigung auf die Funktion haben um diese im Reporting zu sehen. Mittels dieser Funktion ist es möglich einen Report zu favorisieren. Mehr Informationen über favorisierte Reports finden Sie im [Reportinghandbuch](#) im Abschnitt [Report Favoriten](#).

Nähere Informationen zu den Funktionen finden Sie im [Benutzerhandbuch](#), Abschnitt [Taskfunktionen in EPworX](#).

6.4 Formulare

Formulare stellen ein wesentliches Element in *EPworX* dar. Alle prozessspezifischen Informationen werden in einem solchen Formular verwaltet. Es können für jeden Prozess (und dessen Aktivitäten) ein bis mehrere Formulare existieren, die die lokalen Daten des konkreten Geschäftsfalles enthalten.

In der Formularliste stehen Ihnen neben den in Kapitel [Tabellendarstellung](#) beschriebenen Funktionen, die folgenden Funktionen zur Verfügung:

- Neues Formular erstellen
- Im Formulareditor bearbeiten
- Formular anzeigen
- Report
- View erzeugen

Formulare, deren Formularklasse nicht geladen werden kann, werden wie inaktive Tabelleneinträge (siehe Kapitel [Tabellendarstellung](#)) angezeigt. Erstellung bzw. Bearbeitung eines Formulars erfolgt mit dem Formular-Editor, der im folgenden beschrieben wird.

6.4.1 Formulareditor

Der Formulareditor dient zur Erstellung und Bearbeitung von Formularen, siehe Abb. [6.6](#).

Zur Bearbeitung gibt es vier Registerkarten:

- Layout: Darstellung des Formulars und Möglichkeit, das Layout zu erstellen bzw. verändern
- Schema: enthält die Datenbank-Schema Informationen zum Formular
- Bindings: XForms Bindings (Sichtbarkeiten, Mussfelder, etc.)
- Source: HTML-Quelltext, Ansicht und Bearbeitung

Erstellen eines neuen Formulars

Wenn Sie ein neues Formular erstellen möchten, gibt es zwei Möglichkeiten:

- Gehen Sie zur Registerkarte Schema und erstellen Sie die Formularfelder, wobei jeweils Id, Beschriftung, und Typ festgelegt werden. Mit der Menüfunktion *Neues Layout erstellen* kann nun ein Standard-Layout erzeugt werden. Wechseln Sie zur Registerkarte Layout, um es zu überprüfen und ggfls. zu verändern.
- Alternativ können Sie in der Registerkarte Layout die benötigten Elemente mit drag-and-drop ins Formular ziehen. Es wird dann die Detailmaske geöffnet, in der bei „Datenbankfeld“ ein Feld ausgewählt oder (mit +) hinzugefügt werden kann.

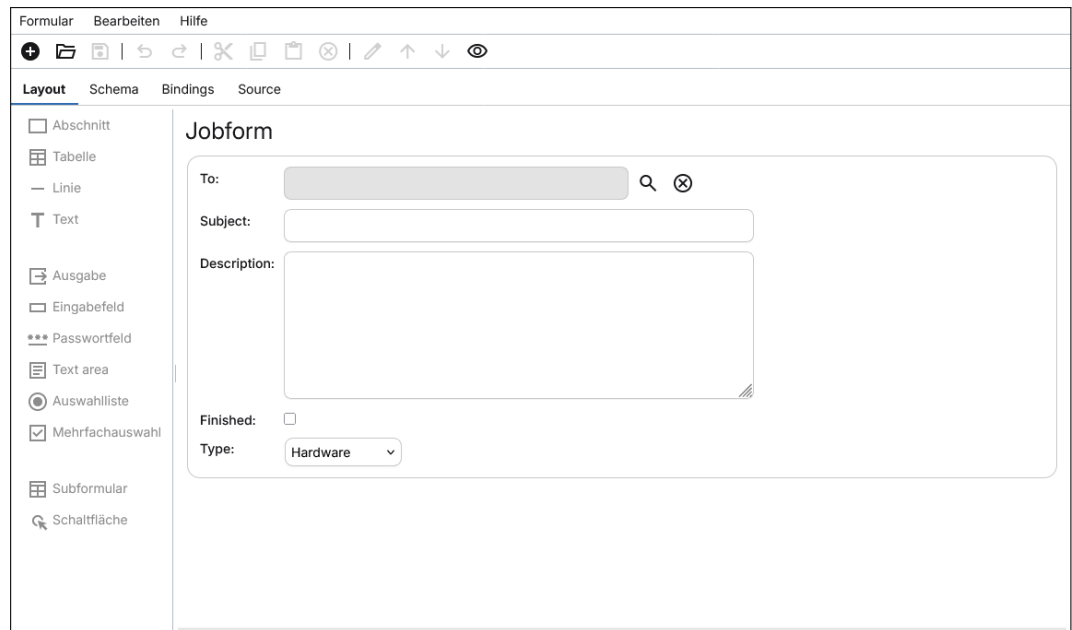


Abbildung 6.6: **Formulareditor**

Abschließend legen Sie in den Eigenschaften oder beim Speichern-Dialog den Namen und die Id des Formulars fest.

Im folgenden werden die einzelnen Bereiche des Editors beschrieben:

Registerkarte: Layout

Hier können die Formular-Elemente mittels drag-and-drop in das Formular gezogen werden. Bei Elementen, die sich auf ein Datenbank-Feld beziehen, ist dieses ein Pflichtfeld. In der Feldeigenschaften-Maske kann ein Feld ausgewählt werden oder ein neues erstellt werden.

Der allgemeinen Aufbau eines Formulars (siehe Abb. 6.7): Ein Formular besteht aus mehreren Abschnitten, jeder Abschnitt enthält ein oder mehrere Tabellen. Die Tabellen enthalten dann die eigentlichen Formular-Elemente. Abschnitte und Tabellen können Titel und Rahmen haben. Folgende Elemente gibt es:

- Linie: eine horizontale Linie
- Text: ein statischer Text
- Ausgabe: ein Ausgabefeld, zeigt den Wert des entsprechenden Formularfelds
- Eingabefeld: Feld zur Eingabe eines Wertes.
- Passwortfeld: Feld zur Eingabe eines Passworts (eingegebener Wert wird nicht angezeigt).
- Auswahlliste: Ein Wert aus einer Liste kann ausgewählt werden.

The image shows a form builder interface. On the left is a sidebar with a list of element types, each with a corresponding icon: Abschnitt (Section), Tabelle (Table), Linie (Line), Text, Ausgabe (Output), Eingabefeld (Input field), Passwortfeld (Password field), Text area, Auswahlliste (Dropdown list), Mehrfachauswahl (Multiple choice), Subformular (Subform), and Schaltfläche (Button). The main canvas on the right displays a preview of a form titled 'Mein Titel'. The form structure includes: 'Abschnitt 1' containing 'Tabelle 1.1' with an 'Input' field and a 'Datum:' field with a calendar icon; 'Tabelle 1.2' with a 'Datum und Zeit:' field containing a calendar icon and a clock icon; 'Abschnitt 2' containing 'Tabelle 2'; and a large grey rectangular area at the bottom.

Abbildung 6.7: **Formular-Struktur**

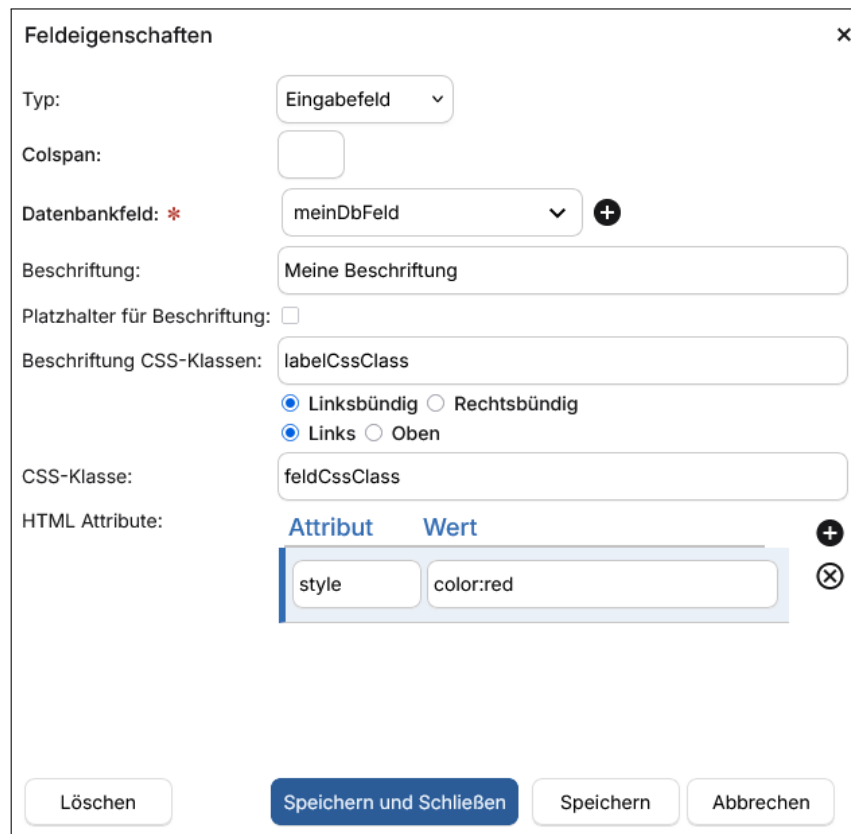
- Mehrfachauswahl: Mehrere Werte aus einer Liste können ausgewählt werden.
- Subformular: eine eingebettete Tabelle (1:n Beziehung zum aktuellen Formular)
- Schaltfläche: eine Schaltfläche mit Angabe einer Aktion

Je nach Elementtyp öffnet sich beim Erstellen des Elements die für das Element angepasste Detailmaske (siehe Abb. 6.8), in der die Beschriftung, das Formularfeld, die Label-Positionierung und weitere HTML-Attribute angegeben werden können.

Detailmaske **Feldeigenschaften**

Über die Feldeigenschaften lassen sich folgende Einstellungen durchführen:

- **Abschnitt, Tabelle:**
 - Titel: Hier kann eine Überschrift für das Element definiert werden.
 - Zusammenklappbar: Mit Hilfe dieser Checkbox können Sie festlegen, ob der Abschnitt oder die Tabelle zusammenklappbar ist.
 - Rahmen: Ist die Checkbox aktiviert, wird das Element mit einem Rahmen angezeigt.
 - CSS-Klassen: Hier können Sie die CSS-Klassen für das Element festlegen.
 - Spalten (nur bei Tabelle): Style Property *grid-template-columns*; Informationen dazu finden Sie [hier](#)
 - Gruppe: Hier kann ein *block* ausgewählt sein. Nähere Informationen zum Spezialwert *block* sind im Abschnitt [Registerkarte: Schema](#) zu finden.



Feldeigenschaften

Typ: Eingabefeld

Colspan: 1

Datenbankfeld: * meinDbFeld

Beschriftung: Meine Beschriftung

Platzhalter für Beschriftung: ☐

Beschriftung CSS-Klassen: labelCssClass

☒ Linksbündig
 ☐ Rechtsbündig
☒ Links
 ☐ Oben

CSS-Klasse: feldCssClass

HTML Attribute:

Attribut	Wert
style	color:red

Löschen Speichern und Schließen Speichern Abbrechen

Abbildung 6.8: Feldeigenschaften

- **Text:**

- Colspan: Das Attribut erlaubt es, eine Tabellenzelle nach rechts über mehrere Spalten auszudehnen.
- Beschriftung: Entspricht dem *label* Element oder in diesem Fall nur ein gewöhnlicher Text.
- CSS-Klasse: Hier können Sie die CSS-Klassen für das Element festlegen.
- HTML Attribute: In dieser Tabelle können Sie die Eigenschaften eines HTML-Elements über die HTML-Attribute definieren.

- **Ausgabe, Eingabefeld, Passwortfeld, Textarea:**

- Typ: Auswahl des Elementtyps.
- Colspan: Analog zu *Text*.
- Datenbankfeld: Hier können Sie ein Datenbankfeld auswählen oder hinzufügen.
- Beschriftung: Analog zu *label* Element.
- Platzhalter für Beschriftung: diese Checkbox ermöglicht die Erstellung eines Platzhalters für die Beschriftung, selbst wenn der Beschriftungstext leer ist.

- Beschriftung CSS-Klassen: Hier können Sie die CSS-Klassen für das *label*-Element festlegen. Zusätzlich kann man mittels Radio-Buttons (*Linksbündig*, *Rechtsbündig*, *Links*, *Oben*) die Position der Beschriftung leicht definieren.
- CSS-Klasse: Analog zu *Text*.
- HTML Attribute: Analog zu *Text*.
- **Auswahlliste, Mehrfachliste:** Die zusätzliche Felder für diese Elemente sind:
 - Darstellung: Hier können Sie definieren wie die Auswahlliste angezeigt werden soll.
 - Zeilen: Diese Einstellung wird nur angeboten, wenn *Select list* als Darstellung ausgewählt ist. Damit wird die Größe *size* der Select-Liste definiert (die Nummer der angezeigten Elemente).
- **Subformular:** Die zusätzlichen Felder für dieses Element sind:
 - Subformular: Hier können Sie ein bestehendes Subformular auswählen bzw. ein neues hinzufügen.
 - Editierbar: Ist die Checkbox aktiviert, wird das Subformular als editierbar definiert. In diesem Fall kann man zusätzlich noch die sichtbare Spalten festlegen. Wird die Checkbox nicht aktiviert, wird das Subformular als nicht editierbar definiert. Für nicht editierbare Subformulare ist es möglich eine *Knoten-Id* zu definieren. Es handelt sich hier um ein *configid* Element, welches ein Verweis auf eine in einem GUI-Konfigurations-XML definierte Tabelle ist.
- **Schaltfläche:** Das zusätzliche Feld für dieses Element ist:
 - Aktionen: In dieser Tabelle können die Aktionen für die Schaltfläche definiert werden (*send*, *setvalue*, *reset*).

Registerkarte: Schema

Hier werden die einzelnen Felder (obere Tabelle) und die Subformulare (untere Tabelle) tabellarisch dargestellt. Editieren erfolgt direkt in der Tabelle, Hinzufügen und Löschen mit dem Toolbar, Verschieben der Einträge in der oberen Tabelle mit dem Haupt-Toolbar. Spalten der oberen Tabelle:

- **Id:** Der Name für das Feld in der Datenbank und Java-Klasse.
- **Beschriftung:** Der Name im GUI, wird beim Layout-Generieren als Default verwendet, außerdem bei einer tabellarischen Anzeige des Formulars oder bei der Spaltenauswahl im Reporting.
- **Beschreibung:** Frei wählbarer Text, der das aktuelle Feld beschreibt.
- **Typ:** Der Typ des Formularfeldes. Welchem Java-Typ ein Eintrag der Auswahlliste entspricht, entnehmen Sie bitte der Tabelle [6.1](#).

Bei Angabe der Typen sind jeweils die Datenbankspezifika zu beachten: z.B. erlaubt Oracle 8 bei varchar 4000 Zeichen.

Ein Spezialwert ist "block": Felder mit diesem Typ werden nicht zur Java Klasse und Datenbanktabelle hinzugefügt sondern dienen zur Strukturierung des Formulars, z.B. um Bereiche dynamisch auszublenden.

- Werte: Angabe der möglichen Werte bei Auswahllisten. Die Liste kann editiert werden, indem man auf die aktuellen Werte klickt (initial "-").
- Länge: Länge des Feldes in der Datenbank.
- Fremdschlüssel: Diese Checkbox kann aktiviert werden, um einen Fremdschlüssel auf das Feld OID eines anderen Formulars oder eines Persistents des Packages `com.groiss.org` zu erzeugen. Die Checkbox ist nur dann aktiv, wenn als Typ eine Formularklasse (z.B. `com.groiss.forms.myform_1`) oder ein Persistent angegeben wurde (z.B. `com.groiss.org.User`).
- Index: Mit Hilfe dieser Checkbox können Sie einen Index auf das ausgewählte Feld erzeugen lassen.

Hinweise: Eine Besonderheit sind die *java.lang* Typen, erklärt am Beispiel *int* und *java.lang.Integer*: Beide Typen werden als decimal(10) in der Datenbank abgelegt. Bei *int* ist jedoch der Default-Wert 0, bei *java.lang.Integer* ist dieser null. Der Vorteil an *java.lang.Integer* liegt darin, dass z.B. bei Formularen das Feld nicht vorausgefüllt ist und somit auch ein Mussfeld-Check durchgeführt werden kann (0 hingegen ist bereits ein gültiger Wert und der Mussfeld-Check ignoriert dieses Feld).

Normale Datumsfelder haben die Eigenschaft, dass beim Öffnen und Speichern in verschiedenen Zeitzonen die Datumswerte unterschiedlich sind, da keine Zeitzone-Information mitgespeichert wird. Ist dieses Verhalten nicht gewünscht, so gibt es neben den Typen *date* und *dateTime* die Sondertypen *UTCdate* und *UTCdateTime*, die es erlauben, ein Datum unabhängig von der Server- oder Clientzeitzone zu speichern und wieder auszulesen. Das Datum wird dabei immer mit der Zeitzone UTC abgespeichert. In der Subformular-Tabelle ist der Typ (ein Formulartyp), die Id (ein Integer) und eine Beschriftung und Beschreibung anzugeben.

6. Workflowelemente

Typ (xmltype)	DB-Typ	Java-Typ	Kurzbeschreibung
string	VARCHAR	java.lang.String	Text mit bis zu 4000 Zeichen
date	DATE (Oracle)	java.util.Date	Nur Datum wird angezeigt inkl. Datumsauswahl-Dialog
	DATETIME (SQL-Server)		
UTCdate	DATE (Oracle)	java.util.Date	Nur Datum wird angezeigt inkl. Datumsauswahl-Dialog, jedoch unabhängig von Zeitzone (siehe nähere Beschreibung bei <i>Hinweise</i>)
	DATETIME (SQL-Server)		
dateTime	DATETIME	java.util.Date	Datum und Uhrzeit werden angezeigt inkl. Datumsauswahl-Dialog
UTCdateTime	DATE (Oracle)	java.util.Date	Datum und Uhrzeit werden angezeigt inkl. Datumsauswahl-Dialog, jedoch unabhängig von Zeitzone (siehe nähere Beschreibung bei <i>Hinweise</i>)
	DATETIME (SQL-Server)		
time	VARCHAR(8)	java.lang.String	Uhrzeit als Text (z.B. 15:00:00)
int	DECIMAL(10)	int	Ganzzahl (32 bit), Default-Wert ist 0
long	DECIMAL(20) BIGINT (SQL-Server)	long	Ganzzahl (64 bit), Default-Wert ist 0
double	DOUBLE PRECISION	double	Reelle Zahl (64 bit), Default-Wert ist 0
decimal	DECIMAL	long	Reelle Festkomma-Zahl, Default-Wert ist 0. Wenn für das Attribut <i>Länge</i> nur Vorkommastellen angegeben wurden (Bsp.: 3)
		double	Wenn für das Attribut <i>Länge</i> sowohl Vorkommastellen, als auch Nachkommastellen (Bsp.: 3,4) angegeben wurden.
boolean	DECIMAL(1)	boolean	0 oder 1, Default-Wert ist 0
longString	CLOB (Oracle) VARCHAR(MAX) (SQL-Server)	java.lang.String	Text über 4000 Zeichen
binary	BLOB (Oracle)	byte[]	XForms erlauben die Definition des Typs <i>binary</i> ; sollte aber nur via API befüllt werden
	VARBINARY(MAX) (SQL-Server)		
java.lang.Integer	DECIMAL(10)	java.lang.Integer	Ganzzahl (32 bit), Default-Wert ist NULL (weitere Details unter <i>Hinweise</i>)
java.lang.Long	DECIMAL(20) BIGINT (SQL-Server)	java.lang.Long	Ganzzahl (64 bit), Default-Wert ist NULL
java.lang.Double	DOUBLE PRECISION	java.lang.Double	reelle Zahl (64 bit), Default-Wert ist NULL
java.lang.Boolean	DECIMAL(1)	java.lang.Boolean	0 oder 1, Default-Wert ist NULL
Klassenname	DECIMAL(20) – oid VARCHAR(200) – Klassenname	Klasse	Diese Klasse muss das Interface <code>com.groiss.store.Persistent</code> implementieren, oder die Subklasse einer Klasse sein, die dieses Interface implementiert. Im Formular wird die <i>oid</i> gehalten. Angezeigt wird das Ergebnis der <i>toString()</i> -Methode des entsprechenden Objekts. Optional, eine zusätzliche Tabellenspalte (mit dem Namen (<i>column_class</i>) und diesem Typ wird generiert, falls eine Java-Klasse angegeben wird, die ein Interface, eine Abstrakte Klasse oder das Interface <code>com.groiss.store.HasSubclasses</code> implementiert.

Tabelle 6.1: Typ eines Formularfeldes und dessen Entsprechung in der Datenbank bzw.zugehörigen Java-Klasse

Registerkarte: Bindings

Hier können XForms Bindings definiert werden, zur Definition von Bindings siehe die XForms Dokumentation. Jeder Eintrag enthält die folgenden Attribute:

- Sub-Binding von: dient zur Schachtelung von Bindings, im übergeordneten Binding kann bei nodeset eine Expression angegeben werden, die nodesets der Sub-Bindings schränken diese dann weiter ein. Beispiel:

```
<xf:bind nodeset="/data/form/subform[@id=1]/form[position() != last()]">  
  <xf:bind nodeset="timehours" ...
```

Das erste Binding selektiert das erste Subformular, das Sub-Binding ein Feld darin.

- Nodeset: die Formularfelder, die von diesem Binding betroffen sind
- Id: optionale Id eines Bindings
- Required: eine XPath Expression, die angibt, wann das Feld oder die Felder Muss-Felder sind.
- Readonly: eine XPath Expression, die angibt, wann das Feld nur lesbar ist.
- Relevant: eine XPath Expression, die angibt, wann das Feld sichtbar ist.
- Constraint: eine XPath Expression, die angibt, wann der Feldwert gültig ist.
- Calculate: eine XPath Expression zur Berechnung des Feldinhalts.

Registerkarte: Source

Hier kann der Quelltext direkt geändert werden. Beachten Sie, dass Änderungen dazu führen können, dass das Formular im Layout-Tab nicht mehr bearbeitbar ist. Falls Sie die Struktur des Formulars hier ändern und das Formular dem vorgegebenen Schema nicht entspricht, wird das Formular nicht mehr im Layout-Reiter bearbeitbar sein.

Die Menüstruktur

Das Menü Formular enthält vor allem die Standard Funktionen zu Öffnen, Speichern und Beenden:

- **Neu:** Ein neues, leeres Formular wird erzeugt.
- **Öffnen:** Ein bestehendes Formular kann zur Bearbeitung ausgewählt werden.
- **HTML parsen:** Der Quelltext wird geparkt und das Schema wird dem Formular entsprechend angepasst: Kommt ein bestehendes Feld nicht mehr im Formular vor, wird es aus dem Schema entfernt; wurde ein neues Formularfeld gefunden, wird es zum Schema hinzugefügt, es ist dann der Typ und ev. weitere Felder zu ergänzen.

Diese Funktion ist immer dann notwendig, wenn das Formular nicht über den Layout-Reiter bearbeitet wird und Felder hinzugefügt oder entfernt werden.

- **Speichern:** Beim Speichern werden folgende Aktionen durchgeführt: Wurde noch kein Name oder keine Id vergeben, so wird ein Dialog geöffnet, in dem Name, Id, Version und Typ des Formulars (Standard, Dokument oder Ordner) eingegeben werden kann. Danach wird am Server:
 - die Java-Klasse erzeugt und im Verzeichnis *forms* von *EPworX* abgelegt.
 - der HTML Quelltext im Applikationsverzeichnis unter `classes/appl-id/forms/formid_formversion.xhtml` abgelegt (falls kein anderer Pfad definiert wurde).
 - die Datenbanktabelle erzeugt oder angepasst.
- **Speichern unter....:** wie oben, es wird aber eine Kopie mit neuem Namen angelegt
- **Vorschau:** Eine Vorschau des Formulars wird in einem eigenen Fenster geöffnet
- **Eigenschaften:** Die Eigenschaften Maske wird geöffnet - entspricht der Maske beim Anwenden der Funktion Bearbeiten in der Liste der Formulartypen (siehe Abschnitt [Registerkarte: Allgemein](#)).
- **Tabellenspalten:** Hier kann man definieren, welche Spalten in der Tabellenansicht (Subformular oder Formulartabelle) dargestellt werden.
- **HTML downloaden:** Herunterladen des HTML-Quelltextes.
- **Beenden:** Beendet den Formular-Editor.

Das Menü Bearbeiten enthält die Standard-Funktionen Rückgängig, Wiederherstellen, Ausschneiden, Kopieren, Einfügen und Löschen. Darüberhinaus:

- **Feldeigenschaften:** Öffnet das Eigenschaften-Fenster für das gerade im Layout selektierte Element.
- **Nach oben, Nach unten:** Verschieben des selektierten Elements nach oben bzw. unten (im Layout und bei Formularfeldern möglich)
- **Layout:** Es kann der Formulartitel festgelegt werden, außerdem können in einem Script Block Javascript Anweisungen und in einem CSS-Block CSS-Anweisungen definiert werden.
- **Neues Layout erstellen:** Es wird ein Standard-Layout mit den gegebenen Formularfeldern erstellt.
- **Einstellungen:** Zwei Formular-Editor Einstellungen können gesetzt werden: SQL nicht ausführen: wenn angeklickt, werden die Datenbank-Anweisungen nicht ausgeführt. Form Handler ausführen: Die Form-Handler Methoden `beforeShow` und `onShow` werden ausgeführt.

Das Hilfe-Menü öffnet die Hilfe-Seite für den Formular-Editor.

View-Formular mit Formular-Editor erstellen

Um ein neues View-Formular (= Formular, das von einem Basisformular ableitet) mit Formular-Editor zu erstellen, müssen folgende Schritte durchgeführt werden:

- Ein neues Formular mit Formular-Editor erstellen
- Über Menüpunkt *Formular* die Eigenschaften öffnen
- ID und Name eingeben und in das Feld *Basisformular* die ID und Versionsnummer des Basisformulars eintragen:

<formularId>_<versionsnummer>

- Formular speichern und im Formular-Editor den *Schema*-Reiter öffnen
- Neue Formularfelder hinzufügen wobei die Felder gleiche IDs haben müssen wie die Felder des Hauptformulars.
- Schließlich über Menüpunkt *Bearbeiten* neues Layout erstellen und Formular danach speichern.

6.4.2 Anzeigen



Wenn Sie die Funktion *Anzeigen* aktivieren, öffnet sich die Objekt-Detailansicht des ausgewählten Formulars. Allerdings befinden Sie sich sofort auf der Registerkarte *Vorschau*.

6.4.3 View erzeugen



In *EPworX* haben Sie im Rahmen der *Erweiterten Suche* die Möglichkeit unabhängig von der Formularversion eines Formulars nach Inhalten in dessen einzelnen Formularfeldern zu suchen. Damit diese Funktion genutzt werden kann, muss ein Datenbank-View über alle Formularversionen dieses Formulars erzeugt werden. Dieser Datenbank-View enthält all jene Formularfelder, die in allen Formularversionen vorkommen.

Nach dem Aktivieren der Funktion *View erzeugen* öffnet sich eine neue HTML-Seite mit den folgenden Informationen:

1. **View erzeugen: formularId:** Hier ist das SQL-Statement zu sehen, das in der Datenbank abgesetzt wird, um den Datenbank-View zu erzeugen.
2. **View erzeugen:** Ist zum ausgewählten Formular noch kein Datenbank-View vorhanden, ist diese Schaltfläche sichtbar. Nach dem Sie diese aktiviert haben, wird der Datenbank-View erzeugt.
3. **Bestehenden View ersetzen:...**: Ist zum ausgewählten Formular bereits ein Datenbank-View vorhanden, ist diese Schaltfläche sichtbar. Nach dem Sie diese aktiviert haben, wird der bestehende View durch den Neuen ersetzt.

Datenbank-View über alle Formularversionen mit der Id 'Jobform' erzeugen

View erzeugen: form_Jobform

```
create view form_Jobform as  
select oid,subject,recipient,description,finished,type  
from form_Jobform_1
```

View erzeugenAbbrechen

Abbildung 6.9: **Funktion: View erzeugen**

6.4.4 Bearbeiten

Die Objekt-Detailansicht von Formularen gliedert sich in die folgenden Bereiche:

- Allgemein
- Formularfelder
- Datenbanktabelle
- Rechte

6. Workflowelemente

- Standardberechtigungen
- Vorschau
- OrdnerEinstellungen
- Historie
- Zugriff
- Referenzen

Registerkarte: Allgemein

Allgemein Formularfelder Datenbanktabelle Rechte Standardberechtigungen Vorschau OrdnerEinstellungen Historie Zugriff Referenzen

Id: * Jobform

Name: * Jobform 118n: Jobform

Version: * 1

Applikation: * Default ▾

Typ: * Standard ▾

Maskentyp: * XFORMS ▾

Beschreibung: **HTML** Text

Aktiv: ☒

Im DMS verwendbar: ☐

Versionierung: Gemäß Konfiguration ▾

Sortierattribute: 🔍

Namensattribute: recipient 🔍

Suchattribute: 🔍

Schlüsselattribute: 🔍

Event-Handler: ✔️

Basisformular:

XHTML-Datei: mask/example/Jobform.xhtml ⬇️

Breite: 0 **Höhe:** 0

Löschen Speichern und Schließen Speichern Abbrechen

Abbildung 6.10: Objekt-Detailansicht: Formulare

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Id:** Die eindeutige Kurzbezeichnung des Formulars (kann nicht mehr über GUI geändert werden). Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).

Hinweis: Mit Hilfe der Servlet-Methode

`com.groiss.upgrade.FormIdChange.showMask` können Sie die Id eines Formulars ändern. Falls die Checkbox `keep old class` aktiviert wurde, bleiben die Versionseinträge erhalten.

- **Name:** Der Name des Formulars. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#). Der I18n-Link neben diesem Feld ermöglicht direkt die Anpassung der Ressource (falls in Applikationsdetailmaske - Registerkarte *Eigenschaften* festgelegt). Die Änderungen werden nach Aktivieren der Schaltfläche *Speichern* in der Ressource Datei (Strings) dieser Applikation gespeichert (siehe Abschnitt [Ressourcen Editor](#)).
- **Version:** Die Versionsnummer des Formulars (kann nicht mehr geändert werden). Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).
- **Applikation:** Gibt an, zu welcher Applikation das Formular gehört.
- **Typ:** Der Typ bestimmt in welchem Kontext das Formular verwendet werden soll (kann nicht mehr geändert werden). Folgende Typen werden unterschieden:
 - *Standard:* Das Formular kann
 - * an Prozesse mit Hilfe des Prozesseditors angehängt und bei den entsprechenden Tasks verwendet werden,
 - * im *EPworX* DMS erzeugt/bearbeitet werden oder
 - * in einer Formulartabelle unabhängig vom Prozess- bzw. DMS-Kontext verwendet werden.
 - *Dokument:* Dieses Formular wird verwendet, um im DMS die Metadaten von Dokumenten zu beschreiben.
 - *Ordner:* Dieses Formular wird verwendet, um im DMS die Metadaten von Ordnern zu beschreiben.
- **Maskentyp:** Der Maskentyp bestimmt, welches Format für das Formular verwendet wird. Folgende Typen werden unterschieden:
 - XHTML
 - XFORMS
- **Beschreibung:** Freier Text, der im DMS bei der Funktion *Neu* bzw. beim Drag & Drop bei der Auswahl des Formulartyps als Tooltip angezeigt wird. *EPworX* bietet hierbei die Möglichkeit, den Text in diesem Feld entsprechend zu formatieren wie es z.B. aus Microsoft Word bekannt ist, indem Sie den Reiter *HTML* auswählen. Im Reiter *Text* wird dann der erzeugte HTML-Code angezeigt.

- Aktiv: siehe Kapitel [Objekt aktivieren / deaktivieren](#)
- Im DMS verwendbar: Ist diese Checkbox angekreuzt, kann das Formular im DMS verwendet werden. View-Formulare können nicht im DMS verwendet werden.
- Versionierung: Hier können Sie angeben, wann eine Version von einem Formular erstellt werden soll. Diese Einstellung gilt nur für Formulare, die im DMS verwendet werden können, und wird nur wirksam, wenn das Formular nicht als *Prozessformular* verwendet wird. Im letzteren Fall erfolgt bei jedem Weiterleiten automatisch eine Versionierung des Prozessformulars. Es stehen Ihnen vier Möglichkeiten für die Konfiguration der Versionierung zur Verfügung:
 - *gemäß Konfiguration*: Ist diese Option ausgewählt, erfolgt die Versionierung so, wie es in der Serverkonfiguration unter dem Punkt *DMS* (siehe [Installationshandbuch](#)) eingestellt ist.
 - *nicht automatisch*: Ist diese Option ausgewählt, muss der Benutzer das Formular explizit versionieren (siehe Funktion *Version erzeugen* im [Benutzerhandbuch](#)).
 - *bei Bearbeiterwechsel*: Ist diese Option ausgewählt, erfolgt immer dann eine Versionierung des Formulars, wenn sich der Bearbeiter des Formulars geändert hat.
 - *bei jeder Änderung*: Ist diese Option ausgewählt, erfolgt bei jeder Änderung des Formulars eine Versionierung.
- Sortierattribute: Wählen Sie hier jenes Attribut aus, nach dem sortiert werden soll. Dabei ist folgende Angabe möglich:

["+" | "-"] colid { ", " ["+" | "-"] colid } *

Eine durch Bestrich getrennte Liste von Spalten-Ids (colid) kann angegeben werden, wobei jede Spalten-Id einen Prefix haben kann:

- + für aufsteigende Sortierung (Standardverhalten)
- - für absteigende Sortierung
- Namensattribute: Der Inhalt der hier ausgewählten Attribute wird als Namensattribut verwendet. Weiters kann eine Regular Expression angegeben werden, z.B. {formfield} (display_text), {formfield2}. Die geschwungene Klammer ist notwendig, um beim Anzeigen von mehreren Werten die Formularfelder zu identifizieren, d.h. vorheriges Beispiel könnte folgende Ausgabe erzeugen: Hans (Vorname), Moser. *Hans* und *Moser* sind Werte aus dem Formularfeld, *Vorname* ist ein selbst eingegebener Text (= display_text). Weiters gibt es die Möglichkeit einer Formatierungsangabe, z.B. {datefield, date}, {datefield, datetime} oder {datefield, UTCdate}. Mit dieser Definition können Sie den Wert des angegebenen Datumfeldes als reines Datum oder als Datum mit Zeitangabe anzeigen. Nähere Infos dazu finden Sie in der *EPworX* APIDoc unter `com.groiss.ds.StringExpression`.

- Suchattribute: Hier können Sie angeben, nach welchen Attributen gesucht werden soll. Dies wird unter anderem wirksam, wenn man in einem DMS Ordner die Kurzsuche im Toolbar dazu konfiguriert (siehe Abschnitt [Registerkarte: Ordner-einstellungen](#)) und über das Suchfeld nach einem bestimmten Begriff sucht. Wird kein Suchattribut festgelegt, wird standardmäßig der Name des DMS Objekts für die Suche herangezogen.
- Schlüssel-Attribute: Hier können Formularfelder als Schlüsselattribute angegeben werden, die für die Duplikatsprüfung herangezogen werden (z.B. bei XML Export/Import).
- Event-Handler: Hier kann eine Klasse bzw. eine durch Beistrich getrennte Liste von Klassen eingetragen werden, die das Java-Interface

- `com.groiss.dms.FormEventHandler`,
- `com.groiss.dms.XHTMLFormEventHandler` oder
- `com.groiss.dms.DocumentEventHandler`

implementieren. Damit hat der Applikationsprogrammierer die Möglichkeit auf bestimmte Ereignisse im Zusammenhang mit der Formularbearbeitung reagieren zu können.

- Basisformular: Dieses Feld zeigt den Namen des Basisformulars an, von dem das aktuelle Formular abgeleitet wurde.
- XHTML-Datei: Referenz zu einer XHTML Datei im Klassenpfad.
- Breite und Höhe: Spezifiziert die Größe der zu öffnenden HTML Seite.
- Dateifilter: Dieses Feld steht nur beim Typ *Dokument* zur Verfügung und erlaubt die Definition einer mit Beistrich getrennten Liste von Erweiterungen bzw. Mimetypes, die als Filter bei der Dateiauswahl gesetzt werden (bei den DMS-Funktionen *Neu* und *Ersetzen* über die Schaltfläche *Durchsuchen*). Erweiterungen müssen immer mit einem Punkt beginnen! Mimetypes bieten den Vorteil Wildcards (*) anzugeben wie im nachfolgenden Beispiel zu sehen. Es ist auch möglich Erweiterungen und Mimetypes zu mischen. Definitions-Beispiele:
 - `.doc,.docx => zeigt MS Word-Dokumente an`
 - `image/png => zeigt PNG-Dateien an`
 - `image/* => zeigt alle Bilddateien an`
 - `application/msword,.docx => zeigt alle Word-Dokumente an`

Wenn Sie die Schaltfläche **HTML downloaden** aktivieren, können Sie das aktuelle Formular als HTML-Datei lokal auf den Rechner abspeichern. Dort kann es von Ihnen bearbeitet, und anschließend durch die Funktion *HTML parsen* (im Formular-Editor) wieder in das System geladen werden.

6. Workflowelemente

[Allgemein](#) **Formularfelder** [Datenbanktabelle](#) [Rechte](#) [Standardberechtigungen](#) [Vorschau](#) [Ordneereinstellungen](#) [Historie](#) [Zugriff](#) [Referenzen](#)

com.dec.avw.appl.itsm_sw_change_1

Id	Name	Beschreibung	Typ	Länge
category	Kategorie		java.lang.String	1
ci	CI		com.dec.avw.appl.itsm_ci_1	
ciclass	CI Klasse		com.dec.avw.appl.itsm_ci_class_1	
citype	CI Typ		com.dec.avw.appl.itsm_ci_type_1	
code_review_agent	Eingeplant für Code Durchsicht		com.groiss.org.User	
code_review_needed	Code Durchsicht		java.lang.Boolean	
comment_ci	Changelog Text		java.lang.String	4000
descr	Beschreibung		java.lang.String	4000
design_agent	Eingeplant für Design		com.groiss.org.User	
design_needed	Design		java.lang.Boolean	
developer_agent	Eingeplant für Entwicklung		com.groiss.org.User	
docu_change_agent	Eingeplant für Technical writer		com.groiss.org.User	
docu_change_needed	Anpassungen in Doku		java.lang.Boolean	
effort	Aufwand		double	
noctext	Kein Changelog Text		boolean	
priority	Priorität		java.lang.String	1
solutiondescr	Lösungsbeschreibung		java.lang.String	4000
state	Status		java.lang.String	1
subj	Betreff		java.lang.String	200

Anzahl der Einträge: 19

Java-Klassen neu generieren

Delete

Save and close

Save

Schließen

Abbildung 6.11: Formular-Details: Formularfelder

Registerkarte: Formularfelder

Diese Registerkarte zeigt eine Auflistung aller Formularfelder des aktuellen Formulars (Name, Typ, Länge, Fremdschlüssel und Index). Die Bedeutung der Icons für *Fremdschlüssel* und *Index* ist wie folgt:

- rotes Icon: Für dieses Feld wurde kein Index/Fremdschlüssel erzeugt
- grünes Icon: Für dieses Feld wurde ein Fremdschlüssel/Index definiert
- graues Icon: Für dieses Feld kann kein Fremdschlüssel erzeugt werden

Mit Hilfe der Schaltfläche *Java-Klassen neu generieren* kann eine neue Java-Klasse erzeugt werden, auch wenn keine bestehende Klasse vorhanden ist. In diesem Fall wird eine neue Klasse im *forms*-Verzeichnis der *EPworX* Installation gespeichert. Bei einer bereits vorhandenen Klasse wird die bestehende ersetzt.

Registerkarte: Datenbanktabelle

Allgemein
Formularfelder
Datenbanktabelle
Rechte
Standardberechtigungen
Vorschau
OrdnerEinstellungen
Historie
Zugriff
Referenzen

Tabellendefinition:

```

create table form_itsm_test_plan_1(
  oid BIGINT not null primary key,
  task BIGINT,
  transactionId BIGINT,
  acl BIGINT,
  avwcreatedBy BIGINT,
  avwcreatedAt TIMESTAMP(0),
  avwchangedBy BIGINT,
  avwchangedAt TIMESTAMP(0),
  xformsinvalid DECIMAL(1),
  ci BIGINT,
  ci_class varchar(100),
  ciclass BIGINT,
  citype BIGINT,
  citype_class varchar(100),
  name VARCHAR(300),
  pdftemplate_tp BIGINT,
  pdftemplate_tr BIGINT,
  pi BIGINT,
  succ_testplan BIGINT,
  vers decimal(10)
);

```

Fremdschlüssel:

Bedingung	Referenzierende Spalte	Fremdtabelle	Fremdspalte
FORM_ITSM_TEST_PLAN_1_PDFTEMPLATE_TP	PDFTEMPLATE_TP	FORM_ITSM_TEST_TEMPLATES_1	OID
FORM_ITSM_TEST_PLAN_1_PDFTEMPLATE_TR	PDFTEMPLATE_TR	FORM_ITSM_TEST_TEMPLATES_1	OID

Referenziert durch:

Bedingung	Referenzierende Tabelle	Referenzierende Spalte
FORM_ITSM_TEST_RUN_1_TESTPLAN	FORM_ITSM_TEST_RUN_1	TESTPLAN

Löschen
Speichern und Schließen
Speichern
Schließen

Abbildung 6.12: Formular-Details: Datenbanktabelle

Diese Registerkarte zeigt das SQL-Statement für die Erzeugung der Tabelle für das Formular. Weiters werden Informationen zu erzeugten Fremdschlüssel und die *Create Index* Statements angezeigt (siehe Schritt 3 des Form-Wizard). Im Bereich *Referenziert durch* werden die Fremdschlüssel (definiert in anderen Formularen) angezeigt, die auf das aktuelle Formular referenzieren. Im Bereich *Fremdschlüssel* werden die Fremdschlüssel des aktuellen Formulars angezeigt, die auf andere Formulare oder Persistents (Package com.groiss.org) referenzieren.

Registerkarte: Rechte

siehe Kapitel [Formulare und Berechtigung](#)

Registerkarte: Standardberechtigungen

siehe Kapitel [Formulare und Berechtigung](#)

Registerkarte: Vorschau

Der Inhalt dieser Registerkarte ist die Darstellung des Formulars.

Registerkarte: OrdnerEinstellungen

Ist dieses Formular vom Typ *Ordner*, so kann man hier das Design der Ordnerinhalts-tabelle verändern (siehe Abbildung 6.13). Es besteht die Möglichkeit

- Spalten hinzuzufügen, zu ändern, zu löschen bzw. die Anordnung zu verändern
- Funktionen hinzuzufügen, zu löschen bzw. die Anordnung zu verändern
- Formulare hinzuzufügen, zu löschen und deren Verwendung zu setzen (*Erlaubt* bzw. *Nicht erlaubt*)

Diesbezügliche Änderungen, die beim Stammdatenobjekt selbst vorgenommen werden, gelten für alle Instanzen dieses Objekts.

Informationen des Dialoges *OrdnerEinstellungen*:

- **Spalten:** Spalten, die in der DMS-Objekt-Tabelle angezeigt werden.
- **Funktionen:** Toolbarfunktionen, die in der DMS-Objekt-Tabelle angezeigt werden.
- **Formulare:** Formulare, die von diesem DMS-Ordner verwendet (Radio-Button *Erlaubt*) bzw. nicht verwendet (Radio-Button *Nicht erlaubt*) werden dürfen.
- **Hinzufügen:** Mit dem Aktivieren dieser Schaltfläche wird der jeweilige Hinzufüge-Dialog angezeigt, mit dem Sie z.B. neue Spalten zur DMS-Objekt-Tabelle des aktuellen Ordners hinzufügen können. Wie Sie das tun können, wird weiter unten in diesem Abschnitt beschrieben.
- **Bearbeiten:** Diese Funktion steht nur bei *Spalten* zur Verfügung und erlaubt das Editieren der einzelnen Spalteneinträge (analog zu *Hinzufügen*).
- **Entfernen:** Mit dem Aktivieren dieser Funktion werden die zuvor ausgewählten Einträge aus der Liste gelöscht.
- **Nach oben:** Mit dem Aktivieren dieser Funktion wird der zuvor markierte Eintrag um eine Position nach oben verschoben. Bei *Spalten* und *Funktionen* werden sie in der DMS-Objekt-Tabelle bzw. im Toolbar im aktuellen Ordner um eine Position nach links verschoben.
- **Nach unten:** Mit dem Aktivieren dieser Funktion wird der markierte Eintrag um eine Position nach unten verschoben. Bei *Spalten* und *Funktionen* werden sie in der DMS-Objekt-Tabelle bzw. im Toolbar im aktuellen Ordner um eine Position nach rechts verschoben.



Tabbed interface: Allgemein, Formularfelder, Datenbanktabelle, Rechte, Standardberechtigungen, Vorschau, **Ordner-einstellungen**, Historie, Zugriff, Referenzen

Spalten: *

- Name
- Zusatzdaten
- Typ
- Größe
- Zuletzt geändert am
- Erstellt von
- Zuletzt geändert von
- Gesperrt durch
- Eigenschaften

Funktionen: *

- new
- delete
- space
- cut
- copy
- link
- paste
- space
- replace

Formulare: *

☒ Erlaubt ☐ Nicht erlaubt

Standard wiederherstellen

Löschen | Speichern und Schließen | Speichern | Abbrechen

Abbildung 6.13: **Formular-Details: Ordner-einstellungen**

Spalten hinzufügen Mit der Funktion *Hinzufügen* neben der Spaltenliste des Dialoges *Ordner-einstellungen* (siehe Abbildung 6.13) können Sie für den aktuellen Ordner neue Spalten festlegen, die in der DMS–Objekt–Tabelle für diesen Ordner angezeigt werden. Wenn Sie diese Funktion aktivieren, erscheint der Dialog *Spalten* (siehe Abbildung 6.14).

Informationen des Dialoges *Spalten*:

- **Id:** Hier können Sie vom System vorgegebene Spalten eingeben, die bestimmten Eigenschaften von DMS–Objekten entsprechen.
- **Name:** Die Spaltenüberschrift für die neue Spalte in der DMS–Objekt–Tabelle. Der l18n-Link neben diesem Feld ermöglicht direkt die Anpassung der Ressource (falls in Applikationsdetailmaske - Registerkarte *Eigenschaften* festgelegt). Die Änderungen werden nach aktivieren der Schaltfläche *Speichern* in der Ressource

The screenshot shows a configuration window for columns. The fields are as follows:

- Id:** * (required), dropdown menu showing 'changed'.
- Name:** text input with 'ep:changed_at', followed by 'l18n: Zuletzt geändert am'.
- Lokalisieren:** checked checkbox.
- Icon:** empty text input, followed by 'Vorschau:'.
- Colspan:** input '1', **Rowspan:** input '1', **Zeile:** input '0'.
- Sichtbar:** checked checkbox, **Filterbar:** checked checkbox, **Gruppierbar:** checked checkbox.
- Nicht ausblendbar:** unchecked checkbox, **Sortierbar:** checked checkbox.
- Typ:** empty text input.
- Javascript Klasse:** empty text input.
- Komponentenschlüssel:** empty text input.
- Buttons:** 'Ok' (blue) and 'Abbrechen' (light blue).

Abbildung 6.14: **Spalten**

Datei (Strings) dieser Applikation gespeichert (siehe Kapitel [Ressourcen Editor](#)). Dieser Link erscheint nur, wenn der aktuelle Benutzer das Recht *Administration* besitzt und die Checkbox *Lokalisieren* aktiviert wurde!

- **Lokalisieren:** Wenn diese Checkbox aktiviert ist, wird der eingegebene *Name*, sofern im Ressource-Bundle vorhanden, übersetzt.
- **Icon:** Hier können Sie eine Pfad zu einem Icon angeben, das anstelle des Namen angezeigt werden soll.
- **Colspan, Rowspan und Zeile:** Mit diesen Attributen können Sie festlegen, wie die Spalte in der Tabelle angezeigt werden soll, z.B. in einer zweiten Zeile über mehrere Spalten.
- **Sichtbar:** Ist diese Checkbox deaktiviert, ist die Spalte nicht sofort sichtbar, kann jedoch über die Spaltenauswahl ganz rechts außen im Tabellenkopf hinzugefügt werden.
- **Filterbar:** Mit dieser Checkbox geben Sie an, ob die Spalte für den Filtermechanismus verwendet werden kann.
- **Gruppierbar:** Wenn aktiviert, kann die Spalte nach ihren übereinstimmenden Werten gruppiert werden.
- **Nicht ausblendbar:** Diese Checkbox gibt an, ob die Spalte über den Columnpicker (Spaltenauswahl) ausgeblendet werden kann (nur im Smartclient).

- **Sortierbar:** Mit Hilfe dieser Checkbox können Sie festlegen, ob nach dieser Spalte sortiert werden kann.
- **Gruppierbar:** Wenn aktiviert, kann die Spalte nach ihren übereinstimmenden Werten gruppiert werden.
- **Typ:** Hier können Sie einen Spaltentyp definieren, indem Sie einen der folgenden Werte angeben: string, date, dateTime, number (für Zahlen ohne Komma) oder decimal (für Zahlen mit Komma + Darstellung in entsprechenden Form gemäß Decimal-Format Konfiguration).
- **Javascript Klasse:** Durch Angabe eines Pfades auf eine JS-Klasse (Widget) können Sie die Darstellung eines Spaltenwertes beeinflussen. Ein Beispiel dazu finden Sie unter *ep/widget/smartclient/dms/columns/Name*.
- **Ok:** Mit dem Aktivieren dieser Schaltfläche werden Ihre Eingaben übernommen. Im Anschluss daran gelangen Sie zum Dialog *Ordneereinstellungen* (siehe Abbildung 6.14) zurück.
- **Abbrechen:** Mit dem Aktivieren dieser Schaltfläche werden Ihre Eingaben ignoriert und Sie gelangen zum Dialog *Ordneereinstellungen* (siehe Abbildung 6.14) zurück.

Funktion hinzufügen Mit der Funktion *Hinzufügen* neben der Funktionsliste des Dialoges *Ordneereinstellungen* (siehe Abbildung 6.13) können Sie für den aktuellen Ordner neue Toolbarfunktionen festlegen. Wenn Sie diese Funktion aktivieren, erscheint der Dialog *Funktion* (siehe Abbildung 6.15).

Abbildung 6.15: **Funktion**

Informationen des Dialoges *Funktion*:

- **Aktions-Id:** Angabe eines Action Keys, der in *EPworX* definiert ist (z.B. cut, insert, copy, link, paste, startProcess, etc.). Weiters ist es möglich eine Kurzsuche hinzuzufügen, indem man die Schlüsselwörter *search* hinzufügt. Damit werden ein Eingabefeld, die Funktion *Suchen* und die Funktion *Alle Einträge* automatisch generiert.
- **Funktion:** Hier können Sie eine Task-Funktion auswählen.

- **In Aufklappmenü:** Ist diese Checkbox aktiv, wird die Funktion in einem Aufklappmenü angezeigt (nur im Smartclient).
- **Hinzufügen:** Mit dem Aktivieren dieser Schaltfläche werden Ihre Eingaben übernommen. Im Anschluss daran gelangen Sie zum Dialog *OrdnerEinstellungen* (siehe Abbildung 6.13) zurück.
- **Abbrechen:** Mit dem Aktivieren dieser Schaltfläche werden Ihre Eingaben ignoriert und Sie gelangen zum Dialog *OrdnerEinstellungen* (siehe Abbildung 6.13) zurück.

Formular hinzufügen Mit der Funktion *Hinzufügen* neben der Formularliste des Dialoges *OrdnerEinstellungen* (siehe Abbildung 6.13) können Sie für den aktuellen Ordner festlegen, welche Formulare erlaubt bzw. nicht erlaubt sind. Es können hierbei aber immer nur Formulare eines Berechtigungstyps hinzugefügt werden, entweder erlaubt oder nicht erlaubt. Ist der Radio-Button *Erlaubt* ausgewählt, werden in der Dropdown-Liste der verfügbaren Formulare nur jene Formulare angezeigt, die in dieser Liste definiert sind. Ist der Radio-Button *Nicht erlaubt* ausgewählt, werden in der Dropdown-Liste der verfügbaren Formulare all jene Formulare aufgelistet, die nicht in dieser Liste definiert sind.

6.4.5 Report

Für ein oder mehrere Formulare kann ein Report analog zu einem Applikationsreport erstellt werden (siehe Abschnitt [Report](#)).

6.5 Prozesse

Ein Prozess beschreibt den Ablauf eines gesamten Geschäftsprozesses. Sie können Prozesse in *EPworX* über den grafischen Prozesseditor (siehe Kapitel [Grafische Prozessdefinition](#)) oder in WDL (siehe Kapitel [WDL](#)) erstellen.

In der Werkzeugleiste stehen Ihnen neben den in Kapitel [Tabellendarstellung](#) beschriebenen Funktionen, folgende Funktionen zusätzlich zur Bearbeitung von Prozessen zur Verfügung:

- Neuen Prozess erstellen
- Im Prozesseditor bearbeiten
- Report
- WDL / XWDL laden

Die Objekt-Detailansicht von Prozessen gliedert sich in die folgenden Bereiche:

- Allgemein
- Source

- Graph
- Komponenten
- Formular-Sichtbarkeiten
- Eskalation
- Funktionen
- OrdnerEinstellungen
- Dokumenten-Berechtigungen
- Historie
- Zugriff
- Referenzen
- Entscheidungsunterstützung

6.5.1 Neuen Prozess erstellen



Durch das Aktivieren dieser Funktion wird der Prozesseditor gestartet. Damit können Sie einen neuen Prozess erstellen und diesen abspeichern. Der Prozesseditor wird in Kapitel [Grafische Prozessdefinition](#) beschrieben.

6.5.2 Im Prozesseditor bearbeiten



Durch das Aktivieren dieser Funktion wird der Prozesseditor gestartet. Der aktuell selektierte Prozess wird dargestellt, und kann bearbeitet werden.

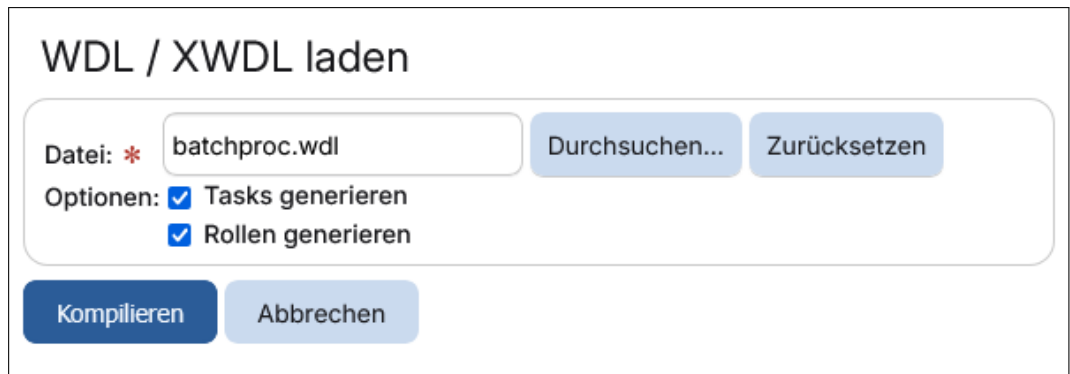
6.5.3 WDL / XWDL laden



Mit Hilfe dieser Funktion können Sie Prozesse, deren WDL–Definition bzw. XWDL–Definition in einer Datei vorliegt, in das System laden und damit einen neuen Prozess erstellen.

Nach dem Aktivieren der Funktion öffnet sich ein HTML-Dialog (siehe Abbildung [6.16](#)) mit den folgenden Informationen:

- **Datei:** Mit Hilfe der Schaltfläche *Durchsuchen* wählen Sie die Datei aus, die die zu ladende Prozess–Definition enthält.
- **Tasks generieren:** Ist diese Checkbox angekreuzt, werden alle Tasks, die im WDL–Script verwendet werden, aber dem System noch nicht bekannt sind, erzeugt.
- **Rollen generieren:** Ist diese Checkbox angekreuzt, werden alle Rollen, die im WDL–Script verwendet werden, aber dem System noch nicht bekannt sind, erzeugt.



WDL / XWDL laden

Datei: * batchproc.wdl Durchsuchen... Zurücksetzen

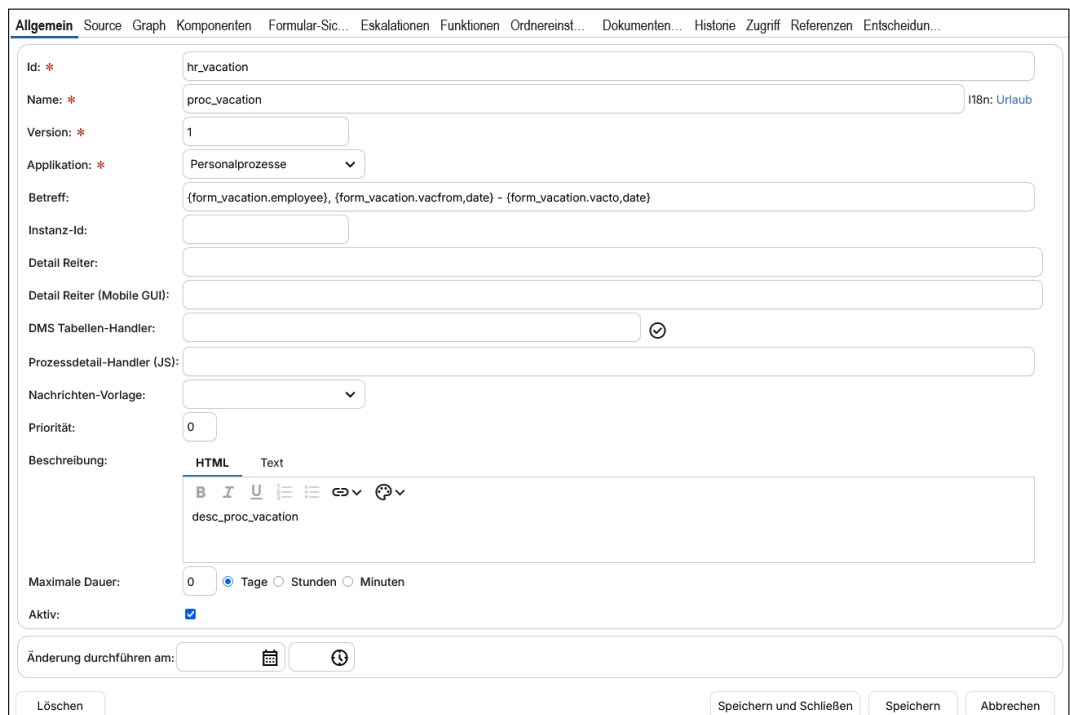
Optionen: ☒ Tasks generieren
☒ Rollen generieren

Kompilieren Abbrechen

Abbildung 6.16: **WDL / XWDL laden**

Durch das Aktivieren der Schaltfläche **Kompilieren** wird das Übersetzungsprogramm (der Compiler) gestartet. Danach meldet das System, ob der Prozess eingetragen wurde oder ob Fehler aufgetreten sind.

6.5.4 Registerkarte: Allgemein



Allgemein Source Graph Komponenten Formular-Sic... Eskalationen Funktionen OrdnerEinst... Dokumenten... Historie Zugriff Referenzen Entscheidun...

Id: * hr_vacation

Name: * proc_vacation 118n: Urlaub

Version: * 1

Applikation: * Personalprozesse

Betreff: {form_vacation.employee}, {form_vacation.vacfrom,date} - {form_vacation.vacto,date}

Instanz-Id:

Detail Reiter:

Detail Reiter (Mobile GUI):

DMS Tabellen-Handler: ✓

Prozessdetail-Handler (JS):

Nachrichten-Vorlage: ▼

Priorität: 0

Beschreibung: **HTML** Text
B I U
desc_proc_vacation

Maximale Dauer: 0 ☒ Tage ☐ Stunden ☐ Minuten

Aktiv: ☒

Änderung durchführen am: 📅 🕒

Löschen Speichern und Schließen Speichern Abbrechen

Abbildung 6.17: **Objekt-Detailansicht: Prozesse**

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Id:** Die eindeutige Kurzbezeichnung des Prozesses. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).
- **Name:** Der Name des Prozesses. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#). Der I18n-Link neben diesem Feld ermöglicht direkt die Anpassung der Ressource (falls in Applikationsdetailmaske - Registerkarte *Eigenschaften* festgelegt). Die Änderungen werden nach Aktivieren der Schaltfläche *Speichern* in der Ressource Datei (Strings) dieser Applikation gespeichert (siehe Abschnitt [Ressourcen Editor](#)).
- **Version:** Die Versionsnummer des Prozesses. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).
- **Applikation:** Die Applikation, in der der Prozess läuft.
- **Betreff:** Formularfeld oder Betrefftext, das/der zur Laufzeit der Prozessinstanz angezeigt wird. Dabei kann weiters ein regulärer Ausdruck angegeben werden.
- **Instanz Id:** Hier kann die Instanz Id festgelegt werden, die jede gestartete Prozessinstanz eindeutig identifiziert. Hierbei besteht auch die Möglichkeit einen regulären Ausdruck im folgenden Format anzugeben:

`{ letter* "{" ("n" | "nn" | "ny" | "nny" | "y" | "yy" | "ou") [ "," formatter ] "}" letter* }*`

Erklärung:

- letter: beliebiges Zeichen
- n: nächste Nummer
- nn: nächste Nummer für diesen Prozess
- ny: nächste Nummer pro Jahr
- nny: nächste Nummer für diesen Prozess für dieses Jahr
- y: Jahr mit 2 Stellen
- yy: Jahr
- ou: Organisationseinheit
- formatter: "date", "datetime", "UTCdate" oder ein Java decimal-format-pattern (optional)

Hinweis: Falls die Instanz Id Leerzeichen enthält und der Parameter *webdav.show.subject* (nähere Informationen dazu im [Installationshandbuch](#)) in der Konfigurationsdatei *ep.conf* auf 1 gesetzt wurde, können DMS-Dokumente, die der Prozessinstanz angehängt wurden, nicht geöffnet werden und folgender Fehler wird angezeigt: Fehler 1002: Das Dokument konnte nicht gefunden werden.

- Detail Reiter: Hier können Sie ein durch Beistrich getrennte Liste von Tab-ID's definieren, welche Reiter in der Ansicht der Prozessinstanz angezeigt werden sollen. Wird nichts angegeben, werden die Standard-Reiter *Formular*, *Dokumente*, *Notizen*, *Historie* und *Prozess* angezeigt. Vordefinierte ID's sind (siehe auch Klasse *ApplicationAdapter* in APIDoc):
 - forms: Prozessformular(e)
 - documents: Dokumente
 - notes: Notizen
 - history: Prozesshistorie
 - process: Prozessbild
 - mails: Ansicht zum Lesen/Erstellen von E-Mails
 - plantab: Zeitplan des Prozesses
 - info: Infobereich, der bei jedem Tab angezeigt wird und Informationen über den aktuellen Task und Prozess enthält
- Detail Reiter (Mobile GUI): Hier können Sie ein durch Beistrich getrennte Liste von Tab-ID's definieren, welche Reiter in der mobilen Ansicht der Prozessinstanz angezeigt werden sollen. Wird nichts angegeben, werden die Standard-Reiter *Formular*, *Dokumente*, *Notizen*, *Historie* und *Prozess* angezeigt. Vordefinierte ID's sind (siehe auch Klasse *ApplicationAdapter* in APIDoc):
 - forms: Prozessformular(e)
 - mblDocuments: Dokumente
 - mblNotes: Notizen
 - mblHistory: Prozesshistorie
 - mblProcess: Prozessbild
 - mblMails: Ansicht zum Lesen von E-Mails
- DMS Tabellen-Handler: Hier können Sie eine eigene Klasse definieren, die für die Anzeige der DMS Tabelle des Prozesses zuständig ist (Reiter *Dokumente* der Prozessinstanz). Nähere Details dazu können im [Programmierhandbuch](#) im Abschnitt [Using the DMS API](#).
- Prozessdetail-Handler (JS): Hier können Sie ein eigenes DOJO-Widget zum Laden der Prozess Detail Reiter angeben.
Dieses Widget muss *ep/widget/smartclient/wl/ProcessDetailsHandler* implementieren. Nähere Informationen dazu finden Sie im [Programmierhandbuch](#).
- Nachrichten-Vorlage: Hier können Sie eine bestehenden Nachrichten-Vorlage auswählen, die beim Versenden von Benachrichtigungs-E-Mails (Notification) angewendet wird. Nähere Informationen dazu finden Sie im [Benutzerhandbuch](#) unter dem Stichwort *E-Mail-Benachrichtigung*. Wurde keine *Nachrichten-Vorlage* ausgewählt, wird standardmäßig die Vorlage mit der Id *notification* verwendet. Alternativ können Sie auch eigene Nachrichten-Vorlagen definieren, die dann ebenso auswählbar sind (siehe Abschnitt [Nachrichten-Vorlagen](#)).

- **Priorität:** Die Priorität des Prozesses.
- **Beschreibung:** Freier Text. *EPworX* bietet hierbei die Möglichkeit, den Text in diesem Feld entsprechend zu formatieren wie es z.B. aus Microsoft Word bekannt ist, indem Sie den Reiter *HTML* auswählen. Im Reiter *Text* wird dann der erzeugte HTML-Code angezeigt.
- **Max. Dauer:** Hier bestimmen Sie die maximale Bearbeitungszeit des Prozesses in Tagen, Stunden oder Minuten.
- **Aktiv:** siehe Kapitel [Objekt aktivieren / deaktivieren](#)
- **Änderung durchführen am:** siehe Kapitel [Änderungen zu späterem Zeitpunkt](#)

In dieser Ansicht steht Ihnen weiters die Funktion *118n* zur Verfügung.

6.5.5 Registerkarte: Source

In der Registerkarte *Source* wird das WDL-Script des ausgewählten Prozesses angezeigt (siehe Abbildung [6.18](#)).

Mit Hilfe der Schaltfläche **BPMN anzeigen** öffnet sich ein neues Fenster, in dem die BPMN-Definition des Prozesses angezeigt wird. Durch Aktivieren der Schaltfläche **BPMN downloaden** können Sie die BPMN-Definition in Ihrem Dateisystem abspeichern. Nähere Informationen dazu finden Sie im [Programmierhandbuch](#) im Kapitel [BPMN](#).

Wenn Sie die Schaltfläche **XWDL anzeigen** aktivieren, öffnet sich ein neues Fenster, in dem die XWDL-Definition des Prozesses angezeigt wird. Wenn Sie die Schaltfläche **XWDL downloaden** aktivieren, können Sie die XWDL-Definition in Ihrem Dateisystem abspeichern.

6.5.6 Registerkarte: Graph

In der Registerkarte zeigt die grafische Darstellung des Prozesses, wie sie im Prozesseditor aussehen würde. Durch Aktivieren der rechten Maustaste öffnet sich ein Kontextmenü mit der Funktion *Grafik herunterladen*, die es erlaubt den angezeigten Prozess als Bild herunterzuladen und im Dateisystem zu speichern.

6.5.7 Registerkarte: Komponenten

Diese Registerkarte zeigt Ihnen, welche Tasks, Formulare, Rollen, Prozesse und Webservices (Webservice Operationen und durch das Webservice importierte Dateien) mit der aktuellen Prozessversion gelinkt sind (siehe Abbildung [6.19](#)). Durch Aktivieren eines Links innerhalb dieses Reiters (z.B. eines Tasks) wird wiederum ein neues Fenster mit der Detailansicht des ausgewählten Objekts geöffnet.

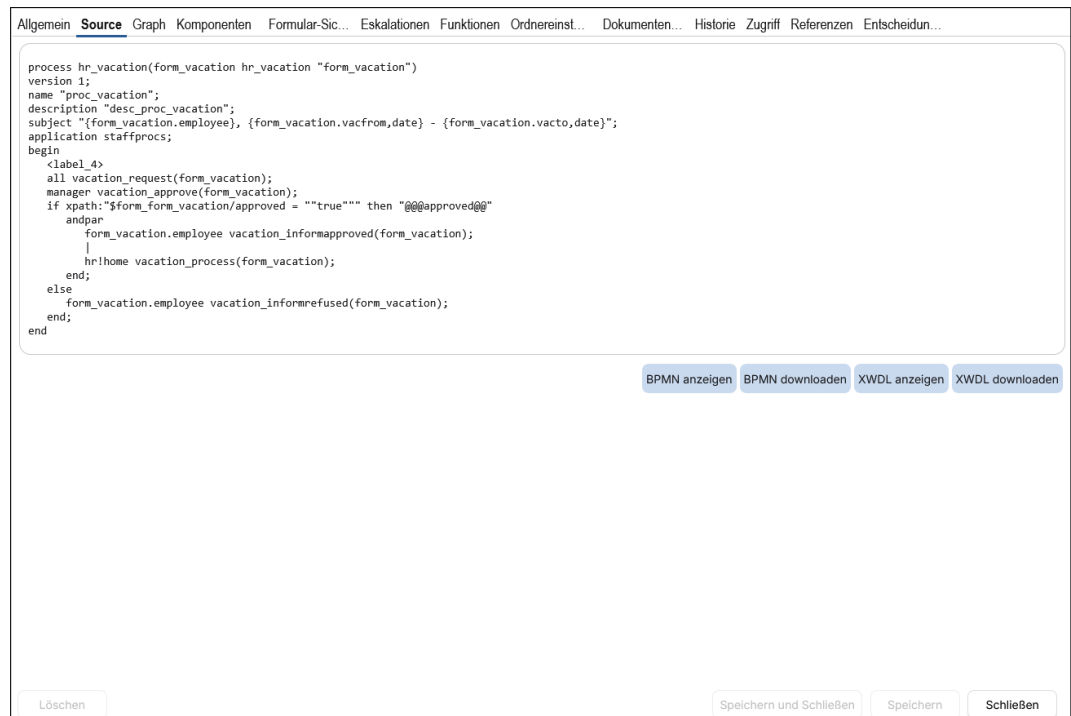


Abbildung 6.18: WDL-Anzeige

6.5.8 Registerkarte: Formular-Sichtbarkeiten

Die Registerkarte *Formulare* zeigt Ihnen eine Übersicht über alle, dem Prozess zugeordneten Formulare. Für jedes Formular wird eine weitere Registerkarte dargestellt, in dem die Sichtbarkeiten der Formularfelder in den einzelnen Tasks aufgelistet werden. In dieser Übersicht scheint ein Task nur auf, wenn ihm bei der Prozessdefinition ein Formular zugeordnet wurde.

Wenn Sie die Sichtbarkeit eines Formularfeldes in einem Task ändern wollen, aktivieren Sie den Link des entsprechenden Tasks. Dadurch wird die HTML-Seite *Anzeigemodi von Formularfeldern* angezeigt.

Informationen der HTML-Seite *Anzeigemodi von Formularfeldern*:

- **Formulartyp:** Die unter *Formularfeld* aufgelisteten Formularfelder beziehen sich auf dieses Formular.
- **Task:** Es handelt sich um ein Formular, das diesem Task zugeordnet ist.
- **Prozess:** Diesem Prozess ist der vorher genannte Task zugeordnet.
- **Sichtbarkeit übernehmen von...:** Hier kann die Sichtbarkeit von einem anderen Task in diesem Prozess übernommen werden, sofern es Unterschiede bzgl. Formular-Sichtbarkeiten zwischen den Tasks gibt.

6. Workflowelemente

Formulare:

Id	Name	Version	Beschreibung
hr_vacation	Urlaub	1	Form for vacations.

Tasks:

Id	Name	Version	Beschreibung
vacation_approve	Genehmigen	1	desc_task_approve_vacation
vacation_informapproved	Genehmigt	1	desc_task_informapproved_vacation
vacation_informrefused	Abgelehnt	1	desc_task_informrefused_vacation
vacation_process	Verbuchen	1	desc_task_process_vacation
vacation_request	Anfordern	1	desc_task_request_vacation
copy	Kopie	1	

Rollen:

Id	Name	Beschreibung
all	Alle	
home	Home	
manager	Abteilungsleitung	

Prozesse:

Id	Name	Version	Beschreibung
----	------	---------	--------------

Webservice Operationen:

Id	Name	Version	Beschreibung
Webservice	Operation	Art	

Importierte Dateien:

Id	Name	Version	Beschreibung
Pfad			

Löschen Speichern und Schließen Speichern Schließen

Abbildung 6.19: **Komponenten**

- **Formularfeld:** Name des Formularfeldes, dessen Sichtbarkeit festgelegt werden soll.
- **Unsichtbar:** Ist dieser Radio-Button aktiviert, wird das Formularfeld nicht angezeigt.
- **Lesen:**
 1. **Gesperrt:** Ist dieser Radio-Button aktiviert, kann das Formularfeld nicht geändert werden und wird grau hinterlegt.
 2. **Text:** Ist dieser Radio-Button aktiviert, kann das Formularfeld auch nicht geändert werden, jedoch wird es nicht grau hinterlegt. Bei XForms ist diese Einstellung nicht wirksam, da XForms diese nicht unterstützen!
- **Schreibbar:**
 1. **Optional:** Ist dieser Radio-Button aktiviert, kann das Formularfeld geändert werden.
 2. **Muss:** Ist dieser Radio-Button aktiviert, ist das Formularfeld änderbar und es muss befüllt werden.

Gibt es zudem ein Subformular, dessen Sichtbarkeit gesetzt werden sollte, gliedert sich die Information der HTML-Seite *Anzeigemodi von Formularfeldern* wie folgt:

- *Formulartyp, Task, Prozess* und *Sichtbarkeit übernehmen von* sind gleich.

6. Workflowelemente

Allgemein Source Graph Komponenten Formular-Si... Eskalationen Funktionen Ordneinst... Dokumenten... Historie Zugriff Referenzen Entscheidun...										
SW-Änderung (swc)										
Formularfelder			Tasks							
↑	Pos. Id	Name	Anfordern	Design	Durchsicht	Design Implementieren	Code Durchsicht	Testen	Dokumentation	erzeuge
1	category	Kategorie	man	man	man	man	man	man	man	
2	ci	CI	rw	rw	rw	rw	rw	rw	rw	
3	ciclass	CI Klasse	man	man	man	man	man	man	man	
4	citype	CI Typ	man	man	man	man	man	man	man	
5	code_review_agent	Eingeplant für Code Durchsicht	rw	rw	rw	rw	ro	ro	ro	
6	code_review_needed	Code Durchsicht	rw	rw	rw	man	ro	ro	ro	
7	comment_ci	Changelog Text	inv	inv	inv	rw	rw	rw	rw	
8	descr	Beschreibung	rw	rw	rw	rw	rw	rw	rw	
9	design_agent	Eingeplant für Design	rw	ro	ro	ro	ro	ro	ro	
10	design_needed	Design	rw	ro	ro	ro	ro	ro	ro	
11	developer_agent	Eingeplant für Entwicklung	rw	rw	rw	ro	ro	ro	ro	
12	docu_change_agent	Eingeplant für Technical writing	rw	rw	rw	rw	rw	rw	ro	
13	docu_change_needed	Anpassungen in Doku	rw	rw	rw	rw	rw	man	ro	
14	effort	Aufwand	rw	rw	rw	rw	rw	rw	rw	
15	nocitext	Kein Changelog Text	inv	inv	inv	rw	rw	rw	rw	
16	priority	Priorität	man	man	man	man	man	man	man	
17	solutiondescr	Lösungsbeschreibung	inv	inv	inv	rw	rw	rw	rw	
18	state	Status	rw	rw	rw	rw	rw	rw	rw	
19	subj	Betreff	man	man	man	man	man	man	man	
20	com.dec.avw.appl.itsm_change_1	Change 1	inv	inv	inv	rw	rw	rw	rw	
21	section_solution	section_solution	inv	inv	inv	rw	rw	rw	rw	

Abbildung 6.20: Formularsichtbarkeiten

- Tabelle: Name des Subformulars, dessen Sichtbarkeit festgelegt werden soll.
- Unsichtbar: Ist dieser Radio-Button aktiviert, wird die Tabelle des Subformulars nicht angezeigt.
- Nur Lesen: Ist dieser Radio-Button aktiviert, wird die Tabelle des Subformulars angezeigt, jedoch kann diese nicht verändert werden.
- Optional: Ist dieser Radio-Button aktiviert, wird die Tabelle des Subformulars angezeigt und kann über den Button *Neuer Tabelleneintrag* verändert werden.
- Kein Einfügen/Löschen: Ist dieser Radio-Button aktiviert, können keine weiteren Einträge der Tabelle hinzugefügt bzw. gelöscht werden.

Weiters gibt es die Möglichkeit die Sichtbarkeit eines Formularfeldes für alle Tasks festzulegen. Dafür müssen Sie auf den Link des entsprechenden Formularfeldes klicken und es öffnet sich die HTML-Seite *Anzeigemodi von Formularfeldern*. Diese ist ähnlich der HTML-Seite für Formularfelder, jedoch werden hier die Sichtbarkeiten für Tasks anstatt für Formularfelder festgelegt.

Durch das Aktivieren der Schaltfläche **Vorschau** öffnet sich ein neues Fenster, in dem das Formular bei Anwendung der aktuell gespeicherten Sichtbarkeiten dargestellt wird. Durch das Aktivieren der Schaltfläche **Ok** werden Ihre Änderungen bezüglich der Sichtbarkeiten der Formularfelder gespeichert.

Durch das Aktivieren der Schaltfläche **Schließen** werden Ihre Änderungen verworfen und die HTML-Seite *Anzeigemodi von Formularfeldern* wird geschlossen.

6.5.9 Registerkarte: Eskalationen

Eskalationsobjekte können Sie verwenden, um auf Zeitüberschreitungen bzw. das vorzeitige Erreichen eines Zeitpunktes bei der Ausführung von Prozesse reagieren zu können. Es ist möglich verschiedene Aktionen zu definieren, die bei einer Zeitüberschreitung ausgeführt werden sollen. Der *EPworX* –Timer *Escalations* ist für die Prüfung der Zeitüberschreitungen verantwortlich. Läuft dieser Timer nicht, erfolgt auch keine Prüfung der Zeitüberschreitungen!

Allgemein

Eskalationstyp: * Prozessfähigkeit ▾

Schritt: ▾

Zeitabstand: * 0 Tage ▾

Aktion *

☐ E-Mail senden

Aktuelle Akteure ☐

Empfänger: ▾

Nachrichten-Vorlage: ▾

☒ Methode aufrufen

Java Methode: * com.groiss.demo.DemoEscalation.fire() ✓

☐ Schritt starten

Schritt: ▾

Beschreibung: ▾

Löschen Speichern und Schließen Speichern Abbrechen

Abbildung 6.21: Objekt-Detailansicht: Eskalation

Die Registerkarte *Eskalation* zeigt zunächst alle zu diesem Prozess definierten Eskalationsobjekte. Sie können diese mit den Funktionen der Werkzeugleiste bearbeiten.

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Eskalationstyp:** Hier können Sie zwischen folgenden Eskalationstypen wählen:

1. **Prozessfälligkeit:** Diese Eskalation bezieht sich auf die Fälligkeit des Prozesses, das beim Prozessstart angegeben wurde.
 2. **Aktivitätsfälligkeiten:** Wenn dieser Eskalationstyp gewählt wurde, wird die Eskalation für irgendeinen Task im Prozess ausgeführt, wenn das Fälligkeitsdatum (des aktiven Tasks) überschritten wurde. Für jeden Task kann im Reiter *Allgemein* die *Maximale Dauer* festgelegt werden, auf die dieser Eskalationstyp reagiert.
 3. **Aktivitätsfälligkeit:** Im Gegensatz zum Typ *Aktivitätsfälligkeiten* wird bei diesem Eskalationstyp genau ein Schritt (Task) ausgewählt, für den die Eskalation ausgelöst werden soll.
 4. **Aktivität nicht angenommen:** Diese Art der Eskalation wird dann ausgelöst, wenn der Task eine gewisse Zeit (*Zeitabstand*) im Rollen–Arbeitskorb verweilt. Dabei ist zu beachten, dass dieser Typ **nur** im Rollen–Arbeitskorb funktioniert.
 5. **Aktivität ungesehen:** Dieser Eskalationstyp reagiert auf den aktuellen Schritt, wenn dieser im Arbeitskorb bzw. Rollenarbeitskorb als *Ungesehen* gesetzt ist. Dies ist z.B. der Fall, wenn der Task im Arbeitskorb landet und der Benutzer ihn noch nicht angesehen hat.
 6. **Aktivität nicht fertig:** Wurde ein Task gestartet und dieser in dem angegebenen Zeitraum (= Zeitabstand) nicht beendet, kann dieser Eskalationstyp verwendet werden, um auf ein solches Ereignis zu reagieren. Dieser Eskalationstyp wird beim Eintreten einer Eskalation für den ausgewählten Schritt im *Arbeitskorb*, *Rollen-Arbeitskorb*, *Wiedervorlage* oder *Rollen-Wiedervorlage* ausgelöst.
 7. **Batch nicht fertig:** Dieser Typ bezieht sich auf nicht beendete Batch-Schritte in einem Prozess. Der Schritt wird über den Step-Label spezifiziert.
 8. **Sync nicht erfolgt:** Analog zu *Batch nicht fertig*, jedoch wird hierbei auf Sync-Schritte reagiert.
 9. **Eingangsnachricht nicht erhalten:** Dieser Typ bezieht sich auf nicht beendete RECEIVE-Schritte (= Eingangsnachricht) in einem Prozess. Der Schritt wird auch hier über den Step-Label spezifiziert.
- **Schritt:** Falls mehrere Schritte im Prozess den gleichen Task verwenden, können Sie hier den Schritt auswählen, auf den die Eskalation angewendet werden soll. Ein Schritt ist nicht auswählbar für Eskalationstyp *Prozessfälligkeit* bzw. *Aktivitätsfälligkeiten*.
 - **Zeitabstand:** Dieser Zeitabstand gibt an, wann eine Zeitüberschreitung vom *Escalations*–Timer festgestellt werden soll. Dieser Wert kann auch negativ sein, um vorzeitig auf eine gewisse Deadline reagieren zu können. Dies macht jedoch nur bei *Sollzeiten* Sinn! Ist dieser Zeitabstand überschritten wird die definierte Aktion ausgeführt. Dabei können Sie zwischen *Stunden*, *Tage* und *Arbeitstage* auswählen. Als Nicht-Arbeitstage werden standardmäßig immer Samstag und Sonntag angenommen. Sie können weitere Nicht-Arbeitstage unter *Konfiguration* → *Kalender*

festlegen (siehe Kapitel [Konfiguration](#)).

Beispiel:

Wenn 2 Arbeitstage (48h) eingegeben wurde, heute Donnerstag 16 Uhr ist und der Prozess am darauffolgenden Montag um 16 Uhr die Fälligkeit hat, wird die Eskalation ausgelöst (Annahme: nur Samstag und Sonntag sind Nicht-Arbeitstage).

- **Aktion:** Es werden drei Arten von Aktionen unterschieden:
 - *E-Mail senden:* Tritt bei der Bearbeitung eines Tasks eine Zeitüberschreitung auf, wird eine entsprechende E-Mail an die im Feld *Empfänger* eingetragene E-Mail Adresse geschickt. Ist die Option *Aktuelle Akteure* ausgewählt, wird an die Akteure des aktuellen Tasks eine E-Mail versandt, sofern beim Benutzer eine E-Mailadresse eingetragen wurde. Ist der aktuelle Akteur des Tasks eine Rolle, so werden an alle Benutzer, die die Rolle (in dieser Organisationseinheit) besitzen, per E-Mail benachrichtigt. Wurde keine *Nachrichten-Vorlage* ausgewählt, wird standardmäßig die Vorlage mit der Id *escalation* verwendet. Alternativ können Sie auch eigene Nachrichten-Vorlagen definieren und diese hier dann auswählen (siehe Abschnitt [Nachrichten-Vorlagen](#)).

Damit das Verschicken der E-Mail funktioniert, muss in der Serverkonfiguration unter *Kommunikation* im Feld *SMTP Host* ein gültiger Mailserver eingetragen sein (siehe [Installationshandbuch](#)). Auf dieser HTML-Seite kann im Feld *Mail Sender* eine gültige E-Mail Adresse angegeben werden, die dann als *Sender* der durch die Zeitüberschreitung verschickten E-Mails aufscheint. Wird in diesem Feld kein Wert eingetragen, wird als Sender der Default-Wert *enterprise@Hostname* des *EPworX* –Servers eingetragen.

Beispiel:

`enterprise@lima.groiss.com`

- *Methode aufrufen:* Hier kann eine Java-Methode angegeben werden, die im Falle einer Zeitüberschreitung ausgeführt werden soll. Dabei besonders sollen die Methoden *SystemAction.increasePriority(String offset)* und *SystemAction.untake()* erwähnt werden, die als Aktion verwendet werden können. Mit der Methode *SystemAction.increasePriority(String offset)* können Sie die Prozesspriorität um einen Wert erhöhen (offset), den Sie als Parameter übergeben. Mit der Methode *SystemAction.untake()* ist es möglich einen Task in den Rollen-Arbeitskorb zurückzugeben.
 - Beispiel:**
`com.groiss.wf.SystemAction.increasePriority("2")`
 - *Schritt starten:* Tritt bei der Bearbeitung eines Tasks eine Zeitüberschreitung auf, wird der im Feld *Schritt* ausgewählte Schritt gestartet. Eskalationsschritte können im Prozesseditor unter dem Menüpunkt *Prozess/Eskalationen* festgelegt werden (siehe Abschnitt [Eskalationen](#)).
- Beschreibung: Freier Text

6.5.10 Registerkarte: Funktionen

Diese Maske ist vom Aufbau her analog zu [Registerkarte: Funktionen](#). Funktionen, die hier zugeordnet werden, sind nur in der *Historie* des Prozesses sichtbar, d.h. man ordnet auf diese Weise nicht die Funktion allen Tasks eines Prozesses zu, selbst dann nicht, wenn bei der Funktion *zu allen Tasks* angeklickt ist.

6.5.11 Registerkarte: Ordneinstellungen

In diesem Reiter können Sie die DMS-Ordneinstellungen für diesen Prozess vornehmen. Diese Funktion ist dabei äquivalent zu den Einstellungen, die bei Formularen vom Typ *Ordner* getätigt werden können (siehe Abschnitt [Registerkarte: Ordneinstellungen](#)). Es besteht die Möglichkeit

- Spalten hinzuzufügen, zu ändern, zu löschen bzw. die Anordnung zu verändern
- Funktionen hinzuzufügen, zu löschen bzw. die Anordnung zu verändern
- Formulare hinzuzufügen, zu löschen und deren Verwendung zu setzen (*Erlaubt* bzw. *Nicht erlaubt*)

6.5.12 Registerkarte: Dokumenten-Berechtigungen

In diesem Reiter können Sie Berechtigungen pro Prozessschritt für Dokumente erstellen und somit das Standard Berechtigungsverhalten von *EPworX* für den Prozess ändern. Neben den Standardfunktionen zum Erstellen, Bearbeiten und Löschen von Berechtigungseinträgen bietet dieser Reiter eine Kopie-Funktion, um Einträge zu duplizieren.

Die Semantik der Berechtigungen ist wie folgt:

Ein Eintrag in der Tabelle *Dokumenten-Berechtigungen* entspricht einem Berechtigungseintrag mit der fiktiven Rolle *Prozessakteur-von-Schritt-x-in-Prozess-y*. Wenn der Benutzer Akteur von Schritt x ist, dann sind die für diesen Schritt definierten Berechtigungen wirksam. Die Auswertung von negativen Berechtigungen zieht dabei stärker als die der positiven. Es gibt zudem 2 Besonderheiten:

1. Ist keine Dokumenten-Berechtigung definiert, gilt weiterhin das Standard Berechtigungsverhalten von *EPworX*.
2. Schritt und Recht können leer sein was soviel bedeutet wie, dass dann die Berechtigung für alle Schritte bzw. für alle Rechte gilt.

Informationen der Registerkarte *Allgemein* eines Dokumenten-Berechtigungen Eintrags:

- **Schritt:** Optionale Auswahl eines interaktiven Prozessschrittes, d.h. es können keine Systemschritte ausgewählt werden, die automatisch von der Workflow Engine ausgeführt werden (system, batch, etc.). Das nachfolgende definierte Recht wird nur in diesem Prozessschritt angewendet. Wird kein Schritt ausgewählt, dann gilt das nachfolgend definierte Recht für alle Schritte des Prozesses.

Allgemein

Schritt: vacation_informapproved ▼

Recht: Objekte ansehen (Default) ▼

Formulartyp: hr_recruiting (Personalauswahl 1) ▼

Zugriff: * Erlaubt ▼

Löschen Speichern und Schließen Speichern Abbrechen

Abbildung 6.22: **Objekt-Detailansicht: Dokumenten-Berechtigungen**

- **Recht:** Auswahl des Rechts, das für das Dokument (= Formulartyp) im optional ausgewählten Schritt angewendet werden soll. Ist kein Recht ausgewählt, werden alle auswählbaren Rechte für den ausgewählten Schritt angewendet.
- **Formulartyp:** Auswahl des Formulartyps, auf den die Berechtigung angewendet werden soll.
- **Zugriff:** Entweder *Erlaubt* für die Definition als positive Berechtigung oder *Nicht erlaubt* für negative Berechtigung.

Anwendungsbeispiele:

Definition des Berechtigungen Eintrags: docperm(step, right, formtype, access)

1. In Schritt inform dürfen alle Dokumente nur angesehen werden:
 - docperm(*, *, *, allowed)
 - docperm(inform, edit, *, not allowed)
 - docperm(inform, delete, *, not allowed)
2. Alle Dokumente sind immer read-only, außer Dokumente vom Typ x:
 - docperm(*, view, *, allowed)
 - docperm(*, edit, x, allowed)

- docperm(*, delete, x, allowed)

3. In Schritt step2 dürfen nur Dokumente vom Typ y hinzugefügt werden:

- docperm(*, edit, *, allowed)
- docperm(*, delete, *, allowed)
- docperm(step1, create, *, allowed)
- docperm(step2, create, y, allowed)
- docperm(step3, create, *, allowed)

6.5.13 Registerkarte: Entscheidungsunterstützung

Über Klassifikatorzuweisungen wird festgelegt, wie ein Formularfeld klassifiziert werden soll. So wird festgelegt, welche Felder zur Klassifizierung von welchem Ausgabefeld herangezogen werden, für welche Tasks das geschehen soll und welcher Klassifikator verwendet werden soll.

Klassifikatorzuweisungen sind persistente Objekte und werden daher in der Datenbank gespeichert. Nach einem Neustart des Systems sind sie nach wie vor verfügbar.

Klassifikatorzuweisungen einer Prozessdefinition verwalten

Beim Bearbeiten einer Prozessdefinition in der Administration kann erkannt werden, dass der Reiter ganz rechts für das Verwalten der Klassifikatorzuweisungen genutzt werden kann.

Allgemein Source Graph Komponenten Formular-Sic... Eskalationen Funktionen OrdnerEinst... Dokumenten... Historie Zugriff Referenzen Entscheidung...				
Name	Aktiv	Ausgabe	Klassifikator Eingabe	Task
com.groiss.ml.classifier.impl.DecisionTree				
Anfordern	●	Urlaub Kommentar	Urlaub MitarbeiterIn, STANDARD Organisationseinheit	Anfordern
approved	●	Urlaub Genehmigt	Urlaub Tage, Urlaub Vertretung, Urlaub Urlaub von, Urlaub bis	Genehmigt
substitution	●	Urlaub Vertretung	Urlaub Genehmigt von, Urlaub Kommentar, Urlaub Tage, Urlaub MitarbeiterIn, Urlaub notapprovedreason, Urlaub Organisationseinheit, Urlaub type, Urlaub Urlaub von, Urlaub bis, STANDARD Prozessstartzeitpunkt, STANDARD Prozessersteller, STANDARD Organisationseinheit	Genehmigen
com.groiss.ml.classifier.tree.manual.ManualTreeClassifier				
vacation	●	Urlaub MitarbeiterIn	Urlaub Tage	Genehmigen

Löschen
Speichern und Schließen
Speichern
Schließen

Abbildung 6.23: Die Klassifikatorzuweisungen

In Abbildung 6.23 kann erkannt werden, dass Klassifikatorzuweisungen hinzugefügt, bearbeitet und gelöscht werden können. Zudem ist auch ein Auslösen des Trainingsvorganges möglich. Genau ein Klassifikator gehört zu einer Klassifikatorzuweisung.

Neue Klassifikatorzuweisungen hinzufügen

Es gibt zwei unterschiedliche Arten von Klassifikatorzuweisungen:



- Die erste beschreibt einen Klassifikator, der nicht mit maschinellem Lernen zu tun hat. Das Wissen wird von einem Experten eingefügt. Die Baumstruktur stellt dann einen Entscheidungsbaum dar.



- Die zweite ist das Hinzufügen einer Zuweisung, die auf Techniken des maschinellen Lernens zurückgreift. Es ist nicht möglich, genaue Regeln vorzugeben, aber es können die Felder und klassifikatorspezifische Variablen gesetzt werden. Mehr zu diesem Thema kann in Sektion [Klassifikatorzuweisungen bearbeiten](#) nachgeschlagen werden.

Klassifikatorzuweisungen bearbeiten

Die Bearbeitung von Klassifikatorzuweisungen basierend auf dem maschinellen Lernen unterscheidet sich von dem mit den Expertenregeln.

Klassifikator basierend auf Expertenregeln

Name: vacation

Aktiv: ☐

Ausgabefeld: Urlaub | MitarbeiterIn

Tasks: Taskauswahl bearbeiten

+ Knoten hinzufügen ✖ Knoten löschen ✖ Knoten rekursiv löschen

▼ Wert beibehalten. -> Urlaub | Tage

 > Urlaub | Tage <= 10 -> Wert beibehalten. -> Urlaub | Tage

 > Urlaub | Tage > 10 -> nobody

Löschen Speichern und Schließen Speichern Abbrechen

Abbildung 6.24: Klassifikator basierend auf Expertenregeln

Abbildung 6.24 zeigt die Schnittstelle zum Bearbeiten von Bäumen mit Expertenregeln. Im oberen Bereich können die folgenden Eingabefelder ausgemacht werden:

- Name: der Name der Klassifikatorzuweisung

- Aktiv: legt fest, ob diese Klassifikatorzuweisung im Moment verwendet werden soll.
- Ausgabefeld: das zu klassifizierende Feld.
- Tasks: Auswahl von Tasks, für die die Klassifikatorzuweisung verwendet werden soll. Beim Klicken auf den zugehörigen Button wird ein Dialog (Abbildung 6.25) zur Auswahl dargestellt.

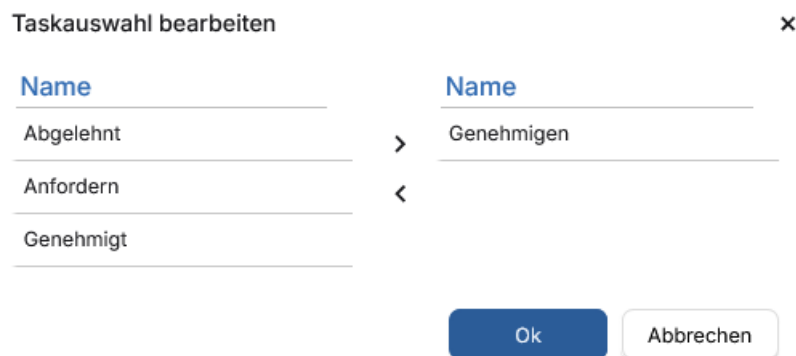


Abbildung 6.25: Taskauswahl für den Baum mit Expertenregeln

Die Semantik der Zeilen kann wie folgt erklärt werden:



- Zweigknoten: Der ausgewählte Knoten beschreibt ein Attribut, an dem geteilt wird. Ist dies der Fall, so muss der Knoten zumindest einen Kindknoten haben.
- Aufteilungsattribut: Zeit an welchem Attribut geteilt wird.
- Blattknoten: Dieser Knoten beschreibt keine Attribute, an denen geteilt werden kann und erlaubt daher auch keine Kindknoten.
- Zweigausswahl: Auf der linken Seite des Pfeils ist das Auswahlkriterium eines Zweiges zu erkennen. Der Wurzelknoten hat nie eine Auswahl.

Bearbeiten des Baumes

Um die Knoten verwalten zu können, wurden die Folgenden Aktionen hinzugefügt:



- Knoten hinzufügen: Fügt dem selektierten Knoten einen Kindknoten an. Dies ist nur möglich wenn an diesem Knoten ein Attribut definiert wurde, an dem geteilt werden soll.
- Knoten löschen: Entfernt den gerade ausgewählten Knoten. Diese Aktion ist nur verfügbar, wenn der ausgewählte Knoten genau einen Kindknoten aufweist. Hat der Knoten einen Kindknoten, so wird dieser an die Position des soeben gelöschten Knotens gehoben. Der Wert dieses Knotens muss dann eventuell angepasst werden.



- Knoten rekursiv löschen: Entfernt den ausgewählten Knoten mitsamt allen seinen Nachfolgern.

Um einen Knoten zu bearbeiten, muss dieser doppelgeklickt werden. Ein Dialog öffnet sich dann.

Knoten bearbeiten ×

Auswahl: Urlaub | Genehmigt = true ▼

Beschriftung: ☐ ▼

Aufteilung: ☒ Urlaub | Vertretung ▼

Ok **Abbrechen**

Abbildung 6.26: Blattknoten bearbeiten im Baum

Knoten bearbeiten ×

Auswahl: Urlaub | Tage (num) <= 10

Beschriftung: ☐ ▼

Aufteilung: ☒ Urlaub | Tage (num) ▼

Wert für Aufteilung: 5

Ok **Abbrechen**

Abbildung 6.27: Knoten bearbeiten mit numerischem Attribut zur Aufteilung

Abbildung 6.26 zeigt die Bearbeitungsansicht eines Blattknotens. In diesem Fall ist wird an einem nominellen Feld geteilt. Abbildung 6.27 zeigt die Bearbeitungsansicht eines Zweigknotens, wo an einem numerischen Feld geteilt wird. Ein Spezialfall einer Aufteilung an einem numerischen Wert kann in Abbildung 6.28 gesehen werden, wo Datum- und Zeitwert zum Einsatz kommen. Im Falle eines numerischen Attributes wird die Aufteilung am angegebenen Wert ausgeführt ($\leq$, $>$).

- Auswahl: Beschreibt, wie dieser Zweig selektiert wird. Für nominale Attribute ist die Operator vorgegeben und der Wert muss ausgewählt werden. Bei numerischen Attribute ist der Wert vorgegeben und der Operator kann zwischen $\leq$ und $>$

Knoten bearbeiten

Beschriftung: ☐ [Dropdown]

Aufteilung: ☒ Urlaub | Urlaub von (num) [Dropdown]

Wert für Aufteilung: 07-08-2026 [Calendar Icon] 09:15 [Clock Icon]

Ok Abbrechen

Abbildung 6.28: Knoten bearbeiten, es wird nach Datum geteilt

gewählt werden.

- **Beschriftung:** Legt fest, wie die Instanz klassifiziert werden soll, wenn dieser Knoten der ausschlaggebende Blattknoten ist (letzter Knoten im Pfad). Wenn die Checkbox nicht ausgewählt ist, so wird der Wert des Ausgabefeldes nicht beeinflusst.
- **Aufteilung:** Mit der Selektion eines Feldes wird dargestellt, dass es sich um einen Zweigknoten handelt. In diesem Fall muss dieser Knoten zumindest einen Kindknoten aufweisen.
- **Wert für Aufteilung:** Wird nur dargestellt, wenn es sich beim Aufteilungsattribut um ein numerisches Attribut handelt. In diesem Fall wird eine Textbox dargestellt. Ist das Attribut ein Datumsattribut, so werden entsprechende Eingabefelder angeboten.

Klassifikator für maschinelles Lernen

Da sich die List der möglichen Eingabefelder bei der Auswahl des Ausgabefeldes ändert, ist es nicht möglich ein Feld sowohl als Eingabe- als auch als Ausgabefeld auszuwählen. Zur Auswahl gibt es Formularfelder und generierte Felder.

Für die Tasks ist die Situation ähnlich. Es ist nicht erlaubt, mehrere Klassifikatorzuweisungen für ein Ausgabefeld und einen Task zu erstellen. Wird ein Ausgabefeld ausgewählt, so werden die Listen zum Auswählen der Tasks automatisch aktualisiert, um nur die verfügbaren Tasks anzuzeigen. Wird ein Ausgabefeld bereits in Kombination mit jedem Task verwendet, so wird dieses nicht mehr angeboten.

Weiters muss das Ausgabefeld nicht nominal sein. In solch einem Fall wird es zu einem nominalen Feld umgewandelt, da die Klassifizierungsalgorithmen nur mit diesen Feldern als Ausgabe umgehen können.

In Abbildung 6.29 ist zu sehen, dass der Name der Klassifikatorzuweisung *approve vacation* ist und drei Eingabefelder aus dem Formular verwendet. Es wurde auch nur ein Task ausgewählt.

Die Klassifikatoroptionen können als Zeichenkette gegeben werden, wobei die Namen der Argumente einen Bindestrich vorgestellt bekommen. Wird ein Entscheidungsbaum

Name:

Aktiv: ☒

Eingabefelder

↑ Quelle	Feld
Urlaub	Genehmigt von
Urlaub	Kommentar
Urlaub	MitarbeiterIn
Urlaub	notapprovedreason

Ausgabefeld:

Tasks

Name
Abgelehnt
Anfordern
Genehmigen

Klassifikator:

Klassifikator-Optionen:

Abbildung 6.29: Eine Klassifikatorzuweisung

verwendet, so kann mithilfe dieses Optionenfeldes das Pruning konfiguriert werden.

Klassifikatorzuweisungen kopieren



Die zurzeit ausgewählten Klassifikatorzuweisungen werden kopiert und in der Datenbank gespeichert. Auch das PMML-Feld wird kopiert. Um ungünstige Zustände zu vermeiden werden neue Klassifikatorzuweisungen inaktiv gesetzt. Der Name wird mit dem Präfix *copy_of_* versehen.

Klassifikatorzuweisungen umwandeln



Manuelle Klassifikatorzuweisungen können zu nicht-manuellen umgewandelt werden und umgekehrt, wenn Entscheidungsbäume verwendet werden. Ein Anwendungsfall wäre das Bearbeiten oder Feintunen und Einfrieren eines maschinell gelernten Baumes.

Optionen für Pruning

Wie bereits beschrieben, kann Pruning mit dem Optionen-Feld konfiguriert werden. In den folgenden Punkten wird erklärt, wie die bereits implementierten Pruningmethoden konfiguriert werden können.

• Pessimistic Post Pruning

Für diese Art des Prunens kann der Penalty-Multiplikator gegeben werden. Dieser muss ein positiver Wert sein (Gleitkomma). Der Standardwert kann in der *EPworX*-Konfiguration geändert werden.

- **Reduced Error Pruning**

Hier kann konfiguriert werden, wie viel Prozent der Trainingsmenge tatsächlich zum Lernen verwendet werden sollen. Dieser Wert muss ein ganze Zahl kleiner oder gleich 100 sein. Der andere Teil der Trainingsmenge wird für das Pruning genutzt. Auch hier kann der Standardwert in in der *EPworX* -Konfiguration geändert werden.

- **Minimal Error Pruning**

Dieser Art des Prunings kann der Typ der Schätzung als Argument mitgegeben werden. Unterschiedliche Werte erfordern hier unterschiedliche Argumente.

Klassifikatorzuweisungen im Prozessablauf verwenden

Klassifikatorzuweisungen können auch vollautomatisch im Prozessablauf integriert werden, entweder als Systemschritt oder als Preprocessing- oder Postcondition-Methode eines Tasks. Dazu bietet das *EPworX* -API die Klasse `com.groiss.ml.classifier.ClassificationAction` an. Eine genau Beschreibung der Methoden dieser Klasse finden Sie in der API-Dokumentation bzw. bei der Methodenauswahl im Systemschritteditor des Prozesseditors.

6.5.14 Report

Für einen oder mehrere Prozesse kann ein Report analog zu einem Applikationsreport erstellt werden (siehe Abschnitt [Report](#)).

6.5.15 Meilensteine

Ein Meilenstein erlaubt die Definition eines vordefinierten Zeitpunktes (fix oder relativ zum Prozessstart bzw. errechnetem Ende des Prozesses), der bei einem Prozessschritt in einem Zeitplan des Prozesses angewendet werden kann. Ein Meilenstein-Objekt besteht aus folgenden Attributen:

- **Id:** Die eindeutige Kurzbezeichnung für den Meilenstein. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).
- **Name:** Der Name des Meilenstein. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#). Der I18n-Link neben diesem Feld ermöglicht direkt die Anpassung der Ressource (falls in Applikationsdetailmaske - Registerkarte *Eigenschaften* festgelegt). Die Änderungen werden nach Aktivieren der Schaltfläche *Speichern* in der Ressource Datei (Strings) dieser Applikation gespeichert (siehe Abschnitt [Ressourcen Editor](#)).
- **Applikation:** Die Applikation, in der das Meilenstein-Objekt gespeichert ist.

- **Intervall:** Hier können Sie ein Intervall angeben, dass entweder *Vom Beginn* bzw. *Vom Ende* eines Prozesses berechnet wird. Die Berechnung erfolgt nur bei Prozessinstanzen für jenen Prozess-Schritt, für den im Zeitplan dieses Meilenstein-Objekt zugewiesen wurde. Beim Wert *Vom Beginn* wird das Prozessstart-Datum herangezogen, bei *Vom Ende* wird die errechnete Prozessfälligkeit verwendet. Subprozesse stellen dabei einen Sonderfall dar: Beim Wert *Vom Beginn* wird das errechnete Prozessstart-Datum des Subprozesses aufgrund des Prozessplans herangezogen sowie bei *Vom Ende* die errechnete Prozessfälligkeit des Subprozesses. Wird ein ganzzahliger Wert angegeben, so wird dieser mit der Einheit s für Sekunden abgespeichert. Es besteht jedoch auch die Möglichkeit die Einheiten für den Wert selbst festzulegen wie z.B. 3d 4h 30m (= 3 Tage, 4 Stunden und 30 Minuten).
- **Berechnungsmodell:** Das Berechnungsmodell bezieht sich nur auf den Wert aus dem Feld *Intervall*:
 - **Arbeitszeit:** Die Berechnung der Arbeitszeit erfolgt aufgrund der Einstellungen in *Konfiguration/Kalender*. Hier können Sie *Arbeitszeit-Beginn*, *Arbeitszeit-Ende*, die *Arbeitszeit pro Tag* und *Arbeitsfreie Tage* festlegen.
Erklärung anhand eines vereinfachten Beispiels: Als Arbeitszeit wurde 09:00 bis 17:00 definiert mit 8h Arbeitszeit pro Tag. Für den ersten Schritt im Prozess wurde im Zeitplan ein Meilenstein definiert mit dem Intervall 1h vom Beginn. Der Prozess wurde um 17:00 gestartet - somit ist das Fälligkeitsdatum für diesen Schritt 10:00 des nächsten nicht-arbeitsfreien Tages.
 - **Kalender-Zeit:** Als Berechnungsgrundlage wird in diesem Fall der 24 Stunden Tag angewendet.
Erklärung anhand eines vereinfachten Beispiels: Als Arbeitszeit wurde 09:00 bis 17:00 definiert mit 8h Arbeitszeit pro Tag. Für den ersten Schritt im Prozess wurde im Zeitplan ein Meilenstein definiert mit dem Intervall 1h vom Beginn. Der Prozess wurde um 17:00 gestartet - somit ist das Fälligkeitsdatum für diesen Schritt 18:00.
- **Formularfeld:** Definition eines Formularfelds vom Typ *date* bzw. *dateTime*, das im Gegensatz um Intervall als fixes Datum für diesen Meilenstein herangezogen wird. Die Definition erfolgt in folgender Form: <procform_id>.<formfield>

Hinweis: Zeitpläne für Prozesse können im Kontext des Prozesseditors erstellt werden (siehe dazu Abschnitt [Prozesspläne](#))!

6.5.16 Plantypen

Plantypen werden dazu verwendet, um Zeitpläne für Prozesse zu typisieren. Ein Plantyp-Objekt besteht aus folgenden Attributen:

- **Id:** Die eindeutige Kurzbezeichnung für den Plantyp. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).

- **Name:** Der Name des Plantyps. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#). Der I18n-Link neben diesem Feld ermöglicht direkt die Anpassung der Ressource (falls in Applikationsdetailmaske - Registerkarte *Eigenschaften* festgelegt). Die Änderungen werden nach Aktivieren der Schaltfläche *Speichern* in der Ressource Datei (Strings) dieser Applikation gespeichert (siehe Abschnitt [Ressourcen Editor](#)).
- **Position:** Die Position in der Auswahlliste der Plantypen beim Erzeugen eines Zeitplans.
- **Applikation:** Die Applikation, in der das Plantyp-Objekt gespeichert ist.

Hinweis: Zeitpläne für Prozesse können im Kontext des Prozesseditors erstellt werden (siehe dazu Abschnitt [Prozesspläne](#))!

6.6 Funktionsgruppen

Funktionsgruppen dienen dazu, um (Task-)Funktionen oder Reports zu gruppieren. Sie können nur dann gelöscht werden, wenn sie keiner *Task-Funktion* bzw. einem *Report* zugewiesen wurden.

The screenshot shows a web-based form for editing a 'Funktionsgruppe' (Function Group). The 'Allgemein' (General) tab is active. The form includes the following fields and controls:

- Id: ***: A text input field containing 'offerqueries'.
- Name: ***: A text input field containing 'offerqueries'. To its right is a blue link labeled 'I18n: offerqueries'.
- Sortierattribut:**: An empty text input field.
- Applikation: ***: A dropdown menu with 'CRM' selected.
- Als Untermenü anzeigen:**: An unchecked checkbox.

At the bottom of the form, there are four buttons: 'Löschen' (Delete), 'Speichern und Schließen' (Save and Close), 'Speichern' (Save), and 'Abbrechen' (Cancel).

Abbildung 6.30: Funktionsgruppe

Eine Funktionsgruppe besteht aus folgenden Elementen (Tab *Allgemein*):

- **Id:** Die eindeutige Kurzbezeichnung der Funktionsgruppe. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).
- **Name:** Der Name der Funktionsgruppe. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#). Der I18n-Link neben diesem Feld ermöglicht direkt die Anpassung der Ressource (falls in Applikationsdetailmaske -

Registerkarte *Eigenschaften* festgelegt). Die Änderungen werden nach Aktivieren der Schaltfläche *Speichern* in der Ressource Datei (Strings) dieser Applikation gespeichert (siehe Abschnitt [Ressourcen Editor](#)).

- **Sortierattribut:** Hier können Sie ein beliebiges Sortierattribut angeben, das zur Sortierung mit anderen Funktionsgruppen herangezogen wird. Hat Funktionsgruppe F1 das Sortierattribut B und Funktionsgruppe F2 das Sortierattribut A, dann wird zuerst die Funktionsgruppe F2 angezeigt und darunter F1 (z.B. im Smartclient bei den globalen Funktionen).
- **Applikation:** Die Applikation, in der die Funktionsgruppe gespeichert ist.
- **Als Untermenü anzeigen:** Ist diese Checkbox aktiviert, werden alle Funktionen, die dieser Funktionsgruppe zugeordnet sind, in einem Untermenü angezeigt. Ist sie nicht aktiviert, wird nur eine Trennlinie zwischen den Funktionen angezeigt, aber der Name der Funktionsgruppe scheint nicht auf. Diese Checkbox wirkt sich nur im Smartclient bei den globalen und Task-Funktionen aus!

6.7 GUI-Konfiguration

Die GUI Konfiguration erlaubt es, den Navigationsbereich (Arbeitskorb, DMS, etc.) einfach anzupassen und die entsprechenden Berechtigungen über den Reiter *Zuordnungen* an Benutzer bzw. Rollen zu vergeben.

Hinweis: Die Beschreibung der XML-Struktur ist im [Programmierhandbuch](#) im Kapitel [The Elements of the Configuration File](#) zu finden.

6.7.1 Registerkarte: Allgemein

Die Definition der GUI-Konfiguration erfolgt in der Registerkarte *Allgemein* (siehe Abbildung [6.31](#)). *Id*, *Name* und *Applikation* sind Pflichtfelder, eine Beschreibung kann optional hinzugefügt werden. Die weitere Definition erfolgt entweder als XML-Baum oder durch Angabe einer Methode in das URL Feld. Bei Angabe einer solchen (Servlet-)Methode wird bei Auswahl dieser GUI-Konfiguration ein Redirect auf diese Methode ausgeführt.

Wird ein Baum definiert (der Default-Fall), wird dieser in einem XML-File im *classes* Verzeichnis der Applikation abgelegt. Der Dateiname entspricht der Id mit der Endung *.xml*.

Über den Toolbar können Sie nun folgende Aktionen zum Erstellen eines XML-Baumes durchführen:

- **Neu:** Hinzufügen von neuen Knoten wie z.B. Arbeitskorb, DMS, Funktionen, etc. (siehe Abschnitt [Knoteneigenschaften](#)).
- **Bearbeiten:** Bearbeiten von bestehenden Knoten; gleiche Funktionalität durch Doppelklick auf das Element.

- **Löschen:** Löschen von bestehenden Knoten, Subknoten werden dadurch mitgelöscht.
- **Ausschneiden und Einfügen:** Mit Hilfe dieser beiden Funktionen können Sie einen bestehenden Baumknoten ausschneiden und an einer anderen Stelle im Baum wieder einfügen.
- **Nach oben verschieben:** Verschiebt den ausgewählten Knoten um eine Stelle nach oben in der gleichen Ebene.
- **Nach unten verschieben:** Verschiebt den ausgewählten Knoten um eine Stelle nach unten in der gleichen Ebene.
- **Eigenschaften:** In diesem Fenster können diverse Einstellungen zur Darstellung des Navigationsbereiches getätigt werden (siehe Abschnitt [Eigenschaften](#)).
- **Funktionen:** Dieser Bereich erlaubt das Erstellen von Funktionen, die nicht im Baum angezeigt werden sollen. Dies können Aktionen, Objektauswahlen, Tabellen und Zusatzdaten sein (siehe Abschnitt [Funktionen](#)).
- **Vorschau:** Mit Hilfe dieser Funktion wird eine Vorschau der aktuell zusammengestellten GUI-Konfiguration angezeigt, wie sie auch die Benutzer sehen würden.

Knoteneigenschaften

Durch Auswählen der Funktion *Neu* bzw. *Bearbeiten* öffnet sich das HTML-Fenster *Knoteneigenschaften*, in dem neue Knoten hinzugefügt bzw. bestehende bearbeitet werden können (siehe Abbildung [6.32](#)). Knoten bestehen immer aus einer *Id* (siehe unten) und einem *Namen*, wobei der Name übersetzt angezeigt werden kann oder auch nicht.

Fixe Elemente eines Knotens sind somit:

- **Id:** Eindeutige Identifizierung des Knoten. Wenn keine Id angegeben wurde, vergibt sie *EPworX* automatisch.
- **Name:** Name, der angezeigt wird. Ist in den meisten Fällen ein Mussfeld.
- **Lokalisieren:** Durch Aktivieren dieser Checkbox wird der eingetragene Name übersetzt (ohne @), sofern der String im Default- bzw. in einem angegebene Resource-Bundle vorhanden ist. Der I18n-Link neben diesem Feld ermöglicht direkt die Anpassung der Ressource (falls in Applikationsdetailmaske - Registerkarte *Eigenschaften* festgelegt). Die Änderungen werden nach Aktivieren der Schaltfläche *Speichern* in der Ressource Datei (Strings) dieser Applikation gespeichert (siehe Abschnitt [Ressourcen Editor](#)).
- **Referenz zu:** Hier können Sie eine Referenz auf einen anderen Knoten in der selben oder einer anderen GUI-Konfiguration definieren. Dazu geben Sie die ID der GUI-Konfiguration mit dem Knoten, der referenziert werden soll in folgender Form an: `<xmlid>.<knotenid>`
Bei Angabe einer Referenz werden alle Einstellungen des Referenzknoten zur

6. Workflowelemente

The screenshot shows the 'Allgemein' tab of a software interface. It features several input fields: 'Id' (value: itsm), 'Name' (value: ITSM), 'Beschreibung' (empty), 'Applikation' (dropdown: ITSM), 'Typ' (radio buttons: Dojo, React, Mobil), 'Datei' (text: jar:file:/C:/Users/nlugic/Desktop/Test Installationen/@ep installations/ep12/ep12_docu/local/appls/itsm/lib/itsm-12.0-SNAPSHOT.jar!/itsm.xml), and 'Url' (empty). Below these are four tabs: 'Anzeigen' (selected), 'Eigenschaften', 'Funktionen', and 'Vorschau'. Under the 'Tree' section, a hierarchical tree view is displayed, starting with '<Root>' and branching into 'Aufgaben', 'Tests', and 'CMDB'. 'CMDB' further branches into 'CI Klassen', 'CI Typen', 'CI's', '<Empty>', 'CI Relationstypen', 'Orte', 'Wertelisten', 'Dokumente', and 'Kalender'. At the bottom of the form are four buttons: 'Löschen', 'Speichern und Schließen', 'Speichern', and 'Abbrechen'.

Abbildung 6.31: Registerkarte: Allgemein



Laufzeit geladen, jedoch können Sie mit den Attributen aus dem aktuellen Knoten diese überschreiben. Ganz rechts außen finden Sie das Anzeige-Icon, mit dem Sie die Attribute des Referenzknoten in schreibgeschützter Ansicht ansehen können.

- Inklusive Kind-Knoten: Dieses Attribut ist Teil der Referenz auf einen anderen Knoten und dient dazu, dass alle Unterknoten des Referenzknoten geladen werden.
- Zusammenklappbar: Ist diese Checkbox aktiviert, wird der Knoten und dessen Subknoten als Baum dargestellt und erlaubt das auf-/zuklappen des Baums, ansonsten werden alle Knoten geöffnet angezeigt.
- Default: Wenn diese Option aktiviert ist, dann ist der jeweilige Knoten die Startseite, die nach dem Login im Hauptfenster geladen wird (sollte nur 1x vergeben werden!). Diese Option wird nicht auf Knoten *Strukturierter Arbeitskorb* ausgewertet!
- Sichtbar über Rollen: Mit Hilfe der Auswahlliste *Sichtbar über Rollen* können Rollen festgelegt werden. Benutzer, die diese Rollen besitzen, haben damit Zugriff

Knoteneigenschaften

Label

Node

Arbeitskorb

Strukturierter Arbeitsk

DMS

Tabelle

Prozess starten

Funktionsliste

Funktion

Report

Benutzerdefiniert

XML-Fragment

Id:

Name: *

☐ Lokalisieren

Referenz zu:

Zusammenklappbar: ☒

Sichtbar über Rollen:

Sichtbar über Rechte:

Parameter:

Hilfekontext:

Bei Anzeigen neu laden: ☐

onClick:

Widget:

Komponentenschlüssel:

Iconpfad:

Inklusive Kind-Knoten: ☐

Default: ☐

Version:

Klassenname: *

Administration: ☐

Spalten:

Name

Beschreibung

CI Klasse

CI Typ

Spaltenauswahl: **Spaltenfilter verwenden:**

Abbildung 6.32: Knoteneigenschaften

auf diesen Knoten, d.h. der Knoten ist in dem Navigationsbereich sicht- und/oder auswählbar. Die Sichtbarkeit des Knotens kann man zusätzlich auf eine Rolle in einer bestimmten OE einschränken.

- **Sichtbar über Rechte:** Mit Hilfe der Auswahlliste *Sichtbar über Rechte* können Rechte festgelegt werden. Benutzer, die diese Rechte besitzen, haben damit Zugriff auf diesen Knoten, d.h. der Knoten ist in dem Navigationsbereich sicht-

und/oder auswählbar. Es können mehrere Rollen und mehrere Rechte gleichzeitig ausgewählt werden. Für den Fall, dass sowohl (*Sichtbar über Rollen* als auch *Sichtbar über Rechte*) angegeben sind, muss der Benutzer mindestens **ein Recht und eine Rolle** zugewiesen haben, damit der Knoten sichtbar ist.

- **Parameter:** Hier können eigene Parameter (*<params>*) definiert werden, die beim Absetzen des Server-Requests mitgeschickt werden (= beim Anzeigen der entsprechenden Knotendetails).
- **Hilfecontext:** Hinzufügen von kontextsensitiver Hilfe. Der Kontext besteht aus der Anwendungs-ID und dem auf der Hilfeseite definierten Anker. Für mehr Informationen bitte im [Application Development Guide](#), Kapitel [Using context sensitive help in applications](#) nachschlagen.
- **Bei Anzeigen neu laden:** Ist dies aktiviert, wird der Knoten bei jedem Aufruf neu geladen.
- **onClick:** Hier können Sie eine onClick-Aktion angeben, die ausgeführt und im Arbeitsbereich angezeigt wird.
- **Widget:** Hier können Sie ein Widget angeben, das im Navigationsbereich angezeigt werden soll.
- **Komponentenschlüssel:** Definieren Sie hier den Komponentenschlüssel des Knotens. Dieser Komponentenschlüssel wird zum Identifizieren der GUI-Komponente verwendet.
- **Iconpfad:** Der Iconpfad für diesen Knoten. Wenn kein Icon definiert ist, wird ein Standard-Icon verwendet.

Folgende Knotentypen stehen zur Auswahl:

- **Label:** Frei wählbarer Text, der im Navigationsbaum angezeigt wird; kann z.B. für Gruppenteilung verwendet werden.
- **Node:** Definition eines Links im Navigationsbaum. Dabei können folgende Attribute festgelegt werden:
 - **Standard-Knoten:** Durch Aktivieren dieser Schaltfläche wird eine Liste mit Standard-Knoten angezeigt. Wird ein Knoten ausgewählt, werden die entsprechenden Einstellungen übernommen.
 - **Zielfenster:** Der Inhalt dieses Feldes bestimmt, wo diese Ausgabe angezeigt wird (z.B. *_top*). Wird in dieses Feld nichts eingetragen, wird die Ausgabe im rechten Frame bzw. im Arbeitsbereich angezeigt.
 - **URI:** Dieses Feld bietet die Möglichkeit händisch eine URL einzutragen. Diese URL wird standardmäßig verwendet, falls keine onClick-Aktion angegeben wurde.
 - **Aktualisierungs-Link:** Durch die Aktivierung dieser Checkbox wird im benutzerdefinierten Knoten ein Aktualisierungs Icon angezeigt.

- **Arbeitskorb:** Hier kann der Arbeitskorb festgelegt werden. Dabei können folgende Attribute gesetzt werden:
 - **Schnellzugriff:** Hier können Sie eine Tastenkombination angeben, um den Arbeitskorb mit dieser anzusteuern (Shortcut). Eine Liste der Tasten ist [here](#) zu finden. Beispieldefinition: CTRL+SHIFT+A.
 - **Applikation:** Mit dieser Auswahl können Sie festlegen, welche Prozesse der ausgewählten Applikation im Arbeitskorb angezeigt werden sollen. Ist dieser Parameter nicht gesetzt, werden alle Prozesse aller Applikationen angezeigt.
 - **Typ:** Festlegung des Arbeitskorbtyps (z.B. Arbeitskorb, Rollen-Arbeitskorb, Wiedervorlage, Rollen-Wiedervorlage, etc.). Dabei können mehrere Typen gleichzeitig für einen Knoten ausgewählt werden. Nähere Informationen zu den Typen finden Sie im [Programmierhandbuch](#) (Tabelle *Worklist types*).
 - **Spalten:** Setzen der Spalten, die in der Tabelle angezeigt werden sollen. Dabei können Spalten hinzugefügt, bearbeitet, gelöscht und in ihrer Reihenfolge verändert werden. Spalten bestehende aus folgenden Elementen:
 - * **Id:** Die ID einer Spalte, wobei eine vordefinierte ID ausgewählt werden kann. Bei Auswahl einer vordefinierten ID, werden die Felder *Name* und *Lokalisieren* vorausgefüllt. Dieses Feld ist ein Pflichtfeld. Bei der Id kann weiters eine Option angegeben werden, welcher Reiter beim Aktivieren des Links im Arbeitskorb geöffnet werden soll, z.B. *process-form0* - es wird die Spalte Prozess angezeigt, jedoch mit dem Link auf das 1. Formular in der Reiteransicht anstatt dem Standardlink auf die Prozesshistorie.
 - * **Name:** Frei wählbarer Name, der im Header der Tabellenspalte angezeigt wird.
 - * **Lokalisieren:** Durch Aktivieren dieser Checkbox wird der eingetragene Name übersetzt (ohne @), sofern der String im Default- bzw. in einem angegebene Ressource-Bundle vorhanden ist. Der I18n-Link neben diesem Feld ermöglicht direkt die Anpassung der Ressource (falls in Applikationsdetailmaske - Registerkarte *Eigenschaften* festgelegt). Die Änderungen werden nach Aktivieren der Schaltfläche *Speichern* in der Ressource Datei (Strings) dieser Applikation gespeichert (siehe Abschnitt [Ressourcen Editor](#)).
 - * **Icon:** Hier können Sie einen Pfad zu einem Icon definieren, das anstelle des Namen angezeigt werden soll.
 - * **Colspan, RowSpan und Zeile:** Mit diesen Attributen können Sie festlegen, wie die Spalte in der Tabelle angezeigt werden soll, z.B. in einer zweiten Zeile über mehrere Spalten.
 - * **Sichtbar:** Ist diese Checkbox aktiviert, wird die Spalte beim erstem Aufruf des Arbeitskorbs angezeigt, andernfalls ist sie über den Columnpicker (Spaltenauswahl) hinzufügbare.
 - * **Filterbar:** Mit dieser Checkbox geben Sie an, ob die Spalte für den Filtermechanismus verwendet werden kann.

- * Nicht ausblendbar: Diese Checkbox gibt an, ob die Spalte über den Columnpicker (Spaltenauswahl) ausgeblendet werden kann.
 - * Sortierbar: Mit Hilfe dieser Checkbox können Sie festlegen, ob nach dieser Spalte sortiert werden kann.
 - * Gruppierbar: Wenn aktiviert, kann die Spalte nach ihren übereinstimmenden Werten gruppiert werden.
 - * Typ: Hier können Sie einen Spaltentyp definieren, indem Sie einen der folgenden Werte angeben: string, date, dateTime, number (für Zahlen ohne Komma) oder decimal (für Zahlen mit Komma + Darstellung in entsprechender Form gemäß Decimal-Format Konfiguration).
 - * Formularfelder: Mit diesem Feld können Sie definieren, welche Feldwerte eines Formulars für diese Spalte herangezogen werden können. Die Syntax dazu finden Sie im [Programmierhandbuch](#) im Kapitel [Configuring the Worklist Client](#). Wenn in der Arbeitskorb-Tabelle eine Prozessinstanz vorhanden ist, wo diese Syntax keinen Treffer liefert, dann bleibt die Spalte leer.
Hinweis: Es ist nur 1 Formularfeld des gleichen Formulars pro Prozessdefinition/version möglich, d.h.
 - *myproc:1:myf:field1;myproc:1:myf:field2* ist nicht möglich, da field1 und field2 von der selben Prozessdefinition/version gelesen werden,
 - *myproc:1:myf:field1;mypproc2:1:myf:field2* ist möglich, weil field1 und field2 von verschiedenen Prozessdefinitionen/versionen gelesen werden.
 - * Javascript Klasse: Durch Angabe eines Pfades auf eine JS-Klasse (Widget) können Sie die Darstellung eines Spaltenwertes beeinflussen.
 - * Komponentenschlüssel: Definieren Sie hier den Komponentenschlüssel der Spalte. Dieser Komponentenschlüssel wird zum Identifizieren der GUI-Komponente verwendet.
- **Spaltenauswahl:** Hier können Sie definieren, ob eine Spaltenauswahl (Columnpicker) angeboten werden soll. Wenn kein Wert ausgewählt wurde, wird die Standardeinstellung angewendet.
 - **Spaltenfilter verwenden:** Hier können Sie definieren, ob der Filtermechanismus für Tabellen verwendet werden soll. Wenn kein Wert ausgewählt wurde, wird die Standardeinstellung angewendet.
 - **Auswahl:** Definition des Auswahl-Typs für die Arbeitskorbtabelle. Es können folgende Einträge ausgewählt werden:
 - * HIDDEN: Keine Checkboxes werden zur Auswahl angeboten, d.h. ein Eintrag aus der Tabelle kann nicht selektiert werden
 - * ONE: Radiobuttons werden zur Auswahl angeboten anstelle von Checkboxes
 - * MULTI: Checkboxes werden zur Auswahl angeboten
 - * ROWONE: Eine Zeile kann selektiert werden (ohne Checkboxes oder Radiobuttons)

- * ROWMULTI: Mehrere Zeilen können ausgewählt werden (ohne Check-boxen oder Radiobuttons)
- **Druckbar:** Mit dieser Checkbox können Sie festlegen, ob oberhalb der Tabelle die Funktion zum Drucken erscheinen soll oder nicht. Diese Checkbox wirkt sich nur im Smartclient aus!
- **Zeige Inline-Details bei:** Hier können Sie definieren wie die Inline-Details bei einem Prozess angeboten werden. Es können folgende Einträge ausgewählt werden:
 - * COLUMN:ID: Die Inline-Details werden durch einen Klick auf den Pfeil neben der Prozess-ID angezeigt
 - * ROW: Die Inline-Details werden durch einen Klick auf die Zeile angezeigt
 - * OFF: Die Inline-Details werden nicht angezeigt
- **DnD-Handle:** Hier können Sie das Drag and Drop (DnD) Verhalten für die Arbeitskorb-Einträge definieren. Es können folgende Optionen ausgewählt werden:
 - * HANDLE: DnD funktioniert nur mittels dem Handle-Icon, das ganz links bei einem Arbeitskorb-Eintrag angezeigt wird
 - * ROW: DnD ist für die ganze Zeile eingeschaltet
 - * OFF: DnD ist deaktiviert
- **Funktionen:** Hier können entweder vorhandene *Funktionen* hinzugefügt oder *Aktions-Ids* gesetzt werden (z.B. new, cut, copy, myconfig.myaction, etc). Aktions-Ids werden als Aktionen unter *Funktionen* (siehe Abschnitt [Funktionen](#)) definiert.

Mit der Checkbox *Primär-Aktion* können Sie festlegen, welche Funktion als primäre Aktion hervorgehoben werden soll. Primär-Aktionen werden, in der Liste, mit dicker Schrift dargestellt. Dieses Konfiguration wird nur am React-client berücksichtigt.

Mit der Checkbox *In Aufklappmenü* können Sie festlegen, ob die Funktion über das Zahnradsymbol in der Tabelle aufgerufen werden kann.

Weiters gibt es die Möglichkeit Funktionen aus der Liste zu löschen und deren Reihenfolge zu verändern.
- **Toolbar-Darstellung:** Hier können Sie definieren, wie Funktionen im Toolbar dargestellt werden sollen. Dabei gibt es folgende Möglichkeiten:
 - * TEXT: Es wird nur der Name der Funktion angezeigt
 - * ICON: Es wird nur das Icon der Funktion angezeigt
 - * BOTH: Es wird das Icon und der Name der Funktion angezeigt (nur in Smartclient anwendbar)
- **Toolbar-Länge:** Hier können Sie die Anzahl der Funktionen einstellen, die initial im Toolbar angezeigt werden sollen. Um alle Funktionen anzuzeigen, geben sie den Wert 0 ein.
- **Ordnerfunktionen:** Hier können Sie *Aktions-Ids* hinzufügen, die als Funktionen in einem Dropdown-Menü neben dem Worklist-Knoten angezeigt wer-

den. Ein bekanntes Beispiel dafür ist die Erzeugung/Verwaltung von Benutzerordner eines Arbeitskorbs (Aktions-Id: `newUserFolder`, `deleteUserFolder`, `editUserFolder`, etc.), wobei diese Aktions-Ids mit Hilfe der Schaltfläche *Standard übernehmen* geladen werden können.

- **Funktionen für Prozessdetails:** Analog zu *Funktionen*, nur hier definiert man die Funktionen, die in der Prozessdetail-Ansicht angezeigt werden.
 - **Defaultsortierung:** Hier kann eine Spalte und die Sortierrichtung angegeben werden. Die Sortierrichtung wird mit + (aufsteigend) oder - (absteigend) angegeben, plus der Spalten-Id als Wert. Ist ein (Sortier-)Attribut nicht vorhanden, wird die erste (oder angegebene Spalte) standardmäßig aufsteigend sortiert. Beispiel: `+name` bedeutet, dass aufsteigend nach der Spalte mit Id "name" sortiert wird.
 - **Defaultgruppierung:** Mit diesem Parameter kann eine Spalte definiert werden, nach welcher standardmäßig die Einträge gruppiert werden. Als Wert sollte man die Spalten-ID und optional die absteigende Sortierrichtung eingeben. Ist neben der Spalten-ID keine Richtung angegeben, wird von Haus aus aufsteigend sortiert.
 - **Tabellen-Handler:** Tragen Sie hier die Klasse Ihres Worklist-Adapters ein. Nähere Informationen zu Worklist-Adaptoren finden Sie im [Programmierhandbuch](#), Kapitel [Customizing the Worklist](#).
 - **Filter-Id:** Hier können Sie eine Id eines anderen Arbeitskorbes angeben. Alle gespeicherten Filter des referenzierten Arbeitskorbes können dann im aktuellen Arbeitskorb verwendet werden.
 - **Ohne Dokumente/Notizen:** Durch Aktivieren dieser Checkbox wird verhindert, dass Dokumente und Notizen der Prozesse beim Anzeigen des Arbeitskorbes bereits gelesen werden. Aktivieren Sie diese Checkbox nur, wenn keine Dokumente oder Notizen bei Prozesse verwendet werden und die Anzeige der Arbeitskorbtabelle optimiert werden soll!
 - **Keine Benutzerordner Filterung:** Durch Aktivieren dieser Checkbox wird verhindert, dass Einträge in Benutzerordner beim Anzeigen des Arbeitskorbes herausgefiltert werden. Aktivieren Sie diese Checkbox nur, wenn keine Benutzerordner verwendet werden und die Anzeige der Arbeitskorbtabelle optimiert werden soll!
- **Strukturierter Arbeitskorb:** Hier können Sie entweder einen Arbeitskorb/Benutzerordner (Typ *user*) oder einen Vertreter-Ordner (Typ *user substitutes*) definieren, der als Platzhalter dient, mit einer Möglichkeit eine Applikation festzulegen. Die restlichen Felder sind analog zum Knoten *Arbeitskorb*. Beim Typ *user substitutes* gibt es noch zusätzlich die Auswahl *Struktur* mit folgenden Werten:
 - **perFolder:** Nur die/alle Benutzerordner-Bäume der vertretenden Personen werden angezeigt, ohne den Top Level Knoten (= Arbeitskorb selbst). Für jede Person wird ein Benutzerordner-Baum angezeigt.
 - **perUserAndFolder:** Für jede vertretende Person wird ein Baum mit den Arbeitskorb-Knoten (Arbeitskorb + Benutzerordner) angezeigt.

- perUser: Nur der Arbeitskorb - ohne Benutzerordner - der vertretenden Personen wird angezeigt. Für jede Person wird ein Arbeitskorb-Link im Baum angeboten.
- DMS: Dieser Knoten ermöglicht die Erstellung bzw. Anpassung eines DMS-Ordners. Dabei können folgende Attribute angepasst werden:
 - **Spalten:** Analog zu Knotentyp *Arbeitskorb*
 - **Spaltenauswahl:** Analog zu Knotentyp *Arbeitskorb*
 - **Spaltenfilter verwenden:** Analog zu Knotentyp *Arbeitskorb*
 - **Auswahl:** Analog zu Knotentyp *Arbeitskorb*
 - **Druckbar:** Analog zu Knotentyp *Arbeitskorb*
 - **Defaultansicht:** Hier können Sie die Defaultansicht in DMS-Tabellen definieren. Diese beinhaltet folgende Ansichten:
 - * Tabelle
 - * Tabelle mit kleinen Symbolen
 - * Tabelle mit großen Symbolen
 - * Kacheln mit kleinen Symbolen
 - * Kacheln mit großen Symbolen

Mehr Informationen finden Sie im [Benutzerhandbuch](#), Kapitel [Elemente des Dokumentenmanagements](#).

 - **DnD-Handle:** Analog zu Knotentyp *Arbeitskorb*
 - **Funktionen:** Analog zu Knotentyp *Arbeitskorb*
 - **Toolbar-Darstellung:** Analog zu Knotentyp *Arbeitskorb*
 - **Toolbar-Länge:** Analog zu Knotentyp *Arbeitskorb*
 - **Formulare:** In dieser Liste können Sie festlegen, welche Formulartypen für diesen Ordner erlaubt bzw. nicht erlaubt sind. Wenn z.B. die Liste leer und der Radio-Button *Nicht erlaubt* aktiviert ist, sind alle verfügbaren Formulartypen für den Benutzer in der DMS-Tabelle auswählbar. Die Auswahl des gewünschten Radio-Buttons wird immer auf alle Formulare in der Selectlist angewendet.

Dieser Knoten kann, wie auch der Knoten *Benutzerordner*, nur 1x im Navigationsbaum vergeben werden!

- Tabelle: Mit Hilfe dieses Knotens kann ein Link auf eine Tabelle erzeugt werden. Diese Tabelle kann Formulklassen (auswählbar über Icon) anzeigen. Dabei können folgende Attribute angepasst werden:
 - **Version:** Hier können Sie wählen, ob Version 1 oder 2 verwendet werden soll. Version 2 ist standardmäßig aktiviert und erzeugt Tabellen, wie sie im Smartclient (z.B. im Arbeitskorb) verwendet werden. Es können folgende Parameter definiert werden:

- * **Klassenname:** Hier muss eine Formularklasse eingegeben bzw. ausgewählt werden. Mit Hilfe des Lupen-Symbols können Sie aus einer Liste von Formularklassen eine Klassennamen auswählen.
- * **Administration:** Durch Aktivieren dieser Checkbox wird diese Tabelle nur in der *EPworX* Administration verwendet.
- * **Spalten:** Analog zu Knotentyp *Arbeitskorb*
- * **Spaltenauswahl:** Analog zu Knotentyp *Arbeitskorb*
- * **Spaltenfilter verwenden:** Analog zu Knotentyp *Arbeitskorb*
- * **Auswahl:** Analog zu Knotentyp *Arbeitskorb*
- * **Druckbar:** Analog zu Knotentyp *Arbeitskorb*
- * **Funktionen:** Analog zu Knotentyp *Arbeitskorb*
- * **Toolbar-Darstellung:** Analog zu Knotentyp *Arbeitskorb*
- * **Toolbar-Länge:** Analog zu Knotentyp *Arbeitskorb*
- * **Detailfenster-Eigenschaften:** Setzen der Fenster-Eigenschaften wie z.B. width, height, scrollbars etc. getrennt durch Semikolons wie bei der Java-Script-Methode *window.open*.
- * **Sortierbar:** Durch Aktivieren dieser Checkbox wird die Tabelle sortierbar.
- * **Gruppierbar:** Wenn aktiviert, kann die Spalte nach ihren übereinstimmenden Werten gruppiert werden.
- * **Defaultsortierung:** Analog zu Knotentyp *Arbeitskorb*
- * **Defaultgruppierung:** Analog zu Knotentyp *Arbeitskorb*
- * **Registerkarten:** Hier können Sie definieren welche Tabs in der Detailansicht des ausgewählten Objekts angezeigt werden sollen. Die einzelnen Tabs werden mittels einer durch Beistrich getrennten Liste von Knoten im Format `<gui_id>.<nodeid>` festgelegt.
- * **Tabellen-Handler:** Tragen Sie hier die Klasse Ihres Tabellen-Handlers ein. Dieser muss *com.groiss.storegui.ObjectTableAdapter* erweitern.
- * **Formular-Handler:** Hier können Sie einen selbst-geschriebenen Formular-Handler angeben, der *com.groiss.storegui.ObjectFormAdapter* erweitert. Ein Beispiel für einen Formular-Handler finden Sie im [Programmierhandbuch](#).
- * **Detail:** Hier können Sie festlegen, ob die Ansicht in einer Reiter-Ansicht angezeigt werden soll. Dazu geben Sie *com.groiss.storegui.TabbedWindow.showDialog* an.
- * **Tabellen-Implementierung:** Hier können Sie eine eigene Tablemodel Implementierung eintragen. Standard ist *com.groiss.storegui.FormTable*.
- * **Fremdschlüssel:** Hier kann ein Fremdschlüssel definiert werden, welcher die Inhalte einer Tabelle eingrenzt z.B. *userid* um nur Objekte eines bestimmten Users anzuzeigen.
- * **Parametertypen:** Im Feld *Bedingung* können Sie auch Platzhalter (?) definieren, die dynamisch befüllt werden. Jeder Platzhalter benötigt einen Typ welcher in diesem Feld in einer durch Beistrich getrennten Liste angegeben werden kann. Folgende Typen können eingetragen werden:

- Persistent
- Date
- Long
- Double
- Integer
- String
- OIDList

Nähere Informationen zur Anwendung finden Sie im [Programmierhandbuch](#), Abschnitt [Object selection - ep/widget/ObjectSelect](#).

- * **Bedingung:** Dieses Attribut kann dazu genutzt werden, um eine SQL-WHERE Bedingung für die Auswahl zu definieren. Dazu müssen Sie nur den Bedingungsstring ohne WHERE angeben, z.B. *application = (select oid from avw_application where id = 'itsm')*.
- * **Attribute:** Abhängige Objekte durch effiziente Operationen abrufen. Der Inhalt ist eine durch Komma getrennte Liste von Java-Feldern der entsprechenden Java-Klasse. Die Feldnamen müssen persistente Objekte bezeichnen! Normalerweise wird eine BulkQuery pro Feld anstelle einer (einzelnen Datensatz-) Select-Anweisung pro Datensatz und Feld ausgeführt.
- * **Änderungen erlauben:** Wenn diese Eigenschaft aktiviert ist, ist das Einfügen, Aktualisieren und Löschen mit JSONLoader erlaubt.

Version 1 entspricht der Tabelle, die bis einschließlich *EPworX* 8.0 verwendet wurde. Folgende Parameter können definiert werden:

- * **Klassenname:** Analog zu Version 2.
- * **Spalten:** Analog zu Version 2.
- * **Spaltenauswahl:** Analog zu Version 2.
- * **Spaltenfilter verwenden:** Analog zu Version 2.
- * **HTML-Maske:** Angabe einer HTML-Maske, die als Detail-Anzeige für den Tabelleneintrag dient.
- * **Mehrseitige Anzeige:** Durch Aktivieren dieser Checkbox wird der Paging-Mechanismus von *EPworX* für die Tabelle verwendet.
- * **Tabelleneinträge pro Seite:** Individuelle Paging-Größe für die Tabelle. Wenn nicht festgelegt, wird der Benutzerparameter und standardmäßig der Konfigurationsparameter verwendet.
- * **Funktionen:** Analog zu Version 2.
- * **Toolbar-Darstellung:** Analog zu Version 2.
- * **Toolbar-Länge:** Analog zu Version 2.
- * **Toolbar-Element:** Die ID der Symbolleiste (ID des Toolbar-Frames).
- * **Toolbar-Ausrichtung:** Hier können Sie festlegen, ob die Symbolleiste vertikal (*v*) oder horizontal (*h*) angezeigt wird.
- * **Tabellen-Element:** Die ID der Tabelle (ID des Tabellen-Frames).
- * **Detailfenster-Eigenschaften:** Analog zu Version 2.

- * **Keine Suche:** Durch Aktivieren dieser Checkbox wird standardmäßig die gesamte Tabelle angezeigt (z. B. wenn mehr Tabelleneinträge verfügbar sind als angezeigt werden dürfen), und es ist keine Suche möglich.
 - * **Keine Warnungen:** Durch Aktivieren dieser Checkbox wird keine Warnung angezeigt, wenn mehr Tabelleneinträge vorhanden sind, als angezeigt werden dürfen.
 - * **Defaultsortierung:** Analog zu Version 2.
 - * **Registerkarten:** Analog zu Version 2.
 - * **Tabellen-Handler:** Analog zu Version 2.
 - * **Formular-Handler:** Analog zu Version 2.
 - * **Detail:** Analog zu Version 2.
 - * **Tabellen-Implementierung:** Analog zu Version 2.
 - * **Suchattribute:** Für die Suche kann eine durch Komma getrennte Liste von Attributen eingegeben werden.
- **Prozess starten:** Dieser Knoten dient dazu, um eine Liste der zu startenden @ep-Prozesse zu erhalten oder einen Prozess direkt zu starten. Dabei kann zum Beispiel festgelegt werden, wer diese Liste sehen darf und wo sie angezeigt wird (analog zu *Node*). Weiters kann eine *Arbeitskorb-Id* festgelegt werden (z.B. *standard.wl*), die der *worklistId* im XML entspricht. Wenn eine solche Id festgelegt wurde, wird nach dem Starten eines Prozesses der Arbeitskorb mit der zutreffenden Id angezeigt. Das Festlegen einer solchen Id ist nur für das alte GUI möglich! Mit dem Attribut *Mode* können Sie den Startmodus festlegen:
 - **ALL:** Die Liste aller startbaren Prozesse wird angezeigt (Standardverhalten). Bei *Applikationen* können Sie definieren, welche Prozesse aus welcher *Applikation* gestartet werden können. Bitte beachten Sie, dass im Smartclient der Prozess starten Knoten mit dieser Option nicht auf beliebigen Ebenen im GUI-Baum verwendet werden kann - der Knoten wird zwar angezeigt, aber ist nicht ausführbar!
 - **DUEDATE:** Ein Formular wird angezeigt, wo die Fälligkeit und die Organisationseinheit festgelegt werden können. Danach erst wird der Prozess gestartet, wobei die *Id* des Prozesses definiert werden muss.
 - **DIRECT:** Direkter Start eines Prozesses, wobei die *Id* des Prozesses definiert werden muss. Optional kann noch angegeben werden, in welcher *Organisationseinheit* der Prozess gestartet werden soll.
 - **FORM:** Es wird das Prozessformular des definierten Prozesses (= *Id*) zuerst angezeigt und kein Prozess gestartet. Mit der Schaltfläche *Prozess starten* wird erst der Prozess gestartet und die eingegebenen Formulardaten übernommen. Optional kann noch angegeben werden, in welcher *Organisationseinheit* der Prozess gestartet werden soll, sowie im welchen Schritt soll der Prozess starten (es sind nur Schritte mit definierter *Schritt-Id* auswählbar). Diese Optionen sind nur im Smartclient verwendbar!
 - **Funktionsliste:** Mit Hilfe dieses Knoten können alle globalen Funktionen einer oder mehrerer Applikationen angezeigt werden.

- Funktion: Dieser Knoten erlaubt das Hinzufügen einzelner Funktionen (über *Id*) zum Navigationsbaum. Da diese Funktion im Navigationsbaum den Kontext des Hauptfensters nicht kennt, sollten hier globale Funktionen (auf keinen Eintrag anwendbar) bevorzugt werden.
- Report: Durch Hinzufügen dieses Knoten kann der ausgewählte Report (über *Id*) im Navigationsbaum ausgeführt werden (analog zu [Reports](#))
- Benutzerdefiniert:
 - Klasse: Hier kann eine Java Klasse eingetragen werden, die das Java Interface *com.groiss.gui.NavigationTreeNode* implementiert.
 - Aktualisierungs-Link: Durch die Aktivierung dieser Checkbox wird beim benutzerdefinierten Knoten ein Icon angezeigt, um die Knotendaten zu aktualisieren.
- XML-Fragment: Mit Hilfe dieses Knotens kann ein GUI-XML Fragment geladen und im Baum eingefügt werden. Der Dateiname einer XML-Datei (ohne die Dateierweiterung .xml) muss im Feld *Fragment-Name* eingetragen werden. Zur Struktur eines solchen Fragments siehe im [Programmierhandbuch](#) den Abschnitt [XML fragment](#) in [Configuring the Worklist Client](#).

Eigenschaften

Mit Hilfe der GUI-Eigenschaften können spezifische Einstellungen für die GUI-Konfiguration getroffen werden. Folgende Attribute stehen zu Verfügung:

- Titel: Hier kann ein Pattern für die Browser-Titelleiste angegeben werden. Wenn kein Pattern vorhanden ist, wird *@@@ep:productname@@ - {user}* verwendet. Folgende Schlüsselwörter können verwendet werden:
 - *config*: Der Name der GUI-Konfiguration
 - *server*: Servername
 - *user*: Benutzername
 - *context*: Name des angezeigten Inhalts
 - *count*: Bei Arbeitskörben wird die Anzahl der ungelesenen Einträge angezeigt
- Sichtbar über Rollen: Mit Hilfe der Auswahlliste *Sichtbar über Rollen* können Rollen festgelegt werden. Benutzer, die diese Rollen besitzen, haben damit Zugriff auf diese Oberfläche, d.h. die Benutzeroberfläche ist für diese Benutzer sichtbar und verwendbar. Die Sichtbarkeit der Oberfläche kann man zusätzlich auf eine Rolle in einer bestimmten OE einschränken.
- Sichtbar über Rechte: Mit Hilfe der Auswahlliste *Sichtbar über Rechte* können Rechte festgelegt werden. Benutzer, die diese Rechte besitzen, haben damit Zugriff auf diese Oberfläche, d.h. die Benutzeroberfläche ist für diese Benutzer sichtbar.

und verwendbar. Es können mehrere Rollen und mehrere Rechte gleichzeitig ausgewählt werden. Für den Fall, dass sowohl (*Sichtbar über Rollen* als auch *Sichtbar über Rechte*) angegeben sind, muss der Benutzer mindestens **ein Recht und eine Rolle** zugewiesen haben, damit die Oberfläche sichtbar ist.

- Navigations-Typ: Hier kann definiert werden wie das Navigationsmenü in *EPworX* aussehen soll. Folgende Schlüsselwörter können verwendet werden:
 - *COLLAPSIBLE*: Navigationsmenü kann vom Benutzer ein- oder ausgeklappt werden. Beim einklappen wird ein eingeschränktes Menü mit Icons angezeigt.
 - *COLLAPSED*: Navigationsmenü wird als ein eingeschränktes Menü angezeigt. Ein- und ausklappen des Menüs ist nicht möglich.
 - *HIDEABLE*: Navigationsmenü kann vom Benutzer ein- oder ausgeklappt werden. Beim Einklappen wird das Menü versteckt.
 - *HIDDEN*: Navigationsmenü wird nicht angezeigt.

Hinweis: Das Navigationsmenü kann man mit einem Doppelklick auf den Splitter oder mittels klicken des Pfeil-Icons in der Mitte, ein- oder ausklappen.

- HTML-Maske: Angabe einer HTML-Maske, die der Anzeige dient (entspricht Attribut *framepage* in der XML-Datei *standard.xml*).
- HTML-Maske (rechts-nach-links): Analog zu *HTML-Maske* (entspricht Attribut *framepageRTL* in der XML-Datei *standard.xml*).
- Standard-Aktionen: Hier können Sie Aktionen definieren, die immer angezeigt werden. Mit Hilfe der Schaltfläche *Standard übernehmen* können Sie die Standard-Aktionen setzen.
- Aktionen in den Benutzerdetails: Hier können Sie die Funktionen des Benutzerprofils definieren. Mit Hilfe der Schaltfläche *Standard übernehmen* können Sie die Standard-Funktionen setzen.
- Layout widget: Hier können Sie ein Widget angeben, dass die Hauptseite überschreibt, optimalerweise ist dies ein dijit/layout/xx widget (z.B. BorderContainer).
- Layout Komponentenschlüssel: Definieren Sie hier den Komponentenschlüssel für das Layout der Hauptseite. Dieser Komponentenschlüssel wird zum Identifizieren der GUI-Komponente verwendet.
- Widget für Benutzerdetails: Hier können Sie ein Widget angeben, das anstatt des Standard Benutzerprofils angezeigt werden soll.
- Komponentenschlüssel für Benutzerdetails: Definieren Sie hier den Komponentenschlüssel der Benutzerdetails. Dieser Komponentenschlüssel wird zum Identifizieren der GUI-Komponente verwendet.

Funktionen

Mit Hilfe der Funktionen können Sie Elemente definieren, die nicht unmittelbar im Baum sichtbar sind, aber unter anderem von Baumfunktionen verwendet werden können. Alle definierten Funktionen werden in einer Tabelle dargestellt. Über die Toolbarfunktionen können diese hinzugefügt, bearbeitet oder entfernt werden. Alle Funktionen, die hier beschrieben werden, sind auch im [Programmierhandbuch](#) im Abschnitt [Non tree nodes \(<nodes>\)](#) beschrieben.

Funktionsknoten bestehen immer aus einer *Id* (siehe unten) und einem *Namen*, wobei der Name übersetzt angezeigt werden kann oder auch nicht.

Fixe Elemente eines Funktionsknotens sind somit:

- **Id:** Eindeutige Identifizierung des Knoten. Wenn keine Id angegeben wurde, vergibt sie *EPworX* automatisch.
- **Name:** Name, der angezeigt wird. Ist in den meisten Fällen ein Muss-Feld.
- **Lokalisieren:** Durch Aktivieren dieser Checkbox wird der eingetragene Name übersetzt (ohne @), sofern der String im Default- bzw. in einem angegebenen Resource-Bundle vorhanden ist. Der I18n-Link neben diesem Feld ermöglicht direkt die Anpassung der Ressource (falls in Applikationsdetailmaske - Registerkarte *Eigenschaften* festgelegt). Die Änderungen werden nach Aktivieren der Schaltfläche *Speichern* in der Ressource Datei (Strings) dieser Applikation gespeichert (siehe Abschnitt [Ressourcen Editor](#)).
- **Referenz zu:** Hier können Sie eine Referenz auf einen anderen Knoten in der selben oder einer anderen GUI-Konfiguration definieren. Dazu geben Sie die ID der GUI-Konfiguration mit dem Knoten, der referenziert werden soll in folgender Form an: `<xmlid>.<knotenid>`
Bei Angabe einer Referenz werden alle Einstellungen des Referenzknoten geladen, jedoch können Sie mit den Attributen aus dem aktuellen Knoten diese überschreiben. Ganz rechts außen finden Sie das Anzeige-Icon, mit dem Sie die Attribute des Referenzknoten in schreibgeschützter Ansicht ansehen können.
- **Inklusive Kind-Knoten:** Dieses Attribut ist Teil der Referenz auf einen anderen Knoten und dient dazu, dass alle Unterknoten des Referenzknoten geladen werden.
- **Parameter:** Analog der Beschreibung im Kapitel [Knoteneigenschaften](#). Einige *EPworX* Aktionen können mit Parameters angepasst werden. Für mehrere Informationen bitte den Kapitel [Anpassbare Aktionen](#) anschauen.
- **Hilfecontext:** Hinzufügen von kontextsensitiver Hilfe. Der Kontext besteht aus der Anwendungs-ID und dem auf der Hilfeseite definierten Anker. Für mehrere Informationen bitte die [Application Development Guide](#), Kapitel [Using context sensitive help in applications](#) anschauen.
- **Bei Anzeigen neu laden:** Ist dies aktiviert, wird der Knoten bei jedem Aufruf neu geladen.



Knoteneigenschaften
Aktion
Objektauswahl
Tabelle
Zusatzdaten

Id: backlogtable
Name: * ep:contents l18n: Inhalt
☒ Lokalisieren
Referenz zu:
Parameter:
Hilfekontext:
Bei Anzeigen neu laden: ☐
onClick: itsm/BacklogTableAction
Widget:
Komponentenschlüssel:
Iconpfad:

Schnellzugriff:
Standard-Knoten...
Zielfenster:
URI:
Detailfenster-Eigenschaften:
Anwenden auf: ONE
Aktions-Id:
Region:
Bei Item platzieren: ☐
Auf ungespeicherte Änderungen prüfen: ☐

Ok
Abbrechen

Abbildung 6.33: Knoteneigenschaften von Funktion

- Auf ungespeicherte Änderungen prüfen: Ist diese Checkbox aktiviert, wird eine Warnung bei nicht gespeicherten Änderungen bevor die Funktion ausgeführt ist, angezeigt.

Folgende Knotentypen stehen zur Auswahl:

- Aktion: In manchen Fällen wird eine eigene Funktion benötigt. Diese Funktion können Sie dann über eine Aktions-Id z.B. beim Arbeitskorb als Funktion einbin-

den. Dies erfolgt nach der Syntax `<xmlid>.<knotenid>`. Eine Aktion hat folgende Elemente:

- **Schnellzugriff:** Hier können Sie eine Tastenkombination angeben, um die Funktion mit dieser anzusteuern (Shortcut). Eine Liste der Tasten ist [here](#) zu finden. Beispielfunktion: CTRL+SHIFT+A. Die Funktion kann jedoch nur ausgeführt werden, wenn sie im sichtbaren Bereich vorhanden ist, z.B. ist die Funktion eine Toolbarfunktion im Arbeitskorb, muss zuerst die Arbeitskorb-Tabelle angezeigt werden.
 - **Standard-Knoten:** Durch Aktivieren dieser Schaltfläche wird eine Liste mit Standard-Knoten angezeigt. Wird ein Knoten ausgewählt, werden die entsprechenden Einstellungen übernommen.
 - **Zielfenster:** Hier kann das Zielfenster der Funktion definiert werden - standardmäßig wird *right* verwendet für den Arbeitsbereich. Mit dem Wert *ajax* kann eine AJAX Servlet Methode aufgerufen werden die z.B. für Subformtabellen nützlich sein kann.
 - **URI:** Definition einer URI zu einer Funktion, die aufgerufen werden soll. Mittels dem Auswahlssymbol können Sie vordefinierte Links auswählen. Diese URL wird standardmäßig verwendet, falls keine onClick-Aktion angegeben wurde.
 - **Detailfenster-Eigenschaften:** Zusätzliche Fenster-Eigenschaften können hier durch Strichpunkt getrennt definiert werden. Die Syntax ist analog zu jener der Java Skript Methode *window.open()*.
 - **Anwenden auf:** Definition, ob Funktion auf keinen, einen oder mehreren Tabelleneinträgen ausgeführt werden soll. Dazu gibt es folgende Modi:
 - * NONE: Funktion kann ohne Auswahl eines Tabelleneintrags ausgeführt werden
 - * ONE: Funktion kann nur ausgeführt werden, wenn genau 1 Tabelleneintrag ausgewählt wurde
 - * MULTI: Funktion kann ausgeführt werden, wenn einer oder mehrere Tabelleneinträge ausgewählt wurden
 - **Aktions-Id:** Hier können Sie eine eigene Aktions-Id eingeben. Damit kann man bei überschriebenen Aktionen fürs "Drag and Drop" herausfinden welche Aktion eigentlich die Ursprungsaktion ist.
 - **Region:** Hier können Sie festlegen wo die Aktion platziert werden soll. Es gibt drei mögliche Regionen, *top*, *trailing* und *center*. Die Kopfzeile in der Prozessdetailansicht kann mit *top* bearbeitet werden. Mit *trailing* und *center* kann definiert werden, ob die Registerkarten oben oder rechts von der Prozessdetailansicht platziert werden.
 - **Bei Item platzieren:** Durch Aktivieren dieser Checkbox kann die Aktion direkt bei dem Eintrag angezeigt werden.
- **Objektauswahl:** In Formularen kann es hilfreich sein, Referenzen von anderen Objekten im System zu speichern. Dazu kann die Objektauswahl hilfreich sein,

die über eine *searchid* beim Formularelement eingebunden wird. Nähere Informationen dazu finden Sie im [Programmierhandbuch](#) im Abschnitt [Usage of customized DOJO controls](#). Hier können Sie folgende Attribute festlegen:

- **Klassenname:** Hier definieren Sie die Klasse, für die eine Objektauswahl angeboten werden soll. Diese Klasse können Sie händisch eintippen oder über den Auswahldialog eine Formulkasse auswählen. Dieses Feld ist ein Pflichtfeld.
- **Defaultsortierung:** Analog zu Knotentyp *Arbeitskorb*
- **Tabellen-Implementierung:** Hier können Sie das Tabellenmodell definieren (Default: *com.groiss.storegui.FormTable*).
- **Suchattribute:** Definieren Sie hier optional die Suchattribute, nach denen in der Objektauswahl gesucht werden kann. Das Festlegen der Suchattribute erfolgt analog zu den Attributen.
- **Parametertypen:** Im Feld *Bedingung* können Sie auch Platzhalter (?) definieren, die dynamisch befüllt werden. Jeder Platzhalter benötigt einen Typ welcher in diesem Feld in einer durch Beistrich getrennten Liste angegeben werden kann. Folgende Typen können eingetragen werden:
 - * Persistent
 - * Date
 - * Long
 - * Double
 - * Integer
 - * String
 - * OIDList

Nähere Informationen zur Anwendung finden Sie im [Programmierhandbuch](#), Abschnitt [Usage of customized DOJO controls](#).

- **Bedingung:** Dieses Attribut kann dazu genutzt werden, um eine SQL-WHERE Bedingung für die Auswahl zu definieren. Dazu müssen Sie nur den Bedingungsstring ohne WHERE angeben, z.B. *formdept in (select oid from avw_dept where id = 'myoe')*.
- **Attribute:** Hier können Sie die Anzeigeattribute der Formulkasse definieren. Dazu geben sie eine Bestrichgetrennte Liste von Formularfeldern an, das zuvor unter Klassenname ausgewählt wurde. Wird kein Attribut angegeben, werden die Attribute herangezogen, die beim Formular als Namensattribute definiert wurden.
- **Attributsnamen:** Eine, durch Kommas getrennte, Liste von Attributsnamen, wie sie dann vom System für jedes Objekt übertragen wird. Die Namen können internationalisiert werden (z.B. *@@@key@@*).
- **Sortierer:** Hier können Sie einen *Comparator<Persistent>* angeben, der dann für die Sortierung verwendet wird.
- **Änderungen erlauben:** Wenn diese Eigenschaft aktiviert ist, ist das Einfügen, Aktualisieren und Löschen mit JSONLoader erlaubt.

- **Tabelle:** Tabellen, die nicht direkt über eine Baumfunktion erreichbar sein sollen, werden hier definiert (z.B. Subtabellen). Die Attribute sind analog zur Tabelle in Abschnitt [Knoteneigenschaften](#).
- **Zusatzdaten:** In *EPworX* ist es möglich Zusatzdaten an Stammdatenobjekte anzuhängen. Zu diesem Zweck kann diese Funktion verwendet werden. Nähere Informationen dazu finden Sie im [Programmierhandbuch](#) im Abschnitt [Adding tab Additional Info](#). Hier können Sie folgende Attribute festlegen:
 - **Klassenname:** Hier definieren Sie die Klasse, die für Zusatzdaten herangezogen werden soll. Diese Klasse können Sie händisch eintippen oder über den Auswahldialog eine Formulkategorie auswählen. Dieses Feld ist ein Pflichtfeld.
 - **Position:** Über das Attribut "position" kann die Position der Registerkarte *Zusatzdaten* in der Registerkartenliste festgelegt werden (muss eine positive Ganzzahl sein).
 - **Hinzufügen zu:** In diesem Feld geben Sie an, zu welchem Stammdatenobjekt (Klassenname) die Zusatzdaten angehängt werden sollen, z.B. com.groiss.org.User.
 - **Formular:** Hier können Sie das Formulartemplate angeben, das als Zusatzdatenformular dient. Bitte beachten Sie, dass die Datei im Klassenpfad ist und vom Formulartyp XForm ist!
 - **Editierbar:** Mit Hilfe dieser Checkbox können Sie angeben, ob das Zusatzdaten-Formular editierbar ist (= Speichern Schaltflächen sind aktiv) oder nur in schreibgeschützter Ansicht angezeigt werden soll.

6.7.2 Registerkarte: Datei

Nachdem Sie diesen Reiter aktiviert haben, öffnet sich der Editor zum direkten Bearbeiten des XML. Mit Hilfe der Speichern-Schaltflächen werden die Änderungen direkt in die XML-Datei am Dateisystem geschrieben. Mittels Tastatur-Taste *F6* kann das XML Schema geöffnet werden.

6.7.3 Registerkarte: Zuordnungen

Bei der Verwendung verschiedener Client-Konfigurationen können Sie nun festlegen, welcher Benutzer oder welche Rolle diese Konfiguration verwendet (siehe Abbildung [6.34](#)). Der Anwendungsbereich ist entweder ein Benutzer oder eine Rolle. Wenn mehr als ein Eintrag übereinstimmt, wird der mit der höheren Präferenz ausgewählt.

Folgende Beschreibung der Detailmaske:

- **Akteur:** Sie können zwischen einem *Benutzer* oder einer *Rolle* (mit *Organisationseinheit*) wählen.
- **Präferenz:** Es ist möglich, einem Agenten mehr als eine GUI Konfiguration zuzuweisen. Zu diesem Zweck können Sie eine Präferenz festlegen, wobei die Einstellungen mit der höchsten Präferenz beim Login verwendet werden.

Allgemein*

Akteur

☒ Rolle: * Alle ▼

☐ Organisationseinheit: ▼

☐ Benutzer: ▼

Präferenz: * 0

Löschen Speichern und Schließen Speichern Abbrechen

Abbildung 6.34: **Objektdetailansicht: Zuweisungen**

Es gibt Konfigurationen für sowohl mobile als auch Desktop-GUIs. Das Verhalten zwischen ihnen, einschließlich des Wechsels zwischen Konfigurationen, wird im Kapitel [Verhalten zwischen mobiler und Desktop-GUI](#) beschrieben.

Verhalten zwischen mobiler und Desktop-GUI

Reihenfolge der Konfigurationsauswahl:

- **Desktop → Mobil:**

1. Wenn eine mobile Konfiguration für die Desktop-GUI zugewiesen ist (<Desktop-GUI>_mobile).
2. Höchste Präferenz.

- **Mobil → Desktop:**

1. Wenn eine Desktop-Konfiguration für die mobile GUI zugewiesen ist (<Desktop-GUI>).
2. Höchste Präferenz.
3. standard.xml.

Einloggen Mobil:

1. Höchste Präferenz.
2. Zugewiesene Desktop-GUI.

3. standard_mobile.xml (wenn keine GUI-Zuweisung vorhanden ist, weder Desktop noch Mobil).

Einloggen Desktop:

1. Höchste Präferenz.
2. standard.xml.

6.7.4 Anpassbare Aktionen

In *EPworX* können Sie bestimmte Standardaktionen anpassen. Diese Aktionen können mithilfe des *EPworX* GUI-Editors einer GUI-Konfiguration hinzugefügt werden, indem Sie den Menüpunkt Funktionen anklicken.

Als ZIP herunterladen (download)

Sie können eine benutzerdefinierte DMS-Funktion erstellen, die auf *EPworX* Standardaktion download referenziert (*Als ZIP herunterladen*).

Das Erstellen eines Namens des heruntergeladenen Dokuments in der Standardaktion ist wie folgt definiert:

- ein Dokument herunterladen: <file-name>.zip
- einen Ordner herunterladen: <folder-name>.zip
- mehrerer Dokumenten/Ordner herunterladen: <parent-folder-name>.zip

Sie können jedoch Ihr eigenes Namensmuster für die resultierende Zip-Datei hier angeben. Erstellen Sie eine neue Aktion und konfigurieren Sie sie wie in Abbildung 6.35 gezeigt.

Bitte beachten Sie die unterstützten Platzhalter innerhalb der Parameter:

- name: der Name des selektieren Objekts bzw. des Ordners, wenn mehrere Objekte selektiert sind
- ou: die Organisationseinheit, zu der das Objekt oder sein Ordner gehört
- id: die Id der Prozessinstanz, zu der das Objekt gehört
- process: der Name der Prozessdefinition der Prozessinstanz, zu der das Objekt gehört
- subject: der Betreff der Prozessinstanz, zu der das Objekt gehört

Dies dient als Beispiel, wenn Sie eine solche Aktion direkt in der xml-Datei selbst einfügen möchten (was im zweiten Tab des GUI-Editors möglich ist):

```
<action id="myDownloadZIPFunction" ref="admin.download">
  <name>My download ZIP function</name>
  <params>{namePattern: "{process}-{ou}"}</params>
  <apply>MULTI</apply>
</action>
```

Knoteneigenschaften

Aktion
Objektauswahl
Tabelle
Zusatzdaten

Id: customDownloadZIPFunction

Name: * My Download ZIP Function

☐ Lokalisieren

Referenz zu: admin.download **Inklusive Kind-Knoten:** ☐

Parameter: {"namePattern":"{process}-{ou}"}

Hilfekontext:

Bei Anzeigen neu laden: ☐

onClick:

Widget:

Komponentenschlüssel:

Schnellzugriff:

Standard-Knoten...

Zielfenster:

URI

Detailfenster-Eigenschaften:

Anwenden auf: MULTI ▼

Aktions-Id:

Region:

Ok
Abbrechen

Abbildung 6.35: Beispielkonfiguration für die benutzerdefinierte ZIP-Download-Funktion

Neues DMS Formular (dmsnew)

Darüber hinaus ist es möglich, eine Schaltfläche zur DMS Symbolleiste hinzuzufügen, die nur eine Instanz des definierten Formulartyps erstellt. Hierfür erstellen Sie eine neue Aktion in der GUI-Konfiguration, die auf die dmsnew Aktion verweist. Fügen Sie dann die ID des gewünschten Formulartyps zum Attribut *Parameter* hinzu (siehe Abbildung 6.36). Fügen Sie schließlich die Aktion einem DMS Ordner hinzu.

Sie können auch zusätzliche Parameter zu dieser Funktion hinzufügen. Weitere Einzelheiten finden Sie in Abbildung 6.37.

Bitte beachten Sie, dass die unterstützten Platzhalter für Parameter in dieser Funktion

Knoteneigenschaften

Aktion

Objektauswahl

Tabelle

Zusatzdaten

Id:

Name: *

☐ Lokalisieren

Referenz zu: **Inklusive Kind-Knoten:** ☐

Parameter:

Hilfekontext:

Bei Anzeigen neu laden: ☐

onClick:

Widget:

Komponentenschlüssel:

Schnellzugriff:

Standard-Knoten...

Zielfenster:

URI

Detailfenster-Eigenschaften:

Anwenden auf:

Aktions-Id:

Region:

Abbildung 6.36: **Beispielkonfiguration für die benutzerdefinierte Neue DMS-Form-Funktion**

wie folgt sind:

- **formtype:** Dieser Parameter enthält Informationen über den Formulartyp, einschließlich der ID, Version und Objekt-ID.
- **iconClass:** Dieser Parameter gibt die mit der Aktion verbundene Symbolklasse an.

Hier ist ein XML-Beispiel für eine solche Aktion:

```
<action id="myFormNew" ref="dmsnew">
```

Knoteneigenschaften

Aktion
Objektauswahl
Tabelle
Zusatzdaten

Id: myFormNew

Name: * New DMS Form
☐ Lokalisieren

Referenz zu: dmsnew **Inklusive Kind-Knoten:** ☐

Parameter: {"formtype": {"id": "Standarddokument", "version": 1, "objectId": "com.groiss.dms.FormType:250"}, "iconClass": "sclcon scDmsForm scDmsDocument Standarddokument scForm-version-1"}

Hilfekontext:

Bei Anzeigen neu laden: ☐

onClick:

Widget:

Komponentenschlüssel:

Schnellzugriff:

Standard-Knoten...

Zielfenster:

URI

Detailfenster-Eigenschaften:

Anwenden auf: NONE ▼

Aktions-Id:

Region:

Ok
Abbrechen

Abbildung 6.37: **Beispielkonfiguration für die benutzerdefinierte Neue DMS-Form-Funktion mit weiteren Parametern**

```

<name>Neues DMS-Formular</name>
<params>
{"formtype":
{"id": "Standarddokument", "version": 1, "objectId": "com.groiss.dms.FormType:250"},
"iconClass": "sclcon scDmsForm scDmsDocument Standarddokument"}
</params>
</action>

```

Akteuer Ändern (setAgent)

Wenn Sie die Aktion *setAgent* anpassen möchten, um nur die Rollen anzuzeigen, beziehen Sie sich auf Abbildung 6.38. Alternativ können Sie den Parameter „showRolesSelect“:false verwenden, um nur die Agenten anzuzeigen.

Knoteneigenschaften

Aktion
Objektauswahl
Tabelle
Zusatzdaten

Id:
setAgentRolesOnly

Name: *
Set Agent Function with Roles Only

☐ Lokalisieren

Referenz zu:
setAgent

Inklusive Kind-Knoten: ☐

Parameter:
{"showUserSelect":false}

Hilfekontext:

Bei Anzeigen neu laden:
☐

onClick:

Widget:

Komponentenschlüssel:

Schnellzugriff:

Standard-Knoten...

Zielfenster:

URI

Detailfenster-Eigenschaften:

Anwenden auf:
NONE

Aktions-Id:

Region:

Bei Item platzieren: ☐

Auf ungespeicherte Änderungen prüfen: ☐

Ok
Abbrechen

Abbildung 6.38: **Beispielkonfiguration für die benutzerdefinierte Set-Agent-Funktion**

Hier ist ein XML-Beispiel für eine solche Aktion:

```
<action id="setAgent_rolesOnly" ref="setAgent">
  <name>Akteuer Ändern</name>
  <params>{"showUserSelect":false}</params>
  <apply>NONE</apply>
  <checkForChanges>true</checkForChanges>
</action>
```

Vertretungen (substitutions)

Sie können Substitutionen im Endbenutzer-Client definieren, um standardmäßig Rollen einzuschließen, indem Sie eine Aktion mit dem Parameter *includeRoles* definieren (siehe Abbildung 6.39).

Hier ist ein XML-Beispiel für eine solche Aktion:

```
<action id="substitutions" ref="admin.substitutions">
  <params>{includeRoles: true}</params>
</action>
```

6.8 Style-Konfigurationen

Mit Hilfe des *Style Editors* können Sie das Aussehen des Workflow-Clients anpassen und verschiedene Designs zur Verfügung stellen, die von den Benutzern ausgewählt werden können. Um die Auswahl für einen Benutzer einzuschränken, verwenden Sie bitten den Reiter *Zugriff* dafür und vergeben den Benutzern/Rollen das *Objekt ausführen*-Recht (nähere Details zu diesem Reiter siehe [Berechtigungssystem](#)).

Standardmäßig werden von *EPworX* folgende Designs angeboten:

- Hell (bright): Design mit hellblauen Tönen
- Dunkel (dark): Dunkles Design

Um ein neues Design zu erstellen, aktivieren Sie die Funktion *Neu*. Im Style Editor können Sie die Variablen für verschiedene Farben ändern, Systemlogos festlegen und benutzerdefinierte CSS-Anweisungen hinzufügen.

Die Konfigurationen werden als einzelne Dateien gespeichert unter:

local/appls/<appl>/classes/<appl>/themes/<theme_id>.less

EPworX bietet weiters die Möglichkeit ein Standard-Design festzulegen indem die Id der entsprechenden Style-Konfiguration beim Konfigurationsparameter *ep.style.theme* unter *Konfigurationen* → *Weitere Parameter* eingetragen wird. Dieses Design wird standardmäßig angezeigt, sofern das nötige *Objekt ausführen*-Recht beim Style-Konfigurationsobjekt definiert wurde und der Benutzer das *Default*-Design in den Benutzereigenschaften auswählt. Dieses Design wird nicht mit anderen Designs gemischt. Sollte der Benutzer für dieses Design nicht das Recht besitzen, wird standardmäßig das System-Design

Knoteneigenschaften

Aktion

Objektauswahl

Tabelle

Zusatzdaten

Id:

Name: *

Referenz zu:

Parameter:

Hilfekontext:

Bei Anzeigen neu laden:

onClick:

Widget:

Komponentenschlüssel:

☐ Lokalisieren

Inklusive Kind-Knoten: ☐

☐

Schnellzugriff:

Zielfenster:

URI

Detailfenster-Eigenschaften:

Anwenden auf:

Aktions-Id:

Region:

Bei Item platzieren:

Auf ungespeicherte Änderungen prüfen:

Standard-Knoten...

Anwenden auf:

NONE

▼

☐
☐

Ok

Abbrechen

Abbildung 6.39: **Beispielkonfiguration für die Definition einer Substitution**

von EPworX geladen.

Wenn das Design für das ganze System angepasst werden sollte, tragen Sie dieses beim Konfigurationsparameter *ep.system.style.theme* unter *Konfiguration* → *Weitere Parameter* ein. Geben Sie dazu die ID (= Dateiname) eines Designs ein, damit die-

ses nach dem Standard-Styling von *EPworX* und vor den über die Administration via *Style Editor* erstellten Designs eingemischt wird.

6.8.1 Registerkarte: Allgemein

Auf dieser Registerkarte können Benutzer die Designs erstellen und anpassen. Neue Designs benötigen eine *Id* und einen *Name*. Darüber hinaus kann eine *Beschreibung* eingegeben werden. Die Auswahl einer *Applikation* ist obligatorisch.

The screenshot shows the 'Style-Konfiguration' window with the 'Allgemein' tab selected. It contains the following elements:

- Id:** demo_gradient
- Name:** gradient_theme
- Beschreibung:** (empty text area)
- Applikation:** demo
- Datei:** C:\Users\inlogic\Desktop\Test Installationen\ep12\ep12_docu\local\apps\demo\classes\demo\themes\demo_gradient.less
- Settings:** A list of UI elements with corresponding color and style pickers:
 - Textfarbe: white
 - Sekundäre Hintergrundfarbe: #386FA4
 - Hauptnavigationsfarbe: transparent
 - Sekundäre Navigationsfarbe: #386FA4
 - Navigations-Selektionsfarbe: #ffff
 - Navigations-Selektions-Textfarbe: #000000
 - Underlay Farbe: #242424
 - Tabellen-Selektions-Textfarbe: @color_standard
 - Tabellenzellen-Textfarbe: @color_standard
 - Tabellenzellen-Hover-Textfarbe: black
 - Link-Hover-Farbe: @color_link
 - Primär-Aktion Textfarbe: black
 - Eingabefeld-Randradius: 8px
 - Eingabefeld-Hover-Randfarbe: @primary_action_color
 - Gesperrtes Eingabefeld Hintergrundfarbe: #E3E3E3
 - Titel Textfarbe: @color_standard
 - Titel Link-Hover-Farbe: @header_text_color
 - Logo invers: ../images/epworx-inverse.svg
 - Logo invers groß: ../images/epworx-inverse-large.svg
 - Haupt-Hintergrundfarbe: #133C55
 - Randfarbe: #C9C9C9
 - Hauptnavigationstextfarbe: white
 - Sekundäre Navigationsfarbe Akkordeon: (empty)
 - Navigations-Selektions-Randfarbe: transparent
 - Popup-Hintergrundfarbe: #eaf0f7
 - Tabellen-Selektionsfarbe: fadein(main_background_color, 0.9)
 - Tabellenzellen-Hoverfarbe: @grid_selection_color
 - Link-Farbe: inherit
 - Primär-Aktion Hintergrundfarbe: #FFA423
 - Eingabefeld-Randfarbe: @standard_border_color; /* rgb(228, 228, 228) */
 - Container-Randradius: 14px
 - Invalid-Farbe: #ae2e24
 - Gesperrtes Eingabefeld Textfarbe: #545454
 - Titel Linkfarbe: #326cb3
 - Logo: ../images/epworx.svg
 - Logo groß: ../images/epworx-large.svg
 - Logo klein: ../images/epworx-small.svg
- Buttons:** Löschen, Speichern und Schließen, Speichern, Abbrechen

Abbildung 6.40: Style Editor

Die *Id* ist auch gleichzeitig der Dateiname der .less-Datei, die im Applikationsverzeichnis gespeichert wird.

Oben rechts auf Höhe des Seiten-Titels wird die Vorschau-Funktion angeboten, die das angepasste Design so anzeigt, wie es die Nutzer sehen würden.

6.8.2 Registerkarte: Datei

Nach Aktivierung dieser Registerkarte öffnet sich der Editor zur direkten Bearbeitung der CSS-Datei. Sofern bereits Änderungen im Reiter *Allgemein* vorgenommen wurden, werden diese auch hier angezeigt.

6.9 Ressourcen Editor

Dieser Abschnitt beschreibt die Verwendung des *EPworX* Ressourcen Editors. Dieses Tool ermöglicht neben der Adaption von Applikationsressourcen (Strings) die Ansicht im schreibgeschützten Modus von *EPworX* Standardressourcen. Der Ressourcen Editor

kann nur verwendet werden, wenn in der Detailmaske der Applikation eine Ressource eingetragen wurde (siehe Abschnitt [Applikationen - Registerkarte: Eigenschaften](#)). Die Applikation *Default* bildet dabei eine Ausnahme, da hier die Standardressourcen von *EPworX* immer im schreibgeschützten Modus angezeigt werden. Eine neue Sprache kann jedoch immer mittels der Toolbarfunktion *Neue Sprache* hinzugefügt werden (siehe Abschnitt [Toolbarfunktionen](#)).

Hinweis: Der Ressourcen Editor erzeugt beim Speichern jeweils eine CSV-Datei (Strings.xls) und Property-Dateien (abhängig vom eingetragenen Pfad in der Applikationsdetailmaske). Um im Editor bestehende Ressourcen editieren zu können müssen eine CSV- bzw. die Property-Dateien am Dateisystem verfügbar sein. Weitere Informationen über Ressourcen finden Sie im [Programmierhandbuch](#) von *EPworX*.

Durch Aktivieren des Verweises *Ressourcen* im Navigationsbaum wird eine Tabellenansicht mit den Applikationsressourcen (Strings) geladen. Beim Aktivieren dieses Verweises in der Applikation *Default*, wird eine Auswahlseite für Strings und Errors angezeigt (Standard *EPworX* Ressourcen). Die Toolbarfunktionen der Tabellenansicht werden im Abschnitt [Toolbarfunktionen](#) beschrieben. Folgende Spalten sind verfügbar (siehe Abbildung [6.41](#)):

- **Znr:** Entspricht der Zeilennummer
- **Schlüssel:** Diese Spalte beinhaltet alle Schlüssel, die übersetzt werden sollen. Existierende Schlüssel können nicht in dieser Ansicht geändert werden (nur gelöscht).
- **Sprachspalten:** Eine Menge von Spalten wird angezeigt, wobei eine Spalte eine Sprache repräsentiert. Die 1. Sprachspalte entspricht automatisch der Standardsprache (= *Strings.properties*).

Das Tabellenverhalten (Sortierung, Spaltenauswahl, etc.) ist analog zum Standard *EPworX* Verhalten (siehe Abschnitt [Spaltenauswahl, Sortierung und Filter](#)).

6.9.1 Toolbarfunktionen

In diesem Abschnitt werden die diversen Toolbarfunktionen zur Adaptierung der Ressourcen behandelt. Die meisten Funktionen können nicht ausgeführt werden, wenn die Ressourcen aus einer JAR-Datei geladen werden. Folgende Funktionen sind verfügbar:



- **Neue Zeile:** Diese Funktion fügt der Tabelle eine neue Zeile hinzu. Wenn keine Zeile aktiviert wurde, wird die erste leere Zeile auf der letzten Seite aktiviert oder, wenn keine leere Zeile vorhanden ist, eine neue Seite mit einer leeren Zeile angelegt. Wenn eine Zeile ausgewählt ist, so wird die neue Zeile an der entsprechenden Position, abhängig von der Sortierung, eingefügt.
- **Zeile bearbeiten:** Wählen Sie eine Zeile aus und aktivieren Sie diese Funktion um eine Übersicht über den ausgewählten Übersetzungsschlüssel zu erhalten. Diese Ansicht erlaubt weiters die Bearbeitung der Übersetzungen und das



6. Workflowelemente

EPworX

Administration ep_worx - Gemeldet: Jack Jones - Development-Lizenz

Organisation

Applikationen

Applikationsliste

Berechtigungsvergabe

CRM

Default

demo

ITSM

Prozesse

Formulare

Tasks

Funktionen

Rollen

Rechte

Objektklassen

Funktionsgruppen

GUI-Konfigurationen

Style-Konfigurationen

Ressourcen

Wertelisten

Timer

Mailboxen

Reports

Webservice Clients

Suche

Admin-Tasks

Konfiguration

Neue Zeile

Zeile bearbeiten

Zeile löschen

Zeile kopieren

Speichern

Rückgängig

Alle Einträge

ITSM-Strings

↑ Znr

Schlüssel

Englisch

Deutsch

1	abortandarchive	Abort and Archive	Abbrechen und Archivieren
2	addchange	Add Change	Change hinzufügen
3	addinc	Add Incident	Incident hinzufügen
4	additionalinfo	Additional Info	Zusatzinformationen
5	addtoexistproc	Add to an existing process	Zu einem bestehenden Prozess hinzufügen
6	addtnewproc	Add to a new process	Zu einem neuen Prozess hinzufügen
7	archive_btn	Archive	Archivieren
8	archivefollowingprocs	Archive the following processes	Archiviere folgende Prozesse
9	ask_selection_table	Attach email to:	E-Mail anhängen an:
10	assign	Assign	Zuweisen
11	assigner	Assigner	Zuweiser
12	auto_receipt	Auto receipt	Automatische Empfangsbestätigung
13	auto.state.change	Auto change of state (PM)	Automatische Statusänderung (PM)
14	bcc.recipient	BCC recipient	BCC Empfänger
15	bug	Bug	
16	cancel	Cancel	Abbrechen
17	category	Category	Kategorie
18	change_req	Change Request	
19	changelogtext	Changelog text	Changelog Text
20	changespecific	Change specific	Change spezifischer
21	chreqcomp	Change Request Implemented	Change Request implementiert
22	chgsforrelease	Linked Changes for Release	Verknüpfte Changes für Release
23	ciclass	CI Class	CI Klasse
24	ciclassess	CI Classes	CI Klassen

Anzahl der Einträge: 600 | 0 ausgewählt

Abbildung 6.41: Ressourcen Editor Tabelle

Vor- bzw. Zurückspringen in den Zeilen. Änderungen werden erst nach Betätigen der Schaltfläche *Übernehmen* in der Tabelle angezeigt (= temporär gespeichert); persistent gemacht werden die Änderungen durch Aktivieren der Toolbarfunktion *Speichern*.



- **Zeile löschen:** Diese Funktion löscht die ausgewählte Zeile aus der Tabelle.



- **Kopieren:** Die ausgewählte Zeile wird durch Aktivieren dieser Funktion kopiert. Die kopierte Zeile wird an der entsprechenden Stelle, abhängig von der Sortierung, eingefügt.



- **Speichern:** Durch Aktivieren dieser Funktion werden alle getätigten Änderungen in der Tabelle in die entsprechenden CSV- bzw. Property-Dateien gespeichert. Die CSV-Datei und die Property-Dateien werden im gleichen Verzeichnis gespeichert, das im Eigenschaften-Dialog der jeweiligen Applikation angegeben wurde (siehe Abschnitt [Applikationen](#)). Ressourcen, die aus einer JAR-Datei gelesen werden, werden beim Speichern nicht berücksichtigt (diese Sprachen können bereits in der Tabelle nicht bearbeitet werden). Neue Sprachen (= neue Spalte) werden unter dem gleichen Pfad im classes-Verzeichnis der jeweiligen Applikation gespeichert (CSV-Datei und die jeweilige Property-Dateien), wie sie auch in der JAR-Datei vorhanden wären.

Hinweis: Die CSV-Datei wird als *UTF-16LE*-Datei gespeichert.



- **Änderungen verwerfen:** Aktivieren Sie diese Funktion, um alle getätigten Änderungen im Ressourcen Editor seit dem letzten Speichern (Toolbarfunktion *Speichern*) zu verwerfen.



- **Kurzsuche:** Geben Sie einen Wert in das Textfeld ein und aktivieren Sie diese Funktion, um ein eingeschränktes Ergebnis zu erhalten. Diese Suche verhält sich analog zu Standard Kurzsuche von *EPworX*.



- **Alle Einträge:** Aktivieren Sie diese Funktion, um wieder alle Einträge der Tabelle anzuzeigen.



- **Tabelle für Ressourcdateien sortieren:** Aktivieren Sie diese Funktion, wenn die temporäre Sortierreihenfolge der Tabelle für die CSV- bzw. Property-Dateien verwendet werden soll. Die Änderungen werden erst durch Betätigen der Toolbarfunktion *Speichern* persistent gemacht.



- **Neue Sprache:** Ein neuer Dialog öffnet sich, wo eine neue Spalte hinzugefügt werden kann, indem aus der Dropdown-Liste ein Wert ausgewählt und anschließend die Schaltfläche *Erstellen* aktiviert wird. Es werden die Standardsprachen in dieser Liste angezeigt falls keine anderen Sprachen unter *Konfiguration → Lokalisierung → Liste der Sprachen* hinzugefügt wurden. Diese Funktion kann immer ausgeführt werden, auch wenn bestehende Ressourcen aus einer JAR-Datei gelesen werden.

6.9.2 CSV-Datei Konvertierung

Bei Verwendung von CSV-Dateien muss sichergestellt sein, dass sie UTF-16LE codiert sind. Folgende Funktion ermöglicht die Konvertierung von Cp1252 nach UTF-16LE:

```
http://host':port'/context-root'/servlet.method/
com.groiss.reseditor.HtmlMethods.convertXLS?resource=<reurl>
```

Der Parameter *resource* muss dabei der URL der CSV-Datei entsprechen, z.B. für Applikation *myappl* Strings:

```
http://host':port'/context-root'/servlet.method/
com.groiss.reseditor.HtmlMethods.convertXLS?
resource=com/dec/myappl/resource/Strings.xls
```

6.10 Wertelisten

Mit Hilfe von Wertelisten können Sie auf einfache Weise verschiedene Attribute mit Werten zusammenfassen und einer Applikation zuordnen. Wertelisten werden zum Beispiel in Formulare für Auswahllisten verwendet und können auch im Formular-Editor (Registertkarte Schema) über die obligatorische *Id* referenziert werden. Die Wertetabelle des Wertelisten Objekts besteht aus folgenden Elementen:

- **Beschriftung:** Ein beliebiger Text (der auch mit beginnenden @@@ Zeichen für die Anzeige internationalisiert werden kann), der auf der Benutzeroberfläche angezeigt werden soll (z.B. in Auswahlliste im Formular, beim Report Anzeige-Attribut, etc.).

- **Wert:** Der Wert, der in der Datenbank gespeichert wird.
- **Position:** Mit Hilfe dieser Spalte können Sie festlegen, in welcher Reihenfolge die Einträge der Werteliste angezeigt werden. Ist keine Position definiert (= Positionswert 0), werden die Werte alphabetisch nach dem übersetzten Beschriftungstext sortiert.

Mit Hilfe der Schaltfläche *Datei hochladen* können Sie eine CSV-Datei mit den entsprechenden Informationen hochladen. Einträge der Werteliste können entweder überschrieben werden (= Option *Ersetzen*) oder es werden die bestehenden Einträge um jene aus der CSV-Datei ergänzt (= Option *Zusammenführen*), wobei der Inhalt des Feldes *Wert* für die Prüfung herangezogen wird. Das Format der CSV-Datei ist folgendes: Die 1. Zeile besteht aus dem fixen Text `value;label;position`, ab der 2. Zeile werden die Daten angegeben, die importiert werden sollen.

Beispiel:

```
value;label;position
1;Option 1;1
2;Option 2;2
3;Option 3;3
```

Die importierten Werte werden nicht sofort in der Datenbank gespeichert, dazu müssen Sie erst explizit die Speichern-Aktion über die entsprechenden Schaltflächen auslösen!

6.11 Timer

Der Timer in *EPworX* löst zeitgesteuerte Ereignisse aus. Er wird für systeminterne Zwecke verwendet, ist aber auch für applikationsspezifische Tasks verwendbar. Durch Aktivieren des Links **Timer** in der Liste der Administrationstasks gelangen Sie in die Verwaltungsmaske, die eine Liste der eingetragenen Ereignisse darstellt. Es stehen Ihnen die in Kapitel [Tabellendarstellung](#) beschriebenen Funktionen und zusätzliche Funktionen zur Verfügung:

- **Ausführen:** Mit dieser Funktion können Sie den selektierten Timer starten.
- **Timermanager Infos:** Eine Funktion, die interne Statusinformationen des Timermanagers anzeigt.

Die Objekt-Detailansicht von Timern gliedert sich in die folgenden Bereiche:

- Allgemein
- Zugriff
- Historie

Allgemein
Historie
Zugriff

Id: *

BatchManager

Klassenname: *

com.groiss.wf.batch.BatchManager

✓

Parameter:

Applikation: *

Default

Erster Lauf:

29-03-2023

12:59

Periode:

600

?

⌚

Aktiv:

☒

Wesentlich:

☐

Aufhol Pattern:

Beim Start ausführen:

☐

Auf jedem Knoten ausführen:

☐

Thread Id:

BatchManager

Beschreibung:

starts batch jobs

Letzter Lauf:

2026-07-21 11:39:51 - 2026-07-21 11:39:51 (0,002 s)

Passiviert um:

-

Originallauf:

-

Neuer Lauf:

-

Status:

-

Ausführen

Reaktivieren

Lauf abbrechen

Lauf wiederholen

Löschen

Speichern und Schließen

Speichern

Abbrechen

Abbildung 6.42: Objekt-Detailansicht: Timer

6.11.1 Registerkarte: Allgemein

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Id:** eine Id (Kurzbezeichnung) des Eintrags
- **Klassenname:** der Name der Klasse, die die Aktion enthält. Die Klasse muss das Java-Interface `com.groiss.timer.TimerTask` implementieren.

- **Parameter:** ein String-Parameter als Argument für die *run* Funktion.
- **Applikation:** Zuordnung des Timers zu einer Applikation. Aufgrund der Zuordnung wird dieser auch in der Timer-Tabelle der jeweiligen Applikation gelistet.
- **Erster Lauf:** Wann soll die Funktion das erste Mal ausgeführt werden?
- **Periode:** Wiederholungsintervall in Sekunden oder in Form von Cron-Pattern (siehe dazu Abschnitt [Cron-Pattern](#)).
- **Wesentlich:** Ist diese Checkbox aktiviert, wird der Timer im Falle eines Fehlers (z.B. Fehler in Timer, Datenbankverbindung unterbrochen, etc.) zu einem späteren Zeitpunkt nochmals ausgeführt und eine E-Mail an den Administrator gesendet. Dieser spätere Zeitpunkt wird mit dem Parameter *Aufhol Pattern* angegeben. Schlägt der Timer-Lauf beim nächsten Zeitpunkt des *Aufhol Pattern* wieder fehl, wird der nächste Zeitpunkt dieses Patterns verwendet usw. Ist der Timer-Lauf jedoch erfolgreich, wird wieder das Wiederholungsintervall vom Parameter *Periode* verwendet. Eine eigene Komponente (Timer-Watcher) überwacht die als wesentlich gekennzeichneten Timer und führt die zuvor genannten Aktionen durch. Ein Check auf verspätete oder versäumte Timerläufe wird alle 60 Sekunden durchgeführt, jedoch kann diese Periode mit dem Parameter *ep.timerentry.essential.check.seconds* auf der Konfigurationsmaske *Weitere Parameter* verändert werden.
Bei der Verwendung von wesentlichen Timer sollten Sie folgendes beachten:

1. Es wird empfohlen, dass wesentliche Timer in ihrem eigenem Thread laufen, d.h. eine eigene *Thread Id* haben!
2. Es wird nicht empfohlen, dass bei wesentlichen Timer die Checkbox *Beim Start ausführen* aktiviert ist!
3. In einer *EPworX* Cluster-Umgebung sollte die Checkbox *Auf jedem Knoten ausführen* aktiviert sein!

Beispiel:

Eine lang laufende Aktion die jedes Quartal beginnt. Das entsprechende Pattern dafür könnte z.B. `0 3 1 1,4,7,10 *` lauten, was bedeutet: Timerlauf wird um 3 Uhr morgens an jedem ersten Januar, April, Juli und Oktober gestartet. Wäre dieser Timer nicht wesentlich, dann würden Ausfälle (z.B. System ist nicht verfügbar, Netzwerk- oder Datenbankfehler) zum Zeitpunkt des Timerlaufes in einen fehlerhaften oder ungeplanten Lauf resultieren. Der versäumte Lauf müsste dann händisch zu einem passenden Zeitpunkt ausgeführt werden. Wäre in diesem Fall z.B. das Aufhol Pattern `* /5 3-5 * * *` definiert, so würden versäumte Läufe täglich, alle 5 Minuten zwischen 3 Uhr und 5 Uhr morgens nachgeholt werden. Dieses Pattern kann sinnvoll sein, wenn der Timerlauf lange dauert und viel Systemlast erzeugt.

- **Aufhol Pattern:** Wiederholungsintervall in Sekunden oder in Form von Cron-Pattern für Timer, die als *Wesentlich* gekennzeichnet sind.
- **Aktiv:** Nur aktive Einträge werden vom Timer berücksichtigt.

- **Beim Start ausführen:** Soll die Aktion beim Start von *EPworX* ausgeführt werden?
- **Auf jedem Knoten ausführen:** Dieser Timer wird lokal auf jedem Knoten im Cluster ausgeführt.
- **Thread Id:** Mit Angabe einer Thread Id können die Tasks zu einzelnen Threads gruppiert werden.
- **Beschreibung:** Freier Text.
- **Letzter Lauf:** Zeigt an, wann der Timer das letzte Mal ausgeführt wurde (Start- und Endzeit) und wie lange er gebraucht hat (in Sekunden).
- **Passiviert um:** Zeitpunkt, wann der Timer temporär inaktiv gesetzt wurde weil eine Ausnahme im Timer auftrat (nur bei Timer, wo Checkbox *Wesentlich* nicht aktiviert ist).
- **Originallauf:** Zeitpunkt, wann ein wesentlicher Timer seinen nächsten Lauf aufgrund der Angabe im Feld *Periode* hat.
- **Neuer Lauf:** Zeitpunkt, wann ein wesentlicher Timer seinen nächsten Lauf hat. Im Fehlerfall ist es der nächste berechnete Zeitpunkt aufgrund der Angabe im Feld *Aufhol Pattern*.
- **Status:** Der Status eines wesentlichen Timers wird hier angezeigt. In der Regel ist der Status *Geplant* und gibt an, dass der Timer zum Zeitpunkt von *Neuer Lauf* ausgeführt werden soll. Es kann jedoch auch sein, dass ein Timer bereits ausgeführt wird, aber noch nicht beendet ist. In diesem Fall ist der Status *Gestartet*. Ein weiterer Status ist *Gescheitert*, der angibt, dass der Timer-Lauf aufgrund eines Fehlers nicht durchgeführt werden konnte.

Wenn Sie die Schaltfläche **Ausführen** aktivieren, wird der Timer sofort ausgeführt.

Wenn Sie die Schaltfläche **Reaktivieren** aktivieren, wird der Timer aus dem passiven Zustand gelöst.

Die Schaltfläche **Lauf abbrechen** und **Lauf wiederholen** sind nur relevant, wenn ein Timer als *Wesentlich* markiert wurde. Wenn Sie die Schaltfläche **Lauf abbrechen** aktivieren, wird die Planung des nächsten Laufs (= Neuer Lauf) für den wesentlichen Timer unterbrochen und auf den nächsten Zeitpunkt aufgrund der Einstellung beim Feld *Periode* gesetzt.

Wenn Sie die Schaltfläche **Lauf wiederholen** aktivieren, wird ein Zeitpunkt für den neuen Lauf festgelegt, der 1 Minute in der Zukunft liegt.



Durch einen Klick auf dieses Symbol wird ein Popup-Fenster geöffnet, wo Sie entweder die Periode in Sekunden eingeben oder das Cron-Pattern verwenden können (siehe Abschnitt [Cron-Pattern](#)).



Wenn Sie auf dieses Symbol klicken, werden in einem Popup-Fenster die nächsten zehn Läufe des Timers angezeigt.

6.11.2 Cron-Pattern

Das Cron-Pattern kommt ursprünglich aus der UNIX-Welt und wird für Aufgaben verwendet, die in wiederkehrenden Abständen automatisiert durchgeführt werden sollen. *EPworX* verwendet dieses Pattern um Timer nach Belieben starten zu können. Dabei hält sich *EPworX* an den V7-Standard von cron.

Eine Zeile besteht grundsätzlich aus fünf festgelegten Spalten. Diese enthalten die Zeitangaben (Minuten, Stunden, Tage, Monate, Wochentage), wobei die Spalten durch Leerzeichen getrennt werden. Folgend sind die Einträge für die Zeitangaben dargestellt:

Minuten	0-59 und * für alle Minuten
Stunden	0-23 und * für alle Stunden
Tage	1-31 und * für jeden Tag
Monate	1-12 und * für jeden Monat
Wochentage	0-7 und * für jeden Wochentag (0 und 7 für Sonntag)

Es gibt zudem erweiterte Funktionen, die hier kurz erwähnt werden:

- Ein Komma , lässt mehrere Zeitangaben zu
- Ein Bindestrich - gibt einen Zeitraum an
- Ein Schrägstrich / teilt einen Zeitraum ein

Beispiele:

- Jeden Tag um 09:00 und 15:00 Uhr soll der Timer ausgeführt werden: 0 9,15 * * *
- Am 15. jedes Monats um 09:50 Uhr soll der Timer ausgeführt werden: 50 9 15 * *
- Jeden Samstag um 00:00 Uhr soll der Timer ausgeführt werden: 0 0 * * 6
- Der Timer soll jeden Tag alle 30 Minuten ausgeführt werden: */30 * * * *
- Jeden Tag von 8 bis 20 Uhr wird alle 20 min. der Timer ausgeführt: */20 8-20 * * *

Um nähere Informationen über cron zu erhalten, sehen Sie bitte [here](#) nach.

6.11.3 Übersicht über Standard-Timer

In *EPworX* gibt es eine Reihe vordefinierter Timer, die hier nun näher beschrieben werden:

- **BatchManager:** Startet und beendet Batch-Jobs. Wird nur verwendet, wenn Batch-Job Schritte in der Prozessdefinition verwendet werden.
- **BuildTimer:** Baut die aktiven Klassifikatorzuweisungen aller aktiven Prozesse. Dieser Timer wird nur dann benötigt, wenn Ihre Klassifikatorzuweisungen gelernt werden und nicht durch einen manuellen Entscheidungsbaum definiert sind. Folgende optionale Parameter können hier definiert werden:

- **processes:** Die (mit Beistrich getrennten) Ids der zu berücksichtigenden Prozesse
- **assignments:** Die (mit Beistrich getrennten) Namen der zu bauenden Klassifikatorzuweisungen
- **mailTo:** Eine E-Mailadresse, an die ein E-Mail mit den Bewertungsergebnissen der gebauten Klassifikatorzuweisungen verschickt wird
- **accuracyThreshold:** Ein Grenzwert für die Genauigkeit einer gebauten Klassifikatorzuweisung mit der Auswirkung, dass die zuvor genannte E-Mail nur die Bewertungsergebnisse jener Klassifikatorzuweisungen enthält, deren Genauigkeit unter diesem Schwellenwert liegt - oder bei deren Bau es zu einem Fehler gekommen ist. Unterstützte Werte liegen zwischen 0 und 100 und können auch Kommastellen enthalten (allerdings mit '.' als Kommasymbol).

Beispiel-Konfiguration:

```
processes=incident  
assignments=category  
mailTo=heisenberg@groiss.com  
accuracyThreshold=87.5
```

- **CalendarReminder:** Überprüft auf Kalendereinträge, die ihren Erinnerungszeitpunkt erreicht haben und sendet Hinweis-E-Mails für diese Einträge. Aktivieren Sie diesen Timer, wenn Sie den Kalender verwenden.
- **CleanUpDMS:** Löscht leere Ordner und ebenso Rechte, die sich auf DMS-Objekte beziehen, aber nicht mehr erreichbar sind. Aktivieren Sie diesen Timer, wenn Sie das DMS verwenden.
- **ClusterCheck:** Überprüft, ob andere Knoten laufen und teilt eine neuen Cluster Timer zu. Dieser Timer wird nur dann benötigt, wenn der *EPworX* Cluster verwendet wird. Im Feld *Parameter* können Sie eine Toleranzzeit angeben, indem Sie einen positiven Wert eingeben. Der Standardwert ist 60 Sekunden, der in der Implementierung des Timers vordefiniert ist. Für nähere Informationen über *EPworX* Cluster sehen Sie bitte im [Installationshandbuch](#) nach.
- **CurrentSubstitutes:** Überprüft, ob irgendwelche Vertretungseinstellungen aktiviert oder deaktiviert werden müssen aufgrund der eingetragenen Zeitdauer. Dieser Timer wird nur dann benötigt, wenn Benutzer- oder Rollenvertretungen verwendet werden.
- **DeferredUpdate:** Dieser Timer überprüft, ob es aufgeschobene Updates von Masterdaten gibt, für die der Ausführungszeitpunkt erreicht wurde. Wenn dieser Zeitpunkt erreicht wurde, wird ein Update durch diesen Timer ausgelöst. Lassen Sie diesen Timer aktiviert.
- **DeleteUserSessions:** Dieser Timer löscht User-Sessions, die bereits abgelaufen sind. Dabei kann man im Feld *Parameter* in Tagen angegeben werden, wie lange eine User-Session aktiv sein darf, bevor sie der Timer löscht.

- **DocumentTrackerTimer:** Verschickt Nachrichten an Empfänger einer Dokumentenverfolgung, wenn ein Dokument geändert wurde. Dieser Timer läuft standardmäßig alle 10 Minuten. Benutzer werden nur über Dokumentenveränderungen informiert, welche älter als 15 Minuten sind. Diese Verzögerung kann unter *Konfiguration* → *DMS* → *Zeitverzögerung für Dokumentenveränderungsbenachrichtigung* angepasst werden. Für weitere Informationen über die Konfiguration der Verzögerung sehen Sie bitte im [Installationshandbuch](#) nach.
- **Escalations:** Dieser Timer überprüft bei jedem Lauf, ob Eskalationen durchgeführt werden sollen. Für nähere Informationen sehen Sie bitte im Abschnitt [Registerkarte: Eskalationen](#) nach. Dieser Timer wird benötigt, wenn Eskalationen in der Prozessdefinition verwendet werden.
- **EvaluationTimer:** Führt eine Bewertung der aktiven Klassifikatorzuweisungen aller aktiven Prozesse durch. Folgende optionale Parameter können hier definiert werden:
 - processes: Die (mit Beistrich getrennten) Ids der zu berücksichtigenden Prozesse
 - assignments: Die (mit Beistrich getrennten) Namen der zu bewertenden Klassifikatorzuweisungen
 - mailTo: Eine E-Mailadresse, an die ein E-Mail mit den Bewertungsergebnissen der Klassifikatorzuweisungen verschickt wird
 - accuracyThreshold: Ein Grenzwert für die Genauigkeit einer Klassifikatorzuweisung mit der Auswirkung, dass die zuvor genannte E-Mail nur die Bewertungsergebnisse jener Klassifikatorzuweisungen enthält, deren Genauigkeit unter diesem Schwellenwert liegt - oder bei deren Bewertung es zu einem Fehler gekommen ist. Unterstützte Werte liegen zwischen 0 und 100 und können auch Kommastellen enthalten (allerdings mit '.' als Kommasymbol).

Beispiel-Konfiguration:

```
processes=incident  
assignments=category  
mailTo=heisenberg@groiss.com  
accuracyThreshold=87.5
```

- **HeartBeat:** Informiert den Cluster, dass dieser Knoten noch aktiv ist. Dieser Timer wird nur dann benötigt, wenn der *EPworX* Cluster verwendet wird. Im Feld *Parameter* können Sie eine Toleranzzeit (positiver Ganzzahlwert) angeben, wobei der Wert doppelt so groß wie der Intervall-Wert des Timers von allen Knoten sein sollte. Der Standardwert ist 30 Sekunden, der in der Implementierung des Timers vordefiniert ist. Für nähere Informationen über *EPworX* Cluster sehen Sie bitte im [Installationshandbuch](#) nach.
- **IndexRefresh:** Aktualisiert den Index für die Volltextsuche in ORACLE. Dieser Timer wird nur dann benötigt, wenn Sie die Volltextsuche unter ORACLE verwenden. Folgende Parameter können definiert werden:

- **index:** Die (Beistrich getrennten) Namen der Volltextsuchindizes. Defaultwert: `avw_ctxdoccont,avw_ctxfieldvals`
- **memory:** Der zugewiesene Speicher für den Refresh(auf der Datenbank). Defaultwert: 32M
- **threads:** Die Anzahl der zu verwendenden Threads auf der Datenbank. Defaultwert: 1
- **maxtime:** Die maximale Dauer in Minuten. Ist diese Zeit überschritten wird der Refresh abgebrochen und beim nächsten Lauf fortgesetzt. Defaultwert: `CTX_DDL.MAXTIME_UNLIMITED`
- **lockwait:** Oracle bietet mehrere Strategien, um mit DB Locks während des Refreshs umzugehen. Defaultwert: `LOCK_NOWAIT_ERROR`

Für nähere Informationen verweisen wir auf die ORACLE TEXT Dokumentation / CTX_DDL Package / SYNC_INDEX Prozedur.

- **LDAPDirSyncTask:** Synchronisiert Stammdaten mit LDAP Verzeichnis-Servern. Für detaillierte Informationen sehen Sie bitte im Abschnitt [LDAP](#) nach. Wird nur dann benötigt, wenn die periodische LDAP Synchronisierung konfiguriert ist.
- **Log:** Dieser Timer entfernt alle Logeinträge (außgenommen den aktuellsten Eintrag/die letzte Änderung), die älter sind als im Feld *Parameter* eingetragen. Wenn der Parameter einen positiven Wert D enthält, dann werden alle Logeinträge die älter als D Tage sind entfernt. Wenn der Parameter einen Property-String enthält, dann kann für jede Klasse ein positiver Wert (in Tage) festgelegt werden. Der Property-String besteht dabei aus Elementen (durch Zeilenumbruch getrennt) mit folgender Struktur: `classname=Dn`. Hat Dn den Wert 0, dann werden keine Logeinträge entfernt. Bei Verwendung des Zeichen * als Klassenname wird der Wert Dn für alle Logeinträge aller Klassen, die nicht explizit angeführt werden, angewendet, z.B.

`*=30`

`com.groiss.org.User=0`

`com.groiss.org.OrgUnit=1000`

`com.groiss.org.Role=1000`

`com.groiss.org.UserRole=365`

- **LogV2Migration:** Timer zum Migrieren von Log-Einträgen der Tabelle `avw_log` aufgrund der Reimplementierung der Versionierung in *EPworX* (eingeführt im Juni 2017). Dieser Timer wird benötigt, da eine Migration der Tabelle `avw_log` während eines Datenbank-Upgrades aufgrund der großen Anzahl von Einträgen (mehrere Millionen) nicht performant durchführbar wäre. Zu diesem Zwecke kann die Migration in definierten Phasen erfolgen, die über den Parameter *duration* (in Sekunden) gesteuert wird. Falls diese Dauer überschritten wird, stoppt der Timer die Ausführung automatisch. Der 2. Parameter *lastMigratedOid* wird zu

Performance-Zwecke verwendet, um Abfragen auf die Tabelle `avw_log` zu beschleunigen. Er wird nach jedem Timer-Lauf vom Timer selbst angepasst und soll daher nicht manuell geändert werden. Ist der Timer aktiviert und die Migration vollständig durchgeführt, so wird der Timer anschließend wieder automatisch auf inaktiv gesetzt.

- **MailGetter:** Ladet die Nachrichten herunter und führt die konfigurierte Aktion aus. Es werden immer alle E-Mails einer Mailbox behandelt und bei erfolgreicher Bearbeitung einer Nachricht die definierte Aktion ausgeführt. Bei E-Mails, die aufgrund eines Fehlers nicht korrekt behandelt werden können, wird nach Abarbeitung aller (erfolgreichen/nicht erfolgreichen) E-Mails eine Fehlermeldung ausgegeben, sofern unter *Konfiguration/Kommunikation/Administrator E-Mail Adresse* eingetragen der Administrator notifiziert und die Information der fehlerhaften E-Mails ins Serverlog geschrieben. Für nähere Informationen bezüglich Mailbox-Objekt sehen Sie bitte im Abschnitt [Mailboxen](#) nach. Dieser Timer wird nur benötigt, wenn Mailboxinhalte automatisch verarbeitet werden sollen.
- **MailQueueTimer:** Der Timer iteriert in einer bestimmten Zeit (standardmäßig 10 min.) über die Mail-Queue und versucht die entsprechenden E-Mails zu versenden. Schlägt das Senden für einen Eintrag fehl, wird bei diesem eine Statusmeldung hinterlegt, ansonsten wird der Eintrag aus der Mail-Queue entfernt. Nähere Informationen zur Mail-Queue finden sie im Abschnitt [Mail-Queue](#).
- **PasswordExpiration:** Überprüft, ob das Kennwort eines Benutzers in Zukunft abläuft und sendet ein Warnungs-E-Mail an den Benutzer. Die Einstellungen für die Gültigkeitsdauer und die Warnfrist sind in der Konfiguration unter dem Punkt *Password Policy* durchzuführen. Die Nachrichten-Vorlage für das Versenden des E-Mails ist in der Default-Applikation unter der Id `pwdExpiration` definiert.
- **ProcessStartTimer:** Der Timer startet den angegebenen Prozess in der angegebenen Organisationseinheit. Dazu müssen im Textfeld *Parameter* die Attribute *process* und *ou* wie folgt angegeben werden. Optional können Sie noch den Parameter *startNote* angeben (kann ein i18n-Key sein, z.B. `<app-id>:<resourcestring>`), um eine Prozessnotiz mit dem angegebenen Betreff zu erzeugen:

```
process=<procdef_id>
ou=<oe_id>
startNote=<notiz_betreff>
```

Beispiel:

```
process=jobproc
ou=gi
startNote=Gestartet durch Timer
```

- **RecycleBin:** Dieser Timer löscht alle Einträge aus dem DMS-Papierkorb aller Benutzer, die älter als n Tage sind. Die Tage können als positive Ganzzahl im Feld *Parameter* eingegeben werden. Nähere Informationen zum Papierkorb finden Sie im [Benutzerhandbuch](#).
- **ReportSubscriptionTimer:** Dieser Timer führt jene Reports aus, für die ein aktives Abonnement definiert wurde. Nähere Informationen über Report-Abonnements finden Sie im [Reportinghandbuch](#).
- **ReportTimer:** Dieser Timer führt gespeicherte Reports in regelmäßigen Abständen aus. Dabei sind folgende Parameter (gleichnamiges Feld) notwendig:
 - **query:** Hier können Sie eine Liste von ID's von gespeicherten Reports angeben, die ausgeführt werden sollen, z.B. `query=bsp_01,bsp_02`. Die Reports müssen einen Exporter verwenden, der *FileReportingExporter* implementiert, wie zum Beispiel Excel-, CSV-, Chart- oder PDF Exporter.
 - **action:** Hier können Sie angeben, wie der/die Report/s erzeugt werden sollen:
 - * **email:** Report wird an definierte E-Mailadresse versendet (= target).
 - * **dms:** Report wird im DMS an definierter Stelle gespeichert (= target).
 - * **file:** Report wird auf dem Dateisystem an definierter Stelle gespeichert (= target).
 - * **none:** Report wird ausgeführt, aber ohne weitere Aktion (z.B. notwendig für Reports mit Eskalationen Exporter)
 - **target:** Abhängig von der definierten Aktion, muss entweder eine E-Mailadresse, ein Pfad zu einem DMS-Ordner (z.B. COMMON/Reports) oder ein Pfad im Dateisystem angegeben werden (absolut oder relativ zum Server Root-Verzeichnis).
 - **filename:** Hier können Sie den Dateinamen (ohne Dateiendung) der exportierten Datei angeben. Wurde bei *query* mehr als nur einen gespeicherten Report angegeben, dann wird `_1`, `_2` etc. angehängt. Der standardmäßige Dateiname lautet: `<QueryId>_yyyy-MM-dd HH:mm:ss`
 - **nonempty:** Wenn dieser Parameter *true* ist, werden Reports ohne Ergebnis nicht versendet oder gespeichert.
 - **templateid:** Hier können Sie optional die Id einer Nachrichten-Vorlage angeben (siehe Abschnitt [Nachrichten-Vorlagen](#)). Standardmäßig wird die Vorlage mit der Id *reportTimer* aus der Applikation *default* verwendet, falls keine *templateid* angegeben wurde.
 - **Parameter:** Bei jeder Bedingung einer Gespeicherten Anfrage kann ein Parametername angegeben werden. Für jede Bedingung, die als „Parameter bei Ausführung“ definiert ist, muss entweder ein Standardwert definiert oder ein Wert und optional ein Operator wie im folgenden Beispiel angegeben werden:

```
<paramname>_value=test  
<paramname>_operator=like
```

Reports mit unvollständigen Bedingungen werden vom Report-Timer nicht ausgeführt.

Beispiel-Konfiguration:

```
query=report1
action=email
target=heisenberg@groiss.com
filename=Report1
nonempty=true
id_value=15
id_operator==
subj_value=Error
subj_operator=like
```

Sie können jedoch auch einen eigenen ReportTimer schreiben, indem Sie die Klasse `com.groiss.reporting.ReportTimer` erweitern. Nähere Informationen dazu finden Sie im *EPworX* API.

- **SeenObjectCleaner:** Entfernt alle Seen-Informationen, die nicht mehr benötigt werden. Die Seen-Informationen werden dazu benutzt, um anzuzeigen, ob ein Arbeitskorbeintrag neu ist (=unseen) oder nicht. Lassen Sie diesen Timer aktiviert.
- **Suspension:** Dieser Timer überprüft alle Einträge der Wiedervorlage, ob diese zurück in den Arbeitskorb kommen. Weiters werden auch aktive Warten Schritte geprüft, ob dessen Zeitdauer bereits überschritten ist (siehe Abschnitt [Ereignis Warten](#)). Lassen Sie diesen Timer aktiviert.
- **TokenRefresh:** Der Timer aktualisiert Zugriffstokens in OAuth Autorisieren. Zugriffstokens, deren Restlaufzeit weniger als die Hälfte der ursprünglichen Gültigkeitsdauer beträgt, werden erneuert.
- **UserSynchronizationTimer:** Synchronisiert Benutzer anhand der konfigurierten Benutzer-Synchronisierungsklasse. Für weitere Informationen über diese Konfiguration sehen Sie bitte im [Installationshandbuch](#) nach.

6.12 Mailboxen

Hier können Sie Mailboxen definieren. Es handelt sich dabei um Mailboxen auf einem Mail-Server auf die mit dem Protokoll IMAP4 oder POP3 zugegriffen wird. Eingehende Mail in solchen Mailboxen können auf unterschiedlichste Art verarbeitet werden. Häufig werden Workflows mit Daten aus der E-Mail gestartet.

Allgemein
Aktion
Mailbox anzeigen
Historie
Zugriff

Id: *
test_mailbox

Applikation: *
ITSM

Server: *
mail.server.com

Benutzer: *
itsmtest

Passwort:

Autorisierer:

E-Mail Adresse: *
itsmtest@company.com

Ordner:

Mail Protokoll:
IMAP

Art der Kommunikation:
Unverschlüsselt

Mit Timer abrufen:
☒

SMTP Host:

Art der SMTP Kommunikation:
Unverschlüsselt

Beschreibung:

Mails downloaden

Löschen
Speichern und Schließen
Speichern
Abbrechen

Abbildung 6.43: Objekt-Detailansicht: Mailbox

6.12.1 Registerkarte: Allgemein

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Id:** Jede Mailbox muss eine eindeutige ID besitzen.
- **Applikation:** Zuordnung der Mailbox zu einer Applikation. Aufgrund der Zuordnung wird diese auch in der Mailboxen-Tabelle der jeweiligen Applikation gelistet.
- **Server:** Mail-Server in der Form Hostname oder Hostname:Port).
- **Benutzer:** Benutzer der Mailbox

- **Passwort:** Das Passwort für die Mailbox. Bleibt leer wenn ein Autorisierer verwendet wird.
 - **Autorisierer:** E-Mail Autorisierer. Mehr Informationen über Autorisierer finden Sie im [Installationshandbuch](#) im Kapitel [OAuth 2.0 authentication](#).
 - **E-Mail Adresse:** Die E-Mail Adresse der Mailbox - diese Adresse wird auch bei Verwendung des E-Mail Tabs bei Prozessen (Definition des Tabs siehe Abschnitt [Registerkarte: Allgemein](#)) herangezogen.
 - **Ordner:** Ordner, auf den zugegriffen wird
 - **Mail Protokoll:** IMAP oder POP3
 - **Art der Kommunikation:** Es gibt 3 Arten:
 - Unverschlüsselt (plain): Die Kommunikation ist unverschlüsselt.
 - Verschlüsselt (encrypted): Die Kommunikation ist verschlüsselt.
 - Mit Zertifikat (trusted): Die Kommunikation ist SSL verschlüsselt mit MailServer-Authentifizierung. Hierzu muss das Zertifikat des Mail Server als vertrauenswürdigen Zertifikat in den Keystore importiert werden. (Siehe Kapitel [Zertifikatsmanagement](#))
 - **Mit Timer abrufen:** MailTimer liest diese Mailbox und führt Aktion automatisch durch. Für das Senden von E-Mails über die Mailbox kann jedoch die entsprechende SMTP Einstellung gemacht werden, sonst wird diejenige aus der Administration unter *Konfiguration/Kommunikation* herangezogen.
 - **SMTP Host:** Server für die ausgehende E-Mails in der Form Hostname oder Hostname:Port).
 - **Art der SMTP Kommunikation:** Es gibt 4 Arten:
 - Unverschlüsselt (plain): Die Kommunikation ist unverschlüsselt.
 - Verschlüsselt (encrypted): Die Kommunikation ist verschlüsselt.
 - STARTTLS (encrypted): Die Kommunikation ist mittels Transport Layer Security (TLS) verschlüsselt.
 - Mit Zertifikat (trusted): Gleich wie bei *Art der Kommunikation*.
- Hinweis:** Neben diesem Feld ist die Funktion mit der ein E-Mail an die E-Mail-Adresse des Administrators gesendet wird.
- **Beschreibung:** Freier Text.

Wenn Sie die Schaltfläche **Mails downloaden** aktivieren, wird die definierte Aktion auf alle eingehenden E-Mails sofort ausgeführt. Es werden immer alle E-Mails einer Mailbox behandelt und bei erfolgreicher Bearbeitung einer Nachricht die definierte Aktion ausgeführt. Bei E-Mails, die aufgrund eines Fehlers nicht korrekt behandelt werden können, wird nach Abarbeitung aller (erfolgreichen/nicht erfolgreichen) E-Mails eine Fehlermeldung ausgegeben und die Information der fehlerhaften E-Mails ins Serverlog geschrieben.

6.12.2 Registerkarte: Aktion

Diese Registerkarte teilt sich in den Bereich *Allgemein* und den Aktionsdetails wobei Sie folgende Einstellungen tätigen können:

- **Auf diese Absender einschränken:** Dieses Feld dient dazu, dass die ausgewählte Aktion nur bei den eingetragenen Absendern ausgeführt wird. Falls nichts eingetragen wird, wird die Aktion immer ausgeführt, vorausgesetzt der Junk-Filter unter *Konfiguration/Kommunikation* greift nicht. Ansonstent wird das E-Mail vom Server unbehandelt entfernt. In dieses Feld können E-Mail Adressen bzw. ein Pattern angegeben werden. Das Trennzeichen ist eine neue Zeile. Beispiele:
 - **groiss** - Wenn E-Mail-Adresse den String 'groiss' beinhaltet, dann wird Aktion ausgeführt
 - **groiss.com* - Wenn E-Mail-Adresse den String 'groiss.com' zum Schluss beinhaltet, dann wird Aktion ausgeführt
 - *max.muster@** - Wenn E-Mail-Adresse den String 'max.muster@' am Anfang beinhaltet, dann wird Aktion ausgeführt
 - *max.muster@groiss.com* - Wenn E-Mail-Adresse genau dem String 'max.muster@groiss.com' entspricht, dann wird Aktion ausgeführt
- **Aktion:** Hier definieren Sie, welche Aktion beim Eingang eines E-Mails ausgeführt werden soll:
 - *Prozess starten:* Wenn diese Aktion ausgewählt wurde, wird der angegebene *Prozess* in der angegebenen *OE* gestartet und das E-Mail an den Prozess angehängt, falls es zuvor keine automatische Zuordnung über das *Betreff Pattern* gab (siehe [Installationshandbuch](#), Abschnitt [Communication](#)). Wenn ein *Empfangsbestätigungstext* angegeben wurde, wird dieser beim Eingang des E-Mails automatisch an den Absender versandt (nur beim Starten eines neuen Prozesses).
 - *Andere Aktion:* Eine Java Klasse, die das Interface `com.groiss.mail.MailHandler` implementiert, siehe entsprechendes Beispiel im Programmierhandbuch. Mit der Checkbox *Automatisch an Prozess zuweisen* können Sie definieren, ob das E-Mail aufgrund des definierten *Betreff Pattern* (siehe [Installationshandbuch](#), Abschnitt [Communication](#)) automatisch zu einem Prozess zugeordnet werden soll bevor die definierte Aktion ausgeführt wird.
 - *Keine*

6.12.3 Registerkarte: Mailbox anzeigen

Beim Wechseln auf diese Registerkarte wird der Inhalt der Mailbox angezeigt.

6.13 Web-Dienste

Dieser Abschnitt behandelt die Erstellung von Webservice Server/Client Objekten, die beim Prozesseditor in einem der 3 Web-Dienste-Knoten definiert werden können (siehe Kapitel [Eigenschaften eines Web-Dienste-Knoten definieren](#)). Um überhaupt Webservice Server/Client Objekte erzeugen zu können, müssen entsprechende WSDL-Dateien im Klassenpfad von *EPworX* vorhanden sein.

Zu beachten ist, dass die WSDL-Dateien dem [WS-I Basic Profile 1.1](#) entsprechen müssen. Webservice Server/Client Objekte können pro Applikation definiert werden.

6.13.1 Webservice Clients

Mit Hilfe der Client Objekte kann definiert werden, welcher Web-Dienst mit welchen Parametern aufgerufen wird (OUT-Parameter) bzw. welche Parameter vom Web-Dienst erwartet werden (IN-Parameter). Client Objekte werden dazu benötigt, um Daten für einen anderen Server, für eine weitere Verarbeitung, bereitzustellen.

Die Objekt-Detailansicht gliedert sich in folgende Registerkarten:

- Allgemein
- Operationen
- Historie
- Zugriff

Registerkarte: Allgemein

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Id:** Die ID des Webservice Client Objekts, die frei vergeben werden kann. Die Id muss pro Applikation eindeutig sein.
- **WSDL-Datei:** Der Pfad zu einer im Klassenpfad befindlichen WSDL-Datei.
- **Webservice:** Hier wird eine Auswahl der Webservices angeboten, die in der WSDL-Datei definiert wurden. Eine Auswahl ist nur möglich, wenn zuvor der korrekte Pfad auf eine *WSDL-Datei* angegeben wurde.
- **Port:** Abhängig von dem ausgewählten *Webservice* kann hier der entsprechende Port ausgewählt werden, der in der WSDL-Datei definiert wurde.
- **URL:** Die URL des Web-Dienstes, der aufgerufen werden soll. Wenn keine URL angegeben wurde, wird jene aus der WSDL-Datei verwendet.
- **Benötigte Module:** Falls erforderlich, kann hier eine durch Kommas getrennte Liste von SOAPWebServiceModuleProvidern eingetragen werden. Dies ermöglicht das Hinzufügen benutzerdefinierter Interceptors, Features und Eigenschaften zum Webservice.

The screenshot shows a software interface with a tabbed menu at the top: 'Allgemein' (selected), 'Operationen', 'Historie', and 'Zugriff'. Below the tabs is a form with the following fields and values:

- Id: ***: tempconvert
- WSDL Datei: ***: tempconvert.wsdl
- Webservice: ***: TempConvert
- Port: ***: TempConvertSoap12
- URL:**: https://www.w3schools.com/xml/tempconvert.asmx
- Benötigte Module:**: (empty)
- Applikation: ***: Test

At the bottom of the form are four buttons: 'Löschen', 'Speichern und Schließen', 'Speichern', and 'Abbrechen'.

Abbildung 6.44: Registerkarte: Allgemein

- **Applikation:** Die Applikation, in der das Client Objekt gespeichert werden soll. Standardmäßig ist jene Applikation vorausgewählt, von der aus die Toolbarfunktion *Neu* aufgerufen wurde.

Nachdem die Informationen in der Registerkarte *Allgemein* gespeichert wurden, muss ein *Operation*-Objekt in der Registerkarte *Operationen* erzeugt werden. In diesem *Operation*-Objekt können dann die Ein- bzw. Ausgangsparameter festgelegt werden (IN-/OUT-Parameter).

Registerkarte: Operationen

In dieser Registerkarte wird eine Tabelle aller Operationen zu diesem Client Objekt angezeigt. Neben den Standard-Toolbarfunktionen gibt es noch eine Funktion *Webservice-Operation aufrufen*, mit der man die Web-Dienst Operation testweise mit den entsprechenden Ausgangsparameter aufrufen kann.

Um eine Operation zu erstellen, aktivieren Sie die Toolbarfunktion *Neu*. Anschließend wird ein neuer Dialog geöffnet, wo man eine *Operation* (definiert in der WSDL-Datei) auswählen kann (siehe Abbildung 6.45). Anschließend kann mittels der Funktion *XML generieren* ein XML generiert werden, das im gleichnamigen Feld gespeichert wird. Nachdem ein *Operation*-Objekt angelegt wurde, können die Ein- bzw. Ausgangsparameter in den entsprechenden Registerkarten angegeben werden. Beim Client Objekt

Abbildung 6.45: **Das Operation-Objekt**

sind Ausgangsparameter (= OUT-Parameter) jene, die dem Web-Dienst zur Verarbeitung (beim anderen Server) übermittelt werden. Eingangsparameter (= IN-Parameter) sind jene, die vom Web-Dienst erwartet werden (z.B. Statusmeldung über Verarbeitung). Ein Parameter wird definiert durch eine *Id*, einen *Namen* und einen *Pfad (XPath)*, wobei alle 3 Felder Mussfelder sind. Als Namespace-Prefix im XPath-Ausdruck können alle Prefixes verwendet werden, welche im Root-Element der WSDL-Datei definiert sind. Parameter können entweder händisch mittels der Toolbarfunktion *Neu* erzeugt werden oder sie werden automatisch mittels der Funktion *Parameter generieren* erzeugt. Es ist dabei nicht möglich Parameter mit gleicher ID mehrfach anzulegen.

6.13.2 Webservice Server

Mit Hilfe der Server Objekte können Webservice am Server zur Verfügung gestellt werden. Andere Systeme können diese aufrufen.

Die Objekt-Detailansicht gliedert sich in folgende Registerkarten:

- Allgemein
- Operationen
- Historie
- Zugriff

Die Objekt-Detailansicht für *Allgemein* und *Operationen* Registerkarten:

- Allgemein: Beinhaltet die selben Felder wie *Webservice Clients*, nur mit dem Unterschied, dass keine URL angegeben werden kann.
- Operationen: Analog zu *Webservice Clients*, jedoch mit dem Unterschied, dass die Funktion *Webservice-Operation aufrufen* nicht verfügbar ist. Als Eingangsparameter (= IN-Parameter) werden in diesem Fall jene bezeichnet, die vom Web-Dienst zur Verarbeitung empfangen werden und Ausgangsparameter (= OUT-Parameter) als jene, die dem Web-Dienst übermittelt werden (z.B. Statusmeldung über Verarbeitung). Optional kann noch ein *Nachrichten-Handler* angegeben werden, der das Interface *com.groiss.ws.server.MessageHandler* implementieren muss. Wenn ein *Nachrichten-Handler* eingetragen wurde, kann diese Operation nicht mehr in einem Prozess verwendet werden.

6.14 Nachrichten-Vorlagen

In diesem Abschnitt wird die Definition von Nachrichten-Vorlagen beschrieben, die in *EPworX* an zahlreichen Stellen verwendet werden können. Folgende Ereignisse verwenden Nachrichten-Vorlagen:

- Prozessablauf (Systemschritt, Preprocessing, Postcondition, etc.)
- Eskalation
- Benachrichtigung über neue Prozesse im Arbeitskorb (Notification)
- Benachrichtigung bei Prozessen, die verfolgt werden (Funktion *Verfolge den Prozess*)
- Timer PasswordExpiration
- ReportTimer
- Änderung in einem DMS-Dokument (definierbar über DMS-Funktion *Verfolge Dokumentenänderungen*)
- API-Programm

EPworX bietet für die meisten dieser Ereignisse bereits Standard-Vorlagen, die in der Applikation *default* gespeichert sind und nicht gelöscht werden können. Neben den Standard-Vorlagen ist es aber auch möglich pro Applikation eigene Vorlagen zu definieren. Folgende Standard-Vorlagen sind verfügbar:

- **Standard-Vorlage (documentTracker):** Diese Vorlage kann bei der DMS-Funktion *Verfolge Dokumentenänderung* ausgewählt werden. Nähere Informationen zu dieser Funktion entnehmen Sie bitte dem [Benutzerhandbuch](#).
- **Eskalation (escalation):** Eskalations-E-Mails verwenden standardmäßig diese Vorlage wobei es egal ist, ob diese beim Eskalationsobjekt ausgewählt wurde oder auch nicht. Nähere Informationen sind im Abschnitt [Registerkarte: Eskalationen](#) vorhanden.

- **Benachrichtigung (notification):** Benachrichtigungen über neue Arbeitskorb-Einträge verwenden standardmäßig diese Vorlage. Nähere Informationen dazu finden Sie im [Benutzerhandbuch](#) unter dem Stichwort *E-Mail-Benachrichtigung*.
- **Prozessverfolgung-Vorlage (processTracker):** Diese Vorlage wird von der Arbeitskorb-Funktion *Verfolge den Prozess* standardmäßig verwendet. Nähere Informationen zu dieser Funktion entnehmen Sie bitte dem [Benutzerhandbuch](#).
- **Vorlage für Warnung über Ablauf des Passworts (pwdExpiration):** Der Timer *PasswordExpiration* verwendet diese Vorlage um ein E-Mail an die Benutzer zu versenden, deren Passwort in der eingestellten Frist ablaufen.
- **Report-Timer-Vorlage (reportTimer):** Diese Vorlage wird standardmäßig vom ReportTimer verwendet, wenn ein E-Mail versendet wird. Nähere Informationen zum ReportTimer entnehmen Sie bitte aus dem Abschnitt [Übersicht über Standard-Timer](#).
- **Benachrichtigung zu empfangener E-Mail (incomingMailNotification):** Diese Vorlage kann man bei einem Prozess benutzen, um den Benutzer zu informieren, dass ein neues E-Mail dem gegebenen Prozess hinzugefügt wurden.

Eine Beschreibung zum API ist im [Programmierhandbuch](#) im Abschnitt [E-Mails](#) vorhanden.

Hinweis: Eine Übersicht über die Versandarten und welche Vorlagen wie verwendet werden ist im Abschnitt [Übersicht über Ereignisse und Versandarten](#) zu finden!

6.14.1 Registerkarte: Allgemein

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Id:** Jede Nachrichten-Vorlage, die in der Administration von *EPworX* definiert wird, benötigt eine Id, die frei vergeben werden kann. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#).
- **Name:** Der Name der Nachrichten-Vorlage. Genauere Informationen zu Ids und Namen finden Sie im Kapitel [Ids und Namen](#). Der I18n-Link neben diesem Feld ermöglicht direkt die Anpassung der Ressource (falls in Applikationsdetailmaske - Registerkarte *Eigenschaften* festgelegt). Die Änderungen werden nach Aktivieren der Schaltfläche *Speichern* in der Ressource Datei (Strings) dieser Applikation gespeichert (siehe Abschnitt [Ressourcen Editor](#)).
- Beschreibung: Freier Text, der die Nachrichten-Vorlage näher beschreibt.
- Applikation: Gibt an, zu welcher Applikation die Vorlage gehört.
- Aktiv: Mit Hilfe dieser Checkbox geben Sie an, ob die Vorlage aktiv/inaktiv ist. Aktive Vorlagen sind bei den diversen Auswahlfeldern auswählbar und werden für den E-Mail-Versand verwendet. Inaktive Vorlagen sind weder auswählbar noch werden sie für den E-Mail-Versand herangezogen.

The screenshot shows a web-based form for creating a message template. The form is organized into a tabbed interface with the following sections:

- General (Allgemein):**
 - Id:** notification
 - Name:** notification
 - Beschreibung:** (empty text area)
 - Applikation:** Default
 - Aktiv:** ☒
 - Mime-type:** HTML
 - Empfänger:** (empty list with add and remove icons)
 - Absender:** (empty text field)
 - Reply-To:** (empty text field)
 - Zum Prozess hinzufügen:** ☐ **Nachricht in Journal protokollieren:** ☐
 - Betreff:** \${notificationSubject}
 - URL zum Nachrichteninhalt:** mask/messages/Notification.xhtml
 - Content Editor:** A rich text editor with 'HTML' and 'Text/Quellcode' tabs. The 'HTML' tab is active, showing a toolbar with bold, italic, underline, list, link, and image icons.
- Buttons:**
 - Löschen** (Delete)
 - Speichern und Schließen** (Save and Close)
 - Speichern** (Save)
 - Abbrechen** (Cancel)

Abbildung 6.46: **Nachrichten-Vorlage**

- **Mime-type:** Mit dieser Auswahl geben Sie an, in welchem Format die Nachricht versendet werden soll. Ist der Wert *HTML* ausgewählt, so können Sie ganz unten im Reiter *HTML* einen HTML-formattierten Text eingeben, der bei der Nachricht verwendet wird. Ist der Wert *Text* ausgewählt, ist nur der Reiter *Text/Quellcode* aktiv, wo Sie einen PLAIN-Text angeben können. Beim E-Mail-Versand wird bei Verwendung von *HTML* ein HTML-Bodypart (text/html) erzeugt, bei *Text* nur ein PLAIN-Bodypart (text/plain).
- **Empfänger:** In der Empfängerliste können Sie mehrere To-, CC- und BCC-Empfänger mit den entsprechenden Funktionen hinzufügen bzw. entfernen. Folgende Empfänger-Typen können definiert werden:
 - **Benutzer:** Hier können Sie einen *EPworX* Benutzer auswählen. Sofern der Benutzer eine E-Mail-Adresse eingetragen hat, wird ein E-Mail an diese

Adresse versendet, ansonsten wird dieser Empfänger ignoriert.

- Rolle: Hier können Sie eine *EPworX* Rolle und Organisationseinheit auswählen. Alle Benutzer, die dieser Rolle (in dieser Organisationseinheit) angehören, bekommen ein E-Mail (sofern beim Benutzer eine E-Mail-Adresse spezifiziert wurde).
 - E-Mail: Hier können Sie eine gültige E-Mail-Adresse eintragen.
 - Akteur eines Prozessschritts: Hier können Sie den Label eines Prozessschritts angeben. Der Label ist dabei die Id eines Schrittes innerhalb der Prozessdefinition. Abhängig vom Akteur (Benutzer oder Rolle) wird an diese(n) ein E-Mail versendet.
 - Formularfeld: Hier können Sie in WDL-Syntax ein Formularfeld eines Prozessformulars angeben. Die Syntax ist dabei `<formid>.<fieldid>` (`<formid>`=Id des lokalen Formulars in der Prozessdefinition). Nähere Informationen zur WDL-Syntax finden Sie im Abschnitt [Prozessdefinition](#).
 - Aktuelle Akteure des Prozesses: Bei diesem Empfänger-Typ wird abhängig von den derzeit aktiven Akteuren (Benutzer oder Rolle) eines Prozesses ein E-Mail an diese versendet (auch an die Vertreter der aktuellen Akteure!).
 - Dokumenten-Erzeuger: Bei diesem Empfänger-Typ wird eine Nachricht an den Akteur versendet, der das Objekt erzeugt hat. Dieser Empfänger wird nur bei DMS-Objekten ermittelt! Ein Beispiel dafür wäre, wenn eine Dokumenten-Verfolgung für ein Dokument eingerichtet wurde und das Dokument von einem anderen Benutzer verändert wird, dann erhält immer der Erzeuger eine Nachricht, dass sich das Dokument geändert hat.
- Absender: Hier können Sie eine beliebige Absender-Adresse eintragen. Bleibt dieses Feld leer, werden die Standard-Einstellungen aus der Konfiguration genommen (siehe Konfiguration/Kommunikation/Mail Sender).
 - Reply-To: Eine oder mehrere durch Komma getrennte E-Mail-Adressen können hier angegeben werden, an die eine Antwort auf die E-Mail geschickt werden soll (falls diese unterschiedlich zum Absender sein soll).
 - Zum Prozess hinzufügen: Ist diese Checkbox aktiv und die Vorlage wird im Kontext eines Prozesses verwendet, dann wird ein entsprechendes E-Mail-Objekt im Ordner *Emails* der Prozessinstanz gespeichert. Nähere Informationen dazu finden Sie im Benutzerhandbuch im Abschnitt *E-Mails*. Falls die Vorlage nicht im Kontext eines Prozesses verwendet wird, hat diese Checkbox keinerlei Auswirkung.
 - Nachricht in Journal protokollieren: Mit diese Checkbox können Sie festlegen, ob ausgehende Nachrichten im *Mail Journal* gespeichert werden sollen. Ein Eintrag im *Mail Journal* wird erst erzeugt, wenn die Nachricht versendet werden konnte. Nähere Informationen zum *Mail Journal* finden Sie im Abschnitt [Mail Journal](#).
 - Betreff: Hier können Sie einen Betreff für die Nachricht definieren. Der Betreff kann

- ein einfacher Text sein,
 - Platzhalter besitzen, die durch ein Ressourcen-Bundle ersetzt werden (siehe Abschnitt [Ressourcen Editor](#)) oder
 - auch XPath-Ausdruck beinhalten, um z.B. auf Werte aus einem Formular zuzugreifen. Nähere Informationen zu XPath-Ausdrücke sind im [Programmierhandbuch](#) zu finden (z.B. im Kapitel [Office templates](#)). Alternativ können Sie sich die Verwendung auch in den Standard-Vorlagen ansehen.
- URL zum Nachrichteninhalt: Alternativ zu einem Nachrichten-Text können Sie auch eine Pfad zu einer HTML-Maske angeben, die für den Versand verwendet werden soll. Die URL ist relativ zum classes-Verzeichnis (von *EPworX* oder der Applikation) anzugeben. Somit haben Sie die Möglichkeit komplexere Masken zu definieren, die beim E-Mail-Versand als HTML-Bodypart (text/html) hinzugefügt werden. Auch hier können Platzhalter für Übersetzungen und XPath-Ausdrücke definiert werden. Wenn eine URL eingetragen wurde, kann kein Nachrichten-Text in HTML bzw. Text/Quellcode erzeugt werden und somit wird auch der Wert von *Mime-type* nicht ausgewertet.
 - Nachrichten-Text (HTML oder Text/Quellcode): Analog zum Betreff kann hier ein Nachrichten-Text angegeben werden (= Bodypart). Ist der *Mime-type* auf den Wert *Text* gesetzt, so ist der Reiter *HTML* inaktiv. Wurde eine *URL zum Nachrichteninhalt* angegeben, sind beide Reiter inaktiv.

6.14.2 Übersicht über Ereignisse und Versandarten

Die nachfolgende Tabelle soll einen Überblick über das Zusammenspiel von Vorlagen und Ereignissen geben:

6.15 Testfälle

Dieser Abschnitt beschreibt die Definition von Testfälle für bestehende Prozesse, mit denen die Funktionsfähigkeit der erstellten Prozessdefinition getestet werden kann. Mit Hilfe der Toolbarfunktion *Neu* können Sie für einen ausgewählten Prozess Testschritte definieren und diese anschließend ausführen lassen. Bevor aber Testschritte erzeugt werden können, benötigt der Testfall zusätzliche Daten:

- **Name:** Frei wählbarer Name des Testfalls.
- **Prozess:** Prozessdefinition, für die Testschritte erstellt werden sollen. Es werden nur Prozessdefinitionen der aktuellen Applikation angeboten, in dessen Kontext der Testfall erzeugt wird.
- **Beschreibung:** Freier, optionaler Text zum beschreiben des Testfalls.
- **Startakteur:** Hier können Sie einen Benutzer auswählen, der immer als Startakteur des Prozesses fungiert. Wird hier nichts ausgewählt, so ist der Startakteur jener gemäß der Prozessdefinition.

6. Workflowelemente

Definition der Vorlage	Standard-Vorlage (Id)	Akteure	Versandart	Sprache
Benachrichtigung über neue Prozesse im Arbeitskorb (Notification):				
Auswahl der Vorlage pro Prozessdefinition	notification	Werden im Code zur Laufzeit gesetzt	Abhängig von Konfigurationsparameter <i>Default Aktion bei Mailversand</i>	Abhängig von Benutzer-Lokale
Eskalation:				
Auswahl der Vorlage pro Eskalation	escalation	Werden im Code zur Laufzeit gesetzt	Abhängig von Konfigurationsparameter <i>Default Aktion bei Mailversand</i>	Default-Lokale von <i>EPworX</i>
Benachrichtigung bei Prozessen, die verfolgt werden:				
fix	processTracker	Benutzer, der ein Verfolgungsobjekt angelegt hat (Funktion <i>Verfolge den Prozess</i>)	Abhängig von Konfigurationsparameter <i>Default Aktion bei Mailversand</i>	Benutzer-Lokale
ReportTimer:				
Im Timer-Feld <i>Parameter</i> setzbar	reportTimer	Gemäß Definition in Vorlage oder aus Timer-Feld <i>Parameter</i>	Abhängig von Konfigurationsparameter <i>Default Aktion bei Mailversand</i>	Default-Lokale von <i>EPworX</i>
Systemschritt:				
Im Methodenaufruf	keine	Gemäß Definition in Vorlage oder im Java-Code setzbar	Mail-Queue Eintrag wird erstellt und durch MailQueueTimer versendet	Default-Lokale von <i>EPworX</i>
Änderung in einem DMS-Dokument:				
Auswahl der Vorlage oder Ad-Hoc Vorlage	documentTracker	Gemäß Definition in Vorlage oder der Empfängerliste bei DMS-Funktion <i>Verfolge Dokumentenänderungen</i>	Mail-Queue Eintrag wird erstellt und durch MailQueueTimer versendet	Default-Lokale von <i>EPworX</i>

Tabelle 6.2: Übersicht über Ereignisse und Versandarten

Nachdem diese Daten vollständig angegeben wurden, können Testschritte im Hauptfenster (Prozess-Debugger) erzeugt werden, das in mehrere Bereiche aufgegliedert ist (siehe Abbildung 6.47). Mit Hilfe der Schaltfläche *Schließen* können Sie das Hauptfenster beenden. In den folgenden Abschnitten werden die einzelnen Bereiche des Hauptfensters näher beschrieben.

Hinweis: Damit die Testfälle ausgeführt werden können, muss in der Konfigurationsdatei *ep.conf* die Ausführung von GROOVY-Skripts mit dem Parameter *ep.scripts.enable=true* aktiviert werden!

6. Workflowelemente

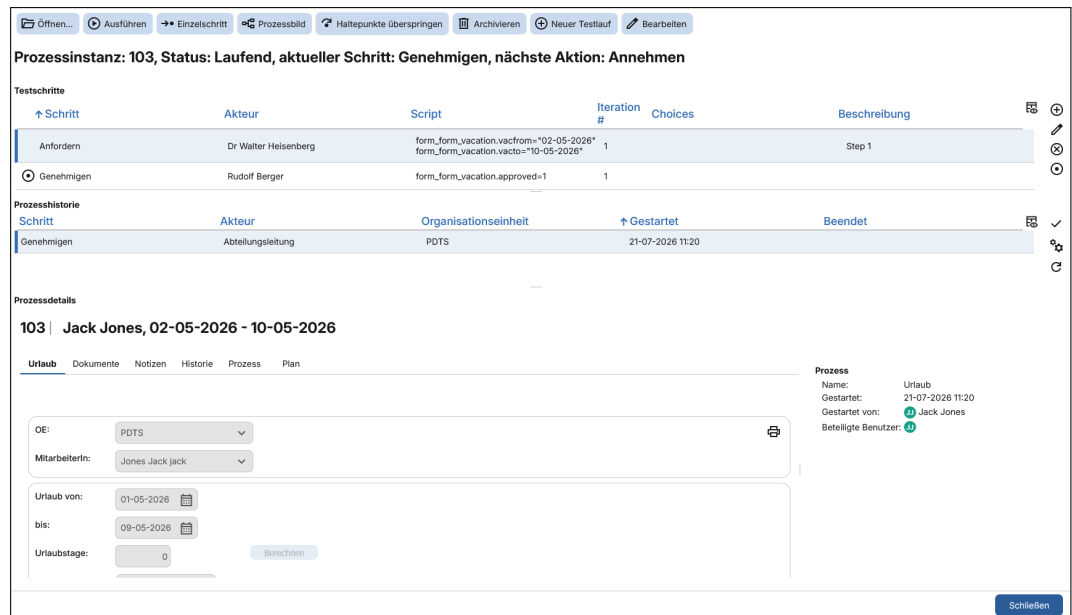


Abbildung 6.47: Prozess-Debugger (Testfall)

6.15.1 Toolbar

Der Toolbar beinhaltet diverse Funktionen zur Ausführung und Bearbeitung eines Testfalls:



- **Öffnen:** Durch Aktivieren dieser Toolbarfunktion wird ein neuer Dialog geöffnet, wo eine bereits bestehende Prozessinstanz geladen werden kann. Diese Prozessinstanz kann anschließend aufgrund der festgelegten Testschritte abgearbeitet werden.



- **Ausführen:** Mit Hilfe dieser Schaltfläche wird eine neue Prozessinstanz gestartet bzw. eine bestehende fortgeführt und aufgrund der definierten Testschritte automatisch durchgespielt. Wurde für einen Testschritt ein Haltepunkt definiert, wird bei diesem Schritt die Ausführung angehalten. Durch nochmaliges Aktivieren der Schaltfläche *Ausführen* werden die nachfolgenden Testschritte bis zum Ende bzw. nächsten Haltepunkt abgearbeitet.



- **Einzelschritt:** Alternativ zur Funktion *Ausführen* können Sie auch jeden Testschritt einzeln durchspielen, entweder von Beginn an oder ab einem gewissen Haltepunkt.



- **Prozessbild:** Durch Aktivieren dieser Funktion wird ein neues Fenster mit dem Prozessbild geöffnet. Dabei werden neben dem Hauptprozess auch die Subprozesse in eigenen Reitern angezeigt. Wird die Funktion *Einzelschritt* ausgeführt und ist das Prozessbild-Fenster offen, so wird der gerade ausgeführte Schritt im Prozess umrandet dargestellt.



- **Haltepunkte überspringen:** Ist diese Funktion aktiviert, werden definierte Haltepunkte beim Klick auf die Funktion *Ausführen* übersprungen.



- **Archivieren:** Da beim Testdurchlauf eine Prozessinstanz in der Datenbank angelegt wird, können Sie mit Hilfe dieser Funktion diese Instanz wieder aus dem System entfernen.



- **Neuer Testlauf:** Mit Hilfe dieser Funktion können Sie einen neuen Testlauf starten, egal ob der bestehende bereits fertig durchlaufen wurde oder nicht. Im darauffolgenden Bestätigungsdialog werden Sie gefragt, ob die durch den Testlauf erzeugte Prozessinstanz auch archiviert werden soll.



- **Bearbeiten:** Durch diese Funktion öffnet sich der Detail-Dialog eines Testfalls, wo *Name*, *Prozess*, *Beschreibung* und *Startakteur* geändert werden können. Die Bedeutung dieser Felder wurde bereits im Abschnitt [Testfälle](#) beschrieben.

Unterhalb des Toolbars befindet sich eine Informationsleiste, die folgende Elemente beinhaltet:

- **Prozessinstanz:** Die Id der Prozessinstanz wird hier angezeigt, die in diesem Testlauf verwendet wird.
- **Status:** Der Status der Prozessinstanz.
- **aktueller Schritt:** Der Prozessschritt, der gerade ausgeführt wird.
- **nächste Aktion:** Die nachfolgende Aktion, die ausgeführt werden soll.

6.15.2 Testschritte



Testschritte werden über die entsprechenden Toolbarfunktionen erstellt, bearbeitet und entfernt. Mit Hilfe der Toolbarfunktion *Haltepunkt Ein/Aus* können Sie für den ausgewählten Testschritt-Eintrag einen Haltepunkt setzen bzw. wieder entfernen. Die Darstellung eines Haltepunktes erfolgt als Kugel am Rande eines Eintrags. Ist ein Haltepunkt definiert, wird bei diesem Schritt die Ausführung angehalten (siehe dazu auch Toolbarfunktion *Ausführen*).

Ein Testschritt besteht aus folgenden Elementen (siehe Abbildung [6.48](#)):

- **Schritt:** Hier definieren Sie den Prozessschritt, für den der Testschritt ausgeführt werden soll. Es werden nur Prozessschritte der festgelegten Prozessdefinition bzw. die Schritte der Subprozesse angeboten.
- **Akteur:** Hier können Sie einen Benutzer auswählen, der immer als Schrittakeur fungiert. Wird hier nichts ausgewählt, so ist der Schrittakeur jener gemäß der Prozessdefinition.
- **Script:** In diesem Feld können Sie ein GROOVY-Skript zum Setzen von Formularwerte angeben. Die Werte werden automatisch während des Testlaufes gespeichert, d.h. man muss sich im Testschritt nicht explizit darum kümmern! Formularwerte werden mit Hilfe folgender Syntax gesetzt:

Testschritt

×

Schritt: *

Anfordern

▼

Akteur:

Dr Walter Heisenberg

▼

Script:

```
form_form_vacation.vacfrom="02-05-2026"
form_form_vacation.vacto="10-05-2026"
```

Choices:

Haltepunkt:

☐

Iteration #:

1

▲

▼

Beschreibung:

Step 1

Ok

Abbrechen

Abbildung 6.48: **Testschritt**

```

form_<formid1_in_procdef>.<field1>=<value1>
form_<formid1_in_procdef>.<field2>=<value2>
form_<formid2_in_procdef>.<field3>=<value3>
...

```

Die Formular-Id entspricht der Id, die bei der Prozessdefinition für das Prozessformular vergeben wurde. Der Wert (value) ist entweder ein String (unter Anführungszeichen gestellt) oder eine Zahl. Bei einem Datumsfeld gibt man den

Wert als String gemäß dem Datumsformat in der *EPworX* Konfiguration an (= Datumsformat, welches auch vom Datepicker verwendet wird). Bei Persistent-Feldern (z.B. ObjectSelects) kann entweder die Id (z.B. Benutzer-Id) angegeben werden oder auch `<classname>:<oid>`, beides unter Anführungszeichen gestellt. Ist das Persistent ein anderes *EPworX* Formular, dann muss entweder mit `store.get()` zuerst das Persistent geladen (und als Wert zugewiesen) werden oder man gibt auch hier `<classname>:<oid>` an.

Für die Manipulation von Subformulare gibt es folgende Möglichkeiten:

- Hinzufügen eines Subformulars: Mit Hilfe der Methode `addSubform()` können neue Subformular-Einträge hinzugefügt werden. Der Parameter `<subformid>` gibt an, welches Subformular betroffen ist und entspricht dem numerischen Wert, der im Formulartemplate (XHTML, XForm) angegeben wurde.

Syntax:

```
form_<formid_in_procdef>.addSubform(<subformid>)
```

- Subformularwerte anpassen: Um Subformularwerte ändern zu können, wird folgende Syntax angewendet, wobei `<subformid>` die numerische Id des Subformulars ist, die im Formulartemplate definiert wurde und `<subformpos>` die Position des Eintrags in der Subformulartabelle angibt, beginnend mit 0.

Syntax:

```
form_<formid_in_procdef>[<subformid>][<subformpos>].<field>=<value>
```

- Subformular entfernen: Mit Hilfe der Methode `deleteSubform()` wird ein Subformular-Eintrag entfernt. Der Parameter `<subformid>` gibt an, welches Subformular betroffen ist und entspricht dem numerischen Wert, der im Formulartemplate (XHTML, XForm) angegeben wurde. Der Parameter `<subformpos>` gibt die Position des Eintrags in der Subformulartabelle an, beginnend mit 0.

Syntax:

```
form_<formid_in_procdef>.deleteSubform(<subformid>,<subformpos>)
```

- Choices: Mit Hilfe dieses Feldes können Sie den entsprechenden Choice-Zweig festlegen, der im Testlauf ausgeführt werden soll, sofern in der Prozessdefinition Choice-Konstrukte verwendet werden. Der Choice-Knoten (`<choice_node_label>`) und die einzelnen Choice-Zweige (`<choice_branch_label>`) müssen einen Label besitzen, der mittels Prozesseditor definiert werden kann. Die Definition des Choice-Zweiges erfolgt immer im vorhergehenden Testschritt mit folgender Syntax:

```
<choice_node_id>=<choice_branch_id>
```


- **Haltepunkt:** Ist diese Checkbox aktiviert, wird für den aktuellen Testschritt ein Haltepunkt festgelegt. Ist ein Haltepunkt definiert, wird bei diesem Schritt die Ausführung angehalten (siehe dazu auch Toolbarfunktion *Ausführen*).
- **Iteration #:** Gibt es mehrere Testschritte mit dem gleichen Prozessschritt, muss hier die entsprechende Iterationsnummer ausgewählt werden, beginnend mit 1. Dies kann z.B. nötig sein, wenn ein Schleifenkonstrukt in der Prozessdefinition definiert wurde und ein Prozessschritt mehrmals ausgeführt werden soll oder auch bei Parfor-Schritten.
- **Beschreibung:** Frei wählbarer Text, der den Testschritt beschreibt.

Mit Hilfe der Schaltfläche *Ok* wird ein Testschritt angelegt und der Tabelle *Testschritte* hinzugefügt. Durch die Schaltfläche *Abbrechen* werden alle Änderungen verworfen und der Dialog *Testschritt* geschlossen.

6.15.3 Prozesshistorie und Prozessdetails

Die Prozesshistorie stellt die Detailansicht jeder einzelnen Prozessinstanz dar. Sie zeigt sämtliche Schritte (Ablauf der Tasks), die ein ausgewählter Prozess bereits durchlaufen hat. Die Prozesshistorie-Einträge werden durch das Abhandeln der Testschritte erzeugt (Toolbarfunktionen *Ausführen* oder *Einzelschritt*). Die Bereiche Prozesshistorie und Prozessdetails korrelieren miteinander, d.h. durch Selektion eines Prozesshistorie-Eintrags werden die entsprechenden Prozessdetails geladen. Zusätzlich gibt es noch weitere Anzeigoptionen für die Prozesshistorie-Tabelle, die im entsprechenden Toolbar aktiviert werden können:



- **Beendete Schritte anzeigen:** Initial werden in der Prozesshistorie nur aktive Schritte angezeigt. Durch Aktivieren dieser Funktion werden auch alle abgeschlossenen Schritte angezeigt.



- **Systemschritte anzeigen:** Mit dem Aktivieren dieser Funktion werden alle Systemschritte, also auch Schritte, die vom WFMS automatisch ausgeführt werden, dargestellt. Diese Zeilen sind in einer anderen Schriftart hinterlegt.



- **Aktualisieren:** Mit Hilfe dieser Funktion können Sie die Prozesshistorie manuell aktualisieren.

7. Prozessdefinition

In diesem Kapitel wird das Vorgehen bei der Definition von Prozessen beschrieben. In *EPworX* gibt es zwei Möglichkeiten der Prozessdefinition:

1. grafische Definition mittels Prozesseditor,
2. Definition des Prozesses als Script in der Workflow Definition Language (WDL)

Beide Formen haben die gleiche Mächtigkeit und sind austauschbar, d.h. sie können einen grafisch definierten Prozess als WDL-Script speichern, dieses weiter bearbeiten, das

WDL-Script laden, und den Prozess im Prozesseditor weiter verändern.¹

Im Folgenden wird zuerst die Skriptsprache WDL beschrieben, da hier die einzelnen Sprachelemente leichter formalisierbar sind.

7.1 WDL

In diesem Abschnitt wird die Syntax der Sprache und die Semantik (Bedeutung) der einzelnen Sprachkonstrukte beschrieben. WDL (Workflow Definition Language) ist die Prozessbeschreibungssprache von *EPworX*. Die Sprache ähnelt einer strukturierten Programmiersprache und ermöglicht die Definition von WDL-Prozessen. Ein WDL-Prozess besteht aus Prozesskopf, Deklarationsteil und Anweisungsteil.

Beispiel:

```
process jobproc()
name "jobproc";
description "simple process";
version 1;
subject f.subj;
forms f.Jobform;
begin
  <label_order> all order(f);
  loop
    f.recipient a_task(f);
    exit when xpath:"$form_f/finished = 'true'";
```

¹Das grafische Layout und Annotationen werden bei Notationswechsel nicht beibehalten

```
end;  
label_order:user inform(f);  
end;
```

Die Prozessdefinition beginnt mit dem Schlüsselwort `process`, gefolgt von der Prozess-Id und einer Argumentliste (= Prozesskopf). Danach folgt der Deklarationsteil als Menge von Key-Value Paaren, z.B. `version 1;`.

Der Anweisungsteil beginnt mit dem Schlüsselwort `begin` und endet mit `end`. Dazwischen wird die Struktur des Workflows, bestehend aus Taskaufrufen, Subprozessen, automatischen Schritten und den verschiedenen Kontrollkonstrukten beschrieben.



Hinweis: Im WDL werden Eskalationen, die für einen Prozess definiert wurden, nicht berücksichtigt!

7.1.1 Lexikalische Konventionen

In WDL gelten die folgenden lexikalischen Regeln:

- Ids

Ids sind Bezeichner für Tasks, Rollen, Benutzer u.ä. In WDL gelten dafür folgende Konventionen:

Ids können mit einem Buchstaben, \$, / oder \ beginnen, danach können weitere der genannten Zeichen oder Ziffern kommen. Die Gesamtlänge von Ids darf 80 Zeichen nicht überschreiten.

Hinweis: Falls die Akteurs-Id keine gewöhnliche Id ist, muss die Id mit einem Ausrufezeichen beginnen.

Beispiel:

```
!right.user@xy.com do_something(f);
```

- Strings

Strings sind in doppelte Anführungszeichen eingeschlossene Zeichenketten. Doppelte Anführungszeichen innerhalb von Strings müssen verdoppelt werden.

Beispiele:

```
"Das ist ein String."
```

```
"Das ist ein String mit zwei ""Anführungszeichen""."
```

- Kommentare

Zwischen „/*“ und „*/“ werden alle Zeichen ignoriert. Damit lassen sich insbesondere auch mehrzeilige Kommentare realisieren.

²

²Die Kommentare werden beim Laden des WDL-Scripts nicht ins System übernommen.

- Groß-/ Kleinschreibung

Zwischen Groß- und Kleinschreibung wird im Allgemeinen unterschieden, d.h. if ist ungleich If.

- Schlüsselwörter

Schlüsselwörter sind in der Tabelle 7.1 aufgelistet. Ein Schlüsselwort, das innerhalb eines einfachen Hochkommata steht, wird nicht als Schlüsselwort interpretiert, jedoch als Id.

abort	ad hocTasks	and	andpar	application
autofinish	baseform	batch	begin	branch
call	choice	corr	correlation	current_tx
days	description	do	else	elsif
end	exception	exit	for	forms
gobackonerror	goto	hours	if	in
instanceid	invoke	loop	maxtime	minutes
name	new_tx	newthread	none	not
null	or	orpar	out	owner
parallel	priority	process	raiseEvent	receive
registerForEvent	repeat	reply	skipable	start
startfunction	startnow	subject	success	sync
system	then	unregister	until	version
wait	when	while		

Tabelle 7.1: Schlüsselwörter in WDL

7.1.2 Prozesskopf

Syntax:

```
processdef =
  "process" id "(" [formdecl{ "," formdecl}] ")"
  { pdeclaration ";" }
  "begin" [nodename]
  statseq
  "end" [nodename] .
```

Beschreibung der Syntax:

- Id: interner Name des Prozesses
- Parameterliste: Formulare, die an den Prozess übergeben werden, wenn dieser als Subprozess aufgerufen wird. Die Übergabe erfolgt *by reference*, d.h. die Formulardaten werden nicht kopiert.
- pdeclaration: Deklarationen, siehe unten.
- statseq: Sequenz der Anweisungen (Taskaufrufe etc.).

7.1.3 Deklarationsteil

Im Deklarationsteil finden sich eine Reihe allgemeiner Informationen über den Prozess.

Syntax:

```
pdeclaration =  
  "name" string  
  | "description" string  
  | "version" number  
  | "subject" ( formfield | expressionstring )  
  | "maxtime" number ( "days" | "hours" | "minutes" )  
  | "forms" formdecl { "," formdecl}  
  | "application" application  
  | "instanceid" string  
  | "priority" number  
  | "adhocTasks" adhocTask { "," adhocTask }  
  .
```

Beschreibung der einzelnen Deklarationen:

- name: Name des Prozesses, wie er in Benutzerdialogen auftaucht.
- description: freier Text
- version: Integer, gibt die Version dieser Prozessdefinition an.
- subject: Inhalt des Betreff-Feldes in der Worklist. Kann ein einzelnes Formularfeld (formid.fieldid) sein oder ein Ausdruck der mehrere Felder umfasst. Die Definition ist im Kapitel [Prozesse](#) beschrieben.
- maxtime: maximale Bearbeitungszeit, Angabe in Tagen, Stunden oder Minuten.
- forms: Deklaration der Formulare als Prozesslokale Datencontainer: Die Definition eines lokalen Formulars lautet wie folgt:

```
formid formtype [ "baseform" baseformid ] [ "formname" ]
```

- formid: ist die Id des lokalen Formulars
- formtype: ist ein dem System bekannter Formulartyp
- formname: ist der Anzeigenname des lokalen Formulars (optional)
- baseform: wenn das zu deklarierende Formular ein View-Formular ist, muss das Basisformular angegeben werden.

Beispiel: forms rg rechnung, ls lieferschein, rgkopf rechnungskopf baseform rg;

- application: Angabe der Id der Applikation, zu welcher der Prozess gehört.

- **instanceid**: Id, welche jede Instanz eindeutig identifiziert. Hierbei besteht auch die Möglichkeit einen Nummernschema über einen regulären Ausdruck anzugeben (siehe Kapitel [Prozesse](#)).
- **priority**: Frei vergebbarer Wert, der die Priorität des Prozesses angibt.
- **adhocTasks**: Kann verwendet werden, um zusätzliche Tasks zu deklarieren, die zur Laufzeit des Prozesses programmatisch erzeugt werden können. Syntaktisch sind das Task Statements (siehe unten) ohne deklarierte Akteursliste. Die Akteure werden erst zur Laufzeit angegeben. Jeder adhocTask ist wie folgt definiert:

```
[ "<" labelid ">" ] taskid "(" [ formlist ] ")" [ nodename ]
```

Alle Deklarationen bis auf name, version und application sind optional..

7.1.4 Einfache Anweisungen

Der Anweisungsteil ist der zentrale Teil einer Prozessspezifikation und befindet sich zwischen den beiden Schlüsselwörtern begin und end. Er enthält zumindest eine Anweisung. Anweisungen werden durch ein Semikolon abgeschlossen.

Syntax:

```
statseq = { [ "<" labelid ">" ] statement [ nodename ] ";" }.
```

```
statement =  
(  
    batchstmt  
| branchstmt  
| choice  
| exitstmt  
| gotostmt  
| ifstmt  
| invokestmt  
| loopstmt  
| par  
| parforstmt  
| raiseEvent  
| receivestmt  
| registerForEvent  
| repeatstmt  
| replystmt  
| subproccall  
| sync  
| systemstmt  
| taskstmt  
| unregister
```

```
| whilestmt  
| waitstmt )  
.
```

labelid: Die Id dieses Schrittes innerhalb der Prozessdefinition. Sie muss eindeutig sein und kann als exakte Referenz für diesen Schritt verwendet werden.

nodename: Ein String, der als Anzeigename für das Statement (und die entsprechenden Knoten im Prozesseditor) verwendet wird. Dieser Name muss nicht eindeutig sein. Beispiel:

```
<ordertask> all order(form) "Bestellung anlegen";
```

Im Folgenden werden die verschiedenen Anweisungen näher beschrieben:

Manueller Taskaufruf

Aufruf von manuellen Tasks.

Syntax:

```
taskstmt =  
  ( "none" | agentlist ) taskid "(" [ formlist ] ")" [ "skipable" ].
```

```
agentlist = agent { , agent }.
```

Beschreibung der Syntax:

Akteure (agentlist): es stehen folgende Möglichkeiten für die Angabe von Akteuren zur Verfügung:

- Der Akteur wird direkt mit seinem Benutzer-Id angegeben. Dies sollte aber nur in wohl begründeten Fällen erfolgen, da die Prozessdefinition für gewöhnlich keine spezifischen Referenzen auf einzelne Benutzer beinhalten sollte.
- Der Akteur wird als Id einer Rolle (logische Benutzergruppe) angegeben. Jeder WF-Benutzer, der dieser Rolle angehört, kann als potenzieller Akteur der Aktivität fungieren.
- Der Akteur wird über eine bestimmte Rolle in einer bestimmten Organisationseinheit (OE) definiert. Die Notation dafür lautet: `org_einheit_id!roleid`. Beispiel: `einkauf!sek`.

Die Organisationseinheit wird nur für den aktuellen Schritt geändert. Für den Gesamtprozess bleibt die alte OE bestehen.

- Akteur eines vorherigen Schrittes: `labelid:user` Dieser Ausdruck gibt den Benutzer an, der die Aktivität mit der Label-Id (Kennzeichnung) `labelid` innerhalb der aktuellen Prozessinstanz zuletzt ausgeführt hat. In diesem Fall wird die Organisationseinheit des vorherigen Schrittes übernommen! Beispiel: `antrag_stellen:user`

- Akteurspezifikation aus einem Formularfeld. Im Formularfeld steht entweder direkt eine Benutzer-Id, eine Rollen-Id oder die Angabe einer bestimmten Rolle innerhalb einer bestimmten Organisationseinheit (OE).
- Ein leerer Akteur: "none". Zur Laufzeit muss der kann der Akteur programmtechnisch oder vom Bearbeiter des vorhergehenden interaktiven Schritts gesetzt werden.
- Aufgrund der Definition einer Java-Methode: Eine Java-Methode (z.B. `com.acme.Action.setAgent()`) mit einem der folgenden Rückgabe-Werte:
 - einer Rollen-Id
 - einer Benutzer-Id
 - einer Kombination aus Id der Organisationseinheit und Id der Rolle in Form von `"org_unitid ! roleid"`
 - eines Benutzer-Objekts
 - eines Rolle-Objekts
 - eines Objekts vom Typ `com.groiss.ds.Pair`, wobei `Pair.first` verwendet wird, um den Akteur der `StepInstance` zu definieren und `Pair.second`, um den "Step Agent" der Schrittinstanz festzulegen. Wenn Sie diese Art verwenden, sollte
 - * `Pair.first` ein Benutzer (`com.groiss.org.User`) und
 - * `Pair.second` eine Rolle (`com.groiss.org.Role`) bzw. eine Kombination aus Organisation (`com.groiss.org.OrgUnit`) und Rolle sein.

Aufgrund dieser Definition erscheint der Task zur Laufzeit im Arbeitskorb beim festgelegten Benutzer und falls die Toolbarfunktion "Zurücklegen" im Arbeitskorb aktiviert wird, landet der Task im Rollen-Arbeitskorb der angegebenen Rolle.

- Sequenz von Akteuren: Es kann auch eine durch Beistrich getrennte Liste von Akteuren angegeben werden, wobei die verschiedenen Formen gemischt werden können. Die Definition von mehreren Akteuren bei einem Task bedeutet, dass diese den Task hintereinander in einer Sequenz bearbeiten.³

Die in der Prozessdefinition angegebenen Akteure können zur Laufzeit durch eine Preprocessing-Methode überschrieben werden. Dafür muss beim entsprechenden Task ein entsprechender Methodenaufwurf angegeben werden.

taskid: Die Id eines in der Applikation definierten Tasks. Wenn Sie eine Id festlegen, die nicht bereits einem anderen Task vergeben wurde, dann kann die Option *Tasks generieren* beim Laden der Prozessdefinition verwendet werden. Der Task wird dann mit der Id in der Prozessdefinition erzeugt.

formlist: Eine durch Beistrich getrennte Liste von formids, die in der Argumentenliste des Prozesses definiert oder als lokales Formular deklariert wurden.⁴

³Eine etwaige Preprocessing-Methode wird vor dem 1. Akteur ausgeführt, die Postcondition wird erst nach dem letzten Akteur überprüft.

⁴Zur Laufzeit werden die Icons für diese Formulare im Arbeitskorb beim jeweiligen Task angezeigt. Der Formularinhalt ist somit sicht- und editierbar für Akteure dieses Tasks. Im Kapitel [Tasks](#) wird die Möglichkeit der feingranularen Einschränkung auf Einzelfeldebene pro Task beschrieben.

skipable: Der Task wird übersprungen, wenn zur Laufzeit kein Akteur definiert ist. Wird in der Prozessdefinition als Akteur *none* angegeben (also kein Akteur), wird der Task nur ausgeführt, wenn über eine Preprocessing-Methode ein Akteur gesetzt wurde.

Subprozess-Aufruf

Ein Prozess kann als Teil eines anderen bereits ausgeführten Prozesses aufgerufen werden. Somit können wiederverwendbare und modular aufgebaute Prozesse definiert werden.

Syntax:

```
subproccall =  
  "call" subprocid "(" [formlist] ")".
```

Das Statement `call` instanziert einen Prozess, der bereits im System definiert wurde, durch Angabe von `subprocid` zur Laufzeit des aktuellen Prozesses. Die Ausführung erfolgt synchron. Der aufgerufene Prozess erhält die Kontrolle und wenn dieser beendet wird, fällt die Kontrolle zurück zum aufrufenden Prozess unmittelbar nach dem `call` Statement.

Formulare können dem aufgerufenen Prozess als durch Beistrich getrennte Liste (`formlist`) von Formular-Ids übergeben werden. Die Übergabe erfolgt *by reference*, d.h. die Formulare werden nicht kopiert. Die `formids` beziehen sich auf im Aufrufer bekannte Formularvariable und müssen zu den im Prozesskopf deklarierten Formularen des aufgerufenen Prozesses passen.

Systemschritte

Ein Systemschritt wird dazu verwendet, um eine Java-Methode ohne interaktives Zutun auszuführen. Der Name der Methode wird nach dem Schlüsselwort `system` festgelegt. Eine Liste von Stringliterals innerhalb der Klammern kann der Parametrisierung dienen. Diese Methode wird synchron abgearbeitet auf kurze Abarbeitungszeiten ist zu achten.

In einem Systemschritt kann auch eine Liste von mehreren Java Methoden angegeben werden. Diese Methoden sind zwischen `begin` und `end` einzuschließen und mit `;` zu trennen. Die Methoden werden sequentiell ausgeführt.

Syntax:

```
systemstmt =  
  "system" methodname "(" [ string { "," string } ] ")"  
  |  
  "system" "begin"  
    methodname "(" [ string { "," string } ] ")"  
    { ";" methodname "(" [ string { "," string } ] ")" }  
  "end"  
  .
```

Achten Sie auf vollständige Angabe der Klassennamen, also inkl. Package-Bezeichnung, z.b.: `system com.groiss.demo.Step.exec("p1","p2");`

Hinweis: Nützliche Standard-Methoden sind in der Klasse `com.groiss.wf.SystemAction` verfügbar!

Batch Schritte

Batch Schritte sind ähnlich wie Systemschritte, da sie auch automatisch ohne Interaktion ausgeführt werden. Der wesentliche Unterschied jedoch besteht darin, dass Batch Aktivität asynchron ausgeführt werden und somit eine längere Abarbeitungszeit beanspruchen können. Bei Batch Aktivitäten muss eine Handler-Klasse angegeben werden, damit auf Ereignisse während der asynchronen Abarbeitung reagiert werden kann.

Beschreibung der Syntax:

```
batchstmt =  
  "batch" batchAdapterClassName "(" [ paramstring ] ")"  
    { "startnow" | "newthread" | "retrystart" |  
      "autofinish" | "pollfinish" | "gobackonerror" }.
```

Achten Sie auf vollständige Angabe des Klassennamens, also incl. Package-Bezeichnung, z.b.: `batch com.groiss.demo.DemoBatchAdapter("param")`. Weitere Informationen zu Batch-Jobs finden Sie im [Programmierhandbuch](#) und in der [API-documentation](#).

Warten Schritte

Ein Warten Schritt kann dazu verwendet werden, um einen Prozesslauf für eine gewisse Zeitdauer bzw. bis zu einem gewissen Zeitpunkt anzuhalten. Der Warten Schritt wird durch den Timer *Suspension* automatisch beendet, d.h. der Warten Schritt wird frühestens beendet, wenn die Zeitdauer überschritten ist abhängig davon wann der Timer *Suspension* das nächste Mal ausgeführt wurde. Es besteht auch die Möglichkeit diesen manuell über die Prozesshistorie beenden. Die Zeitdauer bzw. der Zeitpunkt wird nach dem Schlüsselwort `wait` definiert in Form einer Methode, als ein Formularfeld oder als eine Nummer kombiniert mit der Zeiteinheit:

Syntax:

```
waitstmt =  
  "wait"  
    ( methodname "(" [ string { "," string } ] ")"  
      | formfield  
      | number ("workdays" | "days" | "hours" | "minutes" )  
    )
```

Methoden und Formularfelder können entweder *java.util.Date* Objekte, die als absolute Zeitpunkte interpretiert werden, oder ganzzahlige Werte, die als Minuten interpretiert werden, zurückliefern. Kein Warten im eigentlichen Sinne wird durchgeführt, wenn der Zeitpunkt in der Vergangenheit liegt oder die ganzzahligen Werte nicht positiv sind. Bei Angabe einer Methode muss der full-qualified Methodennamen inklusive Paketname angegeben werden, z.B: `wait com.groiss.demo.Step.calcDate("p1","p2");` Zusätzlich ist es möglich Groovy- und XPath-Ausdrücke anzugeben als Dauer bzw. Zeitpunkt. Nähere Informationen darüber finden Sie im [Programmierhandbuch](#) von EPworX.

7.1.5 Kontrollstrukturen

Kontrollstrukturen sind notwendig, um den Kontrollfluss zu definieren. Mittels Kontrollstrukturen kann bestimmt werden, in welcher Reihenfolge Aktivitäten ausgeführt werden sollen. Die wichtigsten Kontrollelemente sind die Sequenz, die Alternative und die Parallelität.

Sequenz

Eine sequentielle Abarbeitung von Aktivitäten erreicht man, indem die einzelnen Aktivitätsaufrufe nacheinander im WDL-Programm aufgelistet werden. Als Trennzeichen wird das Semikolon ; verwendet.

Beispiel:

Zuerst soll die Aktivität `auftrag_erfassen()` von der Rolle `sek` ausgeführt werden. Sobald diese Aktivität beendet ist, soll die Aktivität `erhebung()` von der Rolle `sachbearbeiter` abgearbeitet werden. Danach soll ein Techniker in der Organisationseinheit `waehlamt` die Aktivität `durchschalten()` ausgeführt werden.

```
...
sek auftrag_erfassen(auftrag);
sachbearbeiter erhebung(auftrag);
waehlamt!techniker durchschalten(auftrag);
...
```

Bedingungen

Bedingungen treten in der WDL an folgenden Stellen auf:

- Alternativen: `if`, `choice`
- Schleifen: `while`, `repeat`, `loops` - `exit when`
- Postconditions bei Tasks

Es können Vergleiche zwischen Konstanten und Formularfeldern, Java-Methodenaufrufe, und beliebige logische Verknüpfungen von beiden angegeben werden. Weiters gibt es noch die Möglichkeit mittels Groovy und XPath Bedingungen anzugeben. Nähere Infos

zur Verwendung von Groovy und XPath finden Sie im [Programmierhandbuch](#).

Syntax:

```
cond = expr1 { "or" expr1 } .
```

```
expr1 = expr2 { "and" expr2 }.
```

```
expr2 = [ "not" ] expr3.
```

```
expr3 =  
    "(" cond ")"  
    | methodcall  
    | booleanformfield  
    | formfield relop (number | string | formfield | "null").
```

```
relop = ( "=" | "<>" | "<=" | ">=" | "<" | ">" ).
```

```
formfield = formid "." fieldid.
```

Beispiele:

- f.recipient = null
- f.auftragswert > 100000
- com.groiss.Check.isAvailable("f.betrag") and f.klasse > 3
- (f.empfaenger <> null or f.wert > 10000) and f.klasse = 4
- groovy: form_f.betreff == "Book"
- xpath:\$form_f/subform[@id='1']/form/status = 'ok'

Java-Methoden haben 0 bis n konstante Strings als Parameter und den Rückgabebetyp boolean. Näheres zur Programmierung von Java-Methoden als Bedingungen sind im [Programmierhandbuch](#) beschrieben.

If

if und elseif ermöglichen die bedingte Ausführung von Prozesszweigen.

Syntax:

```
ifstmt =  
    "if" cond  
    "then" [nodename] statseq  
    { "elseif" cond "then" [nodename] statseq }  
    ["else" statseq ]  
    "end".
```

Beschreibung der Syntax

- cond: eine Bedingung wie oben definiert.
- statseq: eine Sequenz von Statements.

Beispiel:

```
if auftrag.betrag <= 2000 then "kleiner Auftrag"
    sachbearbeiter bestaetigung_schreiben()
elsif auftrag.betrag <= 5000 and auftrag.klasse = 4 then "mittlerer Auftrag"
    direktor genehmigen()
elsif ...
    ...
else
    ...
end
```

Choice: gemischte manuelle und automatische Alternativwahl

Choices ermöglichen dem Benutzer einen Prozessweg aus einer vorbestimmten, aber zur Laufzeit eingeschränkten Menge von möglichen Wegen zu selektieren.

Syntax:

```
choice =
    "choice" [ nodename ]
    { branchname [ "," cond ] ":" statseq }
    "end".
```

Beschreibung der Syntax:

Jeder Zweig hat einen Namen (branchname) — hier kann ein beliebiger String eingegeben werden – und eine optionale Bedingung. Zuerst werden die Bedingungen aller Zweige ausgewertet. Die Zweige mit wahren Bedingungen werden dem Benutzer zur endgültigen Auswahl angeboten. Wenn keine Bedingungen angegeben wurden, wird die Auswahl völlig manuell erfolgen.

Beispiel:

```
choice "manuelle Auswahl"
    "sofort bestellen", f.summe < 5000:
        sek bestellen(f);
    "nochmals prüfen":
        sachbearbeiter pruefen(f);
    "ablegen":
        system Archive.insert();
end;
```

While-Schleife

Syntax:

```
whilestmt =  
    "while" cond "do" [ nodename ]  
    statseq  
    "end"
```

Beschreibung der Syntax:

Die Anweisungen im Schleifenkörper einer While-Schleife (solange-bis Schleife) werden wiederholt, solange der Wert der Bedingung (cond) wahr ist. Da die Überprüfung der Bedingung vor Eintritt in den Schleifenkörper erfolgt, wird dieser Schleifentyp auch als *abweisende* Schleife bezeichnet.

Beispiel:

```
while f.genehmigt = 0 do  
    sek korrigieren(f);  
end;
```

Repeat-Schleife

Syntax:

```
repeatstmt =  
    "repeat" [ nodename ]  
    statseq  
    "until" cond.
```

Beschreibung der Syntax:

Die Statements im Schleifenkörper werden, solange wiederholt, bis die Bedingung wahr ist. Schleifenanweisungen werden mindestens ein Mal ausgeführt.

Beispiel:

```
repeat  
    sachbearbeiter daten_erfassen(auftrag);  
    call daten_pruefen(auftrag);  
until antrag.daten_ok = 1;
```

Loop-Schleife

Syntax:

```
loopstmt =  
    "loop" [ nodename ]  
    [ statseq1 ]  
    "exit" "when" cond;  
    [ statseq2 ]  
    "end".
```

Beschreibung der Syntax:

Allgemeine Schleife, bei Erreichen von end wird wieder zu loop gesprungen. Die Statements in statseq1 werden ausgeführt. Die Bedingung von exit when wird ausgewertet und falls das Ergebnis nicht wahr ist, die Anweisungen von statseq2 ausgeführt und die Schleife wird nochmals durchlaufen. Wenn die Auswertung wahr ergibt, wird die Schleife beendet ohne die Anweisungen aus statseq2 nochmals auszuführen.

Parallelität (andpar und orpar)

Innerhalb eines Prozesses können Teile auch parallel ausgeführt werden. Dies ermöglicht erhebliche Zeiteinsparungen, indem Aktivitäten nebenläufig abgearbeitet werden. Andpar und orpar ermöglichen die Angabe einer festgelegten Anzahl von parallelen Zweigen.

Syntax:

```
par =  
  ( "andpar" | "orpar" ) [ nodename ]  
    statseq  
    { "|" statseq }  
  "end" [ "do" paramethod ].
```

Grundsätzlich gilt für die beiden Varianten:

- andpar: Prozess wird fortgesetzt, wenn alle Zweige beendet sind.
- orpar: Prozess wird fortgesetzt, wenn ein Zweig beendet ist, alle anderen Zweige werden abgebrochen.

Der optionale paramethod wird später noch genauer beschrieben.

Beispiel für andpar:

Für die Behandlung eines komplexen Schadensfalles ist immer die Einholung von 3 Gutachten notwendig. Erst danach kann eine Bewertung erfolgen:

```
...  
andpar  
  gutachter1 gutachten_erstellen(s_form1);  
  | gutachter2 gutachten_erstellen(s_form2);  
  | gutachter3 gutachten_erstellen(s_form3);  
end  
s_abteilungleiter bewerten(s_form1, s_form2, s_form3)  
...
```

Beispiel für orpar:

Bei der Ermittlung eines Fluges werden immer Anfragen an mehrere Flugreservierungssysteme gerichtet. Sobald ein passender Flug gefunden wurde, werden die anderen Abfragen abgebrochen (bzw. nicht weiter verfolgt).

```
...
sachbearbeiter flugdaten_erfassen(flugform);
orpar
  clerk check_route1(flugform);
  | clerk check_route2(flugform);
end
...
```

Eine Verallgemeinerung der Parallelität, wo n von m Zweigen abgeschlossen sein müssen, wird mit einem *andpar* realisiert, wo beim Endknoten (join) eine entsprechende Methode *paramethod* angegeben wird:

```
com.groiss.wf.SystemAction.join(n,action)
```

Beide Parameter sind Strings:

- n : Die Anzahl der fertigzustellenden Zweige, damit die gesamte Parallelität als abgeschlossen betrachtet wird.
- *action*: Beinhaltet entweder *none* oder *cancel*. Beim Wert *cancel* werden die verbliebenen aktiven Zweige abgebrochen.

Beispiel für *andpar* mit n von m abgeschlossenen Zweigen:

Für die Behandlung eines minderen Schadenfalles ist die Einholung von nur zwei Gutachten notwendig:

```
...
andpar
  gutachter1 gutachten_erstellen(s_form1);
  | gutachter2 gutachten_erstellen(s_form2);
  | gutachter3 gutachten_erstellen(s_form3);
end do com.groiss.wf.SystemAction.join("2","cancel");
...
```

Wenn die Fertigstellung der Parallelität nicht durch eine fixe Anzahl von fertigen Zweigen bestimmt ist, sondern zur Laufzeit bestimmt werden kann, ist dies durch eine Java-Methode im Endknoten (join) möglich. Nähere Informationen dazu finden Sie im [Programmierhandbuch](#).

Parallel For

Ermöglicht die Aufspaltung in mehrere identische Zweige. Die Anzahl der Zweige ist erst zur Laufzeit und nicht schon zur Prozessdefinitionszeit bekannt.

Syntax:

```
parforstmt =
  "parallel" "for" (localformid "in" formid"."subformtableid ["when" cond]
    | iteratorclass)
  "do" [ nodename ]
    statseq
  "end" [ "do" parmethode ].
```


Mit dieser Kontrollstruktur kann entweder ein paralleler Zweig pro Zeile in einer Subformtabelle erzeugt werden, oder die Generierung der Zweige erfolgt über einen Iterator.

Beschreibung der Syntax:

- `localformid`: neue lokale Variable (frei wählbarer Id), die auf das entsprechende einzelne Subformular im parallelen Zweig verweist.
- `formid`: Id des Hauptformulars.
- `subformtableid`: Id der Subformulartabelle wie im `tablefield` Eintrag bei der Definition des Formulartyps des Hauptformulars angegeben.
- `cond`: Optionale Bedingung, die angibt, ob der Parfor-Zweig durchlaufen werden soll (`true/false` als Rückgabewert).
- `iteratorclass`: Name einer Klasse, die das Interface *com.groiss.wf.ParForIterator* implementiert.

Im Endknoten kann durch Angabe einer Methode eine spezifische Bedingung für das Ende des Parfor Konstrukts angegeben werden. Fehlt diese Angabe, so wird auf das Ende aller Zweige gewartet.

Bei einer fixen Anzahl von Zweigen kann die Methode

```
com.groiss.wf.SystemAction.parforJoin(n,action)
```

verwendet werden. Die Parameter entsprechen dabei jenen, die bei der Methode *com.groiss.wf.SystemAction.join(n,action)* angegeben wurden (siehe Abschnitt *Parallelität* von Kapitel [WDL](#)).

Wenn die Entscheidung über die Fertigstellung des gesamten Parfor Konstrukts völlig dynamisch erfolgen muss, kann eine Java-Methode angegeben werden. Nähere Informationen über Parallelitäten finden Sie im [Programmierhandbuch](#).

Abzweigung

Abzweigungen erlauben die Aufsplittung des Ausführungspfads in einen Hauptpfad und in einen Nebenpfad (den Branch).

Syntax:

```
branchstmt =  
  "branch" [nodename]  
  statseq  
  "end".
```

Beschreibung der Syntax:

Anweisungen im Branch werden parallel zu den Anweisungen im Hauptzweig ausgeführt. Nebenzweige können die Beendigung des Hauptzweiges überdauern.

Beispiel:

```
begin
  ..
  bearbeiter eingabe(f);
  manager kontrolle(f);
  branch "Merkliste"
    pruefer inform(f);
  ...
end;
mitarbeiter erzeuge(f);
...
end
```

Goto

Goto's erlauben ein Abweichen des Prozessablaufes und Sprünge zu anderen Teilen der Prozessdefinition.

Syntax:

```
gotostmt =
  "goto" labelid.
```

Die Ausführung wird am Statement das durch labelid bezeichnet wird fortgesetzt.

Beispiel:

```
<entry> clerk enter(f);
supervisor check(f);
if (f.quality <> "OK") then /* denotes exceptional case */
  goto entry;
end;
worker build(f);
```

In diesem Beispiel bewirkt die goto Anweisung, dass die Tasks enter und check nochmals wiederholt werden, wenn die Qualität nicht ausreichend ist. Eine repeat until Schleife wäre eigentlich die bessere Formulierung dieses Ablaufs, jedoch möchte der Designer wahrscheinlich mit diesem Konstrukt explizit zum Ausdruck bringen, dass die Wiederholung ein Ausnahmefall ist.

Die Verwendung von gotos sollte sparsam und nur mit Bedacht in wohlbegründeten Ausnahmefällen erfolgen. Exzessive Verwendung beeinträchtigt die Lesbarkeit und Wartbarkeit der Prozessdefinition nachhaltig.

7.1.6 Eventmechanismus

Der Eventmechanismus wird verwendet um innerhalb der Engine Events zu erzeugen und zu behandeln. Prozesse die ihr Interesse für Events registriert haben, empfangen solche Events von der Engine ausgelösten Aufruf der registrierten Eventhandler.

Syntax:

Auslösen von Events durch:

```
raiseEvent =  
"raiseEvent" "(" eventName "," "current_tx" [ "," context ] ")".
```

Warten auf Events:

```
sync =  
"sync" "(" eventName [ "," eventhandler [ "," context ] ] ")".
```

Registrieren von Handlern für Events:

```
registerForEvent =  
"registerForEvent" "(" eventName [ "," eventhandler [ "," context ] ] ")".
```

Die Registrierung von Eventhandlern aufheben:

```
unregister =  
"unregister" "(" eventName ")".
```

Der Kontext wird angegeben über:

```
context =  
"parent" | "processInstance" | "mainProcess" | formid [ "." fieldname ].
```

Beschreibung der Syntax

- **eventName:** Der Name des Events.
- **context:** ein Knoten der aktuellen Prozessinstanz oder ein Formular oder ein Formularfeld, das als Kontext-Objekt dienen soll. Im [Application Development Guide](#) findet man Details dazu. Alternativ kann auch das Schlüsselwort *process* angegeben werden, womit der Elternknoten der aktuellen Aktivitätsinstanz als Kontext-Objekt dient.
- **eventhandler:** Eine Java-Klasse, die das Interface `com.groiss.event.IEventHandler` implementiert. Wenn die Angabe fehlt, wird `com.groiss.event.EventHandler` verwendet.

Weitere Informationen zu Events finden Sie im [Programmierhandbuch](#).

7.1.7 Web-Dienste

Die WDL bietet die Möglichkeit Web-Dienste in die Prozessdefinition einzubinden. Über `invoke`-Knoten können Webservices aufgerufen werden. Die Ausführung in einem Zweig kann mittels `receive`-Knoten aufgehalten werden, bis ein Webserviceaufruf von außen erfolgt. Knoten vom Typ `reply` bezeichnen das Senden einer Antwort auf einen zuvor erfolgten Webservice-Aufruf.

Syntax:

Eingangsnachricht (RECEIVE):

```
receivestmt =  
  "receive" [ "start" "process" ] operationspec "(" [incorrparams] ")" ["end"].
```

Antwort (REPLY):

```
replystmt =  
  "reply" operationspec "(" [outparams] ")".
```

Ausgangsnachricht (INVOKE):

```
invokestmt =  
  "invoke" [address "."] operationspec "(" [inoutparams] ")"  
    ["success" statseq]  
    ["exception" statseq]  
  ["end"].
```

Allgemeine Anweisungsteile, die von den Web-Dienste-Knoten verwendet werden:

```
operationspec =  
  serviceid "." operationid.
```

```
incorrparams =  
  incorrparam {"," incorrparam}.
```

```
incorrparam =  
  inparam | corrparam.
```

```
corrparam =  
  ("corr" | "correlation") xpath "=" messagecomp.
```

```
inoutparams =  
  inoutparam {"," inoutparam}.
```

```
inoutparam =  
  inparam | outparam.
```

```
inparam =  
  ["in"] xpath "=" messagecomp.
```

```
outparams =  
  {"," outparam}.
```

```
outparam =  
  ["out"] messagecomp "=" xpath.
```

Beschreibung der Syntax:

- serviceid: Die ID des Web-Dienstes.
- operationid: Die Operations-ID einer Operation des Web-Dienstes.

- address: Die URL des Webservice
- messagecomp: Die ID des (IN/OUT)-Parameters, der Nachricht.
- xpath: Ein XPath-Ausdruck, der das Formularelement bezeichnet, das auf die Nachrichtenkomponente abgebildet wird.
- statseq: Sequenz der Anweisungen.

```
...
invoke mywebservice.SendMessage(
    MessageTemplate="form_ticket/messageTemplate",
    MessageType="$form_ticket/messageType", enterpriseid="$pi/id",
    xexid="$form_ticket/xexid", reporter="$form_ticket/reporter",
    'subject'="$form_ticket/subject", ""0""=SendMessageResult)
exception
    administrator inform(ticket);
end;
...
receive kserver.sendMessage(
    corr "$pi/id"=enterpriseid,
    "$form_ticket/messageTemplate"=messageTemplate,
    "$form_ticket/messageType"=messageType,
    "$form_ticket/enterpriseid"=enterpriseid,
    "$form_ticket/xexid"=xexid,
    "$form_ticket/reporter"=reporter,
    "$form_ticket/subject"='subject',
    "$form_ticket/text"=text,
    "$form_ticket/analyst"=analyst
) waitforincomingmessage;
...
reply kserver.sendMessage('out'=""0 "");
...
```

Weitere Informationen zu Web-Diensten finden Sie im [Programmierhandbuch](#).

7.2 Grafische Prozessdefinition

Zur grafischen Neudefinition bzw. Änderung von Prozessen steht Ihnen der *EPworX* – Prozesseditor in der Prozesskomponente zur Verfügung. Der Prozesseditor unterstützt dabei die Notationen *BPMN* (Business Process Modeling Notation - siehe Abbildung 7.1).

Je nachdem, ob Sie einen neuen Prozess definieren wollen oder einen bereits bestehenden abändern, gehen Sie wie folgt vor:

- Durch einen Mausklick auf den Link *Neuen Prozess erstellen*) öffnet sich der *EPworX* –Prozesseditor in der BPM-Notation.
- Markieren Sie zunächst jenen Prozess, den Sie bearbeiten wollen und wählen anschließend den Link *Im Prozesseditor bearbeiten*. Dadurch wird der *EPworX* –Prozesseditor gestartet und der aktuell ausgewählte Prozess darin dargestellt. Nun können Sie damit beginnen, die Prozessdefinition zu ändern.

7.2.1 Aufbau des Prozesseditors

Der Prozesseditor besteht aus den folgenden Komponenten:

- **Titelleiste:** Diese Leiste enthält den aktuellen Namen des zu bearbeitenden Prozesses.
- **Menüleiste:** In der Menüleiste befinden sich folgende Menüpunkte:
 - Prozess
 - Bearbeiten
 - Ansicht
 - Hilfe
 - Symbolleiste
- **Zeichenbereich:** In diesem Bereich werden die Prozesse grafisch modelliert.
- **Funktionsleiste:** Diese Leiste stellt die erforderlichen Funktionen für die Manipulation von Prozessen zur Verfügung.



Hinweis: Da es Probleme mit Popup-Blocker geben kann, empfehlen wir Ihnen diese für die Dauer der Prozessbearbeitung abzuschalten!

7.2.2 Arbeiten mit dem Prozesseditor

Bevor wir die verschiedenen Elemente und Funktionen des Editors im Detail beschreiben geben wir hier einen Überblick darüber, wie das Bearbeiten eines Prozesses mit dem Editor erfolgt.

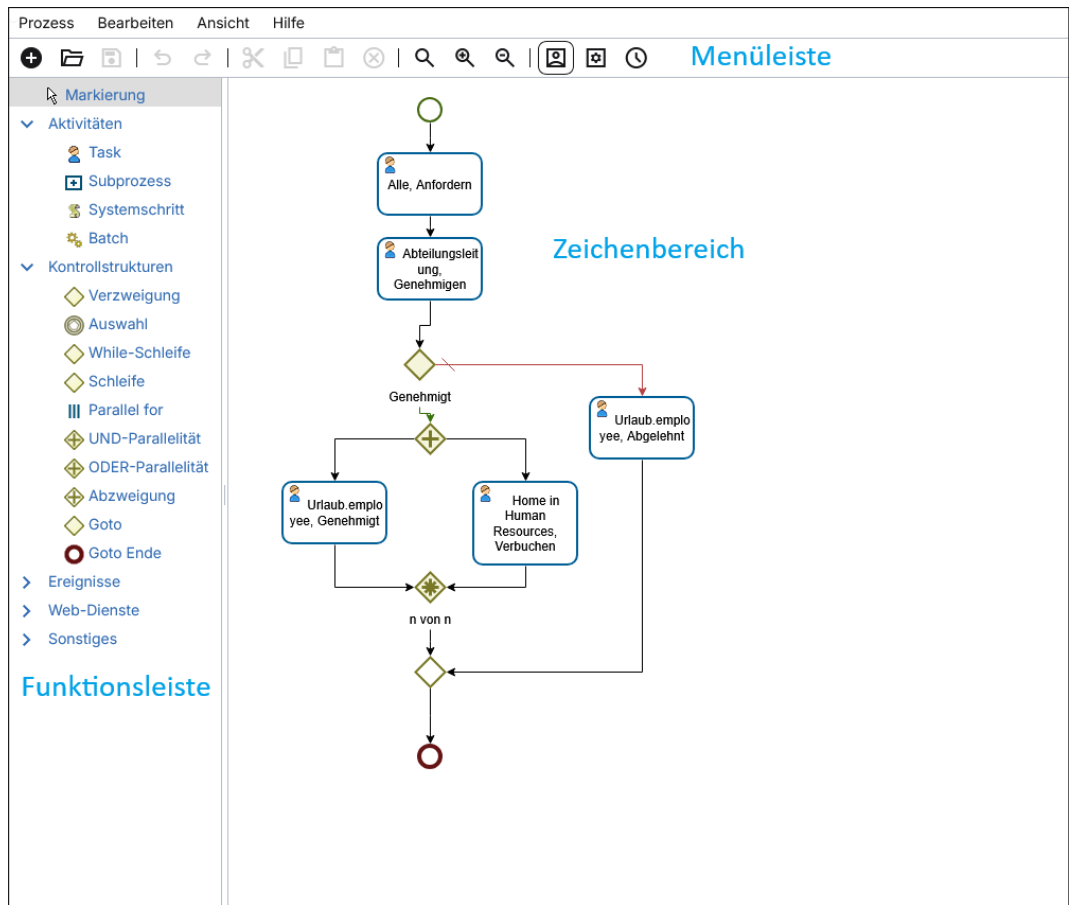


Abbildung 7.1: **Prozesseditor**

Elemente hinzufügen

In der Funktionsleiste links neben dem Zeichenbereich werden Ihnen alle Elemente angezeigt, die Sie Ihrem Prozess hinzufügen können. Zur besseren Übersicht sind diese Elemente in Kategorien unterteilt, z.B. Kategorie *Kontrollstrukturen* zur Modellierung von bedingten Pfaden oder Schleifen.

Um ein neues Element hinzuzufügen gehen Sie wie folgt vor:

- Das gewünschte Element in der Funktionsleiste durch Anklicken auswählen, um in den Einfügemodus zu wechseln.
- Mit der Maus in das Zeichenfenster navigieren. Nun verändert sich der Mauszeiger als Hinweis darauf, dass sich der Editor im Einfügemodus befindet (siehe Abbildung 7.2).
- Ein neues Element können Sie immer nur auf einer Verbindungslinie zwischen zwei bestehenden Elementen einfügen. Dazu bewegen Sie die Maus über die gewünschte Linie (diese wird dann hervorgehoben und auch der Mauszeiger ändert sich wieder) und klicken auf jene Stelle der Linie, an der Sie das ausgewählte



Abbildung 7.2: Einfügemodus bei Auswahl von **Task**

Element einfügen wollen (siehe Abbildung 7.3).

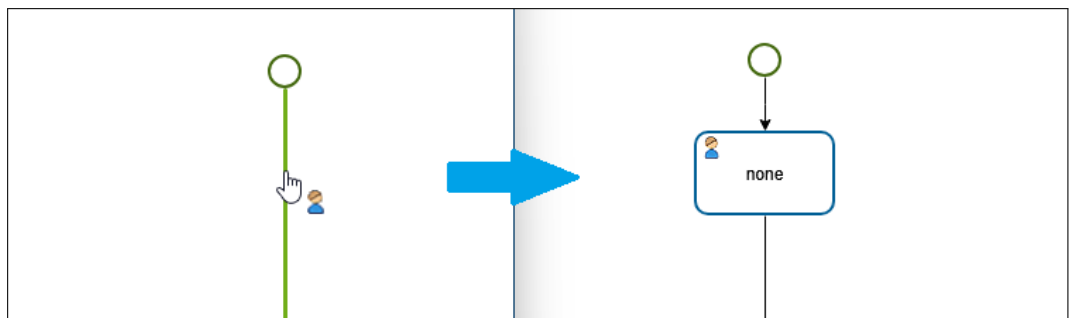


Abbildung 7.3: Einfügen eines **Task**



Hinweis: die einzige Ausnahme von dieser Vorgehensweise ist die Annotation, die Sie hinzufügen können, indem Sie in einen beliebigen leeren Bereich des Zeichenbereichs klicken. Sie können Annotationen aber auch zu einzelnen Elementen hinzufügen. Dazu müssen Sie das Element auswählen und dann den Menüpunkt *Annotieren* aktivieren. Dieser fügt eine Annotation ein, die mittels einer strichlierten Linie mit dem Element verbunden wird.

- Sollte ein Einfügen auf der Linie nicht möglich sein, was nur bei wenigen speziellen Linien der Fall ist, dann bekommen Sie auch das durch eine Änderung des Mauszeigers angezeigt. Außerdem wird in diesem Fall die Linie nicht hervorgehoben (siehe Abbildung 7.4).



Hinweis: den Einfügemodus können Sie jederzeit beenden, indem Sie die *ESC* Taste auf Ihrer Tastatur drücken oder in der Funktionsleiste die Option *Markierung* auswählen.

Der Prozesseditor stellt sicher, dass jede Einfügeoperation zu einem gültigen Prozessmodell führt. So wird z.B. beim Einfügen der Kontrollstruktur *Verzweigung* nicht nur ein einzelnes neues Element hinzugefügt, sondern alle Elemente, die eine solche Verzweigung benötigt, also das Anfangs- und Ende-Element sowie zwei Verbindungslinien für

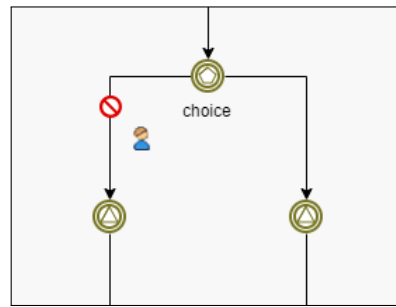


Abbildung 7.4: **Einfügen nicht erlaubt**

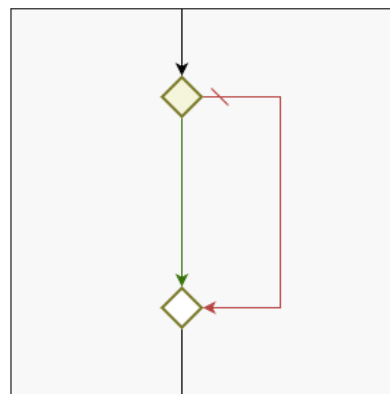


Abbildung 7.5: **Eingefügtes Verzweigung-Konstrukt**

die Wahr- und Falsch-Pfade für diese Bedingung (siehe Abbildung 7.5).

Das ist ein wesentliches Unterscheidungsmerkmal zu vielen anderen Prozessmodellierungstools, die ein solch strukturiertes Vorgehen nicht anbieten. Eine Auflistung und Beschreibung der einzelnen Elemente der Funktionsleiste finden Sie in Kapitel [Die Funktionsleiste](#).



Hinweis: Konstrukte wie *Parallelität* und *Auswahl* können beliebig viele Pfade haben, zu Beginn werden aber nur zwei Pfade eingefügt. Wenn Sie weitere Pfade hinzufügen wollen, dann markieren Sie den Anfangsknoten dieses Konstrukts und aktivieren den Menüpunkt *Zusätzliche Kante*.

Elemente bearbeiten

Elemente besitzen Eigenschaften wie z.B. einen Namen. Um diese Eigenschaften zu bearbeiten haben sie folgende Möglichkeiten:

- Doppelklick auf das Element
- Element auswählen und mittels Klick der rechten Maustaste das Kontextmenü des Elements öffnen und dort den Eintrag *Aktivitätseigenschaften* aktivieren.

- Element auswählen und in der Menüleiste das Menü *Bearbeiten* öffnen und dort den Eintrag *Aktivitätseigenschaften* aktivieren.

Jede dieser Aktionen öffnet ein Detailfenster, in dem die für dieses Element verfügbaren Eigenschaften bearbeitet werden können. Nähere Details dazu finden Sie in Kapitel [Die allgemeinen Eigenschaften eines Knotens](#) und folgende.

Elemente wie z.B. *Task* können Sie auch in der Größe verändert. Dazu müssen Sie das Element auswählen. Es erscheinen nun kleine Quadrate am Rand des Elements, mit dem Sie das Element in seiner Breite (Quadrate an den vertikalen Rändern), seiner Höhe (Quadrate an den horizontalen Rändern) oder in beiden Dimensionen zugleich (Quadrate an den Ecken) verändern können. Dazu müssen Sie den Mauszeiger über ein solches Quadrat bewegen (siehe Abbildung 7.6). Sobald dieser seine Darstellung verändert können Sie die Linie bei gedrückter linker Maustaste durch Bewegen der Maus verschieben.

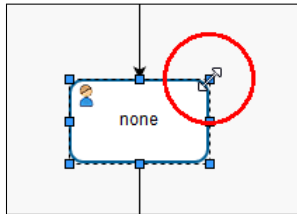


Abbildung 7.6: **Auswahlrahmen mit Angriffspunkten**



Hinweis: wenn ein Element nach seiner Auswahl keine solchen Quadrate anzeigt, dann ist es auch in seiner Größe nicht veränderbar. Dies ist bei allen Elementen der Fall, die durch einen Kreis oder eine Raute dargestellt werden.

Elemente verschieben

Ein bereits vorhandenes Element können Sie einfach per Drag & Drop an eine andere Position verschieben. Dazu klicken Sie auf das Element und halten die rechte Maustaste gedrückt. Nun können Sie die Maus bewegen und das Element durch Lösen der rechten Maustaste an die aktuelle Position verschieben (siehe Abbildung 7.7).

Sie können auch mehrere Elemente zugleich verschieben, indem Sie Ihre Auswahl auf diese Elemente erweitern. Dazu halten Sie die Umschalt-Taste gedrückt und klicken weitere Elemente an. Alternativ können Sie auch mit der Maus in einen leeren Teil des Zeichenbereichs klicken und mit gedrückter linker Maustaste den Mauszeiger bewegen und somit einen Auswahlbereich festlegen.

Verbindungslinien können auch verschoben werden um so z.B. Überschneidungen mit anderen Linien/Elementen zu vermeiden. Dazu müssen Sie die Linie auswählen, indem Sie auf die Linie klicken. Nun werden auf der Linie kleine Quadrate eingeblendet. Mit Ausnahme der Quadrate am Anfang und Ende einer Linie können Sie mit diesen die Ausrichtung der Linie verändert. Dazu müssen Sie den Mauszeiger über ein solches

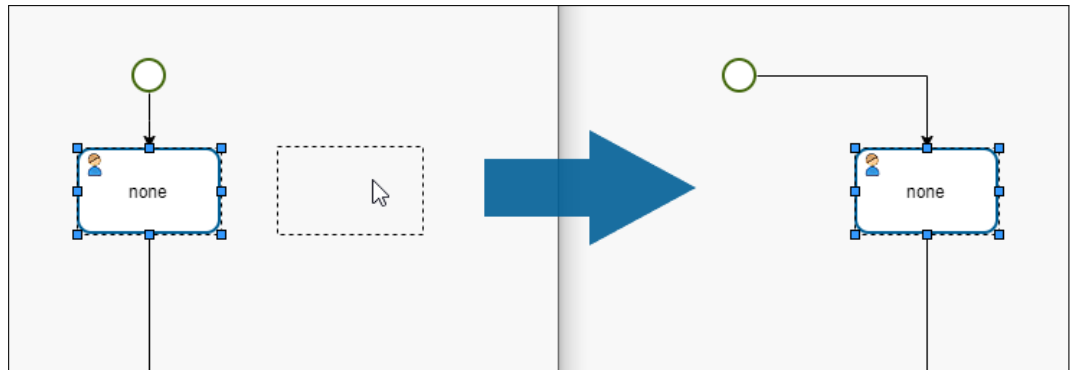


Abbildung 7.7: **Element verschieben**

Quadrat bewegen. Sobald dieser seine Darstellung verändert können Sie die Linie bei gedrückter linker Maustaste durch Bewegen der Maus verschieben – wobei vertikale Linien horizontal und horizontale Linien vertikal geändert werden können (siehe Abbildungen 7.8 und 7.9).

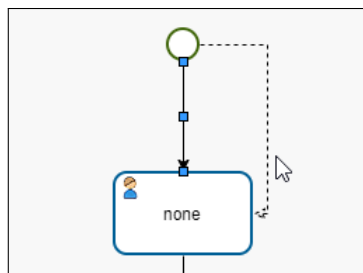


Abbildung 7.8: **Verschieben einer vertikalen Linie**



Hinweis: mit dem Menüpunkt *Automatisch routen* können Sie eine manuell veränderte Linie wieder in ihren Ursprungszustand zurück versetzen. Diese Funktion steht auch im Kontextmenü einer Linie zur Verfügung.

Element löschen

Um ein Element zu löschen, müssen Sie dieses zuerst auswählen und dann entweder die *ENTF* Taste auf ihrer Tastatur drücken oder den Menüpunkt *Löschen* aktivieren.

Zeichenbereich vergrößern

Wenn der Zeichenbereich zu klein wird, dann können Sie diesen vergrößern. Dazu müssen Sie ein beliebiges Element mittels Drag & Drop zu jenem Rand des Zeichenbereichs schieben, den Sie ausdehnen möchten.

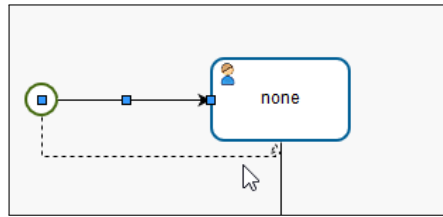


Abbildung 7.9: **Verschieben einer horizontalen Verbindung**



Hinweis: Diese Operation kann manchmal etwas störrisch sein. In diesem Fall bewegen Sie bitte die Maus am inneren Rand des Fensters mit kleinen Bewegungen noch oben und unten bzw. nach links und rechts – dadurch sollte der Zeichenbereich weiter in die gewünschte Richtung wachsen.

Datenmodellierung

Jeder Prozess erfasst und verarbeitet Daten, seien dies Nutzdaten für die Benutzer oder Steuerdaten, um den Prozessablauf zu lenken. In *EPworX* werden diese Daten strukturiert über Formulare modelliert – siehe dazu Kapitel [Formulare](#). Die so definierten Formulare können Sie wie folgt Ihrem Prozess zuordnen:

- Aktivieren Sie im Menü *Prozess* den Menüpunkt *Eigenschaften*
- Im nun angezeigten Eigenschaftsfenster wechseln Sie auf den Reiter *Formulare*
- Klicken Sie auf die Schaltfläche *Neu* (Plus-Icon), füllen Sie die Felder wie in Kapitel [Prozesseigenschaften](#) Abschnitt *Formulare* aus und klicken Sie auf *Ok*
- Jetzt klicken Sie auf *Speichern und Schließen* im Eigenschaftsfenster



Hinweis: durch *Speichern und Schließen* werden nur die in diesem Fenster gemachten Änderungen in den Editor übernommen. Der Prozess selbst wird noch nicht gespeichert, das erfolgt nur über die Funktion *Speichern* des Editors selbst.

Damit diese Formulare durch einen Benutzer gesehen und bearbeitet werden können, müssen Sie diese den jeweiligen interaktiven Schritten, den Tasks, zuordnen. Dazu wählen Sie den gewünschten Task in ihrem Prozessmodell aus und aktivieren die Funktion *Aktivitätseigenschaften*. In dem nun angezeigten Fenster können Sie alle Formulare, die Sie Ihrem Prozess hinzugefügt haben, dem ausgewählten Task zuordnen. Zusätzlich können Sie auch die Sichtbarkeiten für die einzelnen Formularfelder (änderbar/nur lesbar/unsichtbar) in diesem Task festlegen. Mehr Details zu diesem Fenster und seinen Feldern und Funktionen finden Sie in Kapitel [Eigenschaften einer Aktivität definieren](#).

7.2.3 Die Funktionen der Menüleiste

Das Menü *Prozess*

- **Neu:** Mit dieser Funktion können Sie den gegenwärtigen Inhalt im Zeichenbereich löschen und neu mit der Modellierung eines Prozesses beginnen.
- **Öffnen:** Mit dieser Funktion können Sie einen bestehenden Prozess öffnen und ihn weiter bearbeiten.
- **Speichern:** Mit dieser Funktion speichern Sie Ihre bisher gemachten Änderungen unter dem aktuellen Prozessnamen ab. Das System informiert Sie, ob die Sicherung erfolgreich durchgeführt wurde oder nicht. Wenn Schritte im Prozess nicht ausreichend spezifiziert wurden (z.B. kein Task zu einer Aktivität zugewiesen), wird der Prozess zwar gespeichert, aber auf *inaktiv* gesetzt. Der Prozess muss vor seiner Verwendung wieder händisch aktiviert werden (siehe Kapitel [Prozesse](#)).



Hinweis: Das Ausführen dieser Funktion ist nur möglich, wenn Sie zuvor über das Menü *Prozess* die Funktion *Eigenschaften* ausgeführt und für den Prozess einen eindeutigen Namen vergeben haben.

- **Speichern unter ...:** Mit dieser Funktion können Sie die bestehende Prozessspezifikation unter einem neuen Namen abspeichern. Dazu erscheint der in Abbildung 7.10 dargestellte Dialog.

The dialog box is titled 'Speichern unter' (Save As). It contains the following fields and controls:

- Id: ***: Input field containing 'hr_vacation'.
- Name: ***: Input field containing 'proc_vacation'.
- Version: ***: Input field containing '1'.
- Aktiv:**: A checkbox that is checked.
- Buttons**: 'Ok' and 'Abbrechen' (Cancel) buttons at the bottom right.

Abbildung 7.10: **Speichern unter**

- **Eigenschaften:** Mit dieser Funktion werden die Prozesseigenschaften geöffnet (siehe Abschnitt [Prozesseigenschaften](#)).
- **Tasks:** Mit Hilfe der Task-Maske legen Sie die Tasks fest, die Sie beim Ändern eines Akteurs im Arbeitskorb einem Empfänger in der Empfängerliste zuordnen können.
- **Eskalationen:** Die Reaktion des Prozesses bzw. der Prozesselemente auf Timeouts wird mit dieser Funktion spezifiziert (siehe Kapitel [Eskalationen](#)).

- **Prozesspläne:** Die Verwaltung von Zeitplänen für diesen Prozess (siehe Abschnitt [Prozesspläne](#)).
- **Drucken ...:** Prozess wie in Seitenformat definiert drucken (Standarddruckfenster).
- **Grafik herunterladen:** Mit Hilfe dieser Funktion können Sie den angezeigten Prozess als Bild herunterladen.
- **Beenden:** Mit dieser Funktion steigen Sie aus dem Prozesseditor aus. Sollten Sie noch nicht gespeicherte Änderungen gemacht haben, so erscheint ein Dialogfenster, welches den Abbruch des Beenden-Vorgangs oder das Abspeichern des Prozesses ermöglicht.

Das Menü *Bearbeiten*

- **Rückgängig:** Mit dieser Funktion können die letzten n Schritte im Zeichenbereich rückgängig gemacht werden.
- **Wiederherstellen:** Diese Funktion ist analog zu *Rückgängig*.
- **Ausschneiden:** Diese Funktion erlaubt es, Elemente aus einer Stelle im Prozess auszuschneiden und an einer anderen Stelle im Prozess wieder einzufügen. Wenn Sie Elemente ausschneiden, können Sie diese nur 1x wieder einfügen. Es wird hierbei das Original verschoben und keine Kopie erzeugt, was zur Folge hat, dass alle Einstellungen (Akteure, Formular-Sichtbarkeiten, etc.) für die ausgeschnittenen Elemente erhalten bleiben.
- **Kopieren:** Diese Funktion erlaubt es, Elemente aus einer Stelle im Prozess zu kopieren und an einer anderen Stelle im Prozess wieder einzufügen. Wenn Sie Elemente kopieren, können Sie diese beliebig oft wieder einfügen. Dabei werden nicht alle Einstellungen (z.B. Formular-Sichtbarkeiten) für die kopierten Elemente übernommen.
- **Einfügen:** Mit Hilfe dieser Funktion können Sie vorher ausgeschnittene bzw. kopierte Elemente an der ausgewählten Stelle einfügen. Wählen Sie zuerst diese Funktion und fügen Sie dann mit einem Klick auf die gewünschte Stelle das Element ein. Das Element in der Zwischenablage wird neben dem Mauszeiger dargestellt, bis Sie entweder das Element eingefügt haben, eine andere Funktion ausgewählt haben oder auf die Taste *Escape* drücken.
- **Löschen:** Mit dieser Funktion können einzelne Elemente gelöscht werden. Wenn unter diesem Knoten (z.B. Verzweigung) noch weitere Elemente enthalten sind, werden Sie gefragt, ob Sie diesen Knoten wirklich löschen wollen. Klicken Sie einfach auf das Element und wählen Sie dann diese Funktion aus.
- **Aktivitätseigenschaften:** Diese Funktion öffnet die Detailansicht zu dieser Aktivität (Batch), wo sie Akteure und Formulare hinzufügen können.

- **Eskalationen:** Mit Hilfe dieser Funktion können Sie für einen Task-, Synchronisieren- oder Batch-Schritt Eskalationen festlegen, die gefeuert werden sollen. Nähere Informationen dazu finden Sie in den Abschnitten [Registerkarte: Eskalationen](#) und [Eskalationen](#).
- **Taskeigenschaften:** Mit dieser Funktion werden die Task-Eigenschaften für diese Aktivität geöffnet (siehe Kapitel [Tasks](#)) bzw. im Falle eines Subprozesses ein neues Prozesseditor-Fenster mit dem Prozess geöffnet.
- **Planeintrag:** Wenn mindestens ein Zeitplan für diesen Prozess definiert wurde, wird mit dieser Funktion die Maske *Planeintrag* für diesen Prozessschritt angezeigt (siehe Abschnitt [Planeintrag](#)). Eine generelle Beschreibung zu Prozessplänen ist im Kapitel [Prozesspläne](#) vorhanden.
- **Annotieren:** Mit Hilfe dieser Funktion kann ein zuvor ausgewählter Knoten mit einer Anmerkung versehen werden, d.h. es wird eine Verlinkung zwischen Knoten und Anmerkung erzeugt. Machen Sie einen Doppelklick in das Textfeld, um einen Text hinzuzufügen und bestätigen Sie die Eingabe, in dem Sie mit der Maus in einen Bereich außerhalb des Textfeldes klicken. Der Text kann auch internationalisiert werden. In diesem Fall muss er mit @@@ beginnen und mit @@ enden, z.B. @@@myannotation@@.
- **Ausnahmebehandlung hinzufügen/entfernen:** Diese Funktion ist nur für den Knoten *Ausgangsnachricht* (INVOKE) verfügbar und erlaubt die Definition eines Exception Flows bzw. das Entfernen eines bestehenden. Der Exception Flow (Ausnahmebehandlung) wird dann aufgerufen, falls beim INVOKE-Knoten etwas schief läuft (z.B. Server antwortet nicht).
- **Zusätzliche Kante:** Mit Hilfe dieser Funktion können Sie den Elementen *Choice*, *UND-Parallelität* und *ODER-Parallelität* eine weitere Kante (Pfad) hinzufügen. Selektieren Sie dafür das gewünschte Objekt und wählen Sie danach diesen Menüpunkt aus.
- **Alles markieren:** Durch diese Funktion werden alle Elemente im Zeichenbereich markiert.
- **Markierung umkehren:** Diese Funktion markiert alle jene Elemente im Zeichenbereich, die vorher nicht markiert waren.

Das Menü *Ansicht*

- **Zoom:** Diese Funktion beinhaltet folgende 3 Subfunktionen:
 1. *Normale Ansicht:* Der Zeichenbereich wird in der Größe angezeigt, die beim Start des Prozesseditors vorgegeben ist.
 2. *Vergrößern:* Der angezeigte Bereich wird vergrößert.
 3. *Verkleinern:* Der angezeigte Bereich wird verkleinert.

- **Ausrichten:** Mit dieser Funktion können die Elemente im Zeichenbereich entsprechend dem äußersten Element (links, rechts, oben, unten) ausgerichtet werden.
- **Endknoten anzeigen:** Diese Funktion markiert die Endknoten des ausgewählten Elements.
- **Automatisch Routen:** Sie haben beim orthogonalen Routen die Möglichkeit Kanten zu verschieben. Wählen Sie die Kante aus und ziehen Sie sie in die entsprechende Richtung. Damit die Kante wieder automatisch geroutet wird, wählen Sie zuerst die Kante und anschließend diese Funktion aus.
- **Einstellungen:** Mit Hilfe dieser Funktion können Sie folgende Einstellungen tätigen:
 - Am Raster ausrichten: Wenn diese Checkbox aktiviert ist, werden die Elemente und Pfade nach den Rasterlinien ausgerichtet.
 - Raster anzeigen: Ist diese Checkbox aktiviert, wird ein Raster im Zeichenbereich angezeigt.
 - Seitenränder anzeigen: Diese Funktion zeigt die Seitenränder im Zeichenbereich an.
 - Seitenformat: Hier können Sie ein Format auswählen, das für die Druckfunktion relevant ist.
 - Drucker-Zoom: Hier können Sie einstellen, wie vergrößert bzw. verkleinert der Prozess ausgedruckt werden soll.
 - Hilfskanten nicht anzeigen: Wenn diese Checkbox aktiviert ist, dann werden Hilfskante von z.B. einem GOTO nicht angezeigt. Hilfskanten werden im Prozesseditor als hellgraue strichlierte Linie angezeigt.
 - Goto-Hilfe nicht anzeigen: Wenn diese Checkbox aktiviert ist, wird das Hilfefenster beim Einfügen des Gotos nicht angezeigt.
 - Willkommensnachricht nicht anzeigen: Wenn diese Checkbox aktiviert ist, wird die Willkommensnachricht beim Start des Editors nicht angezeigt.

Das Menü *Hilfe*

- **Hilfe:** Hier wird die Hilfe-Seite des Prozesseditors angezeigt (siehe Kapitel [Das HTML-Interface](#)).

Die Symbolleiste

Über die Symbole haben Sie die Möglichkeit, die am häufigsten verwendeten Funktionen schnell zu erreichen, ohne in den vorher beschriebenen Menüs danach zu suchen:

- **Neu:** siehe Abschnitt [Das Menü Prozess](#)
- **Öffnen:** siehe Abschnitt [Das Menü Prozess](#)
- **Speichern:** siehe Abschnitt [Das Menü Prozess](#)

- **Rückgängig:** siehe Abschnitt [Das Menü Bearbeiten](#)
- **Wiederherstellen:** siehe Abschnitt [Das Menü Bearbeiten](#)
- **Ausschneiden:** siehe Abschnitt [Das Menü Bearbeiten](#)
- **Kopieren:** siehe Abschnitt [Das Menü Bearbeiten](#)
- **Einfügen:** siehe Abschnitt [Das Menü Bearbeiten](#)
- **Löschen:** siehe Abschnitt [Das Menü Bearbeiten](#)
- **Normale Ansicht:** siehe Abschnitt [Das Menü Ansicht](#)
- **Vergrößern:** siehe Abschnitt [Das Menü Ansicht](#)
- **Verkleinern:** siehe Abschnitt [Das Menü Ansicht](#)
- **Aktivitätseigenschaften bei Doppelklick auf Knoten anzeigen:** Wenn diese Schaltfläche aktiviert ist und Sie auf einen Knoten einen Doppelklick ausführen, werden die *Aktivitätseigenschaften* angezeigt.
- **Taskeigenschaften bei Doppelklick auf Knoten anzeigen:** Wenn diese Schaltfläche aktiviert ist und Sie auf einen Knoten einen Doppelklick ausführen, werden die *Taskeigenschaften* angezeigt bzw. im Falle eines Subprozesses ein neues Prozesseditor-Fenster mit dem Prozess geöffnet.
- **Planeintrag bei Doppelklick auf Knoten anzeigen:** Wenn mindestens ein Zeitplan für diesen Prozess definiert wurde, diese Schaltfläche aktiviert wurde und Sie auf einen Knoten einen Doppelklick ausführen, so wird die Maske *Planeintrag* für diesen Prozessschritt angezeigt (siehe Abschnitt [Planeintrag](#)). Eine generelle Beschreibung zu Prozesspläne ist im Kapitel [Prozesspläne](#) vorhanden.

Das Kontextmenü

Das Kontextmenü stellt eine schnelle und bequeme Art der Bedienung im Zeichenbereich dar. Sie erreichen dieses Menü, indem Sie im Zeichenbereich ein Element auswählen und dabei die rechte Maustaste betätigen. Das Kontextmenü beinhaltet dabei zum Teil die gleichen Elemente wie die Menüleiste:

- **Ausschneiden:** siehe Abschnitt [Das Menü Bearbeiten](#)
- **Kopieren:** siehe Abschnitt [Das Menü Bearbeiten](#)
- **Löschen:** siehe Abschnitt [Das Menü Bearbeiten](#)
- **Aktivitätseigenschaften:** siehe Abschnitt [Das Menü Bearbeiten](#)
- **Eskalationen:** siehe Abschnitt [Das Menü Bearbeiten](#)
- **Taskeigenschaften:** siehe Abschnitt [Das Menü Bearbeiten](#)
- **Planeintrag:** siehe Abschnitt [Das Menü Bearbeiten](#)

- **Annotieren:** siehe Abschnitt [Das Menü Bearbeiten](#)
- **Ausnahmebehandlung hinzufügen/entfernen:** siehe Abschnitt [Das Menü Bearbeiten](#)
- **Zusätzliche Kante:** siehe Abschnitt [Das Menü Bearbeiten](#)



Hinweis: Um schneller im Prozesseditor arbeiten zu können, werden *Shortcuts* unterstützt. Der jeweilige Shortcut zu einer Funktion wird gleich daneben angezeigt.

7.2.4 Prozesseigenschaften

Mit Hilfe der Prozesseigenschaften-Maske können Sie Einstellungen bezüglich des Prozesses treffen. Die einzelnen Reiter werden hier nun beschrieben:

- **Allgemein:** Analog zu [Prozesse](#), jedoch ist das Feld *Änderung durchführen am* nicht vorhanden.
- **Formulare:** Hier können Sie die Formulare des Prozesses, den Sie gerade bearbeiten (oder bearbeiten möchten), bestimmen. Durch einen Mausklick auf die Schaltfläche *Hinzufügen*, öffnet sich ein neues Fenster (siehe Abbildung 7.11), in dem Sie zunächst ein Formular auswählen und anschließend weitere Optionen zu einem Formular angeben können. Der Dialog *Formular hinzufügen* beinhaltet folgende Informationen:
 - Id: Hier muss die Id für die eindeutige Identifizierung des Prozessformulars eingegeben werden.
 - Name: Der Name für das Prozessformular kann hier festgelegt werden.
 - Typ: In dieser Tabelle werden sämtliche Formulartypen des Systems aufgelistet. Mit dem Button *Neu* rechts neben der Liste können Sie weitere Formulare hinzufügen (siehe Kapitel [Formulare](#)).
 - Mode: Hier können Sie den Verwendungszweck des hinzufügenden Formulars näher spezifizieren.
 - lokal:* Das Formular wird nur innerhalb des Prozesses verwendet und wird somit innerhalb des Prozesses erzeugt.
 - inout:* Das Formular wird dem Prozess beim Aufruf übergeben, im Prozess bearbeitet und danach wieder nach außen weitergeleitet.
 - Basisformular für View: Wählen Sie hier das Basisformular aus, wo eine View hinzugefügt werden soll.
 - Ok: Durch einen Mausklick auf diese Schaltfläche wechseln Sie wieder in das Fenster *Prozesseigenschaften*, wobei das von Ihnen eingegebene Formular nun in der Formularliste des Prozesses aufscheint.
 - Abbrechen: Durch einen Mausklick auf diese Schaltfläche wechseln Sie wieder zum Fenster *Prozesseigenschaften*, jedoch ohne ein Formular eingefügt zu haben.

Durch einen Mausklick auf die Schaltfläche *Entfernen* können Sie ein Formular aus der Liste dieses Prozesses löschen. Mit der Schaltfläche *Bearbeiten* haben Sie die Möglichkeit, das Formular zu bearbeiten. Bei Änderung einer bestehenden Formular-Id werden bei bestehenden Objekten wie z.B. Aktivitäten die Id's automatisch ersetzt. Bei Strukturen, die z.B. eine Bedingung beinhalten, müssen Sie die Id selbst ändern, anderenfalls werden Sie beim Speichern des Prozesses darauf hingewiesen. Vor dem Aufruf der Funktionen *Bearbeiten* bzw. *Löschen* muss ein Formular ausgewählt sein.



Hinweis: Werden View-Formulare als Prozessformulare verwendet, so sollten diese nur bei der Zuordnung zu Tasks verwendet werden, damit die Sichtbarkeitseinschränkungen greifen. Für alle anderen Fälle (z.B. Prozesssteuerung über Formularfelder) sollte immer das entsprechende Basis-Formular verwendet werden!

- **Source:** Analog zu [Prozesse](#).
- **Komponenten:** Analog zu [Prozesse](#).
- **Formular-Sichtbarkeiten:** Analog zu [Prozesse](#).
- **Eskalation:** Analog zu [Prozesse](#).
- **Historie:** Analog zu [Prozesse](#).
- **Zugriff:** Analog zu [Prozesse](#).
- **Referenzen von:** Analog zu [Prozesse](#).

7.2.5 Tasks

Mit Hilfe der Task-Maske (siehe Abbildung [7.13](#)) legen Sie die Tasks fest, die Sie beim Ändern eines Akteurs im Arbeitskorb einem Empfänger in der Empfängerliste zuordnen können. Diese Funktion wird standardmäßig im Arbeitskorb nicht angeboten und muss erst in der GUI-Konfiguration beim Knotentyp

Arbeitskorb → *Funktionen*

hinzugefügt werden (*Action Key*: adHoc). Weiter Details zur GUI-Konfiguration/Knoteneigenschaften finden Sie im Kapitel [Knoteneigenschaften](#).

Task hinzufügen

Folgende Schritte sind notwendig:

1. Wählen Sie im Menü *Prozess* den Eintrag *Tasks* aus. Es öffnet sich die Tabelle *Tasks*. Im Toolbar aktivieren Sie die Funktion *Neu*. Es öffnet sich der in Abbildung [7.13](#) dargestellte Dialog.
2. **Task:** Wählen Sie einen Task aus. Alternativ können Sie auch einen neuen erstellen, der danach auch hinzugefügt wird.

7. Prozessdefinition

Id: * form_vacation

Name:

Typ: *

- hr_evaluation (Bewertung 1)
- hr_expense (Expense 1)
- hr_recruiting (Personalauswahl 1)
- hr_sickness (Krankheit 1)
- hr_timeitem (Time Item 1)
- hr_vacation (Urlaub 1)**
- hr_vacationbalance (vacation_balance 1)
- hr_worktime (Arbeitszeit 1)
- Item-besides-item (Besides-Item 1)

Mode: * ☒ Lokal ☐ InOut

Basisformular:

Ok Abbrechen

Abbildung 7.11: Formular hinzufügen

Allgemein **Source** Graph Komponenten Formular-Sic... Eskalationen Funktionen OrdnerEinst... Dokumenten... Historie Zugriff Referenzen Entscheidun...

```
process hr_vacation(form_vacation hr_vacation "form_vacation")
version 1;
name "proc_vacation";
description "desc_proc_vacation";
subject "{form_vacation.employee}, {form_vacation.vacfrom,date} - {form_vacation.vactto,date}";
application staffprocs;
begin
  <label_4>
    all vacation_request(form_vacation);
    manager vacation_approve(form_vacation);
    if xpath:"${form_vacation/approved} = 'true'" then "@@approved@"
      andpar
        form_vacation.employee vacation_informapproved(form_vacation);
        |
        hr!home vacation_process(form_vacation);
      end;
    else
      form_vacation.employee vacation_informrefused(form_vacation);
    end;
  end;
end
```

BPMN anzeigen BPMN downloaden XWDL anzeigen XWDL downloaden

Löschen Speichern und Schließen Speichern Schließen

Abbildung 7.12: WDL Anzeige

Task: Anfordern

Schrittname:

Schritt-Id: label_4

Icon: Vorschau:

Füllfarbe:

Akteur(e): Alle

Überspringbar: ☐

Verfügbare Formulare:

Hinzugefügte Formulare: Urlaub

Ok Abbrechen

Abbildung 7.13: **Tasks**

3. Schrittname (optional): Bezeichnung des Schrittes, der auch internationalisiert werden kann, indem der zu übersetzende Wert mit @@@ beginnt und mit @@ aufhört, z.B. @@@myname@@.
4. Schritt-Id (optional): Diese muss innerhalb des Prozesses eindeutig sein und unterliegt den gleichen syntaktischen Bedingungen wie eine *EPworX* -Id. Eine eingegebene Schritt-Id wirkt sich bei der Akteurauswahl für andere Aktivitäten aus.
5. Formulare (optional): Aktivieren Sie die Schaltfläche *Hinzufügen* (Pfeil nach rechts). Nun erscheint das hinzugefügte Formular in der Liste *Hinzugefügte Formulare*. (Wenn Sie dem Task mehrere Formulare hinzufügen wollen, dann wiederholen Sie die Schritte entsprechend oft.)
6. Aktivieren Sie die Schaltfläche *Ok*, um den Task der Tabelle hinzuzufügen.

Wenn Sie mehrere Tasks hinzufügen möchten, wiederholen Sie einfach die Schritte 2 bis 5 entsprechend oft, bevor Sie Schritt 6 ausführen.

Task entfernen

Folgende Schritte sind notwendig:

1. Wählen Sie im Menü *Prozess* den Eintrag *Tasks* aus. Es öffnet sich die Tabelle *Tasks*.
2. Wählen Sie in der Tabelle einen oder mehrere Einträge aus.
3. Aktivieren Sie die Toolbarfunktion *Löschen* (rotes Kreuz).

Löschen eines Formulars, das einem Task zugeordnet ist

Die folgenden Schritte sind notwendig:

1. Wählen Sie im Menü *Prozess* den Eintrag *Tasks* aus. Es öffnet sich die Tabelle *Tasks*. Wählen Sie einen Tabelleneintrag aus und aktivieren Sie im Toolbar die Funktion *Bearbeiten*. Es öffnet sich der in Abbildung 7.13 dargestellte Dialog.
2. Wählen Sie das Formular in der Liste *Hinzugefügte Formulare* aus.
3. Aktivieren Sie die Schaltfläche *Entfernen* (Pfeil nach links). Nun erscheint das hinzugefügte Formular in der Liste *Verfügbare Formulare*.
4. Aktivieren Sie die Schaltfläche *Ok*.

Wenn Sie mehrere Formulare löschen wollen, wiederholen Sie die Schritte 1 bis 3 entsprechend oft, bevor Sie Schritt 4 ausführen.

7.2.6 Eskalationen

Diese Funktion ermöglicht die Definition von Eskalationsschritten, die im Falle einer Eskalation ausgeführt werden können. Ein Eskalationsschritt kann entweder ein Task sein, der ausgeführt werden soll oder ein Prozess, der gestartet werden soll. In beiden Fällen muss ein Eskalationsschritt-Objekt angelegt werden, das dann bei der Definition einer Eskalation in der Auswahlliste von *Schritt starten* ausgewählt werden kann (siehe Abschnitt [Registerkarte: Eskalationen](#)).

Eskalation hinzufügen

Folgende Schritte sind notwendig:

1. Wählen Sie im Menü *Prozess* den Eintrag *Eskalationen* aus. Es öffnet sich die Tabelle *Eskalationen*. Im Toolbar aktivieren Sie die Funktion *Neu*. Es öffnet sich der in Abbildung 7.14 dargestellte Dialog.
2. **Task:** Wählen Sie einen Task aus. Alternativ können Sie auch einen neuen erstellen, der danach auch hinzugefügt wird.

Task

Prozess

Task: *

adHoc

Schrittname:

AdHoc

Schritt-Id: *

adhoc_schritt

Akteur(e):

Alle

Verfügbare Formulare:

Hinzugefügte Formulare:

Rechnung

Ok

Abbrechen

Abbildung 7.14: Eskalation

- Schrittname (optional): Bezeichnung des Schrittes, der auch internationalisiert werden kann, indem der zu übersetzende Wert mit @@@ beginnt und mit @@ aufhört, z.B. @@@myname@@.
- Schritt-Id:** Diese muss innerhalb des Prozesses eindeutig sein und unterliegt den gleichen syntaktischen Bedingungen wie eine *EPworX* -Id. Eine eingegebene Schritt-Id wirkt sich bei der Akteurauswahl für andere Aktivitäten aus.
- Akteur(e): Hier können Sie einen oder mehrere Akteure aus den Auswahlmöglichkeiten Rolle, Benutzer, Task, Formularfeld oder Methode festlegen (siehe Abschnitt [Eigenschaften einer Aktivität definieren](#) für mehr Details zur Akteurauswahl). Der Eskalationsschritt wird dann entsprechend der ausgewählten Akteure in den entsprechenden Arbeitskörben angezeigt.
- Formulare (optional): Aktivieren Sie die Schaltfläche *Hinzufügen* (Pfeil nach rechts). Nun erscheint das hinzugefügte Formular in der Liste *Hinzugefügte Formulare*. (Wenn Sie der Eskalation mehrere Formulare hinzufügen wollen, dann wiederholen Sie die Schritte entsprechend oft.)

7. Aktivieren Sie die Schaltfläche *Ok*, um die Eskalation der Tabelle hinzuzufügen.

Wenn Sie mehrere Eskalationen vom Typ *Task* hinzufügen möchten, wiederholen Sie einfach die Schritte 2 bis 6 entsprechend oft, bevor Sie Schritt 7 ausführen.

Alternativ können Sie auch einen Prozess als Eskalation hinzufügen. Dafür führen Sie Schritt 1 durch, wechseln auf den Reiter *Prozess* und wählen im 2. Schritt statt einen *Task* einen *Prozess* aus. Die restlichen Schritte (ausgenommen Akteurauswahl) sind analog zum Reiter *Task*.

Eskalation entfernen

Folgende Schritte sind notwendig:

1. Wählen Sie im Menü *Prozess* den Eintrag *Eskalationen* aus. Es öffnet sich die Tabelle *Eskalationen*.
2. Wählen Sie in der Tabelle einen oder mehrere Einträge aus.
3. Aktivieren Sie die Toolbarfunktion *Löschen* (rotes Kreuz).

7.2.7 Prozesspläne

Mit Hilfe dieser Funktion können Sie für den Prozess Zeitpläne generieren, um realistische Deadlines für den Prozess und die Prozessschritte zu haben. Pro Prozessdefinition kann es mehrere Prozesspläne geben, z.B. je nach Aufgabentyp, der irgendwann in der Prozessanfangsphase festgelegt wurde. Für diese Typisierung dient das Attribut *Plantyp*, das nur 1x pro Prozessdefinition vergeben werden kann. Plantypen werden pro Applikation im Submenü von *Prozesse* verwaltet und müssen vor der Definition eines Prozessplans festgelegt werden (siehe dazu Abschnitt [Plantypen](#)).

Nach Auswahl des Menüpunktes *Prozesspläne* wird ein Dialog geöffnet, in dem alle Prozesspläne dieses Prozesses verwaltet werden können (siehe Abbildung 7.15). Der Toolbar dieses Dialogs beinhaltet neben den Funktionen zum Erstellen und Editieren von Prozessplänen (Neu, Bearbeiten, Löschen) auch eine Kopier-Funktion, um weitere ähnliche Prozesspläne zu erstellen. Beim Aufruf der Funktion *Kopieren* wird zuvor ein Zwischendialog geöffnet, wo ein Plantyp definiert werden muss. Dabei ist zu beachten, dass der Plantyp nicht bereits für einen anderen Prozessplan in dieser Prozessdefinition verwendet wurde! Nach Aktivieren der Schaltfläche *Ok* wird ein weiterer Prozessplan mit dem ausgewählten Plantyp erzeugt, wobei die Planeinträge eine exakte Kopie des Originals sind.



Als weitere Funktion steht *Vergleiche Pläne* zur Verfügung, um mehrere Prozesspläne im Detail miteinander zu vergleichen. Dazu selektieren Sie einfach mehrere Prozesspläne und führen die Funktion aus, die ein neues Fenster mit den entsprechenden Informationen öffnet.

Prozessplan

×

Typ: *

My Plan

▼

Name:

My Plan Entry

Default:

☐

Berechnung:

Vom Beginn ☒ Vom Ende ☐

Dauer:

0

↑ Position	Schritt	Dauer	Beendet	Meilenstein		
1	Anfordern	2h	2h	My Milenstein		
2	Genehmigen	0	0			
3	Abgelehnt	0	0			
4	Genehmigt	0	0			
5	Verbuchen	0	0			
6	andjoin	0	0			

Löschen

Speichern und Schließen

Speichern

Abbrechen

Abbildung 7.15: **Prozesspläne**

Definition eines Prozessplans

Nachdem im Toolbar die Funktion *Neu* bzw. *Bearbeiten* aktiviert wurde, öffnet sich der Dialog *Prozessplan*, der sich wie folgt gliedert (siehe Abbildung 7.15):

- **Typ:** Der Plantyp ist ein obligatorisches Feld und kann pro Prozessdefinition nur 1x in einem Plan verwendet werden. Plantypen werden in der Administration von *EPworX* im Submenü von *Prozesse* definiert (siehe dazu Abschnitt [Plantypen](#)).
- **Name:** Ein Prozessplan kann optional noch einen Namen besitzen, der eine Kurzbeschreibung des Prozessplans darstellt.
- **Default:** Mit dieser Option wird festgelegt, ob der Prozessplan standardmäßig beim Prozessstart angewendet werden soll, d.h. beim Starten des Prozesses wird auf Basis des Default-Prozessplans sofort ein Zeitplan generiert und im Reiter *Plan* dargestellt. Bitte beachten Sie, dass pro Prozessdefinition nur 1 Default-Prozessplan festgelegt werden kann!
- **Berechnung:** Hier wird die Art der Berechnung des Prozessplans definiert. Die

Berechnung des Prozessplans erfolgt entweder *Vom Beginn*, also vom Zeitpunkt des Prozessstarts aus, oder *Vom Ende* und somit dem errechneten Prozessfertigstellungsdatum.

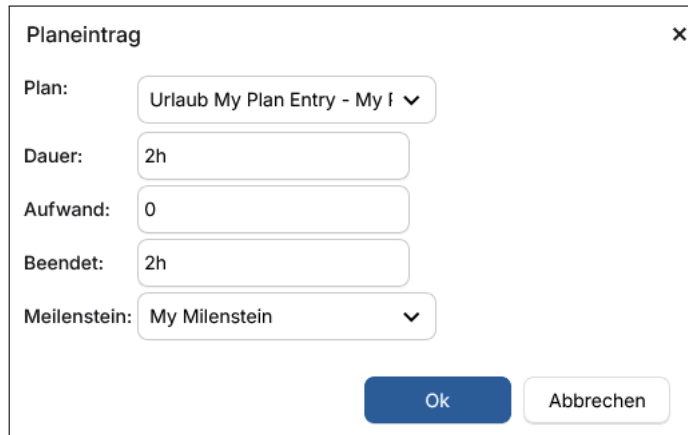
- **Dauer:** Dieses Attribut zeigt die errechnete Dauer der einzelnen Planeinträge von diesem Prozessplan an.
- **Planeinträge:** In dieser Tabelle werden alle Planeinträge aufgrund der vorhandenen Prozessschritte angezeigt. Die Tabelle beinhaltet folgende Informationen:
 - **Position:** Die Position der Prozessschritte aufgrund der y-Ausrichtung im Prozessgraph.
 - **Schritt:** Der Name des Prozessschritts.
 - **Dauer:** Hier wird die Dauer angezeigt, die dieser Prozessschritt in Anspruch nimmt. Die Dauer wird immer aufgrund der definierten Arbeitszeit berechnet (definierbar unter *Konfiguration/Kalender*)! Falls beim Task-Objekt das Attribut *Maximale Dauer* bereits festgelegt wurde, wird beim Generieren der Planeinträge dieser Wert im Feld *Dauer* gespeichert.
 - **Beendet:** Abstand des Endzeitpunkts des Schrittes zum Beginn des Prozesses. Dieser Wert wird entweder automatisch durch die Funktion *Plan berechnen* gesetzt oder kann manuell verändert werden.
 - **Meilenstein:** Der Name des Meilensteins, der auf diesen Planeintrag angewendet werden soll. Die Definition eines Meilensteins erfolgt in der *EPworX* Administration im Submenü von *Prozesse* (siehe Abschnitt [Meilensteine](#)).

Beim Erstellen eines Prozessplans ist diese Tabelle zunächst leer. Durch Aktivieren der Schaltfläche *Speichern* werden Planeinträge generiert, aber noch kein Plan berechnet. Sie können nun die Attribute in den Planeinträgen manuell definieren oder mittels Toolbarfunktion *Plan berechnen* automatisch berechnen lassen. Für die automatische Berechnung muss zuvor für jeden Planeintrag mindestens eine Dauer definiert werden! Um den Dialog zum Bearbeiten eines Plans zu öffnen, führen Sie entweder einen Doppelklick auf einen Planeintrag aus oder selektieren Sie zuerst einen Planeintrag und aktivieren dann die Toolbarfunktion *Bearbeiten* (siehe Abschnitt [Planeintrag](#)).

- **Löschen:** Durch Aktivieren dieser Schaltfläche wird der Prozessplan inkl. Planeinträge gelöscht und der Dialog geschlossen. Die Tabelle *Prozesspläne* wird anschließend aktualisiert.
- **Speichern und Schließen:** Durch Aktivieren dieser Schaltfläche werden die Änderungen am Prozessplan gespeichert und der Dialog geschlossen. Die Tabelle *Prozesspläne* wird anschließend aktualisiert.
- **Speichern:** Durch Aktivieren dieser Schaltfläche werden die Änderungen am Prozessplan gespeichert. Der Dialog *Prozessplan* bleibt aber weiterhin geöffnet.
- **Abbrechen:** Durch Aktivieren dieser Schaltfläche werden die Änderungen am Prozessplan verworfen und der Dialog geschlossen.

Planeintrag

Der Dialog *Planeintrag* besteht aus folgenden Attributen (siehe Abbildung 7.16):



The screenshot shows a dialog box titled "Planeintrag" with a close button (X) in the top right corner. Inside the dialog, there are five labeled input fields: "Plan:" with a dropdown menu showing "Urlaub My Plan Entry - My f"; "Dauer:" with a text input field containing "2h"; "Aufwand:" with a text input field containing "0"; "Beendet:" with a text input field containing "2h"; and "Meilenstein:" with a dropdown menu showing "My Milenstein". At the bottom right of the dialog are two buttons: "Ok" (blue) and "Abbrechen" (white with a grey border).

Abbildung 7.16: **Planeintrag**

- **Schritt oder Plan:** Wird der Planeintrag im Kontext eines Prozessplans geöffnet (siehe Abschnitt [Definition eines Prozessplans](#)), so wird hier der Name des Prozessschritts angezeigt (nicht editierbar). Wird dieser Dialog über die Funktion *Planeintrag* im Prozessgraphen aufgerufen (z.B. über das Kontextmenü eines Prozessschritts), so wird an dieser Stelle eine Auswahlliste für einen *Plan* angezeigt. Nachdem ein Plan ausgewählt wurde, werden die Informationen des entsprechenden Planeintrags angezeigt.
- **Dauer:** Hier wird die Dauer angezeigt, die dieser Prozessschritt in Anspruch nimmt. Falls beim Task-Objekt das Attribut *Maximale Dauer* bereits festgelegt wurde, wird beim Generieren der Planeinträge dieser Wert im Feld *Dauer* gespeichert. Wird ein ganzzahliger Wert angegeben, so wird dieser mit der Einheit s für Sekunden abgespeichert. Es besteht jedoch auch die Möglichkeit die Einheiten für den Wert selbst festzulegen wie z.B. 3d 4h 30m (= 3 Tage, 4 Stunden und 30 Minuten). Die Dauer wird immer aufgrund der definierten Arbeitszeit berechnet (definierbar unter *Konfiguration/Kalender*)!
- **Aufwand:** Hier wird der tatsächliche Aufwand zur Durchführung angegeben, der aber auf die Berechnung des Prozessplans keinerlei Auswirkung hat. Falls beim Task-Objekt das Attribut *Aufwand* bereits festgelegt wurde, wird beim Generieren der Planeinträge dieser Wert im Feld *Aufwand* gespeichert. Wird ein ganzzahliger Wert angegeben, so wird dieser mit der Einheit s für Sekunden abgespeichert. Es besteht jedoch auch die Möglichkeit die Einheiten für den Wert selbst festzulegen wie z.B. 3d 4h 30m (= 3 Tage, 4 Stunden und 30 Minuten). Der Aufwand wird immer aufgrund der definierten Arbeitszeit berechnet (definierbar unter *Konfiguration/Kalender*)!
- **Beendet:** Abstand des Endzeitpunkts des Schrittes zum Beginn des Prozesses.

Dieser Wert wird entweder automatisch durch die Funktion *Plan berechnen* gesetzt oder kann manuell verändert werden. Wird ein ganzzahliger Wert angegeben, so wird dieser mit der Einheit s für Sekunden abgespeichert. Es besteht jedoch auch die Möglichkeit die Einheiten für den Wert selbst festzulegen wie z.B. 3d 4h 30m (= 3 Tage, 4 Stunden und 30 Minuten).

- **Meilenstein:** Optionale Auswahl eines Meilensteins, der auf diesen Planeintrag angewendet werden soll. Die Definition eines Meilensteins erfolgt in der *EPworX* Administration im Submenü von *Prozesse* (siehe Abschnitt [Meilensteine](#)).
- **OK:** Durch Aktivieren dieser Schaltfläche werden die Änderungen des Planeintrags gespeichert und der Dialog geschlossen. Die Tabelle *Planeinträge* wird anschließend aktualisiert, sofern dies der Kontext für die Bearbeitung des Planeintrags ist.
- **Abbrechen:** Durch Aktivieren dieser Schaltfläche werden die Änderungen des Planeintrags verworfen und der Dialog geschlossen.

7.2.8 Die Funktionsleiste

In der Funktionsleiste befinden sich sämtliche Funktionen zur grafischen Modellierung von Prozessen in der Zeichenfläche. Mit der Maus können Sie in der Funktionsleiste den gewünschten Modus auswählen, um z.B. Aktivitäten anzulegen, Verbindungen zwischen Aktivitäten herzustellen oder grafische Objekte zu löschen.

Zum Einfügen einer Aktivität, eines Subprozesses oder Kontrollflusses wählen Sie die entsprechende Funktion aus, und klicken dann auf eine Kante im Prozessgrafen. Beim orthogonalen Routing können die Knoten durch Gedrückthalten der *Shift-Taste* und Ziehen mit der Maus horizontal bzw. vertikal verschoben werden. Ausgangspunkt für die Verschiebungsrichtung ist der ursprüngliche Standpunkt des Knotens.

Die Funktionen im Einzelnen:

- **Markierung:** Den Zeigermodus aktivieren Sie über die Funktion *Markierung*. In diesem Modus können Sie grafische Objekte (Aktivitäten) am Bildschirm neu positionieren und bearbeiten.

Wählen Sie dazu mit der Maus eine Aktivität aus und halten Sie die linke Maustaste gedrückt. Sie können nun die Aktivität an eine beliebige Stelle im Zeichenbereich bewegen. Weiters haben Sie auch die Möglichkeit nach dem Drag und Drop Prinzip Objekte im Zeichenbereich neu zu positionieren bzw. zu löschen.

- **Task:** Im Aktivitätsmodus können Sie neue Aktivitäten anlegen und die Eigenschaften von Aktivitäten spezifizieren. Folgende Vorgangsweise ist dazu erforderlich:
 - Gehen Sie in den Aktivitätsmodus, indem Sie mit der linken Maustaste in der Funktionsleiste auf die Schaltfläche *Aktivität* klicken.

- Positionieren Sie nun den Cursor im Zeichenbereich auf jene Kante, wo Sie eine neue Aktivität einfügen wollen.
 - Drücken Sie die linke Maustaste und eine neue Aktivität wird angelegt.
 - Führen Sie unmittelbar danach einen Doppelklick auf der Aktivität aus, um die Eigenschaften der Aktivität zu definieren. Diese Aktion können Sie auch nachträglich im Zeigermodus ausführen (siehe Zeigermodus oben).
- **Subprozess:** Analog dem Vorgehen beim Einfügen einer Aktivität können Subprozesse eingefügt werden.
- **Systemschritt:** Im Systemschrittmodus können Sie neue Systemschritte anlegen und die Methode für den jeweiligen Systemschritt angeben. Folgende Vorgangsweise ist dazu erforderlich:
 - Gehen Sie in den Systemschrittmodus, indem Sie mit der linken Maustaste in der Funktionsleiste auf die Schaltfläche *Systemschritt* klicken.
 - Positionieren Sie nun den Zeiger im Zeichenbereich an jener Stelle, wo Sie einen neuen Systemschritt einfügen wollen.
 - Drücken Sie die linke Maustaste und ein neuer Systemschritt wird angelegt.
 - Führen Sie unmittelbar danach einen Doppelklick auf den Systemschritt aus, um die Methode, die in diesem Systemschritt ausgeführt werden soll, zu definieren. Diese Aktion können Sie auch nachträglich im Zeigermodus ausführen (siehe Zeigermodus oben).
- **Batchschritt:** Analog zu *Systemschritt*. Allerdings wird hier der Name einer BatchAdapter Klasse eingetragen und es besteht die Möglichkeit, das Verhalten des Batch-Jobs zur Laufzeit detaillierter zu bestimmen.
- **Verzweigung:** Eine Verzweigung besteht aus einem If-Knoten (dargestellt durch ein grünes Kästchen), der die Bedingung enthält, und einem End-Knoten, der die Wege wieder zusammenführt. Vom If-Knoten zum End-Knoten gehen eine rote und eine grüne Kante. Bei der Ausführung des Workflows wird zuerst die Bedingung überprüft. Wenn diese wahr ist, wird die grüne Kante weiterverfolgt, andernfalls die rote.

Wenn Sie im Fall einer falschen Bedingung (rote Kante) nicht gleich eine Aktivität ausführen möchten, sondern die Nachfolgeaktivität wieder von einer Bedingung abhängig machen wollen, so können Sie wieder eine Verzweigung einführen, indem Sie in den Verzweigungsmodus wechseln und auf den roten Pfeil klicken, der die Verzweigungsaktivität mit der Nachfolgeaktivität verbindet.

Durch diese Vorgehensweise wird kein neues *if-end* erzeugt, sondern eine zusätzliche Bedingung in das bereits bestehende *if-end* eingefügt, wodurch im WDL-Script folgende Anweisungen entstehen:

```
if Bedingung 1 Then  
    Aktion 1
```

```
elsif Bedingung 2 Then
    Aktion 2
elsif Bedingung 3 Then
    Aktion 3
else
    Aktion 4
end
```

- **Auswahl:** Jeder Auswahlzweig hat einen Namen und eine optionale Bedingung. Zuerst werden die Bedingungen aller Zweige ausgewertet, die Zweige mit wahren Bedingungen werden dem Benutzer zur Auswahl angezeigt.

Um einen Choicezweig einzufügen, ist folgende Vorgangsweise erforderlich:

- Gehen Sie in den Choicemodus, indem Sie mit der linken Maustaste in der Funktionsleiste auf die Schaltfläche Choice klicken.
 - Positionieren Sie nun den Cursor im Zeichenbereich auf jener Kante, wo Sie einen neuen Choicezweig einfügen wollen.
 - Drücken Sie die linke Maustaste und eine neue Choice wird angelegt.
 - Sie sehen nun einen schwarzen Pfeil, wobei der schwarze Pfeil für einen möglichen Choicezweig steht, zu dem Sie, wie oben beschrieben, Aktivitäten hinzufügen können. Wenn Sie eine weitere Alternative hinzufügen wollen, wählen Sie die Choice aus und aktivieren Sie entweder in der Menüleiste unter *Bearbeiten* den Punkt *Zusatzpfad* oder klicken Sie mit der rechten Maustaste auf die Choice und wählen Sie im Kontextmenü *Zusatzpfad* aus.
- **While-Schleife:** Mit dem While-Konstrukt können Sie eine Schleife erzeugen, die ausgeführt wird, solange eine Bedingung wahr ist. Ähnlich wie bei der Verzweigung muss auch hier eine Bedingung angegeben werden. Aus dem *While* führt, wenn die Bedingung falsch (geworden) ist, ein roter Pfeil zur nächsten Anweisung. An den grünen Pfeil können Sie z.B. eine Aktivität setzen, die den Körper der While-Schleife bildet und bei wahrer Bedingung ausgeführt wird.
 - **Schleife:** Eine Loop-Schleife führt eine Anweisung aus und prüft danach, ob eine Bedingung wahr ist. An den schwarzen, aus der Loop führenden Pfeil können Sie z.B. eine Aktivität setzen, die mindestens einmal ausgeführt wird. Ist die nachfolgende Bedingung wahr, wird die Schleife verlassen, andernfalls wird die Aktivität nochmals ausgeführt. Die Aktivität nach dem roten Pfeil wird nur ausgeführt, wenn die Bedingung für den Schleifenabbruch falsch ist. Die Aktivität nach dem grünen Pfeil wird ausgeführt, sobald die Abbruchbedingung für die Schleife erfüllt ist.
 - **Parallel for:** Mit diesem Kontrollkonstrukt kann man über Subformulare oder über einen Iterator iterieren. Das Kontrollkonstrukt beginnt mit einem *parfor*-Knoten und endet mit einem *end*-Knoten in der WD-Notation. In der BPM-Notation wird dieses Kontrollkonstrukt als BPMN-Subprozess dargestellt. Wenn in diesem Fall innerhalb des Rahmens geklickt wird, aber nicht auf ein Element darin, dann wird

der Rahmen selektiert. So kann z.B. das Konstrukt verschoben oder gelöscht werden. Ein Doppelklick öffnet denselben Dialog wie ein Doppelklick auf den Startknoten des *Parallel for*.

Weitere Informationen zu diesem Konstrukt entnehmen Sie dem WDL-Teil des Kapitels [Prozessdefinition](#).

- **UND-Parallelität:** Dieses Konstrukt ermöglicht die Synchronisation paralleler Prozesspfade. Zwischen den speziellen Knoten *par* und *andjoin* können beliebig viele parallele Exekutionspfade erzeugt werden. Wenn Sie weitere Pfade hinzufügen wollen, wählen Sie die Parallelität (*par*) aus und aktivieren Sie entweder in der Menüleiste unter *Bearbeiten* den Punkt *Zusatzpfad* oder klicken Sie mit der rechten Maustaste auf die Parallelität und wählen Sie im Kontextmenü *Zusatzpfad* aus. Die Weiterführung des Prozesses erfolgt erst, wenn alle Pfade abgearbeitet sind.
- **ODER-Parallelität:** Dieses Konstrukt dient zur Formulierung einer ODER-Parallelität. Es unterscheidet sich von der UND-Parallelität dadurch, dass nur einer der parallelen Pfade beendet werden muss. Die Aktivitäten auf den anderen Pfaden werden dann abgebrochen.
- **Abzweigung:** Mit einer Abzweigung (Branch) können Sie einen Zweig im Prozess definieren, der unabhängig davon, ob der Prozess bereits beendet wurde oder nicht, abgearbeitet wird.
- **Goto:** Mit Goto können Sie zu einer beliebigen Aktivität eines Prozesses springen, ohne dass dabei eine Bedingung geprüft wird und erfüllt sein muss. Um auf eine andere Aktivität zu verweisen, klicken Sie auf die Pfeilspitze der durchgezogenen Kante und ziehen Sie diese bei gedrückter linker Maustaste über den gewünschten Zielknoten und lassen Sie dann die Maustaste los. Die gestrichelte Linie vom Goto aus zeigt den ursprünglichen Verlauf des Prozesses an. Wenn die durchgezogene Kante auf eine Aktivität zeigt, wird der Label dieser Aktivität im Goto-Eigenschaftsfenster angezeigt. Im Goto-Eigenschaftsfenster können Sie den *Label des Zielknoten* angeben. Wenn die durchgezogene Kante bereits auf ein Element im Prozesseditor zeigt und sie den Label im Goto verändern, wird die Änderung im Zielknoten auch übernommen.

Mit der Sonderform *Goto Ende* können Sie automatisch zum Prozessende springen. Innerhalb eines *Parfor*-Konstrukts ist diese Form des Goto's jedoch nicht erlaubt!



Hinweis: In der BPM-Notation gibt es den Sonderfall, dass die durchgezogene Kante nicht die Grenzen (= den Rahmen) eines *Parallel for* überschreiten darf und zwar weder ein- noch ausgehen (auch nicht bei geschachtelten *Parallel for*).

Achtung: Durch die Verwendung von Goto kann es vorkommen, dass Sie, ohne es zu wollen, Endlos-Schleifen einbauen und der Endzustand Ihres Prozesses nie erreicht wird!

- **Ereignisse:** Zu den Ereignissen gehören *Raise-Ereignis*, *Sync-Ereignis*, *Register-Ereignis*, *Unregister-Ereignis* und der *Warten* Schritt, der einen Sonderfall darstellt. Der Eventmechanismus wird verwendet, um innerhalb der Engine Events zu erzeugen und zu behandeln. Dabei kann ein Event durch die Ausführung eines Prozesses oder via API erzeugt werden. Ein Event wird von allen Prozessinstanzen, die sich dafür registriert haben, empfangen. Dadurch wird der Event Handler, der durch den Empfänger spezifiziert wurde, gestartet.

Ein Event wird durch einen Namen und ein optionales Kontext-Objekt identifiziert. Wenn die Prozessinstanz, die den Event auslöst, ein Kontext-Objekt definiert, dann stimmt die Handler-Registrierung nur, wenn bei der Registrierung des Handlers:

- das gleiche Kontext-Objekt oder
- kein Kontext-Objekt angegeben wurde.

Als Kontext-Objekt kann entweder ein Formular oder ein Formularfeld angegeben werden. Alternativ kann auch das Schlüsselwort *process* angegeben werden, womit die aktuelle Prozessinstanz (OID) als Kontext-Objekt dient. Enthält ein angegebenes Formularfeld ein *com.groiss.store.PersistentObject*, so wird beim Prüfen des Kontexts auf Gleichheit von oid und Klassenname geprüft, ansonsten wird als Kontext das Ergebnis der *toString*-Methode des übergebenen Kontextobjektes herangezogen.

Weitere Informationen zu Events finden Sie im [Programmierhandbuch](#).

- **Web-Dienste:** Bei den Web-Diensten können folgende Knoten ausgewählt werden:
 - Ausgangsnachricht (INVOKE): Ist dieser Knoten ausgewählt, so wird zur Laufzeit der definierte Web-Dienst aufgerufen und die entsprechenden Daten mitgesendet. Falls es dabei zu einem Fehler kommt und eine *Ausnahmebehandlung* hinzugefügt wurde, so wird diese ausgeführt (siehe Definition einer Ausnahmebehandlung in Abschnitt [Das Menü Bearbeiten](#)).
 - Eingangsnachricht (RECEIVE): Ist dieser Knoten ausgewählt, so wird zur Laufzeit auf Daten (des zuvor aufgerufenen) Web-Dienstes gewartet. Bekommt dieser die Daten, werden diese entsprechend der Definition abgearbeitet.
 - Antwort (REPLY): Ist dieser Knoten ausgewählt, so wird im Falle einer positiven Abarbeitung des Knoten *Eingangsnachricht*, eine Antwort an den Genserver gesendet.

Die Eigenschaften-Dialoge der einzelnen Knoten werden im Abschnitt [Eigenschaften eines Web-Dienste-Knoten definieren](#) beschrieben.

- **Anmerkung:** Wenn Sie diese Funktion ausgewählt haben, können Sie an einer beliebigen Stelle im Zeichenbereich ein Textfeld positionieren, wo Sie Anmerkungen hinzufügen können. Machen Sie einen Doppelklick in das Textfeld um einen Text hinzuzufügen und bestätigen Sie die Eingabe mit *Return*.

7.2.9 Die allgemeinen Eigenschaften eines Knotens

Für jeden Schritt besteht die Möglichkeit einen *Schrittname* und eine *Schritt-Id* zu definieren. Der Schrittname ist ein frei vergebbarer Name für den Schritt, der auch internationalisiert werden kann, indem der zu übersetzende Wert mit @@@ beginnt und mit @@ aufhört, z.B. @@@myname@@. Wenn nichts eingegeben wurde, wird der Standard-Schrittname verwendet. Dieses muss innerhalb des Prozesses eindeutig sein und unterliegt den gleichen syntaktischen Bedingungen wie eine *EPworX* -Id.

7.2.10 Eigenschaften einer Aktivität definieren

Wählen Sie eine Aktivität aus und führen Sie mit der linken Maustaste einen Doppelklick darauf aus, wenn die Funktion *Aktivitätseigenschaften bei Doppelklick auf Knoten anzeigen* aktiviert ist oder klicken Sie mit der rechten Maustaste auf den Knoten und rufen Sie im Kontextmenü *Aktivitätseigenschaften* auf. Es erscheint nun am Bildschirm die Maske *Eigenschaften für Aktivitäten* (siehe Abbildung 7.17), mit deren Hilfe Sie die Defaultwerte einer Aktivität überschreiben und bestimmte Eigenschaften definieren können.

Folgende Eigenschaften können bestimmt werden:

- **Task:** Legt die Aktivität fest durch Einfügen eines Tasks.
- **Schrittname:** Bezeichnung des Schrittes, der auch internationalisiert werden kann, indem der zu übersetzende Wert mit @@@ beginnt und mit @@ aufhört, z.B. @@@myname@@.
- **Schritt-Id:** Diese muss innerhalb des Prozesses eindeutig sein und unterliegt den gleichen syntaktischen Bedingungen wie eine *EPworX* -Id. Eine eingegebene Schritt-Id wirkt sich bei der Akteurauswahl für andere Aktivitäten aus.
- **Icon:** Das Icon, durch das die Aktivität im Prozesseditor dargestellt wird. Wird hier kein Wert angegeben, so wird das Default Icon von *EPworX* verwendet. Ein Icon wird wie eine Ressource behandelt, die im Klassenpfad vorhanden ist. Beispiel: Pfad *lang/default/images/pred/nodes/event_register.png* liefert Icon des Knoten *Event registrieren*.
- **Füllfarbe:** Hier können Sie eine Hintergrundfarbe für den Schritt definieren. Die Farbe kann entweder mittels *Color picker* ausgewählt oder auch direkt als hexadezimaler Code eingegeben werden.
- **Akteur(e):** Die Namen der Akteure (Rolle, Benutzer, Akteur des Tasks, Formularfeld, Methode).
- **Überspringbar:** Ist diese Checkbox aktiviert und kein Akteur eingetragen, wird diese Aktivität ausgelassen.
- **Formulare:** Gibt an, welche Prozessformulare in dieser Aktivität verwendet werden. Die Reihenfolge der Prozessformulare kann über die Sortierbuttons neben der Liste verändert werden, d.h. das oberste Formular wird in der Reiter-Ansicht im Arbeitskorb ganz links angezeigt.

Task: Anfordern

Schrittname:

Schritt-Id: label_4

Icon: Vorschau:

Füllfarbe:

Akteur(e): Alle

Überspringbar: ☐

Verfügbare Formulare:

Hinzugefügte Formulare: Urlaub

Ok Abbrechen

Abbildung 7.17: **Eigenschaften für Aktivitäten**

Folgende Funktionen stehen zur Verfügung:

- **Neue Aktivität:** Hierbei öffnet sich die Detailansicht zu einem neuen Task, wo Sie der Aktivität diesen neu erstellten Task hinzufügen können (siehe Abschnitt [Tasks](#)).
- **Auswahl:** Durch einen Mausklick auf diese Schaltfläche wird ein Fenster angezeigt, das es Ihnen ermöglicht eine bereits in der Taskverwaltung definierte Aktivität auszuwählen. Dafür gehen Sie wie folgt vor:
 1. Wählen Sie eine Aktivität aus der Liste aus.
 2. Klicken Sie auf die Schaltfläche *Ok*.
 3. Wenn Sie auf die Schaltfläche *Abbrechen* klicken, kehren Sie zum vorigen Fenster zurück und Ihre Eingaben werden verworfen.
- **Akteur hinzufügen:** Durch einen Mausklick auf die Schaltfläche *Akteur hinzufügen* wird ein neues Fenster angezeigt, das es Ihnen ermöglicht einen Akteur für

die aktuelle Aktivität auszuwählen.

Wählen Sie aus der Liste, die Ihnen dann angeboten wird, einen Akteur aus. Bevor Sie dies tun, müssen Sie angeben wie der Akteur definiert wurde, d.h. ein Akteur kann sein:

- *Benutzer*: Registrierter Benutzer.
- *Rolle*: Spezifiziert eine Rolle. Eine Organisationseinheit kann zusätzlich ausgewählt sein, allerdings gilt diese nur für den aktuellen Schritt, die OE des Prozesses wird nicht verändert.
- *Akteur des Tasks*: Es wird der Akteur einer früheren Aktivität genommen.
- *Formularfeld*: Die Rolle wird zur Laufzeit einem Formularfeld entnommen, dessen Inhalt dann einer Rollen- oder Benutzer-Id entsprechen muss.
- *Methode*: Angabe einer Java-Methode, die als Akteur dient (kein Groovy-Skript). Rückgabewert muss ein *Agent* oder ein *String* in WDL-Syntax sein. Nähere Informationen finden Sie auf der Seite [187](#).

Durch einen Mausklick auf die Schaltfläche *Ok* wird der ausgewählte Akteur in die Liste der Akteure der vorigen HTML-Seite aufgenommen, und Sie kehren dorthin zurück.

Durch einen Mausklick auf die Schaltfläche *Abbrechen* kehren Sie zur vorigen HTML-Seite zurück, ohne einen Akteur zu übernehmen.

- **Akteur entfernen**: Um einen bestimmten Akteur zu entfernen, müssen Sie diesen aus der Liste der Akteure durch einen Mausklick auswählen, und anschließend mit der Maus auf die Schaltfläche *X* klicken.
- **Formular hinzufügen**: In der Tabelle *Verfügbare Formulare* werden alle Formulare aufgelistet, die für den konkreten Prozess (im Menü *Prozess* Menüpunkt *Eigenschaften/Attribute*) definiert wurden. Durch Auswählen eines Formulars und Aktivierung der Pfeil-Schaltfläche können Sie Formulare zur Aktivität hinzufügen. Die Formulare, die einer Aktivität hinzugefügt wurden, stehen rechts in der Tabelle *Hinzugefügte Formulare*.
- **Formular entfernen**: Aus der Liste der hinzugefügten Formulare können Sie ein Formular entfernen, wenn Sie die Schaltfläche *X* rechts neben dieser Liste betätigen.
- **Formular-Sichtbarkeiten**: Mit Hilfe des Schraubenschlüssel-Symbols auf der rechten Seite neben der Liste *Hinzugefügte Formulare* können Sie die Formular-Sichtbarkeiten für diese Aktivität festlegen, indem Sie ein Formular aus der Liste auswählen und auf dieses Symbol klicken (Analog zu Kapitel [Prozesse](#)).
- **OK**: Durch Aktivierung des Befehls *Ok* bestätigen Sie Ihre Eingaben und beenden das Eingabefenster.
- **Abbrechen**: Durch Aktivierung des Befehls *Abbrechen* verwerfen Sie Ihre Eingaben und beenden das Eingabefenster.

7.2.11 Bedingungen für Ifs, Choice, Schleifen

Für die genannten Konstrukte kann nach Doppelklick auf das grafische Symbol eine Bedingung eingegeben werden. Zu diesem Zwecke steht ein Editor zur Erstellung von Bedingungen zur Verfügung (siehe Abschnitt [Bedingungseditor](#)). Zur Laufzeit wird die Bedingung vom System automatisch ausgewertet.

Weiters können noch der Schrittname und die Schritt-Id spezifiziert werden. Der Schrittname ist ein frei vergebbarer Name für den Schritt, der auch internationalisiert werden kann, indem der zu übersetzende Wert mit @@@ beginnt und mit @@ aufhört, z.B. @@@myname@@. Wenn nichts eingegeben wurde, wird der Standard-Schrittname verwendet. Diese muss innerhalb des Prozesses eindeutig sein und unterliegt den gleichen syntaktischen Bedingungen wie eine *EPworX* -Id.

7.2.12 Methoden eines Systemschrittes bestimmen

Wählen Sie einen Systemschritt und führen Sie mit der linken Maustaste einen Doppelklick darauf aus.

Es erscheint nun am Bildschirm die Maske *Eigenschaften Systemschritt*, mit deren Hilfe Sie die Methoden, die bei der Ausführung eines Systemschrittes ausgeführt werden sollen, definieren können.

Neben 2 unterschiedlichen Eingabe-Arten, die in den nachfolgenden Abschnitten beschrieben werden, können weiters noch ein Pfad zu einem Icon (das im Klassenpfad von *EPworX* liegt und als Ressource behandelt wird), die Füllfarbe, der Schrittname und die Schritt-Id spezifiziert werden. Der Schrittname ist ein frei vergebbarer Name für den Schritt, der auch internationalisiert werden kann, indem der zu übersetzende Wert mit @@@ beginnt und mit @@ aufhört, z.B. @@@myname@@. Wenn nichts eingegeben wurde, wird der Standard-Schrittname verwendet. Diese muss innerhalb des Prozesses eindeutig sein und unterliegt den gleichen syntaktischen Bedingungen wie eine *EPworX* -Id.

Die Syntax von Systemschritten ist im Abschnitt [Systemschritte](#) der textuellen Prozessdefinition beschrieben. Für die Eingabe werden 2 Arten angeboten:

Der Reiter *Editor*

Hier können Java-Methoden einfach per Mausklick angefügt, bearbeitet, entfernt oder ihre Reihenfolge verändert werden. Beim Hinzufügen und Editieren einer Methode öffnet sich der Editor-Dialog (siehe Abbildung [7.18](#)), in welchem standardmäßig nützliche Methoden von *EPworX* angezeigt werden. Um eigene Methoden in der Liste anzuzeigen, müssen diese mit der Annotation *@CallableMethod* versehen werden (weitere Details dazu im [Programmierhandbuch](#), Kapitel [Adding methods to the system step editor](#)). Durch Auswahl eines Eintrags wird - sofern vorhanden - ein Bereich mit Parametern angeboten, den die Methode entgegennehmen kann. Durch Aktivieren der Schaltfläche

Ok wird die ausgewählte Methode entweder hinzugefügt oder eine bestehende aktualisiert.

↑ Methode	Beschreibung	Funktionsgruppe
addDocument	Fügt ein Dokument dem Prozessordner hinzu	Dokumente
addFolder	Erstellt einen neuen Ordner mit dem angegebenen Namen und fügt ihn zum Prozess hinzu.	Dokumente
create	Systemschritt oder Postcondition zur Erstellung eines neuen Dokuments	Dokumente
exportDocument	Kopiert ein Dokument vom Prozess-Ordner in den Dateipfad.	Dokumente

Parameter Ordnername:

Ok Abbrechen

Abbildung 7.18: **Systemschritt-Editor**

Der Reiter *Eingabe*

In diesem Reiter können Methoden händisch deklariert und bearbeitet werden. Um eine Methode hinzuzufügen, muss diese mit dem voll-qualifizierten Methodennamen plus optionaler Parameter eingefügt werden. Wenn eine Sequenz von Methoden ausgeführt werden soll, dann sind die Methoden durch ; zu trennen. Die korrekte Eingabe der Methoden kann mithilfe des Prüf-Icons neben dem Eingabefeld auf ihre Gültigkeit geprüft werden. Verändert sich das Icon in ein rotes Rufzeichen, so ist die Methode syntaktisch falsch. Ein grüner Haken deutet darauf hin, dass die Methode korrekt ist. Zudem auf die Verfügbarkeit im Klassenpfad geprüft.

7.2.13 Die Eigenschaften eines Batchschrittes bestimmen

Wählen Sie einen Batchschritt aus und führen Sie mit der linken Maustaste einen Doppelklick darauf aus.

Es erscheint nun am Bildschirm die Maske *Eigenschaften Batchschritt*, mit deren Hilfe Sie die BatchAdapter Klasse, deren Methoden bei der Ausführung eines Batchschrittes aufgerufen werden sollen, definieren können. Im Feld *Parameter* können Sie die Parameter angeben, die der BatchAdapter-Klasse übergeben werden (in der WDL werden diese Parameter innerhalb der runden Klammern angezeigt). Die Klasse selbst wird ohne runde Klammern angegeben. Variationen des Verhaltens der Laufzeit eines

Batch-Jobs können über die Checkboxes angegeben werden. Mehr dazu im [Programmierhandbuch](#) und in der [API-documentation](#).

Weiters können noch ein Pfad zu einem Icon (das im Klassenpfad von *EPworX* liegt und als Ressource behandelt wird), die Füllfarbe, der Schrittname und die Schritt-Id spezifiziert werden. Der Schrittname ist ein frei vergebbarer Name für den Schritt, der auch internationalisiert werden kann, indem der zu übersetzende Wert mit @@@ beginnt und mit @@ aufhört, z.B. @@@myname@@. Wenn nichts eingegeben wurde, wird der Standard-Schrittname verwendet. Diese muss innerhalb des Prozesses eindeutig sein und unterliegt den gleichen syntaktischen Bedingungen wie eine *EPworX*-Id.

Die Syntax von Batchschritten ist im Abschnitt [Batch Schritte](#) der textuellen Prozessdefinition beschrieben.

7.2.14 Die Eigenschaften eines Subprozesses bestimmen

Wählen Sie einen Subprozess aus und führen Sie mit der linken Maustaste einen Doppelklick aus.

Es erscheint nun am Bildschirm die Maske *Eigenschaften* für Subprozesse, mit deren Hilfe Sie die Eigenschaften eines Subprozesses definieren können.

Folgende Eigenschaften können definiert werden:

- **Prozess:** Der Name des Subprozesses.
- **Schrittname:** Frei wählbarer Name für diesen Schritt, der dann angezeigt werden soll; kann auch internationalisiert werden, indem der zu übersetzende Wert mit @@@ beginnt und mit @@ aufhört, z.B. @@@myname@@.
- **Schritt-Id:** Diese muss innerhalb des Prozesses eindeutig sein und unterliegt den gleichen syntaktischen Bedingungen wie eine *EPworX*-Id.
- **Icon:** Pfad zum Anzeigen eines alternativen Icons, das im Klassenpfad von *EPworX* liegt und als Ressource behandelt wird.
- **Füllfarbe:** Hier können Sie eine Hintergrundfarbe für den Schritt definieren. Die Farbe kann entweder mittels *Color picker* ausgewählt oder auch direkt als hexadezimaler Code eingegeben werden.
- **Formulare:** Formulare, die dem Subprozess übergeben werden.

Folgende Funktionen stehen zur Verfügung:

- **Auswahl:** Durch das Anklicken der Schaltfläche mit der Lupe wird Ihnen eine Liste aller zur Verfügung stehenden Prozesse angezeigt, aus der Sie einen auswählen können. Sie erreichen dies, indem Sie:
 - einen Prozess mit der Maus auswählen, und anschließend die Schaltfläche *Ok* anklicken.

- Durch das Anklicken der Schaltfläche *Abbrechen* wird der Auswahlvorgang abgebrochen und das Auswahlfenster für Prozesse geschlossen.
- Formular hinzufügen: In der Tabelle *Verfügbare Formulare* werden alle Formulare aufgelistet, die für den konkreten Prozess (im Menü *Prozess* Menüpunkt *Eigenschaften/Attribute*) definiert wurden. Durch Auswählen eines Formulars und Aktivierung der Pfeil-Schaltfläche können Sie Formulare zur Aktivität hinzufügen. Die Formulare, die einer Aktivität hinzugefügt wurden, stehen rechts in der Tabelle *Hinzugefügte Formulare*.
- Formular entfernen: Aus der Liste der hinzugefügten Formulare können Sie ein markiertes Formular entfernen, wenn Sie die Schaltfläche *X* rechts neben dieser Liste betätigen.
- OK: Durch Aktivierung des Befehls *Ok* bestätigen Sie Ihre Eingaben und beenden das Eingabefenster.
- Abbrechen: Durch Aktivierung des Befehls *Abbrechen* verwerfen Sie Ihre Eingaben und beenden das Eingabefenster.

7.2.15 Eigenschaften eines Parallel For definieren

Wählen Sie einen *parfor*-Knoten aus und führen Sie mit der linken Maustaste einen Doppelklick darauf aus. Es erscheint nun am Bildschirm die Maske *Parallel For* mit deren Hilfe Sie bestimmte Eigenschaften definieren können.

Folgende Eigenschaften können bestimmt werden:

- **für jedes Subformular in:** Wenn dieser Radio-Button aktiviert ist, wird das Parallel For für die Subformular-Einträge eines Formulars ausgeführt, wie es im WDL-Teil beschrieben ist (siehe Fall 1, Abschnitt *Parallel For* unter [WDL](#)). Aus der Dropdown-Liste können Sie nun das entsprechende Subformular (Hauptformular.Subformular.Id) auswählen.
 - Id des Formulars in der Schleife: Die Id des zuvor ausgewählten Subformulars innerhalb des Parallel For Konstrukts.
 - Name des Formulars in der Schleife: Name des Subformulars innerhalb des Parallel For Konstrukts.
 - Bedingung: Hier können Sie eine Methode (mit boolean Rückgabewert) oder eine einfache Bedingung angeben, wann ein Parfor-Zweig gestartet werden soll, z.B. `localform.isopen= 0` (*localform* ist die ID des Subformulars im Knoten und *isopen* ist ein Feld im Formular). Durch Aktivieren des Bearbeitens Icons können Sie im neu geöffneten Fenster mit Hilfe eines Editors Bedingungen zusammenstellen, die dann in dieses Feld eingetragen werden (siehe Abschnitt [Bedingungseditor](#)).
- **Iterator:** Wenn dieser Radio-Button aktiviert ist, wird das Parallel For für diese Klasse ausgeführt, wie es im WDL-Teil beschrieben ist (siehe Abschnitt [Parallel For](#)).

- **Schrittname:** Frei vergebbarer Name für den Schritt, der auch internationalisiert werden kann, indem der zu übersetzende Wert mit @@@ beginnt und mit @@ aufhört, z.B. @@@myname@@. Wenn nichts eingegeben wurde, wird der Standard-Schrittname verwendet.
- **Schritt-Id:** Diese muss innerhalb des Prozesses eindeutig sein und unterliegt den gleichen syntaktischen Bedingungen wie eine *EPworX* -Id.

Im Endknoten eines Parallel For kann zudem eine Methode angegeben werden (siehe Abschnitt [Parallel For](#)).

7.2.16 Eigenschaften einer UND-/ODER-Parallelität oder eines *Parallel for* Endknoten bestimmen

Wählen Sie den End-Knoten einer UND-, ODER-Parallelität oder PARFOR aus und führen Sie mit der linken Maustaste einen Doppelklick aus. Es erscheint nun am Bildschirm die Maske *UND-Parallelität Join*, *ODER-Parallelität Join* oder *Parallel for join* mit deren Hilfe Sie folgende Eigenschaften definieren können.

Folgende Eigenschaften können bestimmt werden:

- **Methodenaufruf:** Wenn die Fertigstellung der Parallelität nicht durch eine fixe Anzahl von fertigen Zweigen bestimmt ist, sondern zur Laufzeit bestimmt werden kann, ist dies hier durch Angabe einer Java-Methode möglich. Nähere Informationen dazu finden Sie im [Programmierhandbuch](#).
- **Schrittname:** Frei vergebbarer Name für den Schritt, der auch internationalisiert werden kann, indem der zu übersetzende Wert mit @@@ beginnt und mit @@ aufhört, z.B. @@@myname@@.. Wenn nichts eingegeben wurde, wird der Standard-Schrittname verwendet.
- **Schritt-Id:** Diese muss innerhalb des Prozesses eindeutig sein und unterliegt den gleichen syntaktischen Bedingungen wie eine *EPworX* -Id.

Die Syntax ist im Abschnitt [Parallelität \(andpar und orpar\)](#) für UND-/ODER-Parallelität bzw. [Parallel For](#) für *Parallel for* der textuellen Prozessdefinition beschrieben.

7.2.17 Eigenschaften eines Ereignisses definieren

Wählen Sie ein Ereignis aus und führen Sie mit der linken Maustaste einen Doppelklick darauf aus. Es erscheint nun am Bildschirm die Maske des ausgewählten Ereignisses mit deren Hilfe Sie bestimmte Eigenschaften definieren können.

Dabei gibt es folgende Ereignisse:

- **Auslösen (Raise):** Ist dieses Ereignis ausgewählt, so wird ein Event ausgelöst.
- **Synchronisieren (Sync):** Ist dieses Ereignis ausgewählt, so wartet der Prozess an dieser Stelle auf den Event.

- Registrieren (Register): Ist dieses Ereignis ausgewählt, so registriert sich der Prozess für den Event.
- Deregistrieren (Unregister): Ist dieses Ereignis ausgewählt, so wird an dieser Stelle im Prozess die Registrierung für den Event aufgehoben.
- Warten (Wait): Dieses Ereignis ist ein Sonderfall und wird in Abschnitt [Ereignis Warten](#) beschrieben.

Folgende Eigenschaften können bestimmt werden:

- **Event Name:** Der Name des Events.
- **Event Handler:** Eine Java-Klasse die das Interface *com.groiss.event.EventHandler* implementiert. Wenn kein Wert eingegeben wurde wird die Default-Implementierung eingetragen.
- **Kontext:** spezifiziert einen Knoten in der aktuellen Prozessinstanz oder ein Formular oder Formularfeld, das als Kontext-Objekt dient. Die folgenden Optionen sind verfügbar:
 - Übergeordneter Schritt: der Kontext ist ein unmittelbarer Elternknoten der aktuellen Aktivitätsinstanz, das kann eine Prozessinstanz oder ein Parfor-Knoten sein.
 - Prozess: die innerste (Sub-)Prozessinstanz, die von der aktuellen Aktivitätsinstanz aus erreichbar ist.
 - Hauptprozess: die äußerste Prozessinstanz, die von der aktuellen Aktivitätsinstanz aus erreichbar ist.
 - Systemweit: keine Einschränkung auf ein Kontext-Objekt.
 - ein Formular (optional mit Feld): das Formular mit der angegebenen Id, das von der aktuellen Aktivitätsinstanz aus erreichbar ist, oder das für dieses Formular spezifizierte Feld.
- **Schrittname:** Frei vergebbarer Name für den Schritt, der auch internationalisiert werden kann, indem der zu übersetzende Wert mit @@@ beginnt und mit @@ aufhört, z.B. @@@myname@@. Wenn nichts eingegeben wurde, wird der Standard-Schrittname verwendet.
- **Schritt-Id:** Diese muss innerhalb des Prozesses eindeutig sein und unterliegt den gleichen syntaktischen Bedingungen wie eine *EPworX* -Id. Die Schritt-Id ist auch für Prozessescalationen vom Typ *Sync nicht erfolgt* relevant (siehe Abschnitt [Registrierkarte: Eskalationen](#)).
- **Icon:** Pfad zum Anzeigen eines alternativen Icons, das im Klassenpfad von *EPworX* liegt und als Ressource behandelt wird.

Folgende Funktionen stehen zur Verfügung:

- **OK:** Durch das Aktivieren der Schaltfläche *Ok* werden Ihre Eingaben übernommen und der Dialog wird geschlossen.

- **Abbrechen:** Durch das Aktivieren der Schaltfläche *Abbrechen* werden Ihre Eingaben verworfen und der Dialog geschlossen.

Die Syntax von Ereignissen ist im Abschnitt [Eventmechanismus](#) der textuellen Prozessdefinition beschrieben. Weitere Details zum Thema Ereignisse finden Sie im [Application Development Guide](#).

Ereignis *Warten*

Ein Warten Schritt kann dazu verwendet werden, um einen Prozesslauf für eine gewisse Zeitdauer bzw. bis zu einem gewissen Zeitpunkt anzuhalten. Der Warten Schritt wird durch den Timer *Suspension* automatisch beendet, d.h. der Warten Schritt wird frühestens beendet, wenn die Zeitdauer überschritten ist abhängig davon wann der Timer *Suspension* das nächste Mal ausgeführt wurde. Es besteht auch die Möglichkeit diesen manuell über die Prozesshistorie beenden. Folgende Eigenschaften können bestimmt werden:

- **Dauer:** In diesem Bereich können Sie entweder ein Zeitintervall angeben oder einen Ausdruck:
 - Zeitintervall: Hier können Sie einen ganzzahligen positiven Wert festlegen, wie lange ein Warten Schritt dauern soll. Als Zeiteinheiten stehen Ihnen dabei Minuten, Stunden, Tage und Arbeitstage zu Verfügung.
 - Ausdruck: Hier können Sie einen Ausdruck angeben, der einen gewissen Zeitpunkt (also ein Datum) oder ein Zeitintervall in Minuten definiert. Möglich sind Methoden, Formularfelder, XPath- oder Groovy-Ausdrücke (siehe dazu Definition in WDL im Kapitel [WDL](#)).
- **Schrittname:** Frei vergebbarer Name für den Schritt, der auch internationalisiert werden kann, indem der zu übersetzende Wert mit @@@ beginnt und mit @@ aufhört, z.B. @@@myname@@. Wenn nichts eingegeben wurde, wird der Standard-Schrittname verwendet.
- **Schritt-Id:** Diese muss innerhalb des Prozesses eindeutig sein und unterliegt den gleichen syntaktischen Bedingungen wie eine *EPworX* -Id.

Folgende Funktionen stehen zur Verfügung:

- **OK:** Durch das Aktivieren der Schaltfläche *Ok* werden Ihre Eingaben übernommen und der Dialog wird geschlossen.
- **Abbrechen:** Durch das Aktivieren der Schaltfläche *Abbrechen* werden Ihre Eingaben verworfen und der Dialog geschlossen.

Die Syntax von Warte-Schritten ist im Abschnitt [Warten Schritte](#) der textuellen Prozessdefinition beschrieben.

7.2.18 Eigenschaften eines GOTO-Knoten definieren

Wählen Sie den GOTO-Knoten aus und führen Sie mit der linken Maustaste einen Doppelklick aus. Es erscheint nun am Bildschirm die Maske *Goto* mit deren Hilfe Sie folgende Eigenschaften definieren können.

Folgende Eigenschaften können bestimmt werden:

- **Schrittname:** Frei vergebbarer Name für den Schritt, der auch internationalisiert werden kann, indem der zu übersetzende Wert mit @@@ beginnt und mit @@ aufhört, z.B. @@@myname@@. Wenn nichts eingegeben wurde, wird der Standard-Schrittname verwendet.
- **Schritt-Id:** Diese muss innerhalb des Prozesses eindeutig sein und unterliegt den gleichen syntaktischen Bedingungen wie eine *EPworX* -Id.
- **Label des Zielknotens:** Hier können Sie das Label des Zielknotens manuell angeben. Dieses Feld wird automatisch befüllt, wenn Sie die durchgezogene Kante des GOTO auf ein Element im Prozesseditor ziehen. Wird das Label hier verändert, so wird die Änderung auch im Zielknoten übernommen.

Die Syntax ist im Abschnitt [Goto](#) der textuellen Prozessdefinition beschrieben.

7.2.19 Eigenschaften eines Web-Dienste-Knoten definieren

Wählen Sie einen beliebigen Web-Dienste-Knoten aus und führen Sie mit der linken Maustaste einen Doppelklick darauf aus. Es erscheint nun am Bildschirm die Maske des ausgewählten Knotens mit deren Hilfe Sie bestimmte Eigenschaften definieren können. Für jeden Schritt gibt es einen *Schrittname*, der frei vergebbar ist und der auch internationalisiert werden kann, indem der zu übersetzende Wert mit @@@ beginnt und mit @@ aufhört, z.B. @@@myname@@. Wenn nichts eingegeben wurde, wird der Standard-Schrittname verwendet. Weiters gibt es eine *Schritt-Id*. Diese muss innerhalb des Prozesses eindeutig sein und unterliegt den gleichen syntaktischen Bedingungen wie eine *EPworX* -Id. Die Schritt-Id ist beim Knoten *Eingangsnachricht* für Prozessskalationen vom Typ *Eingangsnachricht nicht erhalten* relevant (siehe Abschnitt [Registrierkarte: Eskalationen](#)).

Folgende Eigenschaften können für den Knoten *Ausgangsnachricht* bestimmt werden:

- **Webservice Operation:** Sie können aus der Auswahlliste eine bestehende Webservice Client Operation auswählen, die zuvor (für die Applikation) definiert wurde (siehe Kapitel [Webservice Clients](#)).
- **Schrittname:** Frei vergebbarer Name für den Schritt, der auch internationalisiert werden kann, indem der zu übersetzende Wert mit @@@ beginnt und mit @@ aufhört, z.B. @@@myname@@.. Wenn nichts eingegeben wurde, wird der Standard-Schrittname verwendet.
- **Schritt-Id:** Diese muss innerhalb des Prozesses eindeutig sein und unterliegt den gleichen syntaktischen Bedingungen wie eine *EPworX* -Id.

- **Icon:** Pfad zum Anzeigen eines alternativen Icons, das im Klassenpfad von *EPworX* liegt und als Ressource behandelt wird.
- **Füllfarbe:** Hier können Sie eine Hintergrundfarbe für den Schritt definieren. Die Farbe kann entweder mittels *Color picker* ausgewählt oder auch direkt als hexadezimaler Code eingegeben werden.
- **Adresse:** Auswahl einer Adresse, um den Web-Dienst aufzurufen. Sie haben entweder die Möglichkeit die Adresse aus der WSDL-Datei zu beziehen, einen XPath-Ausdruck anzuwenden oder die URL direkt einzugeben.

Beispiel für XPath-Ausdruck: Lesen aus dem Konfigurationsparametern der Applikation *myappl*:

```
string($configuration_myappl/property[@name='webservice.address'])
```

- **Out-Parameter:** Hier können Sie eine Liste von Parametern als XPath-Ausdruck angeben, die dem Web-Dienst übergeben werden. Die Parameter sind in der WSDL-Datei definiert und wurden bei der Erstellung des Webservice-Clients angegeben.
- **In-Parameter:** Analog zu Out-Parameter, jedoch für Daten, die aus dem Web-Dienst gelesen werden.

Folgende Eigenschaften können für den Knoten *Eingangsnachricht* bestimmt werden:

- **Webservice Operation:** Sie können aus der Auswahlliste eine bestehende Webservice Server Operation auswählen, die zuvor (für die Applikation) definiert wurde (siehe Kapitel [Webservice Server](#)).
- **Schrittname:** Frei vergebbarer Name für den Schritt, der auch internationalisiert werden kann, indem der zu übersetzende Wert mit @@@ beginnt und mit @@ aufhört, z.B. @@@myname@@.. Wenn nichts eingegeben wurde, wird der Standard-Schrittname verwendet.
- **Schritt-Id:** Diese muss innerhalb des Prozesses eindeutig sein und unterliegt den gleichen syntaktischen Bedingungen wie eine *EPworX* -Id.
- **Icon:** Pfad zum Anzeigen eines alternativen Icons, das im Klassenpfad von *EPworX* liegt und als Ressource behandelt wird.
- **Füllfarbe:** Hier können Sie eine Hintergrundfarbe für den Schritt definieren. Die Farbe kann entweder mittels *Color picker* ausgewählt oder auch direkt als hexadezimaler Code eingegeben werden.
- **Prozess starten:** Wenn diese Checkbox aktiviert ist und dieser Knoten der 1. Schritt in der Prozessdefinition ist, dann wird eine neue Instanz des Prozesses erzeugt. Ist diese Checkbox in diesem Fall nicht aktiviert, kann keine Instanz gestartet werden. Wenn dieser Knoten nicht der 1. Schritt ist, dann darf diese Checkbox nicht aktiviert werden!

- **In-Parameter:** Hier können Sie eine Liste von Parametern als XPath-Ausdruck angeben, die vom Web-Dienst gelesen werden. Die Parameter sind in der WSDL-Datei definiert und wurden bei der Erstellung des Webservice-Servers angegeben.
- **Korrelation:** Auch in diesem Reiter können Sie eine Liste von Parametern als XPath-Ausdruck angeben, die bei der Erstellung des Webservice-Servers angegeben wurden. Hier können Mappings definiert werden, mit denen dann eine automatische Zuordnung einer *Eingangsnachricht* zu einer Prozessinstanz gemacht werden können.

Folgende Eigenschaften können für den Knoten *Antwort* bestimmt werden:

- **Webservice Operation:** Sie können aus der Auswahlliste eine bestehende Webservice Server Operation auswählen, die zuvor (für die Applikation) definiert wurde (siehe Kapitel [Webservice Server](#)).
- **Schrittname:** Frei vergebbarer Name für den Schritt, der auch internationalisiert werden kann, indem der zu übersetzende Wert mit @@@ beginnt und mit @@ aufhört, z.B. @@@myname@@.. Wenn nichts eingegeben wurde, wird der Standard-Schrittname verwendet.
- **Schritt-Id:** Diese muss innerhalb des Prozesses eindeutig sein und unterliegt den gleichen syntaktischen Bedingungen wie eine *EPworX* -Id.
- **Icon:** Pfad zum Anzeigen eines alternativen Icons, das im Klassenpfad von *EPworX* liegt und als Ressource behandelt wird.
- **Füllfarbe:** Hier können Sie eine Hintergrundfarbe für den Schritt definieren. Die Farbe kann entweder mittels *Color picker* ausgewählt oder auch direkt als hexadezimaler Code eingegeben werden.
- **Out-Parameter:** Hier können Sie eine Liste von Parameter als XPath-Ausdruck angeben, die dem Web-Dienst übergeben werden. Die Parameter sind in der WSDL-Datei definiert und wurden bei der Erstellung des Webservice-Servers angegeben.

Folgende Funktionen stehen bei allen Web-Dienste-Knoten zur Verfügung:

- **OK:** Durch das Aktivieren der Schaltfläche *Ok* werden Ihre Eingaben übernommen und der Dialog wird geschlossen.
- **Abbrechen:** Durch das Aktivieren der Schaltfläche *Abbrechen* werden Ihre Eingaben verworfen und der Dialog geschlossen.

Für weitere Informationen über die WDL-Syntax sehen Sie bitte im Abschnitt [Web-Dienste](#) nach.

7.2.20 Bedingungseditor

Mit Hilfe des Bedingungseditors können komplexe Bedingungen erstellt werden. Dazu werden 2 Arten der Eingabe angeboten:

Reiter *Editor*

In diesem Reiter ist es möglich Bedingungen in XPath-Syntax grafisch zu definieren. Zu diesem Zwecke können Sie mit Hilfe des Toolbars Bedingungszeilen hinzufügen, entfernen und an der entsprechenden Stelle (Nach oben/unten verschieben) positionieren. Mit Hilfe der Klammersetzung können Sie die Auswertungslogik beeinflussen, indem Sie die nötigen Bedingungszeilen markieren und die Toolbarfunktion *Klammern setzen* aktivieren. Bedingungen werden beim Hinzufügen automatisch mit logisch UND verknüpft, jedoch kann der Operator auf logisch ODER geändert werden, indem man auf den Operator-Button klickt, d.h. Klick auf UND wechselt zu ODER und umgekehrt.

Abbildung 7.19: **Grafischer Editor für Bedingungen**

Eine Bedingungszeile besteht aus folgenden Komponenten:

- Ausdruck 1: In der 1. Dropdownliste wird folgende Auswahl angeboten:
 - Aktueller Benutzer: Zum Zeitpunkt der Bedingungs-Auswertung wird der aktuelle Benutzer (Thread-User) herangezogen.
 - Aktueller Prozess: Zum Zeitpunkt der Bedingungs-Auswertung wird die aktuelle Prozessinstanz herangezogen. In der 2. Dropdownliste muss ein Prozessattribut ausgewählt werden, das für den Vergleich herangezogen wird.
 - Aktuelles Datum: Zum Zeitpunkt der Bedingungs-Auswertung wird das aktuelle Datum herangezogen.
 - Formulare: Alle dem zum Prozess hinzugefügten Formulare werden zur Auswahl angeboten (siehe Definition in Abschnitt [Prozesseigenschaften](#)). In der 2. Dropdownliste werden alle Formularfelder des ausgewählten Formulars angezeigt und es muss ein Formularfeld für den Vergleich ausgewählt werden.
 - Konfigurationen: Gibt es für Applikationen eine eigene Konfiguration, wird diese hier zur Auswahl angeboten (siehe Definition in Abschnitt [Registerkarte: Eigenschaften](#)). In der 2. Dropdownliste muss ein Konfigurationsparameter für den Vergleich ausgewählt werden.

- **Operator:** Eine Dropdownliste mit verschiedenen Operatoren, um beide Ausdrücke miteinander zu vergleichen.
- **Ausdruck 2:** Hier kann entweder eine Definition wie in Ausdruck 1 erfolgen oder ein direkter Vergleich mit einem bestimmten Wert. Der Wert kann abhängig vom Ausdruck 1 entweder frei eingegeben oder aus einer Dropdownliste ausgewählt werden (z.B. wenn in Ausdruck 1 ein Formularfeld vom Typ Radio Button selektiert wurde, wird in Ausdruck 2 eine Liste von möglichen Optionen dieses Radio Buttons angeboten). Bitte beachten Sie, dass bei der freien Eingabe unter Umständen beim Wert doppelte Hochkommas verwendet werden müssen (z.B. wenn im Ausdruck 1 ein String Attribut ausgewählt wurde).



Wenn eine Bedingung nicht korrekt definiert wurde bzw. fehlerhaft ist, wird dies durch das Warnung-Icon am Anfang der Zeile angezeigt. Die Bedingungen werden wie eingangs erwähnt in XPath-Syntax abgespeichert. Sollte der gesamte Ausdruck syntaktisch falsch sein oder nicht der XPath-Syntax entsprechen, wird dies mit einer Fehlermeldung in diesem Reiter angezeigt.

Mit Hilfe der Schaltfläche *Ok* werden die Bedingungen gespeichert bzw. in das referenzierte Feld in XPath-Syntax übernommen. Die Schaltfläche *Abbrechen* verwirft die getätigten, aber nicht zuvor gespeicherten Änderungen.

Reiter *Eingabe*

Dieser Reiter bietet die Möglichkeit Bedingungen auf 3 Arten zu definieren:

- **XPath:** Die XML Path Language ist eine vom W3-Konsortium entwickelte Abfragesprache, um Teile eines XML-Dokumentes zu adressieren und auszuwerten. Der Reiter *Editor* erzeugt XPath-Syntax, die hier in textueller Form angezeigt werden kann. Es besteht aber auch die Möglichkeit selbst XPath-Bedingungen hinzuzufügen und diese dann - sofern auswertbar - im Reiter *Editor* anzuzeigen. Eine nähere Beschreibung zu XPath-Ausdrücke finden Sie im [Programmierhandbuch](#).
- **WDL / Java Methode:** Die Workflow Definition Language ist die Prozessbeschreibungssprache von *EPworX* und erlaubt eine entsprechende Bedingungsdefinition (siehe Abschnitt [WDL](#)). Alternativ können Sie im Eingabefeld auch einfach eine Java-Methode mit einem boolschen Rückgabewert definieren.
- **Groovy:** Groovy ist eine Programmier- und Skriptsprache, die dynamische und statische Typisierung unterstützt. Eine nähere Beschreibung zu Groovy-Bedingungen finden Sie im [Programmierhandbuch](#).



Jede Bedingung kann mit Hilfe des Prüf-Icons neben dem Eingabefeld auf ihre Gültigkeit geprüft werden. Verändert sich das Icon in ein rotes Rufzeichen, so ist die Bedingung syntaktisch falsch. Ein grüner Haken deutet darauf hin, dass die Bedingung korrekt ist. Im Falle einer Java-Methode wird zudem auf die Verfügbarkeit im Klassenpfad geprüft.

7. Prozessdefinition

Mit Hilfe der Schaltfläche *Ok* wird die eingegebene Bedingung gespeichert bzw. in das referenzierte Feld übernommen. Die Schaltfläche *Abbrechen* verwirft die getätigten, aber nicht zuvor gespeicherten Änderungen.

Bedingung: Editor **Eingabe**

☒ XPath ☐ WDL / Java Methode ☐ Groovy

`$form_form_businessstrip/approved = "true" and com.groiss.xpath.XPathUtil.compareDates("${form_form_businessstrip/tripstart}",$now,"=") or $form_form_businessstrip/employee = $user`

Schrittname: @@@approved@@@

Schritt-Id:

Ok Abbrechen

Abbildung 7.20: **Textueller Editor für Bedingungen**

8. Die Suche in *EPworX*

8.1 Prozesssuche

Die Prozesssuche ermöglicht es Ihnen Prozessinstanzen, in denen Sie als Benutzer involviert waren oder sind, (wieder) zu finden. Eine genaue Beschreibung dieser Funktion entnehmen Sie bitte dem [Benutzerhandbuch](#).

8.2 Dokumentensuche

Mit Hilfe der Dokumentensuche können Sie gezielt nach DMS–Objekten in *EPworX* suchen. Aktivieren Sie den Verweis *Dokumentensuche* in der Gruppe *Suche*. Eine genaue Beschreibung dieser Funktion entnehmen Sie bitte dem [Benutzerhandbuch](#).

8.3 Report Designer

Diese Funktion ermöglicht ebenfalls eine Suche, allerdings mit stark erweiterter Funktionalität. Dieser Eintrag taucht nur auf, wenn Sie das Recht für den Report Designer haben. Eine genaue Beschreibung dieser Funktion entnehmen Sie bitte dem Handbuch [Reporting](#).

8.4 Reports

In dieser Tabelle können Sie gespeicherte Reports bearbeiten und ausführen. Eine genaue Beschreibung dieser Funktion entnehmen Sie bitte dem Handbuch [Reporting](#).

9. Administrationsfunktionen

Die Administrationsfunktionen von *EPworX* werden im Bereich **Admin-Tasks** aufgelistet. Wenn Sie den Link **Konfigurierte Admin-Tasks** aktivieren, werden die von Ihnen hinzugefügten Admin-Tasks im Bearbeitungsbereich dargestellt.

9.1 Server

9.1.1 Servermonitor

EPworX verwendet das Tool *Java Melody* zur Überwachung des Servers und Auswertung von Serverinformationen.

Der Servermonitor bietet dabei folgende Informationen (siehe Abbildung 9.1):

- Überwachungsstatistiken
- Statistiken für HTTP Requests
- Statistiken für SQL-Statements
- Statistiken für HTTP Systemfehler
- Statistiken für System Fehlerlogs
- Aktuelle Anfragen
- Systeminformationen
- Threads

Standardmäßig ist *Java Melody* aktiviert, kann aber entweder über den Abschnitt *Web Komponenten* der *EPworX* Konfiguration mit dem Parameter *Servermonitoring mit Java Melody* deaktiviert werden.

Hinweis: Weitere Informationen zu *Java Melody* finden Sie in der [Installationshandbuch](#) im Kapitel [Web components](#) und in der *Online-Hilfe*, die über einen [Link](#) im Servermonitor erreichbar ist .

9. Administrationsfunktionen

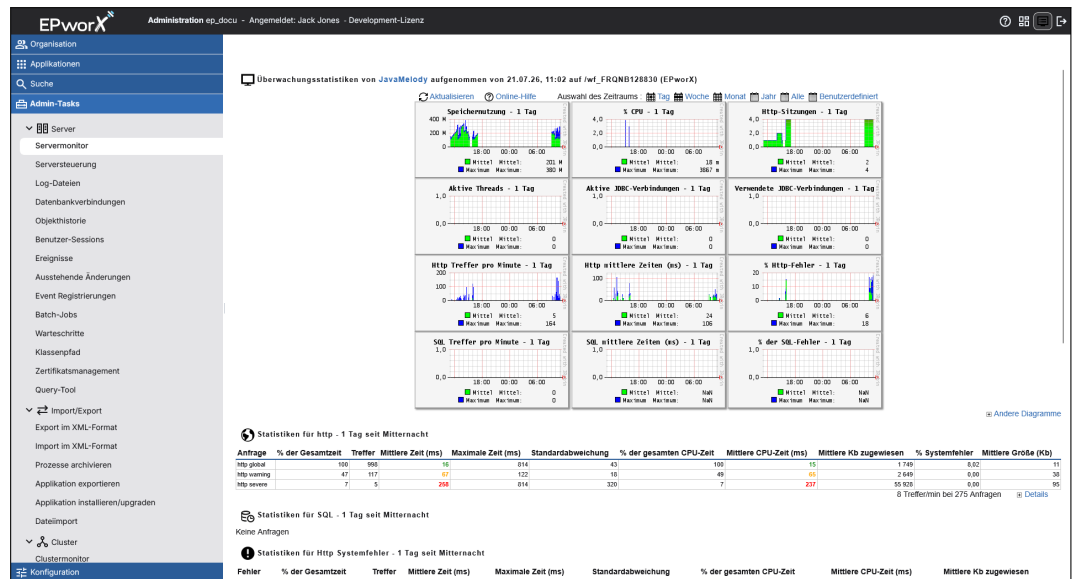


Abbildung 9.1: Servermonitor

9.1.2 Serversteuerung

In der Serversteuerung stehen Ihnen die folgenden Funktionen zur Steuerung und Kontrolle des Servers zur Verfügung:

- **Gestartet:** Gibt an, wann der Server gestartet wurde.
- **Abschalten:** Wenn Sie diese Funktion aktivieren, wird der Server abgeschaltet.
- **Neu starten:** Mit dieser Funktion starten Sie den Server neu.
- **Anmelden deaktivieren/aktivieren...:** Diese Funktion ermöglicht es Ihnen, das Anmelden für normale Benutzer zu deaktivieren bzw. wieder zu aktivieren. Ist das Anmelden deaktiviert, ist es nur einem Systemadministrator, der das Recht *Konfiguration* besitzt, möglich, sich anzumelden.

In der Maske stehen Ihnen die folgenden Informationen zur Verfügung:

- Anmelden aktiviert: Ist dieser Radio-Button aktiviert, gibt es keine Einschränkung beim Anmelden.
- Anmelden deaktiviert: Ist dieser Radio-Button aktiviert, können sich nach Neustart des Servers nur Systemadministratoren anmelden. Gleichzeitig werden - falls für die Applikation definierte - Upgrades von Applikationen durchgeführt. Für nähere Informationen darüber, sehen Sie bitte im API von *EPworX* nach (*ApplicationAdapter.getVersion()* und *ApplicationAdapter.upgrade()*).
- Nachricht: Wenn das Anmelden deaktiviert ist, können Sie hier eine Nachricht angeben, die dem Benutzer angezeigt wird, wenn er versucht sich an das System anzumelden.



- **Datenbankverbindungen neu aufbauen:** Alle Datenbank-Verbindungen (des aktuellen Knotens) werden als alt markiert. Vor jeder Zuordnung einer Verbindung zu einer Transaktion wird auf das Alter geprüft und anschließend die alten Verbindungen geschlossen und neue aufgebaut. Das Markieren umfasst also alle Verbindungen. Der Neuaufbau wird aber verzögert, je nach Verwendung von Verbindungen.



- **Alle Passwörter müssen beim nächsten Login geändert werden:** Wird diese Funktion aktiviert, muss jeder Benutzer beim nächsten Login sein Passwort ändern.



- **Nach neuen JAR-Libraries suchen:** Mit Hilfe dieser Funktion wird nach neuen JAR Dateien gesucht und diese dann dem Klassenpfad hinzugefügt.



- **Health Check:** Diese Funktion überprüft den Status des Systems. Dabei werden folgende Bereiche überprüft:
 - Allgemein: Prüfung auf Synchronität der Uhrzeiten von *EPworX* Server, Datenbank-Server und Client und Prüfung ob das Default-Passwort des sysadm gesetzt ist.
 - Shadow Klassen: In diesem Abschnitt wird geprüft, ob *EPworX* Ressourcen "geshadowed" werden.
 - Dateien und Verzeichnisse in der Konfiguration: Hier werden Konfigurationsparameter wie *Verzeichnis für Formularklassen*, *Server temporäres Verzeichnis*, *KeyStore Datei*, *web.xml*, *avw.java.compiler* und *httpd.jetty.favicon.path* geprüft, ob die Pfade bzw. Dateien am Dateisystem vorhanden sind.
 - Methoden und Klassen in der Konfiguration: Hier werden alle möglichen Methoden und Klassen geprüft, die in der Konfiguration von *EPworX* definiert wurden. Beispiele: *JDBC Treiber Klasse*, *Autorisationsklasse*, *Settings Klasse*, *Archivierungs-klasse*, *Error-Formatter Klasse*, *DMS-Archivierungs-klasse*, *Feiertags-Klasse*, etc.
 - Dateien und Verzeichnisse in den Applikationen: In diesem Abschnitt wird geprüft, ob zu den definierten Formulartypen die entsprechende Template-datei am Dateisystem vorhanden ist. Weiters werden die eingetragenen GUI-Konfigurationsobjekte geprüft, ob die entsprechenden XML-Dateien am Dateisystem vorhanden sind.
 - Methoden und Klassen in den Applikationen: Dieser Bereich prüft alle möglichen Methoden/Klassen, die bei den Applikationsobjekten eingetragen wurden. Beispiele: Prozess DMS-TableHandler, Formular Eventhandler, Methoden bei Tasks, Methoden von Eskalationen, Methoden von Funktionen, Timer-Methoden und Java-Methoden von Mailboxen.
 - Systemtabellen: Prüft, ob alle Systemtabellen vorhanden sind.
 - Formulartabellen: Hier werden alle Formulartabellen aller Applikationen geprüft und eine Liste von Funde ausgegeben wo die Größe der Spalte in der

Datenbank sich von jener des Formularfelds unterscheidet. Mit der Schaltfläche *Erweitere Spaltengrößen* können Sie bei varchar-Spalten die Spaltengröße automatisch an die des Formularfelds anpassen. Das passiert nur für jene Spalten, wo die Größe unterschiedlich ist und die Spaltengröße in der Datenbank kleiner als die Spaltengröße des Feldes des Formulars ist.

- Rollenzuordnungen: In diesem Bereich werden alle fehlerhaften Rollenzuordnungen gelistet, z.B. wenn eine lokale Rolle einem Benutzer zugeordnet ist ohne entsprechende Organisationseinheit.



- **Prüfe Fremdschlüssel:** Diese Funktion überprüft, ob die Fremdschlüssel in der Datenbank korrekt angelegt wurden. Gleichzeitig versucht diese Funktion noch fehlende Fremdschlüssel anzulegen und zeigt dies im Ergebnis an (erzeugt bzw. fehlgeschlagen).



- **Prüfe Hintergrundindizes für Fremdschlüssel:** Diese Funktion prüft, ob die Indizes für die Fremdschlüssel angelegt sind. SQL-Anweisungen für die Erstellung fehlender Indizes werden generiert, aber die Indizes selbst werden nicht automatisch generiert, um Leistungseinbußen zu vermeiden. Bitte mit Vorsicht verwenden.



- **Threads:** Mit Hilfe dieser Funktion können Sie Threaddumps auslösen, indem Sie diesen Link mehrmals aktivieren.



- **Server Info:** Diese Funktion liefert Informationen über den *EPworX* Server wie z.B. die verwendete Lizenz, Datenbank, verwendetes JDK, etc.



- **Client Info:** Diese Funktion liefert Informationen über Client spezifische Dinge wie z.B. verwendete Cookies, Session Variablen, etc.



- **Nachricht an alle angemeldeten Benutzer senden:** Mit dieser Funktion können Sie an alle angemeldeten Benutzer eine Nachricht verschicken. Diese Nachricht wird in einem eigenen Dialog im Smartclient angezeigt, nachdem die Schaltfläche *Senden* aktiviert wurde.



- **Konfiguration neu laden:** Diese Funktion hat die Aufgabe nach Änderungen in den Konfigurationsdateien von *EPworX* (z.B. *ep.conf*) bzw. *appl.prop* einer Applikation zu suchen (jedoch nicht in Konfigurationsparameter-Definition *properties.xml*), die nicht über das GUI erfolgt sind, und neu einzulesen bzw. geänderte Werte in den Speicher von *EPworX* (Configuration-Objekt) zu übernehmen. Anschließend wird für jedes Services (und jede Applikation, deren Applikationsklasse das Interface *com.groiss.component.Service* implementiert) die Methode *reconfigure()* aufgerufen. Nähere Informationen dazu finden sie im [Programmierhandbuch](#) - Kapitel [The Configuration Files](#).

Unter dieser Funktion werden in einer Tabelle für jede Applikation der aktuelle Status der Konfiguration angezeigt sowie Links auf die Konfigurationsdatei(en) angeboten. Die Tabelle beinhaltet folgende Spalten:

- Id: Die Id der Applikation. Durch Aktivieren des Links wird ein neues Fenster geöffnet, wo alle Parameter der Konfigurationsdateien dieser Applikation angezeigt werden. Falls der Parameter *ep.configuration.store.in.database* unter

Konfiguration/Weitere Parameter gesetzt ist, wird neben den Konfigurationsparametern der Datei nachfolgend auch noch die Parameter angezeigt, die in der Datenbank gespeichert werden - dieser Teil ist mit dem Text *Configuration content from DB follows* gekennzeichnet. Nähere Informationen zum Parameter *ep.configuration.store.in.database* finden Sie im [Installationshandbuch](#).

– Ladezeitpunkt: Dieser Zeitstempel gibt an, wann die Konfiguration von *EPworX* in den Speicher geladen wurde.

– Aktuell: In dieser Spalte wird der Status der Konfiguration angezeigt.



* aktuell: Die vorhandene Konfiguration im Speicher des *EPworX* Servers ist konsistent mit jenen aus den Dateien bzw. der Datenbank.



* veraltet: Die vorhandene Konfiguration im Speicher des *EPworX* Servers weicht von jenen aus den Dateien bzw. der Datenbank ab, d.h. die Werte aus den Dateien bzw. der Datenbank sind aktueller als die im Speicher des *EPworX* Servers. In diesem Fall kann mit der Funktion *Konfiguration neu laden* wieder ein konsistenter Zustand erzeugt werden, d.h. die Parameterwerte werden aus den Dateien bzw. der Datenbank neu ausgelesen und im Speicher aktualisiert.



* unbekannt: Es gibt keine Konfiguration für diese Applikation die geladen bzw. konsistent gehalten werden kann.

– Dateien/Änderungszeitpunkt: Hier werden die Pfade zu den Konfigurationsdateien angezeigt und der Zeitpunkt, wann die Dateien geändert wurden. Falls in der Spalte *Änderungszeitpunkt* der Wert N/A angezeigt wird, so ist die Konfigurationsdatei nicht vorhanden.

– Datenbank/Änderungszeitpunkt: Diese Spalten werden nur angezeigt, falls der Parameter *ep.configuration.store.in.database* unter *Konfiguration/Weitere Parameter* gesetzt ist. In der Spalte *Datenbank* wird die ID angezeigt, mit der die Konfiguration in der Datenbank identifiziert wird. Die Spalte *Änderungszeitpunkt* zeigt den Zeitpunkt an, wann die Konfiguration in der Datenbank geändert wurde. Falls in der Spalte *Änderungszeitpunkt* der Wert N/A angezeigt wird, so ist die Konfiguration nicht in der Datenbank vorhanden.



• **Java System Properties:** Diese Funktion öffnet ein Fenster mit einer Liste aller gesetzten Java System Einstellungen.



• **Environment Variables:** Diese Funktion öffnet ein Fenster mit einer Liste aller gesetzten Environment Variablen.



• **Aktueller Status:** Eine Übersicht über die aktuell ausgeführten und wartenden Reports.



• **Reporting Schema anzeigen:** Zeigt das Reporting Schema im XML-Format in einem eigenen Fenster an.



• **Reporting Schema neu laden:** Diese Funktion lädt das Reporting Schema neu.



- **Cache Statistiken anzeigen:** Zeigt Statistiken des Berechtigungs-Caches zu Debugging-Zwecken an.



- **Cache Infos anzeigen:** Zeigt interne Informationen des Berechtigungs-Caches zu Debugging-Zwecken an.

Worklist Cache

Im Bereich *Worklist Cache* wird der Status angezeigt (einstellbar im Konfigurationsabschnitt *Tuning*) und Sie können folgende Tätigkeiten durchführen:



- **Cache aktualisieren:** Das Aktualisieren der Cache-Strukturen ist in den folgenden Fällen nötig: Eintrag von neuen Applikationen, Abteilungen oder Rollen und bei Änderungen im Organisationsbaum. Dafür werden folgende Funktionen angeboten:
 - *Aktualisieren der Org. Strukturen:* Dadurch werden die Organisationsdatenstrukturen des Caches manuell aktualisiert.
 - *Aktivitätsinstanzen aktualisieren:* Die Arbeitskorbeinträge werden neu von der Datenbank gelesen; dies kann aber durch Ausschalten und wieder Einschalten des Caches bewerkstelligt werden.
 - *WLCache reparieren:* Falls Inkonsistenzen im Worklist-Cache eines Knoten N aufgrund der Java Garbage Collection auftreten, können diese mit dieser Funktion repariert werden. Durch Angabe eines Zeitintervalls (z.B. Start und Ende der Garbage Collection Phase des inkonsistenten Knotens) und Auswahl eines anderen Knoten M bekommt Knoten N Informationen von M über die geänderten Stepinstances dieses Zeitraums. Knoten N nutzt diese Informationen, um seinen internen Stand auf den Stand der aktuellen Datenbankdaten dieser Stepinstances zu bringen.

Bei beiden Funktionen wird der Zeitpunkt der letzten Ausführung in eckigen Klammern angezeigt.

Der Worklistcache liest beim Einloggen eines Users dessen aktuelle Rollenzuordnungen und Vertretungen. Damit werden Änderungen bei Rollen eines Users nach dem Einloggen wirksam. Änderungen bei Vertretungen (bei denen ja automatisch durch den *CurrentSubstitutesTask* Fristbeginn oder Fristablauf berücksichtigt werden) spiegeln sich sofort im Cache wieder, ein neuerliches Einloggen des Users oder ein Aktualisieren der Cachestrukturen ist nicht erforderlich.

Das Aktualisieren der Cachestrukturen instantiiert einige der Strukturen doppelt (die alte und die neue Version der Struktur). Es ist mit zusätzlichem Speicherplatzbedarf während der Operation bis zur nächsten Garbage-Collection zu rechnen.

Das API stellt zum Aktualisieren der Cachestrukturen zwei Methoden zur Verfügung:

- `com.dec.avw.wlcache.WLCache.getInstance().refresh()` Aktualisiert die Cachelstrukturen (alle Änderungen an Organisationsdaten); entspricht der manuellen Auslösung der Funktion. Empfohlen wird die Benutzung z.B. nach dem Import eine Menge von Benutzern und deren Rollen.
 - `com.dec.avw.wlcache.WLCache.getInstance().refreshUser(User u)` Geänderte Rollenzuordnungen und Vertretungen eines Benutzers werden im Cache berücksichtigt. Andere Modifikationen der Organisationsdaten (neue Applikationen, Abteilungen, Rollen . . .) bleiben unberücksichtigt.
- **Cache für Benutzer:** Damit können Sie die Konsistenz des Worklist-Caches für einen bestimmten Benutzer testen, indem Sie einen Benutzer auswählen und die Funktion *Anzeigen* aktivieren. Treten für Einträge im Arbeitskorb, Rollen-Arbeitskorb, Wiedervorlage, Rollen-Wiedervorlage bzw. für ausstehende Aktionen Inkonsistenzen auf, werden Ihnen diese angezeigt. Wenn Inkonsistenzen für den jeweiligen Bereich vorhanden sind, werden ein rotes Icon und die betroffenen Elemente angezeigt, ansonsten, wenn es keine Inkonsistenzen gibt, ein grünes Icon. Inkonsistenzen können mittels der Funktion *Aktualisieren der Org. Strukturen für Benutzer* behoben werden.

9.1.3 Log-Dateien

Je nach eingestelltem Log-Level werden die Zugriffe auf das System mitprotokolliert. Mit der Funktion *Logging* haben Sie die Möglichkeit, die aktuellsten Log- und Error-Log-Dateien anzusehen bzw. das aktuelle Log zu initialisieren. Im [Installationshandbuch](#) ist im Kapitel [Konfiguration](#) genauer beschrieben, welche Information bei welchem Log-Level in das Log-File geschrieben wird.

Folgende Tätigkeiten können auf dieser Maske durchgeführt werden:



- **Tabellenhandling:** Das aktuelle Log wird in Fettschrift dargestellt. Es kann zudem nach jeder angezeigten Spalte sortiert werden.
- **Logfile ansehen:** Nachdem ein Tabelleneintrag ausgewählt wurde, kann mit dieser Funktion das Logfile angesehen werden. Ein Doppelklick auf einen Tabelleneintrag führt ebenso diese Funktion aus.
- **Länge des Ausschnitts bzw. Ende des Logfiles ansehen:** Abhängig von dem Wert, der im Eingabefeld steht, wird beim Aktivieren der Funktion *Ende des Logfiles ansehen* das Log mit den letzten n Zeilen angezeigt. Der Standardwert des Eingabefeldes kann in der *Konfiguration* im Abschnitt *Logging* festgelegt werden.
- **Herunterladen:** Mit Hilfe dieser Funktion können Sie die ausgewählte Log-Datei als ZIP- oder Textdatei herunterladen.
- **Log-Datei initialisieren:** Nach Aktivieren dieser Funktion wird die bestehende Log-Datei wegkopiert und eine neue Log-Datei initialisiert.

9.1.4 Datenbankverbindungen

Diese Tabelle zeigt Ihnen aktuell offene Datenbankverbindungen an, die durch Statements ausgeführt werden. Ist ein Statement in der Spalte *Anweisung* kursiv dargestellt, ist die Anweisung bereits beendet, aber die Datenbankverbindung noch offen (d.h. es wurde kein Commit durchgeführt).

Wenn Sie Oracle als Datenbank verwenden, können DB-Verbindungen mit dem ThreadUser und dem busyObject gekennzeichnet werden in der Spalte *in Verwendung*, z.B. *sysadm*Thread[JHttp-23,5,main]*. Diese Kennzeichnung (Label) wird gesetzt, wenn eine Verbindung einem Thread angehängt wird und wird entfernt, wenn die Transaktion beendet wird. Das Label ist im Feld *client_identifizier* von View v\$session verfügbar. Dieses Label erhalten Sie, wenn Sie in der Konfiguration im Bereich *Datenbank* den Parameter *Warnungszeit bei lange reservierten DB Verbindungen (sek)* auf einen positiven Wert setzen.

Im Toolbar stehen Ihnen folgende Funktionen zur Verfügung:



- **Aktualisieren:** Diese Funktion lädt die Tabelle neu.



- **Anweisung abbrechen:** Das Ausführen diese Funktion kann unter Umständen notwendig sein, wenn eine Anweisung zu lange ausgeführt wird und abgebrochen werden muss.



- **Transaktion freigeben:** Mit dieser Funktion können Sie Datenbankverbindungen freigeben, indem die entsprechende User-Transaction zurückgesetzt wird (rollback). Dabei wird jede Datenbankverbindung der User-Transaction zurückgesetzt, geschlossen, an den DB-Pool zurückgegeben und auf den Status *Dubious* gesetzt. Weiters werden die afterCompletion()-Methoden der SessionSynchronisation Beans aufgerufen.



- **Histogramm:** Diese Funktion zeigt die Datenbankverbindungen in einem Histogramm an. Beim Histogramm haben Sie die Möglichkeit *Einheit* und *Spalten* festlegen.



- **Statement-Statistik:** Wenn in der Konfiguration unter *Tuning* die Checkbox *Statement-Statistik* aktiviert wurde, werden hier sämtliche DB-Anweisungen des Systems angezeigt.

9.1.5 Objekthistorie

Diese Funktion ermöglicht es Ihnen, die Veränderungen in den einzelnen Daten anzusehen. Es ist auch möglich sich die Änderungen der einzelnen Objekte, bezogen auf ein bestimmtes Zeitintervall, anzusehen.

Wählen Sie zunächst aus der Liste **Objektklasse** jene Objektklasse aus, deren Änderungen Sie ansehen möchten. Wenn Sie *Alle* auswählen, findet hier keine Einschränkung statt und Sie sehen eine Liste aller Änderungen in allen Objektklassen. Optional können Sie auch einen Zeitraum eingeben, in dem Sie die Änderungen verfolgen wollen. Starten Sie die Suche durch Aktivieren der Schaltfläche **Suche starten**.



In einer Tabelle werden Ihnen nun alle Änderungen der jeweiligen Objektklasse angezeigt. Sie können sich die einzelnen Einträge durch Aktivieren der Funktion **Anzeigen** im Detail ansehen. Mit der Schaltfläche **Zurücksetzen** können Sie die Suchkriterien wieder auf den Ausgangszustand zurücksetzen.

9.1.6 Benutzer-Sessions

Mit Hilfe der Administrationsfunktion *Benutzer-Sessions* können Sie feststellen, wann welche Benutzer im System angemeldet waren.

Sobald sich ein Benutzer am System anmeldet, wird die Anzahl der angemeldeten Benutzer mit der Lizenz gegengeprüft (Concurrent-User). Ist das Anmelden möglich, wird eine sogenannte *Benutzer-Session* generiert. Diese Benutzer-Session ist so lange gültig, bis sich der Benutzer selbst abmeldet (*Logout*). Wird das Logout nicht durchgeführt, ist die Benutzer-Session 24 Stunden gültig und wird danach automatisch beendet. Zur Lizenzüberprüfung werden allerdings nur Benutzer-Sessions herangezogen, die höchstens seit zwei Stunden inaktiv sind.

Sie können sich die Benutzer-Sessions entweder als Benutzerliste oder in Form eines Histogramms ausgeben lassen. Damit die Ausgabe übersichtlich bleibt, können Sie weiters den Zeithorizont einstellen. Als Ergebnis werden Ihnen alle Benutzer-Sessions ausgegeben, welche in diesem Zeithorizont aktiv waren, d.h. wo das Login nach dem Datum (und Uhrzeit) im Datumsfeld **Von**, und das Logout vor dem Datum (und Uhrzeit) im Datumsfeld **Bis**, liegt. Nach Aktivieren der Schaltfläche **Suche starten** wird eine Ergebnistabelle mit allen Benutzer-Sessions aufgrund der angegebenen Bedingungen angezeigt. Mit der Schaltfläche **Zurücksetzen** können Sie die Suchkriterien wieder auf den Ausgangszustand zurücksetzen. Durch einen Doppelklick auf einen Eintrag in der Ergebnistabelle oder mittels Toolbar-Funktion *Details* können die Session-Details angezeigt werden. Dadurch wird ein neues Fenster geöffnet, in welchem Sie Detailinformationen zu dieser Aktivität erhalten (u.a. HTTP-Sessions). Falls gerade ein Thread ausgeführt wird (eine Aktion des Benutzers), wird dieser auch angezeigt. Die Funktion *Aktualisieren* aktualisiert die Daten für die vorhandene Benutzer-Session. Über die Funktion *HttpSession* werden die Session-Details in einem neuen Fenster dargestellt.

Benutzerliste: Für die Ausgabe einer Benutzerliste, müssen Sie eine weitere Einschränkung vornehmen.

- **Angemeldete Benutzer:** Liefert Ihnen alle Benutzer-Sessions, die aktuell aktiv sind.
- **Alle Benutzer:** Liefert Ihnen alle Benutzer-Sessions.
- **Ausgewählte Benutzer:** Liefert Ihnen alle Benutzer-Sessions des ausgewählten Benutzers.

Mit der Option **letzter Zugriff (Minuten)** können Sie abhängig von der ausgewählten Option bei *Benutzerliste* festlegen, welche Benutzer angezeigt werden, wo der letzte Zugriff vor x Minuten durchgeführt wurde.

Sie können die Benutzerliste mit Hilfe des Zeithorizonts weiter einschränken! Folgende Informationen finden Sie in der Ergebnistabelle:

- **Benutzer:** Enthält den Vor- und Nachnamen des Benutzers.
- **Client IP:** Die IP-Adresse des Benutzers.
- **Anmeldung:** Das Initialisierungsdatum der Benutzer-Session, d.h. der Zeitpunkt an dem sich der Benutzer angemeldet hat.
- **Letzter Zugriff:** Der Zeitpunkt an dem der Benutzer das letzte Mal im System aktiv war.
- **Abmeldung:** In dieser Spalte sehen Sie den Abmeldungszeitpunkt. Über die Toolbar-Funktion *Abmelden* können die Sessions beendet und der Benutzer ausgeloggt (auch Clusterweit) werden.
- **Knoten-Id:** Enthält die Cluster Knoten-Id.
- **Trace Level:** Hier wird das gesetzte *Trace Level* angezeigt. Das Trace Level für eine oder mehrere Sessions kann mit der Toolbar-Funktion *Trace Level setzen* definiert oder geändert werden.

Histogramm: Wenn Sie sich für die Ausgabe des Histogramms entscheiden, wird für das jeweilige Intervall die Anzahl der Benutzer-Sessions ermittelt und jeweils der höchste Wert ausgegeben. Liegen in einem Intervall keine Messwerte vor, d.h. es hat kein Login stattgefunden, wird für das Intervall kein Wert ausgegeben. Im Histogramm steht in jeder Zeile der Startzeitpunkt des Intervalls.

- **Stunde:** Das Zeitintervall beginnt jeweils mit 0 Minuten.
- **Tag:** Das Zeitintervall beginnt jeweils um 0 Uhr.
- **Woche:** Die Woche beginnt jeweils mit dem Starttag des Zeithorizonts um 0 Uhr.

9.1.7 Ereignisse

Unter dem Menüpunkt *Ereignisse* können Sie nach protokollierten Systemereignissen von *EPworX* suchen:

- **Von und Bis:** Zeitliche Einschränkung der Ereignisse
- **Typ:** Hier können Sie den Ereignistyp *startup* bzw. *shutdown* festlegen
- **Suche starten:** Mit dieser Funktion können Sie nach allen protokollierten Systemereignissen suchen, abhängig von den Suchoptionen, die getätigt wurde. Wurden keine Einschränkungen festgelegt, werden alle Systemereignisse in einer Tabelle angezeigt.
- **Zurücksetzen:** Diese Funktion setzt die Suchkriterien wieder auf den Ausgangszustand zurück.
- **Alle löschen:** Diese Funktion erlaubt das Löschen aller protokollierten Systemereignisse, die in der Tabelle angezeigt werden.

9.1.8 Ausstehende Änderungen

Hier werden alle Objekte aufgelistet, für die bevorstehende Änderungen definiert wurden (mit der Funktion *Änderungen durchführen am*, siehe Kapitel [Änderungen zu späterem Zeitpunkt](#)).



Sie können sich die Details zum jeweiligen Objekt mit der Funktion **Anzeigen** ansehen. Sie können die ausstehende Änderung entweder direkt in der Detailansicht oder mit der Funktion **Löschen** verwerfen.



9.1.9 Event Registrierungen

Durch das Anklicken des Links *Event Registrierungen* werden Ihnen im Arbeitsbereich alle Event Registrierungen von laufenden Prozessen angezeigt. Die Tabelle enthält die folgenden Informationen:

- Event Registrant: Die Id des Prozesses, der sich für diesen Event registriert hat.
- Event Name: Der Name des Events für den die Registrierung erfolgt ist.
- Kontext: Der Kontext für den Event.
- Handler: Der Event Handler.

In der Werkzeugleiste stehen Ihnen die folgenden Funktionen zur Verfügung:



- *Aktualisieren*: Wenn Sie diese Funktion aktivieren, wird die Tabelle aktualisiert.



- *Ansehen*: Wenn Sie diese Funktion aktivieren, wird die zum Prozess gehörende Prozesshistorie angezeigt. Diese Funktion wird auch beim Doppelklick auf einen Tabelleneintrag ausgelöst.



- *Weiterleiten*: Nach dem Aktivieren dieser Funktion wird der Prozess weitergeleitet (wie im Arbeitskorb).



- *Kurzsuche*: Geben Sie eine Prozess-Id in das Textfeld ein und aktivieren Sie diese Funktion, um ein eingeschränktes Ergebnis zu erhalten.



- *Alle Einträge*: Aktivieren Sie diese Funktion, um wieder alle Einträge der Tabelle anzuzeigen.

9.1.10 Batch-Jobs

Mit dieser Funktion ist es möglich nach Batch-Jobs zu suchen. Als Suchkriterien können sowohl die Prozess-Id als auch der Status eines Batch-Jobs verwendet werden. Weiters kann nach einer gewissen Zeitspanne eingegrenzt werden, wann der Batchschritt gestartet wurde. Nach Aktivieren der Schaltfläche *Suche starten* wird eine Tabelle mit allen Batchschritten aufgrund der angegebenen Bedingungen angezeigt. Durch einen Doppelklick auf einen Eintrag können die Details zu diesem Batchschritt angezeigt und anschließend verändert werden. Durch das Aktivieren der Schaltfläche *Zurückgehen*

werden alle Suchkriterien wieder auf den Ausgangszustand zurückgesetzt.

Nähere Informationen zu Batch-Jobs finden Sie im [Programmierhandbuch](#) im Abschnitt [Batch Processing](#).

9.1.11 Warteschritte

Diese Funktion erlaubt es nach Prozessinstanzen zu suchen mit derzeit ausgeführten Warteschritten (siehe dazu Abschnitt [Ereignis Warten](#)). Als Suchkriterium kann die Prozess-Id angegeben werden. Zusätzlich kann das Suchergebnis auf eine gewisse Zeitspanne eingegrenzt werden, wann der Warteschritt gestartet wurde bzw. bis wann er ausgeführt wird. Nach Aktivieren der Schaltfläche *Suche starten* wird eine Tabelle mit allen Prozessinstanzen aufgrund der angegebenen Bedingungen angezeigt. Mit der Schaltfläche *Zurücksetzen* können Sie die Suchkriterien wieder auf den Ausgangszustand zurücksetzen. Durch einen Doppelklick auf einen Eintrag in der Ergebnistabelle können die Prozessdetails angezeigt werden. Mit der Toolbarfunktion *Endzeitpunkt des Warteschritts ändern* können Sie den Warteschritt entweder sofort beenden oder die Zeitspanne verlängern (mit optionaler Angabe eines Kommentars).

9.1.12 Klassenpfad

Mit dieser Funktion können Sie *Klassen neu laden*, *Klassen auffinden* und *Shadow Klassen anzeigen*. Diese Seite zeigt Ihnen außerdem, welche Verzeichnisse aktuell im Klassenpfad eingebunden sind.

Klassen neu laden

Aus Gründen des Benutzerkomforts für Applikationsprogrammierer wird das Neu laden von Klassen unterstützt. Eine Vorbedingung dafür ist, zwischen Systemklassen und Applikationsklassen (und Ressourcen) zu unterscheiden.

Systemklassen werden vom System-Klassen-Lader geladen und können nicht entladen werden. Diese Klassen befinden sich im Verzeichnis *lib* der Installation.

Die Applikationsklassen befinden sich in den Verzeichnissen *lib* und *classes* der Applikation. Die Formularklassen, die vom System erzeugt wurden, befinden sich im Verzeichnis *forms* der Installation. Diese Klassen werden vom Applikations-Klassen-Lader geladen und können neu geladen werden, ohne den Server neu starten zu müssen. Dies kann auf zwei Arten erfolgen:

1. In der Serverkonfiguration unter Punkt *Tuning* ist die Checkbox *Klassen neu laden* angekreuzt. Soll nun eine Klasse verwendet werden, leert der Applikations-Klassen-Lader seinen Cache, und lädt die Klasse danach automatisch neu.
2. Ist die unter Punkt 1 erwähnte Checkbox nicht angekreuzt, können Sie durch das Aktivieren der Schaltfläche *Klassen neu laden* die Klassen manuell neu laden.

Anmerkung: Das Neu laden einer Applikationsklasse ist nicht möglich, wenn die Instanz einer Klasse aus dem Systemklassenpfad auf die Instanz der Applikationsklasse

verweist. In diesem Fall muss der Server neu gestartet werden, damit die entsprechende Applikationsklasse neu geladen wird.

Klassen auffinden

Wenn Sie nicht mehr wissen, wo eine Klasse, die Sie verwenden, definiert ist, dann geben Sie den Klassennamen (inkl. Package) in das Feld *Klasse oder Ressource suchen* ein und aktivieren Sie das Lupen-Symbol bzw. drücken Sie die Enter-Taste. Wenn die Klasse gefunden wird, wird der Pfad bei *URL* ausgegeben.

Shadow Klassen anzeigen

Diese Funktion zeigt Ihnen alle Klassen und Ressourcen, die im Klassenpfad mehr als einmal zu finden sind. Die Anzeige erfolgt in folgender Syntax:

relativer Pfad der Ressource Anzahl der gefundenen Ressourcen mit diesem relativen Pfad

Absoluter Pfad der vom System verwendeten Ressource

Absoluter Pfad der vom System nicht verwendeten Ressourcen

...

9.1.13 Zertifikatsmanagement

Um SSL verwenden zu können brauchen Sie ein Zertifikat. Zertifikate und die dazugehörigen Private und Public Keys werden im sogenannten Server-Keystore gespeichert. Bevor Sie nun ein Zertifikat generieren können, müssen Sie den Keystore konfigurieren und den SSL-Port festlegen.

Hinweis: Unter dem Punkt *Security* bzw. *HTTP-Server* müssen zunächst die entsprechenden Einstellungen getätigt werden (siehe [Installationshandbuch](#))! Nach einem Server Restart können Sie die Funktionalität des Zertifikatsmanagements nutzen.

Zunächst müssen Sie ein KeyPair bestehend aus Private und Public Key generieren, das gleichzeitig als selbst signiertes Zertifikat gilt. Dieses ist aber nur für Testzwecke zu empfehlen. Für sensible Daten benötigen Sie ein Zertifikat, das von einer Certification Authority (kurz CA) ausgestellt wurde. Um zu einem solchen Zertifikat zu gelangen, müssen Sie allerdings zuerst einen entsprechenden Zertifikations-Request (CR) zu einem KeyPair erzeugen und an diese CA schicken. Diese schickt Ihnen dann ein fertiges Zertifikat, das Sie in *EPworX* integrieren und somit eine sichere Kommunikation sicherstellen können.

Zertifikate werden auch zum Unterschreiben von PDF-Dokumenten verwendet. Hierfür können Benutzer selbst selbstsignierte Zertifikate erstellen, Zertifikatsrequests herunterladen und Zertifikate importieren bzw. diese Aktionen können auch hier in der Administration pro Benutzer durchgeführt werden. Wenn man als Rolle (zum Beispiel Rolle sales) ein Dokument unterschreiben will, muss in der Administration ein Zertifikat für die

entsprechende Rolle angelegt werden. Jeder Benutzer, der die Rolle zugeordnet hat, kann dann - sofern ihm das Passwort bekannt ist - mit diesem Zertifikat Dokumente unterschreiben. Bei Benutzer- und Rollen-Zertifikate wird kein Server-Keystore benötigt (siehe Einstellungen im Konfigurationsbereich *Security*), d.h. ein Erstellen dieser Zertifikate ist immer möglich!

Selbst signiertes Zertifikat generieren

Selbst signiertes Zertifikat generieren

Alias Name: *	selfsigned
Land: *	AT
Firmenname: *	PDTS GmbH
Organisationseinheit: *	PDTS
E-Mail:	
Hostname / Name: *	Dr Walter Heisenberg
Firmensitz: *	Wien
Bundesland: *	Wien
Gültigkeitsdauer des Keys in Tagen (Default=90): *	90
Schlüsselpasswort: *	●●●●●●●●
Passwortbestätigung: *	●●●●●●●●
Schlüssellänge in bit:	☒ 1024 ☐ 2048

OkSchließen

Abbildung 9.2: Formular für selbstsigniertes Zertifikat

Folgende Informationen sind hier von Ihnen anzugeben:

- **Alias Name:** Geben Sie hier eine eindeutige Id für diesen Eintrag ein.
- **Land:** Hier geben Sie den Ländercode des jeweiligen Landes ein. Für Österreich ist dies AT.
- **Firmenname:** Sie müssen hier die vollständige Bezeichnung Ihrer Firma angeben.
- **Organisationseinheit:** Hier wird der Name der Abteilung, für die das Zertifikat erstellt werden soll, verlangt.

- **E-Mail:** Die E-Mail Adresse des verantwortlichen Mitarbeiters ist hier einzutragen.
- **Hostname / Name:** Hier müssen Sie den Hostnamen jenes Rechners angeben, auf dem der *EPworX* Server läuft, für den das Zertifikat ausgestellt werden soll.
- **Firmensitz:** Der Name der Stadt, in der sich ihre Firma befindet, ist hier einzutragen.
- **Bundesland:** Der Name des Bundeslandes, in der sich ihre Firma befindet, ist hier einzutragen.
- **Gültigkeitsdauer in Tagen:** Ein Zertifikat hat eine gewisse Gültigkeitsdauer. Defaultwert ist hier 90 Tage.
- **Schlüssellänge in bit:** Hier können Sie die Länge der zu erzeugenden Schlüssel auswählen.

Nachdem Sie dieses Zertifikat generiert haben, ist SSL prinzipiell betriebsbereit. Es wird jedoch empfohlen, ein offizielles Zertifikat einer CA zu beantragen.

Zertifizierungsrequest erzeugen

Um einen CR zu erzeugen, müssen Sie auf der Seite für das Zertifikatsmanagement das selbst signierte Zertifikat mit einem Doppelklick auswählen. Sie erhalten den Zertifizierungsrequest als Download. Wurde der CR von Ihnen erzeugt, so können Sie damit bei der CA Ihres Vertrauens ein Zertifikat anfordern. Wie Sie das machen, entnehmen Sie bitte der Dokumentation der jeweiligen CA. Testzertifikate erhalten Sie zum Beispiel bei www.secude.com/trustfactory/ oder www.trustcenter.de.

Importieren eines Zertifikats

Wenn Sie von der CA Ihr Zertifikat im PEM-Format oder DER-Format erhalten haben, so müssen Sie dieses in den Keystore importieren. Klicken Sie auf Zertifikat importieren.

Folgende Informationen sind hier von Ihnen anzugeben:

- **Alias Name:** Geben Sie hier die Id des Eintrages an, zu dem der Zertifizierungsrequest erzeugt wurde. Ist das Zertifikat ein fremdes vertrauenswürdiges Zertifikat, so geben Sie eine noch nicht verwendete ID an.
- **Passwort für privaten Schlüssel:** Hier müssen Sie ein beliebiges Passwort für den privaten Schlüssel angeben.
- **Kodierung:** Zertifikate können im PEM (Base64 codiert) oder DER (binär) Format von der CA zugesendet werden.
- **Zertifikatsfeld:** Liegt Ihr Zertifikat im PEM Format vor, kopieren Sie das Zertifikat in dieses Textfeld.

Zertifikat importieren

Alias Name: *

Passwort für privaten Schlüssel: *

Schlüssel: *

Kodierung des Zertifikates : * ☒ Binär (DER, PKCS#7) ☐ Base64 kodiert

Datei mit Zertifikat: *

Kodierung des Zertifikates: * ☐ Base64 kodiert

Fügen Sie Ihr Zertifikat hier ein (inkl. der Kopf- und Fusszeile): *

☐ Zertifikat einer vertrauenswürdigen Organisation

☒ Serverzertifikat

Abbildung 9.3: Zertifikate importieren

- **Zertifikatsdatei:** Wenn Sie ein DER Zertifikat vorliegen haben, so geben Sie hier den Dateipfad Ihres Zertifikats an.
- **Art des Zertifikats:** Ist dieses Zertifikat jenes, mit dem sich der *EPworX* Server authentifizieren soll, so geben Sie hier Serverzertifikat an. Ist das zu importierende Zertifikat ein Zertifikat einer vertrauenswürdigen Organisation (z.B: Mail-Server, CA für Clientzertifikate) so geben Sie bitte dies an.

Um Zertifikatsketten zu importieren, muss das eigentliche Zertifikat an erster Stelle im Textfeld oder der DER-Datei stehen, gefolgt von dem Zertifikat der CA, die das Zertifikat ausgestellt hat. Diese Kette kann beliebig lang sein, bis hin zu der sogenannten Root-CA!

Zertifikat samt KeyPair importieren

Mit Hilfe dieser Funktion können Sie WildCard Zertifikate importieren. Ein Wildcard-Zertifikat ermöglicht Ihnen die Verwaltung mehrerer Subdomains einer Domain mit einem einzigen Zertifikat. Da man statt vieler einzelner Zertifikate nur ein einzelnes Wildcard-Zertifikat benötigt, ist das Wildcard-Zertifikat für viele Organisationen eine kosteneffektive und effiziente Lösung.

Folgende Informationen sind hier von Ihnen im Dialog *Zertifikat samt KeyPair importieren* anzugeben:

- **Alias Name:** Geben Sie hier die Id des Eintrages an, zu dem der Zertifizierungs-

Zertifikat samt KeyPair importieren

Alias Name: *

Passwort für privaten Schlüssel: *

Zertifikatsdatei (.der Format): *

Privatekey Datei (.der Format): *

Abbildung 9.4: **Zertifikat samt KeyPair importieren**

request erzeugt wurde. Ist das Zertifikat ein fremdes vertrauenswürdiges Zertifikat, so geben Sie eine noch nicht verwendete ID an.

- **Passwort für privaten Schlüssel:** Hier müssen Sie ein beliebiges Passwort für den privaten Schlüssel angeben.
- **Zertifikatsdatei:** Geben Sie hier den Pfad zur Zertifikatsdatei an, welche im DER-Format vorliegen muss.
- **Privatekey Datei:** Geben Sie hier den Pfad zur Privatekey Datei an, welche im DER-Format vorliegen muss.

9.1.14 Query-Tool

Das Query-Tool stellt ein einfaches Interface zum lesenden Zugriff auf Datenbanken zur Verfügung. Da die Nutzung dieser Funktionalität mit beträchtlichen Sicherheitsrisiken verbunden sein kann, steht diese nur unter zwei Voraussetzungen zur Verfügung:

- Der Konfigurationsparameter *database.direct.access* hat den Wert *true*. Sie können diesen Parameter per Hand direkt im Konfigurationsfile eintragen oder in der Konfiguration unter *Weitere Parameter* die Checkbox *database.direct.access* aktivieren.
- Der Benutzer muss das Recht *Objekt ausführen* auf alle Objekte besitzen (jeder Benutzer mit der Rolle *sys* hat dieses Recht). Vertretungen werden hier nicht berücksichtigt.

Während das Query-Tool hauptsächlich dazu gedacht ist, ad-hoc SQL-Anfragen an die "lokale" Datenbank von *EPworX* zu stellen, kann es auch dazu benutzt werden, Konnektivitätsprobleme zu anderen Datenbanken zu untersuchen bzw. deren Daten zu inspizieren. Ist der Radio-Button *Lokal* aktiviert, werden die Anfragen gegen das zentrale Datenbank-Schema von *EPworX* ausgeführt.

Ist der Radio-Button *Andere Datenbank* aktiviert, erscheint eine Gruppe von Feldern, die es erlauben, die technischen Eigenschaften der Verbindung zur externen Datenbank anzugeben. Sie können die Hilfefunktion (das Fragezeichen neben dem *JDBC*

URL Feld) verwenden, um die Felder *JDBC Treiber Klasse* und *JDBC URL* mit Mustern für konkrete Werten zu befüllen. Ändern Sie die installationsspezifischen Parameter wie Hostname, Port und Datenbank-IDs auf diejenigen Parameter, die für Ihre Umgebung geeignet sind. Der *JDBC Treiber*, der die JDBC Treiber Klasse enthält, muss im Klassenpfad verfügbar sein. Das erfolgt üblicherweise, indem Sie die entsprechende *.jar Datei ins *lib* Verzeichnis der *EPworX* Installation kopieren. Über das Icon rechts neben dem Feld *JDBC Treiber Klasse* können Sie die Verfügbarkeit der angegebenen Klasse überprüfen. Anschließend tragen Sie die Anmeldedaten in die Felder *Datenbank Userid* und *Datenbank Passwort* ein.

Wenn in *Kommunikation* → *Externe Datenbanken*

eigene Datenbankverbindungen konfiguriert sind, kann man über die *Externe Datenbank* Option eine Verbindung zu einer der dort definierten Datenbanken erstellen. Alle konfigurierten Verbindungen sind in der *Externe Datenbank* Drop-Down Liste zu finden. Weitere Informationen finden Sie unter [Externe Datenbanken](#).

Unabhängig davon, ob die lokale/interne Datenbank oder ob externe Datenbanken angesprochen werden sollen, können Sie die Anfrage (nur *select* Anweisungen sind erlaubt) im Textfeld **Anweisung** eingeben und diese Anfrage durch Aktivieren der Schaltfläche **Ausführen** absenden. Die Ergebnisse der Anfrage werden im unteren Bereich der Seite angezeigt.

Weiters wird nach den Ergebnissen optional der Query-Plan der Anfrage dargestellt. Dies funktioniert nur unter Oracle und nur wenn der Benutzer das *STAT*-Recht hat. Weitere Informationen zum Konfigurieren dieser Funktionalität finden Sie im Abschnitt [Installation/ Database Preparation/Oracle](#) des Handbuchs [Installation and Configuration](#).

Hinweis: Nach einer solchen Änderung ist ein Neustart von *EPworX* erforderlich.

Bitte beachten Sie, dass die Ausgaben von *Query-Plans* ins Logfile geschrieben werden und dass diese Funktionalität nicht gedacht ist, im Produktionsbetrieb dauerhaft eingeschaltet zu sein.

Zusätzlich zu Anfragen kann das **describe** Kommando benutzt werden, um Metainformationen über die Datenbank zu erhalten. Für Details zu den Resultaten sei auf die entsprechende Java API Dokumentation zu *java.sql.DatabaseMetaData* verwiesen.

Das *describe* Kommando kann folgende Formen annehmen:

- *describe DBSchemas* - zeigt die verfügbaren Schemata an
vgl. *DatabaseMetaData.getSchemas()*
- *describe SQLTypes* - zeigt die verfügbaren Datentypen des DBMS an
vgl. *DatabaseMetaData.getTypeInfo()*
- *describe table_name* - zeigt Informationen über die Tabelle an: ¹
 - Spalten: vgl. *DatabaseMetaData.getColumns()*,
 - Primärschlüssel keys: vgl. *DatabaseMetaData.getPrimaryKeys()*,

¹Unter Verwendung von Oracle kann das *describe table_name* Kommando langsam sein (viele Sekunden); da ist etwas Geduld erforderlich.

- Fremdschlüssel: vgl. *DatabaseMetaData.getImportedKeys()* und *DatabaseMetaData.getExportedKeys()*,
- Indizes: vgl.. *DatabaseMetaData.getIndexInfo()*.

Der *table_name* kann auch durch einen *schema_name* oder durch einen *catalog_name* in Kombination mit einem *schema_name* qualifiziert werden, z.B.:

- *describe table_name*
- *describe otherschema.othertable*
- *describe othercatalog.otherschema.othertable*

9.2 Import/Export

9.2.1 Import/Export im XML Format

Die Import/Export Funktionalität erlaubt es Ihnen, Daten von einem *EPworX* Server zu exportieren und auf einem anderen zu importieren. Die Daten werden hierbei in einer XML Datei gespeichert.

Export

Sie können verschiedene Arten von Daten exportieren. Der XML Export zeigt Ihnen eine Auswahl der zur Verfügung stehenden Daten und lässt Sie zusätzliche Optionen wählen. Abbildung 9.5 zeigt die verfügbaren Arten von Daten. Wenn es zusätzlich auswählbare Optionen gibt, wird ein Bereich mit Einstellmöglichkeiten sichtbar (wie in der zuvor genannten Abbildung für Organisationseinheiten zu sehen). Im ersten Teil der Export Maske können Sie eine optionale Beschreibung des Exportes eingeben. Diese Beschreibung wird in die XML Datei integriert und bei einem späteren Import angezeigt.

Mit Hilfe der Checkbox *Referenzierte Benutzer auf sysadm setzen* wird an so vielen Stellen wie möglich versucht Benutzerreferenzen auf den sysadm-Benutzer zu setzen. Wo das z.B. nicht gemacht wird ist bei Berechtigungslisten-Einträge oder auch bei Schrittakteuren einer Prozessdefinition. Auch ist es nicht sinnvoll diese Checkbox bei jeder Export-Art anzuwenden, da Sie die Daten verfälschen kann.

Sie können die folgenden Daten exportieren:

Applikationen Export jeweils einer Applikation mit allen Daten. Dazu zählen die Daten, die mit Prozessen exportiert werden (siehe Punkt Prozesse) und Daten, die zusätzlich im Applikationskontext definiert wurden: Rechte, Objektklassen, Task-Funktionen, Tasks, Formulare und Rollen, GUI-Konfigurationen, etc. Testfälle werden nicht mit Applikationen mitexportiert, dafür gibt es einen eigenen Exporter (siehe weiter unten).

9. Administrationsfunktionen

Abbildung 9.5: Export im XML-Format

Die Checkbox *Inkl. Applikationsverzeichnis* ist standardmäßig nicht aktiviert, das bedeutet, dass das beim Applikationsobjekt eingetragene Applikationsverzeichnis beim Export ignoriert wird, d.h. wird die exportierte Datei am Zielsystem importiert, dann wird das Applikationsverzeichnis am Zielsystem nicht geschrieben bzw. ein bestehendes nicht ersetzt. Soll aber das Applikationsverzeichnis am Zielsystem auf das Verzeichnis gesetzt werden, welches am Quellsystem definiert wurde, dann muss diese Checkbox aktiviert werden.

Mit der Checkbox *Inkl. Reports* können Sie definieren, ob alle der Applikation angehörigen Reports mitexportiert werden sollen oder nicht (standardmäßig werden sie mitexportiert).

Prozesse Export eines oder mehrerer Prozesse mit den darin verwendeten Benutzern, Organisationseinheiten, Tasks, Steps, Formularen, Rollen, Prozess-Interfaces und Zugriffsrechten auf exportierte Objekte, usw. Die Zugriffsrechte werden am Zielsystem nur importiert wenn die benötigten Benutzer, Rollen und Organisationseinheiten bereits existieren.

Option „Inkl. Rollen“: Ist diese Option ausgewählt, werden referenzierte Rollen mitexportiert. Andernfalls werden diese als Referenzen im Export-File vermerkt.

Option „Inkl. Formulartypen“: Ist diese Option ausgewählt, werden referenzierte Formulartypen mitexportiert. Andernfalls werden diese als Referenzen im Export-File vermerkt.

Formulartypen Export von einem oder mehreren Formulartypen. Das Formular-Template wird nicht mitexportiert (ist normalerweise im Applikations-Jar enthalten).

Option „Inkl. Subformulare“: Ist diese Option ausgewählt, werden referenzierte

Formulare mitexportiert. Sonst werden diese als Referenzen im Export-File vermerkt.

Wertelisten Export von einer oder mehreren Wertelisten.

Organisationseinheiten Export aller Organisationseinheiten.

OE-Bäume Export von OE-Bäume und dessen Organisationseinheiten.

Benutzer Export aller Benutzer. Optional können Rollenzuordnungen, Rechtezuordnungen, Benutzereinstellungen und die Dashboard Elemente der Benutzer mit exportiert werden.

Achtung: Benutzereinstellungen enthalten eine Startseite. Sowohl in dieser Startseite als auch in den Dashboard Elementen können OIDs von beliebigen Objekten vorkommen. Diese werden beim Import nicht geändert, wodurch Startseiten und Dashboard Elemente möglicherweise am Importserver nicht mehr funktionieren könnten.

Berechtigungslisten Export aller Berechtigungslisten im System.

Reports Export einer oder mehrerer gespeicherter Anfragen. Die zu exportierenden Objekte können in einem zweiten Schritt ausgewählt werden. Optional können Zugriffsrechte auf die gespeicherten Anfragen mit exportiert werden. Referenzierte Objekte werden nicht mit exportiert (in gespeicherten Anfragen können unterschiedlichste Objekte vorkommen - z.B. Prozessdefinitionen, Tasks, etc.). Am Zielsystem müssen diese Objekte vorhanden sein, damit die gespeicherten Anfragen importiert werden.

Timer Export eines oder mehrerer Timer. Die Timer können in einem zweiten Schritt ausgewählt werden.

LDAP Einstellungen Export aller LDAP Einträge. Exportiert die LDAP Einträge, welche unter

Kommunikation → LDAP

definiert sind.

Externe Datenbank Einstellungen Export aller externen Datenbank Konfigurationen die unter

Kommunikation → Externe Datenbanken

eingetragen sind.

Mail Einstellungen Export aller Mailboxen. Exportiert die Mailboxen, welche unter

Applikation → Mailboxen

eingetragen sind.

Dashboard (Default Elemente) Export von Default Dashboard Elementen. Administratoren können Default Dashboard Elemente erstellen, welche mit diesem Export exportiert werden können. Dashboard Elemente von Benutzern werden hierbei nicht mit exportiert.

Prozessinstanzen Export von Prozessinstanzen (Laufzeitdaten) eines oder mehrerer Prozesse. Hierbei werden alle Prozessinstanz-Schritte, Formularinstanzen, Ad-hoc Schritte, etc. exportiert. Zugriffsrechte auf exportierte Objekte können optional mit exportiert werden. Die Menge der exportierten Instanzen kann durch eine Einschränkung der Prozessstart-Zeit begrenzt werden (z.B. nur Instanzen exportieren, die nach einem bestimmten Datum gestartet wurden). Die Prozesse, von denen Instanzen exportiert werden sollen, können in einem zweiten Schritt ausgewählt werden.

Stammdaten (z.B. Prozessdefinitionen, Benutzer, Rollen) werden beim Exportieren von Prozessinstanzen nicht mit exportiert. Prozessinstanzen werden am Zielsystem nur importiert, wenn die entsprechenden Objekte vorhanden sind. Möchten Sie also Prozessinstanzen von einem Server auf einen anderen bringen, stellen Sie zuvor sicher, dass die Stammdaten am Zielsystem verfügbar sind.

DMS Ordner Export eines DMS Ordners mit Inhalt (Laufzeitdaten). Dokumente, Formulare, Notizen, Weblinks und Unterordner werden mit exportiert (rekursiv). Verknüpfungen zu anderen DMS Objekten werden nicht exportiert. Zugriffsrechte auf die exportierten Daten können optional mit exportiert werden. Darin vorkommende Benutzer, Rollen und Organisationseinheiten werden allerdings nicht exportiert - diese müssen am Zielsystem verfügbar sein, damit die Zugriffsrechte auch dort angelegt werden.

Sie können den "Allgemein" Ordner oder beliebige Benutzer-Ordner exportieren (oder Unterordner dieser Ordner). Wenn Sie einen Benutzer-Ordner exportieren möchten, wählen Sie bitte zuerst den Benutzer aus und danach den entsprechenden Ordner.

Formulare Export von jeweils einer Formularinstanz. Es können alle in *EPworX* vorhandenen Formularinstanzen exportiert werden. Bevor eine Formularinstanz importiert werden kann, muss sichergestellt sein, dass die Formularklasse und alle referenzierten Objekte (z.B. Prozessdefinition, da die Formularinstanz ein Prozessformular ist) am Zielsystem vorhanden sind, d.h. die entsprechende Applikation mitsamt Formulartyp muss vorhanden sein.

Formulartyp: Es werden alle Formularinstanzen des ausgewählten Formulartyps exportiert.

Bedingung: Eine SQL-Bedingung, die spezifiziert welche Formularinstanzen exportiert werden sollen.

Lösche nicht enthaltene Instanzen beim Import: Wenn diese Option ausgewählt ist, verändert sich das Verhalten beim Importieren. Alle Formularinstanzen, des exportierten Formulartyps, die bereits im Zielsystem vorhanden sind, aber nicht im Export-File mitinbegriffen sind, werden beim Importieren gelöscht.

Exportiere referenzierte Instanzen von: Alle in einem Formularfeld referenzierten Formularinstanzen werden mitexportiert. Wenn kein Formulartyp ausgewählt wurde, werden referenzierte Formularinstanzen als Referenzen im Export-File vermerkt.

Testfälle Export von allen Testfällen der ausgewählten Applikation (siehe Abschnitt [Testfälle](#)). Alle referenzierten Objekte (Prozessdefinition, Tasks, Akteure, etc.) werden hier nicht mitexportiert und müssen daher am Zielsystem vorhanden sein!

Klassifikatoren Bei Auswahl der Gruppe *Klassifikatoren* können Sie einen oder mehrere Prozessklassifikatoren für den Export auswählen.

Der Server exportiert die XML Datei zuerst in sein temporäres Verzeichnis. Nachdem der Export abgeschlossen ist, werden Sie gefragt, ob Sie die Datei herunterladen möchten. Speichern Sie die Datei, um sie später am Zielsystem importieren zu können.

Import

Der Import passiert in drei Schritten:

1. Zuerst wird die XML Datei auf den Server hochgeladen.
2. Danach zeigt der Browser Informationen über den Inhalt der Datei an.
3. Dann wird die Datei importiert und Informationen über die importierten Objekte erscheinen im Browser.

Importiert wird dabei immer nach dem Prinzip, dass bei Auftreten eines Fehlers der Import abgebrochen und der Fehler angezeigt wird. Die bereits importierten Objekte werden nicht wieder gelöscht!

Beim Import von Export-Files aus früheren *EPworX* Versionen wird der Benutzer zuerst darauf hingewiesen, dass der Export mit einer früheren *EPworX* Version durchgeführt wurde und weiters aufgefordert, dass eine Applikation für die Standardobjekte ausgewählt werden muss, damit die Zuordnung von Applikationsobjekten (z.B. Prozesse) korrekt durchgeführt werden kann. Beim Import von z.B. Mail-Einstellungen wird diese Auswahl ignoriert, da sie in diesem Fall nicht relevant ist.

Hinweis: Falls ein Export von der default-Applikation gemacht, so werden beim Import alle Standardobjekte von *EPworX* nicht verändert! Diese Standardobjekte werden beim SETUP von *EPworX* angelegt und zu diesen gehören z.B. Rollen (all, sys, home), Formulartypen, Objektklassen, etc.

Import/Export Abhängigkeiten

Wenn Sie Daten von einem Server auf einen anderen Server bringen möchten, ist es nötig, die Dateien in der richtigen Reihenfolge zu importieren. Beim Export ist die Reihenfolge hingegen egal. Laufzeitdaten setzen voraus, dass benötigte Stammdaten bereits am Importsystem vorhanden sind, andernfalls können die Laufzeitdaten nicht importiert werden. Das Gleiche gilt auch für bestimmte Objekte wie z.B. Zugriffsrechte und gespeicherte Reports. Es wird empfohlen, Imports in folgender Reihenfolge durchzuführen:

1. Benutzer (ohne Rechte)

9. Administrationsfunktionen

2. Organisationseinheiten
3. Organisationshierarchien
4. Applikationen, Prozesse, Formulartypen, Wertelisten
5. Benutzer (inkl. Rechte), Berechtigungslisten
6. Prozessinstanzen, DMS Ordner, gespeicherte Reports, Timer, LDAP Einstellungen, Mail Einstellungen, Dashboard Elemente

9.2.2 Prozesse archivieren

Wenn Sie in der Liste der Administrationstasks die Funktion *Prozesse archivieren* aktivieren, dann wird eine neue HTML-Seite angezeigt, die es Ihnen ermöglicht, beendete Prozessinstanzen zu löschen (siehe Abbildung 9.6). Die Prozessinstanzen werden aus *EPworX* –Tabellen gelöscht. Ist eine Archivierungsklasse unter *Konfiguration/Klassen* eingetragen, werden die Prozesse vorher in ein Archivsystem übernommen. Mit Hilfe der Archivierungsklasse `com.groiss.wf.NoArchiver` kann festgelegt werden, dass keine Archivierung möglich ist!

Um beendete Prozesse zu löschen, führen Sie folgende Schritte durch:

1. Wählen Sie den zu löschenden Prozesstyp oder eine Applikation aus (alle Prozesse der Applikation).
2. Wählen Sie ein Datum, bis zu welchem sämtliche Instanzen des ausgewählten Typs gelöscht werden sollen. Prozesse, die am Tag des eingegebenen Tags beendet wurden, werden nicht gelöscht.
3. Wenn noch laufende Prozessinstanzen ebenfalls gelöscht werden sollen, markieren Sie die entsprechende Checkbox.
4. Archivieren Sie die Prozesse, indem Sie auf die Schaltfläche *Archivieren* klicken.

Prozessinstanzen archivieren

Applikation:	ITSM
Prozess:	
Beendet vor:	<input type="calendar"/>
Auch laufende Prozesse archivieren:	☒

ACHTUNG: Es ist keine Archivierungsschnittstelle installiert, die Prozessinstanzen werden nur gelöscht.

Archivieren

Abbildung 9.6: Prozessinstanzen archivieren

9.2.3 Applikation exportieren

Diese Funktion bietet die Möglichkeit eine *EPworX* Applikation in ein ZIP-Archiv zu verpacken (inkl. optionalem Applikations-Export) und mittels Hilfe der Funktion *Applikation installieren* wieder auf einem anderen *EPworX* System zu installieren (siehe Abschnitt [Applikation installieren/aktualisieren](#)). Diese Funktionalität soll sich vor allem an Kunden richten, die eine Applikationen über die *EPworX* Administrationsbenutzeroberfläche erstellen, ohne Java-Code zu schreiben. Durch das Ausführen der Schaltfläche *Export* werden folgende Schritte durchgeführt:

- Alle Dateien aus dem *classes* Ordner des angegebenen Applikationsverzeichnis werden zusammengesammelt und in eine JAR-Datei verpackt, die als Dateiname die ID der ausgewählten Applikation besitzt. Die Definition des Applikationsverzeichnisses ist im Abschnitt [Registerkarte: Allgemein](#) beschrieben. Welche Dateien sich im *classes* Ordner befinden können, ist im [Programmierhandbuch](#) im Kapitel [Organization of Files](#) beschrieben.
- Die zuvor erzeugte JAR-Datei wird innerhalb des ZIP-Archivs im Ordner *lib* platziert.
- Zusätzlich wird auf oberster Ordner-Ebene innerhalb des ZIP-Archivs automatisch die Datei *installation.properties* mit Parametern angelegt, die für einen Applikationsimport nötig sind (siehe dazu [Programmierhandbuch](#), Kapitel [The Configuration File](#)).
- Ist die Checkbox *XML-Export der Applikation erstellen* aktiviert, wird ein XML-Export der ausgewählten Applikation durchgeführt (siehe Abschnitt [Import/Export](#)) und in die JAR-Datei verpackt gemäß der beschriebenen Organisationsstruktur aus dem [Programmierhandbuch](#), Kapitel [Organization of Files](#).
- Die Erzeugung des ZIP-Archivs erfolgt standardmäßig in den Ordner *dist* innerhalb des definierten Applikationsverzeichnisses. Ist die Checkbox *Zip File herunterladen* aktiviert, so wird das ZIP-Archiv zum Download via den Browser angeboten.
- Durch Aktivieren der Checkbox *Sperren* wird angegeben, ob die zu exportierende Applikationen auf dem Zielsystem verändert werden kann. Falls aktiviert, bedeutet das, dass am Zielsystem keine neuen Applikationselemente erstellt bzw. bestehende bearbeitet/gelöscht werden können. Es gibt aber einige Ausnahmen:
 - Applikationsverzeichnis ändern
 - Zugriff und Berechtigungen erstellen/bearbeiten unter folgenden Bedingungen:
 - * Zugriff für Rollen, die zu einer Applikation mit einer höheren Startup Position gehören
 - * Berechtigungen von Rollen auf Objekte, die zu einer Applikation mit einer höheren Startup Position gehören
 - Objekte für die Entscheidungsunterstützung bei einer Prozessdefinition erstellen bzw. ändern

9. Administrationsfunktionen

- Zuordnung von Funktionen im Reiter *Funktionen* bei Prozessdefinition, wenn die Funktion zu einer Applikation mit einer höheren Startup Position gehört
- Zuordnungen in GUI-Konfigurationsobjekten erstellen/bearbeiten, wenn Rolle zu einer Applikation mit einer höheren Startup Position gehört
- Timer (de)aktivieren bzw. Periode setzen
- Zugangsdaten von Mailboxen ändern
- URLs in Webservice Client Objekten ändern
- Hinzufügen einer neuen Sprache im Ressourcen Editor

Die Information, ob eine Applikation gesperrt ist, wird im Manifest der erzeugten JAR-Datei mit folgendem Attribut vermerkt: *Locked: true*

Hinweis: Bei einem Applikations-Upgrade bzw. einem XML-Import ist die Sperre aufgehoben!

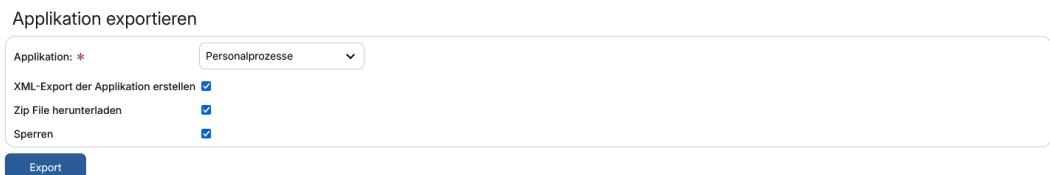


Abbildung 9.7: **Applikation exportieren**

9.2.4 Applikation installieren/aktualisieren

Mit Hilfe dieser Funktion können Sie einerseits eine Applikation installieren und andererseits diese auch aktualisieren. Für beide Arten benötigen Sie eine Archiv-Datei (.zip oder .jar), die idealerweise mit der Administrationsfunktion *Applikation exportieren* erstellt wurde (siehe Abschnitt [Applikation exportieren](#)).

Hinweis: Falls Sie die Archiv-Datei selbst erstellen wollen, beachten Sie, dass diese mit dem Zeichensatz UTF-8 erstellt wird, damit z.B. Umlaute in Datei-/Ordernamen korrekt entpackt werden können!

Installieren

Um eine Applikation zu installieren, stehen Ihnen zwei Optionen zur Verfügung:

- Hochladen: Wenn eine Archiv-Datei im Dateisystem vorhanden ist, können Sie die Applikation mithilfe dieser Option installieren. Beginnen Sie, indem Sie die Archiv-Datei aus dem Dateisystem auswählen. Die Standardoption ist Default Applikationsverzeichnis, das automatisch im Ordner local/appls ein Verzeichnis

mit der Applikations-ID erstellt. In diesem Verzeichnis erfolgt die Installation der Applikation. Alternativ können Sie, wenn gewünscht, eine benutzerdefinierte Zielposition festlegen. Geben Sie dazu im Feld Zielverzeichnis den Speicherort an, an dem die Applikation installiert werden soll.

Klicken Sie abschließend auf die Schaltfläche Installieren, um den Installationsprozess zu starten. Nach Abschluss der Installation werden Sie aufgefordert, den *EPworX* Server neu zu starten. Dies erfolgt über das Info-Icon in der Toolbar.

- Von URL laden: Hier geben Sie die URL zu einer Archiv-Datei auf einem Fremdsystem an, die geladen werden soll. Das weitere Vorgehen ist analog zur Option *Hochladen*.

Applikation installieren/upgraden

☒ Installieren ☐ Upgrade

Datei: *

☒ Hochladen ☐ Von URL laden

☒ Default Applikationsverzeichnis: *appls/application-id*

☐ Zielverzeichnis:

Abbildung 9.8: Applikation installieren/aktualisieren

Upgrade

Um eine Applikation zu aktualisieren, folgen Sie dem gleichen Prozess wie bei einer Neuinstallation (siehe oben beschriebene Anleitung). Nach dem Erstellen der Archiv-Datei können Sie, wie beim Installieren, wählen, ob die Datei vom Dateisystem oder einem externen System geladen werden soll. Eine zusätzliche Option ist *EPworX* hochladen. Mit dieser Option können Sie das *EPworX* System selbst aktualisieren, indem Sie eine SetupDatei (*ep-setup-11.*.jar*) verwenden. Beim Ausführen der Schaltfläche *Upgrade* wird die ausgewählte ArchivDatei in den Ordner *upgrades* des *EPworX* Servers gestellt und der Administrator aufgefordert, den *EPworX* Server neu zu starten. Durch diesen Neustart wird automatisch der Upgrade-Prozess von *EPworX* gestartet. Im Falle der Applikation werden alle Dateien im Applikationsverzeichnis durch diejenigen aus dem Archiv ersetzt. Nach Abschluss des Upgrades kann der *EPworX* Server wie gewohnt neu gestartet werden.

9.2.5 Dateiimport

Der Dateiimport ermöglicht den Import von Daten aus Textdateien ohne aufwendige Programmierung. Um dies durchzuführen, müssen folgende Einstellungen getätigt werden:

- *Import Definition*: Um die Funktion *Dateiimport* nutzen zu können, muss eine Import Definition (*import.xml*) erstellt werden, die im classes-Verzeichnis von *EPworX* oder in einem Applikationsverzeichnis gespeichert werden muss (siehe [Programmierhandbuch](#) - Kapitel [Organization of Files](#)). Nachstehend ein Beispiel für eine Import Definition:

```
<?xml version="1.0" encoding="iso-8859-15" standalone="yes"?>
<importDeclarations>

  <import name="resources">
    <targetClass>com.groiss.calendar.pers.Resource</targetClass>
    <columns>
      <column name="name"/>
      <column name="description"/>
    </columns>
    <keyField>name</keyField>
    <delimiter>;</delimiter>
  </import>

</importDeclarations>
```

Die einzelnen Schlüsselwörter der Import Definition sind im nachfolgenden Unterkapitel *Schlüsselwörter der Import Definition* beschrieben.

- *Datei*: Hier können Sie den Ort auswählen, von wo die Datei hochgeladen werden soll:
 - Hochladen: Wenn diese Funktion gewählt wurde, haben Sie die Möglichkeit einen Pfad anzugeben, welche Datei hochgeladen werden soll
 - Lokal: Diese Funktion erlaubt es, alle Dateien, die im *EPworX* -Verzeichnis liegen, hochzuladen. Die Wurzel ist dabei das *EPworX* -Verzeichnis selbst.
 - Klassenpfad: Bei Auswahl dieser Funktion können nur Dateien hochgeladen werden, die im Klassenpfad vorhanden sind
 - Gemäß Definition: Es wird die Datei hochgeladen, die in der *Import Definition* (Datei *import.xml*) mitsamt Pfadangabe eingetragen ist
- *Mode*: In dieser Dropdown-Liste können Sie zwischen 3 verschiedenen Upload-Modi auswählen:
 - Nur Parsen: Die Datei wird nur geparkt und keine Objekte im System angelegt
 - Ohne Datenbankoperation: Die Datei wird mit bestehenden Objekten im System verglichen
 - Import: Die Datei wird hochgeladen und Objekte werden im System (Datenbank) erzeugt
- *Laden*: Mit Hilfe dieser Funktion wird die ausgewählte Datei hochgeladen

Dateiimport

Import Definition: demo - persons

Mode: Nur Parsen

Datei: Hochladen

Durchsuchen...

Laden

Abbildung 9.9: **Dateiimport**

Schlüsselwörter der Import Definition

- `<import>`: Die Importbeschreibung, die dem Format `<import name="name">` entsprechen muss. Dabei können folgende Attribute definiert werden:
 - `ignoreHeader`: Wenn auf `true` gesetzt, wird 1. Zeile ignoriert.
 - `useOrgData`: Wenn auf `true` gesetzt, werden die `OrgData`-Methode von *EP-worX* verwendet anstatt der Store-Methoden.
- `<targetClass>`: Symbolisiert den zu importierenden Typ (=Zielklasse).
- `<targetCondition>`: Einschränkung der Elemente der `targetClass`. Nur diese Elemente werden mit den importierten verglichen, eventuell nicht mehr vorhandene gelöscht.
- `<keyField>`: Feld der Zielklasse, das den Schlüssel enthält.
- `<importHandler>`: Wenn kein `keyField` angegeben wurde, muss ein Import-Handler angegeben werden, der das Interface `com.groiss.fileimport.FileImportHandler` implementiert.
- `<constants>`: Beinhaltet eine Menge von Konstanten (`<constant name="name" value="wert"/>`). Diese werden zu der Menge der Werte jeder Zeile hinzugefügt.
- `<extensionClass>`: Name der Klasse für zusätzliche Daten von Stammdatenobjekten (Benutzer, OEs)
- `<delimiter>`: Feldtrenner, z.B. ;
- `<escapeMode>`: Gibt an, was zu tun ist, wenn ein zu escapendes Zeichen auftritt. Backslash oder Duplicate, z.B. Sonderzeichen ist quote: dÁrtangnon bzw. d"artangnon
- `<commentchar>`: Jene Zeilen werden ignoriert, die mit diesem Zeichen beginnen.
- `<charset>`: Alle gültigen Java Charsets (default: `StringUtil.getCharset()`);
- `<file>`: Pfad zur Datei.
- `<columns>`: Kann eine Menge von Spalten beinhalten, die zum Importieren sind:
`<column name="name" startcol="1" endcol="10" length="100" [format="dateformat"] [mapping="mappingName"] />`

- `<dateformats>`: Eine Menge von Datumsformate kann angegeben werden:

`<dateformat name="name" timezone="timezone" locale="locale"/>`

Beispiel:

```
<dateformats>
  <dateformat name="date">ddMMyyyy</dateformat>
</dateformats>
```

- `<mappings>`: Definition von Mappings möglich im Format `<mappings name="name">`
`<mapping*><keys><key>M</key></keys><value>1</value></mapping> </mappings>`.

Beispiel:

```
<mappings>
  <mapping name="lang">
    <keys><key>DE</key></keys>
    <value>de_DE</value>
  </mapping>
</mappings>
```

Dateiimport mit Timer

Dateien können mit einem Timer zeitgesteuert importiert werden. Dafür muss man einen Timer definieren. Der Klassenname des Timers ist *com.groiss.fileimport.ImportTimer* und als Parameter soll bei dem Timer eine Kombination von der Applikation-ID und der Import-ID eingetragen werden (`<application_id>:<import_id>`). Import-ID entspricht folgender Angabe im *import.xml*: `<import name="importid">...</import>`.

Zum Beispiel:

myappid:importid

Mehr Informationen über Timer finden Sie in Kapitel [Timer](#).

9.3 Cluster

9.3.1 Clustermonitor

Informationen über die Cluster-Architektur von *EPworX* sind im [Installationshandbuch](#) zu finden. Dort werden auch die einzelnen Attribute eines Clusters bzw. Knotens erklärt.

9.3.2 Server

Dieses Stammdatenobjekt wird aus Gründen der Abwärtskompatibilität zu früheren Versionen von *EPworX*, wo es für den Verteilungsmechanismus relevant war, weiterhin angeboten.

Ab Version 6.1, und der damit verbundenen Einführung der Cluster-Architektur, wird

die Verwaltung von mehreren Servern unnötig. Es ist jedoch weiterhin erforderlich, dass hier ein einziger Server eingetragen wird, nämlich der, der Ihre Installation ausmacht. Sollte dieser Server aus Versehen gelöscht werden, muss er wieder eingefügt werden, und zwar mit den Attributwerten, die im Konfigurationsfile von *EPworX* eingetragen sind.



Hinweis: Die Port-Einstellungen im Server-Objekt werden zum Beispiel für E-Mail Benachrichtigungen oder wenn ein *EPworX* Cluster mit Load-Balancing betrieben wird verwendet. Die Port-Einstellungen für den laufenden Betrieb müssen in der *Konfiguration* von *EPworX* getätigt werden.

9.4 DMS

9.4.1 Volltextsuche

Wird die Volltextsuche in der Systemkonfiguration (siehe [Installationshandbuch](#)) aktiviert, so gilt die Volltextindizierung nur für Objekte, die nach dem Aktivieren erstellt bzw. geändert werden. Um ältere Objekte in die Volltextsuche aufzunehmen, können Sie über diese Funktion alle Formulare und Dokumente, die nach dem unten angegebenen Datum erzeugt oder geändert wurden (bzw. für alle, wenn kein Datum angegeben wird), indizieren.

Weiters kann mit dieser Funktion der Informationsstand über geänderte Namensattribute von Formulartypen aktualisiert werden (Tabelle `avw_formfieldvals`). Die Diskrepanz entsteht durch die Änderung des Namensattributs bzw. Namens-Patterns eines Formulartyps über die entsprechende Administrationsmaske.

9.4.2 Schlagworte

Durch das Anklicken dieses Links wird im Arbeitsbereich eine HTML-Seite angezeigt, mit der Sie eine Liste von Schlagworten verwalten können. Die hier erfassten Schlagworte können im Dokumentenmanagement (siehe [Benutzerhandbuch](#)) einzelnen Dokumenten zugeordnet werden. Diese Schlagworte können in der Dokumentensuche verwendet werden.

9.4.3 Suche in Papierkörben

Mit Hilfe dieser Funktion können Sie als Administrator nach allen gelöschten DMS-Einträgen suchen, die in den Papierkörben der Benutzer liegen. Dabei wird mit Hilfe des Reportings eine Suchmaske angeboten, wo nach folgenden Kriterien gesucht werden kann:

- *Name:* Der Name des Dokuments, nach dem gesucht werden soll.
- *Gelöscht von* und *Gelöscht bis:* Hier können Sie den Zeitraum definieren, in dem die Dokumente gelöscht wurden.

- *Papierkorb von:* Mit Hilfe dieser Auswahl können Sie festlegen, in welchen Benutzerpapierkörben Sie nach Dokumente suchen wollen.
- *Ursprung:* Pfad-Definition, an welchem Speicherort das DMS–Objekt ursprünglich gespeichert war (DMS–Ordner, Prozessinstanz).
- *Formulartyp:* Hier können Sie festlegen, welche Typen die gesuchten Dokumente entsprechen sollen.

Mit der Schaltfläche/Funktion *Ausführen* starten Sie die Suche. Im anschließenden (Report-)Ergebnis werden folgende Spalten angezeigt:

- *Name:* Der Name des DMS–Objektes. Durch Aktivieren des Links wird der Inhalt des Dokuments angezeigt (Ordner sind nicht anklickbar).
- *Ursprung:* Diese Spalte beinhaltet die Information, an welchem Speicherort das DMS–Objekt ursprünglich gespeichert war (DMS–Ordner, Prozessinstanz).
- *Papierkorb von:* Der Name des Benutzers, dem der Papierkorb gehört.
- *Gelöscht am:* Das Datum, an dem das DMS–Objekt gelöscht und somit in den Papierkorb verschoben wurde.
- *Formulartyp:* Der Typ des DMS–Objektes.
- *Gelöscht mit:* Information, ob das Element direkt oder indirekt gelöscht wurde. Indirekt bedeutet, dass ein Ordner gelöscht wurde und das Element Teil dieses Ordners war. In diesem Fall kann das Element nicht alleine wiederhergestellt werden, sondern man muss entweder die Funktion *Ausschneiden* verwenden oder den entsprechenden Ordner wiederherstellen.

Zusätzlich zu den Toolbarfunktionen des Reportings werden noch die Funktionen *Wiederherstellen* und *Ausschneiden* angeboten.



Mit Hilfe der Funktion *Wiederherstellen* ist es möglich, die ausgewählten DMS-Objekte vom Papierkorb an den ursprünglichen Speicherort wiederherzustellen, d.h. die DMS-Objekte werden aus dem Papierkorb und wieder zum ursprünglichen Speicherort verschoben. Ist der ursprüngliche Speicherort nicht mehr vorhanden, wird dies mit einer entsprechenden Fehlermeldung ausgegeben. In diesem Fall können Sie nur mehr die Papierkorb-Funktion *Ausschneiden* verwenden.



Die Funktion *Ausschneiden* erlaubt das Wiederherstellen von ausgewählten DMS-Objekten aus dem Papierkorb an einen beliebigen Speicherort. Dazu wählen Sie zuerst die DMS-Objekte aus, aktivieren die Toolbarfunktion *Ausschneiden*, wechseln in einen beliebigen DMS-Ordner und führen dort die Toolbarfunktion *Einfügen* aus.

Für beide Funktionen muss der Administrator die entsprechenden Berechtigungen besitzen, d.h. hat der Administrator z.B. kein Recht um den Inhalt eines DMS-Ordners zu verändern, wird ihm dies beim Ausführen der Funktionen entsprechend mitgeteilt!

9.5 Reorganisation

In diesem Bereich der Admin-Tasks sind Administrationsfunktionen der Reorganisation zusammengefasst. Zu diesen gehören:

- Rollenzuordnungen ändern
- Prozessinstanzen analysieren
- OE-Historie

9.5.1 Rollenzuordnungen verändern

Wählen Sie im Feld *Von alter Organisationseinheit* jene Organisationseinheit aus, deren Rollen Sie an die Organisationseinheit im Feld *Zu neuer Organisationseinheit* verschieben oder kopieren möchten (siehe Abbildung 9.10). Nachdem Sie die Schaltfläche **Weiter** aktiviert haben, wird Ihnen eine weitere HTML-Seite angezeigt (siehe Abbildung 9.11), in der Sie nun explizit die Rollenzuordnungen von bestimmten Benutzern ändern können.

Rollenzuordnungen ändern

Von alter Organisationseinheit: * Human Resources ▼

Zu neuer Organisationseinheit: * Management ▼

Weiter

Abbildung 9.10: Rollenzuordnungen verändern (1)

Rollenzuordnungen ändern

Benutzer	Rolle	Rolle belassen	Auf die neue Org.-Einheit verlegen	Kopieren
Stefan Schmied	Abteilungsleitung	☒	☐	☐
Dr. Walter Heisenberg	Abteilungsleitung	☐	☒	☐
Stefan Schmied	Home	☐	☐	☒

Anzahl der Einträge: 3

Übernehmen Zurück

Abbildung 9.11: Rollenzuordnungen verändern (2)

Sie können nun festlegen, ob:

- die Rolle bei der alten Organisationseinheit bleibt oder
- die Rolle zur neuen Organisationseinheit verschoben werden soll oder
- die Rolle durch Kopieren auch der neuen Organisationseinheit zugeordnet werden soll.

Durch Aktivieren der Schaltfläche *Durchführen* wird die Rollenzuordnung verändert.

9.5.2 Prozessinstanzen analysieren

Hier wird Ihnen eine Liste aller Prozessinstanzen präsentiert, die derzeit keinen gültigen Akteur haben. Entweder wurde der Akteur oder die Organisationseinheit deaktiviert oder es gibt keine passende Rollenzuordnung usw.

Beispiel: Ein existierender Prozess wird weitergeleitet, der jedoch keinen gültigen Folgeakteur besitzt. Die Prozessinstanz taucht in der Tabelle mit dem Problemstatus *Weiterleiten nicht abgeschlossen* auf. Nun gibt es die Möglichkeit die Detailansicht der Instanz zu öffnen (Klick auf Prozessinstanz-ID) und den Prozess einem anderen Akteur zuzuweisen, der den Prozess korrekt abschließen kann. Dazu muss in der Prozesshistorie auf den letzten aktiven Akteur geklickt und über die Maske *Akteur ändern* die Prozessinstanz einem anderen Akteur zugewiesen werden. Die 3 Fragezeichen (???) in der Historie symbolisieren, dass die Instanz noch keinem Akteur zugewiesen werden konnte; sollte auch nicht über die Funktion *Prozessinstanzen analysieren* durchgeführt werden, da sonst eine weitere Prozessinstanz erzeugt wird.

9.5.3 OE–Historie

Hier wird Ihnen die Möglichkeit angeboten, die Änderungsgeschichte von OEs manuell festzuhalten. Dies kann im Rahmen von Umstrukturierungen der Organisationsstruktur erforderlich sein, wenn Sie wissen wollen, welche OE im Rahmen der Umstrukturierung aus welcher anderen OE hervorgegangen ist.

9.6 Kommunikation

In diesem Bereich der Admin-Tasks sind alle Administrationsfunktionen, die mit der Kommunikation in Zusammenhang stehen, zusammengefasst. Zu diesen gehören:

- Mail-Queue
- Mail Journal
- Autorisierer
- LDAP
- Externe Datenbanken
- Lokale Dienste

9.6.1 Mail-Queue

Die Mail-Queue ist eine Warteschlange, in der E-Mails vor ihrer Verteilung gesammelt und im Fehlerfall (z.B. Server nicht verfügbar) nicht verworfen werden bis ein erfolgreicher Sendeversuch durchgeführt werden konnte. Ein Eintrag in der Mail-Queue besteht aus folgenden Spalten:

- E-Mail: Das E-Mail Objekt, das es zu versenden gilt.

- **Erstellt am:** Das Erstellungsdatum des Mail-Queue Eintrags. Dieses dient als Information, wann der Eintrag erstellt wurde und wird für den *MailQueueTimer* herangezogen, der auf die maximal verstrichene Zeit für Sendeveruche prüft. Diese maximale Zeit kann in der Konfiguration unter *Kommunikation* beim Parameter *Max. Zeit für Mail-Queue Eintrag* gesetzt werden.
- **Letzter Versuch:** Falls es beim Abarbeiten des Eintrags einen Fehler gab, wird der Zeitpunkt des Auftretens gespeichert. Gibt es keinen Fehler, wird der Eintrag aus der Mail-Queue entfernt.
- **Status:** Hier werden Statusmeldungen zum Mail-Queue Eintrag angezeigt. Wird ein neuer Eintrag erstellt, ist der Statustext auf *Neu* gesetzt. Falls es beim Abarbeiten des Eintrags einen Fehler gab, wird die entsprechende Fehlermeldung als Status geschrieben. Gibt es keinen Fehler, wird der Eintrag aus der Mail-Queue entfernt.
- **Referenziertes Objekt:** Hier wird, falls vorhanden, das referenzierte Objekt angezeigt, welches ein beliebiges Persistent in *EPworX* sein kann. In der Regel wird hier ein Objekt vom Formulartyp *E-Mail* angezeigt, wobei der Link den Ordner öffnet, in dem das Objekt gespeichert ist.

Die Einträge werden in der Regel durch den *MailQueueTimer* abgehandelt, können aber auch manuell über die entsprechende Toolbarfunktion versendet werden. Folgende Toolbarfunktionen stehen Ihnen zur Verfügung:



- **E-Mail Details anzeigen:** Wenn ein Eintrag ausgewählt und diese Funktion aktiviert wurde, werden die Details des E-Mails angezeigt.



- **Sofort versenden:** Durch Auswahl eines Eintrags und aktivieren dieser Funktion wird versucht das E-Mail zu versenden. Schlägt das Senden fehl, wird der Benutzer entsprechend hingewiesen und der Status geschrieben.



- **Löschen:** Die ausgewählten Einträge werden mit dieser Funktion aus der Mail-Queue gelöscht.



- **Löschen inkl. referenziertes Objekt:** Die ausgewählten Einträge werden mit dieser Funktion aus der Mail-Queue gelöscht und das referenzierte Objekt wird auch gelöscht. Ist das referenzierte Objekt eine Prozessinstanz (bzw. Aktivitätsinstanz), so wird der Hauptprozess abgebrochen und archiviert!



- **Aktualisieren:** Mit dieser Funktion kann die Tabelle aktualisiert werden.

Ein Mail-Queue Eintrag kann über verschiedene Wege erstellt werden: Zum einen kann das Compose-Window beim E-Mail Tab einer Prozessinstanz aufgerufen werden. Weiters steht Ihnen noch die Möglichkeit zur Verfügung das API von *com.groiss.messaging.MessageTem* zu verwenden. Nähere Details dazu finden Sie im [Programmierhandbuch](#).

9.6.2 Mail Journal

Das Mail Journal ist ein Protokoll über alle ausgehenden E-Mails, bei denen in der Nachrichten-Vorlage die Checkbox *Nachricht in Journal protokollieren* aktiviert wurde (siehe Abschnitt [Registerkarte: Allgemein](#)). Das Mail Journal gliedert sich dabei in 2 Abschnitte: Die Suchfunktion und die Ergebnistabelle.

Mit Hilfe der Suchfunktion können Sie das Mail Journal einschränken. Dabei sind alle Felder mit logisch-UND verknüpft, ausgenommen die Felder *Prozess-Id* und *Dokumentenname*: Diese sind zueinander mit logisch-ODER verknüpft, d.h. man kann entweder nur nach Prozessen oder nur nach Dokumenten suchen. Wird keine Bedingung angegeben und die Schaltfläche *Suche starten* aktiviert, dann werden alle Einträge des Mail Journal in der Ergebnistabelle angezeigt.

Die Ergebnistabelle besteht aus folgenden Spalten:

- **Betreff:** Der Betreff der Nachricht, die versendet wurde.
- **Absender:** Der Versender der Nachricht; in der Regel der Standard *EPworX* Mail Sender oder ein alternativer Absender, wenn dieser in der Nachrichten-Vorlage definiert wurde.
- **Empfänger, CC und BCC:** Eine durch Beistrich getrennte Liste von Empfänger, an die die Nachricht versendet wurde.
- **Gesendet:** Der Zeitpunkt, wann die Nachricht versendet wurde.
- **Kontext:** In dieser Spalte wird das referenzierte Objekt angezeigt, für das eine Nachricht versendet wurde. Im Falle eines Prozesses ist es eine Verknüpfung auf die Prozessdetails, im Falle eines DMS-Dokuments wird der Name des DMS-Dokuments angezeigt mit einer Verknüpfung auf den Ordner, in dem das Dokument liegt.



Mit der Toolbarfunktion **Einträge aus Journal entfernen** können Sie eine Menge von Einträgen löschen, die bis zum angegebenen Datum im Mail Journal abgelegt wurden.

9.6.3 Autorisierer

Mehr Informationen über Autorisierer finden Sie im [Installationshandbuch](#) im Kapitel [OAuth 2.0 authentication](#).

9.6.4 LDAP

Hier können Sie Zugänge zu LDAP (Lightweight Directory Access Protocol) Servern definieren. Diese dienen dem Abgleich der *EPworX* Organisationsdaten mit bestehenden Verzeichnisdiensten. Neben einem vordefinierten Abbildungsmechanismus können auch eigene Synchronisationssemantiken für spezifische LDAP-Schemata realisiert werden. Näheres dazu findet man im [Programmierhandbuch](#).

9. Administrationsfunktionen

Allgemein Verbinden und auflisten Historie Zugriff

Name: *EP_User

Server: *10.205.224.90

Port: *389

Art der Kommunikation: *Unverschlüsselt ▾

Vertrauensniveau: Systemstandard ▾

Richtung: *

☐ nach LDAP

☒ nach EPworX

Suchpfad: CN=Users,DC=dgroiss,DC=local

Benutzer: DGROISS/Adm

Passwort:

Filter: *(objectClass=person)

LDAP Seitengröße: 0

Klassenname: com.groiss.ldap.BasicUserDirectorySyncer ✓

Parameter: sn=surname
givenName=firstName
cn=id
title=title

Beschreibung:

Mit Timer abrufen: ☒

Organisationseinheiten: ☐

OE-Bäume: ☐

Rechte: ☐

Rollen: ☐

Benutzer: ☐

Synchronisieren

Löschen

Speichern und Schließen

Speichern

Abbrechen

Abbildung 9.12: **Objekt-Detailansicht: LDAP**

Registerkarte: Allgemein

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Name:** Name des Servers
- **Server:** Hostname des LDAP-Servers.
- **Port:** Port des LDAP-Servers. Üblicherweise wird Port 389 (für unverschlüsselte Kommunikation und für STARTTLS) bzw. Port 636 (für verschlüsselte Kommunikation) verwendet.

- **Art der Kommunikation:** Eine der folgenden Arten kann hier definiert werden:
 - *unverschlüsselt*: LDAP Verbindung ohne Verschlüsselung (Standard).
 - *verschlüsselt*: Die LDAP Verbindung wird SSL-verschlüsselt.
 - *STARTTLS*: Dies ist eine Erweiterung zur unverschlüsselten Kommunikation. Initial wird eine unverschlüsselte Verbindung über den Port aufgebaut und nach erfolgreichem Verbindungsaufbau auf eine verschlüsselte Kommunikation umgeschaltet ohne einen separaten SSL-Port zu definieren.
- **Vertrauensniveau:** Um eine sichere Kommunikation zu gewährleisten, muss sich der LDAP Server gegenüber *EPworX* authentifizieren. Dies erfolgt durch Prüfung des Zertifikats des LDAP-Servers.

Wenn als Art der Kommunikation *verschlüsselt* oder *STARTTLS* ausgewählt wurde, muss man ein Vertrauensniveau auswählen:

- *Systemstandard*: Der Standard JAVA Trust-Mechanismus wird verwendet. Das Zertifikat des Directory Servers muss ein 'offizielles' sein. Dieser Modus wird für Produktivsysteme empfohlen.
 - *Blind*: Keine wirkliche Prüfung des Serverzertifikats, keine Prüfung des Hostnamens. Diese Einstellung sollte nur in Entwicklungsumgebungen oder für Testzwecke eingesetzt werden, nicht für Produktivsysteme!
 - *Zertifikat in Truststore*: Der *EPworX* Truststore wird verwendet. Wenn das Zertifikat des LDAP-Servers in den *EPworX* Keystore (Truststore) importiert wurde, dann ist es vertrauenswürdig, auch wenn es selbst signiert ist.
- **Richtung:** Richtung der Synchronisation: entweder
 - nach LDAP oder
 - nach **EPworX**
 - Suchpfad: LDAP Suchstring, z.B. *dc=groiss,dc=com*
 - Benutzer: LDAP-Benutzer, z.B. *cn=LDAPAdmin,dc=groiss,dc=com*
 - Passwort: Passwort des Benutzers
 - **Filter:** LDAP Filter: erlaubt die Einschränkung auf bestimmte LDAP-Einträge z.B.: (*objectClass=organizationalPerson*)
 - LDAP Seitengröße: Mit diesem Parameter wird das Ergebnis seitenweise ausgelesen. Wird benötigt, wenn der LDAP-Server die Anzahl der Resultate beschränkt.
 - Klassenname: durch Angabe einer Klasse, die *com.groiss.ldap.DirectorySyncer* implementiert, können eigene Abbildungen proprietärer LDAP-Schemata realisiert werden. *EPworX* bietet eine Basis-Implementierung um Benutzer aus einem Active Directory Server zu importieren.
Dazu ist der Klassenname *com.groiss.ldap.BasicUserDirectorySyncer* einzutragen. Nähere Infos dazu finden Sie gleich unten im Abschnitt *Active Directory Synchronisierung*.

- **Parameter:** Hier können Sie zusätzliche Parameter für eine *Active Directory Synchronisierung* angeben. Nähere Informationen dazu finden Sie im gleichnamigen folgenden Abschnitt.
- **Beschreibung:** Freier Text.
- **Durch Timer ausführen:** Wenn diese Checkbox aktiviert ist, führt der LDAPDir-SyncTask-Timer die Aktion automatisch durch.
- **Organisationseinheiten:** Ist diese Checkbox angekreuzt, wird diese Objektklasse bei der Synchronisation berücksichtigt.
- **OE-Bäume:** Ist diese Checkbox angekreuzt, wird diese Objektklasse bei der Synchronisation berücksichtigt.
- **Rechte:** Ist diese Checkbox angekreuzt, wird diese Objektklasse bei der Synchronisation berücksichtigt.
- **Rollen:** Ist diese Checkbox angekreuzt, wird diese Objektklasse bei der Synchronisation berücksichtigt.
- **Benutzer:** Ist diese Checkbox angekreuzt, wird diese Objektklasse bei der Synchronisation berücksichtigt.

Wenn Sie die Schaltfläche **Synchronisieren** aktivieren, wird die Synchronisation in die angegebene Richtung durchgeführt.

Active Directory Synchronisierung

Wie bereits oben beim Feld *Klassenname* erwähnt, bietet *EPworX* eine Implementierung für einen Abgleich mit einem Active Directory Server an. Dabei können Benutzer aus dem Active Directory nach *EPworX* importiert werden, aber nicht exportiert! Zu diesem Zweck muss die Klasse *com.groiss.ldap.BasicUserDirectorySyncer* im Feld *Klassenname* eingetragen werden. Im Feld *Parameter* müssen dazu Zeilen von Mappings in der Form *LDAP-Attributname=Java-Feldname* eingetragen, z.B.

```
sn=surname
givenName=firstName
samAccountName=id
title=title
description=description
mail=email
telephoneNumber=telNr
userAccountControl=active
```

Bevor ein Synchronisierungsvorgang durchgeführt werden kann (durch einen Timer oder manuell) muss noch ein inaktiver Prototype-Benutzer in *EPworX* angelegt werden. Dieser Benutzer muss als ID den Namen des *EPworX* LDAP-Objekts gefolgt vom Zusatz *_prototype* haben, z.B. *EP_User_prototype*. Bei diesem Prototype-Benutzer können Sie Rechte, Rollenzuordnungen, Benutzer-Einstellungen (User properties) und die

zu übernehmende Sprache definieren, die für neu importierte Benutzer verwendet werden. Nähere Informationen zur Definition eines Benutzers finden Sie im Kapitel [Benutzer](#).

Ausgehend vom angegebenen Suchpfad werden alle gefilterten LDAP-Objekte wie folgt synchronisiert:

- Wenn der Benutzer mit der ID in *EPworX* nicht vorhanden und im Active Directory deaktiviert ist, wird dieser Benutzer nicht importiert.
- Wenn der Benutzer mit der ID in *EPworX* nicht existiert, dieser aber im Active Directory aktiv ist, wird ein neuer Benutzer in *EPworX* angelegt. Die Werte aus dem Active Directory werden gemäß dem Field-Mapping übernommen und mit den Rechten, Rollenzuordnungen, Benutzer-Einstellungen und der Sprache des Prototyp-Benutzers angereichert.
- Wenn ein Benutzer in *EPworX* existiert, der ursprünglich nicht aus dem Active Directory übernommen wurde, dann wird dieser Benutzer ignoriert (nicht verändert).
- Wenn ein Benutzer in *EPworX* existiert, der ursprünglich aus dem Active Directory übernommen wurde, werden die Werte aus den Feldern des Active Directory Eintrages gemäß den Feld-Mappings herangezogen. Wenn es dabei Diskrepanzen gibt, wird der Benutzereintrag in *EPworX* aktualisiert.
- Wenn es einen aktiven Benutzer in *EPworX* gibt, der ursprünglich aus dem Active Directory übernommen wurde, dieser Benutzer aber jetzt nicht mehr (im Active Directory) vorhanden ist, wird der Benutzer in *EPworX* deaktiviert.

Das Feld *active* des importierten Benutzers wird auf der Basis des *userAccountControl* Attributs aus dem Active Directory Eintrag ermittelt. Wenn der LDAP Server kein Active Directory ist, dann ist das *userAccountControl* Attribut normalerweise nicht verfügbar. In diesem Fall wird der importierte Benutzer 'aktiv' gesetzt (weil ja ein korrespondierender LDAP Eintrag existiert).

Für solche LDAP Server (kein AD) kann jedes boolsche LDAP Attribut verwendet im Mapping angegeben werden, um das *active* Feld des importierten Objects zu setzen. Wenn kein Mapping für das *active* Feld angegeben wird, erfolgt keine Aktivierung des Benutzers beim Import.

Registerkarte: Verbinden und auflisten

Wenn Sie auf diese Registerkarte wechseln, wird der Inhalt des LDAP-Servers unter Berücksichtigung von Suchpfad und Filter aufgelistet.

9.6.5 Externe Datenbanken

Hier können Sie die Verbindungen zu externen Datenbanken definieren. Jede Konfiguration erhält eine eindeutige ID und muss mit den richtigen Einstellungen für das JDBC-Framework konfiguriert werden. Die Treiberklasse, die JDBC-URL, der Benutzername und das Kennwort für die entfernte Datenbank müssen angegeben werden.

Stellen Sie sicher, dass sich die entsprechende JAR-Datei für den Datenbank-Treiber im Klassenpfad befindet. Richtig definierte Datenverbindungen können direkt aus *EPworX* im Query-Tool benutzt werden (siehe [Query-Tool](#)). Definierte Verbindungen können mittels Administrationsfunktion *Export im XML* exportiert werden (näheres dazu ist in [Import/Export](#) beschrieben). Weitere Informationen zum Zugriff zu externen Datenbanken über das API finden Sie im [Programmierhandbuch](#).

Registerkarte: Allgemein

Informationen der Registerkarte *Allgemein* (Mussfelder sind fett dargestellt):

- **Id:** Eindeutige ID der Datenbankverbindung.
- **JDBC Treiber Klasse:** Klassenname des JDBC Treibers. Ob der Name richtig ist und ob sich der Treiber in Klassenpfad befindet können Sie mit dem Icon neben dem Feld überprüfen. Mittels *JDBC Hilfeseite* (das Fragezeichen-Icon beim nächsten Feld) können Sie den Klassennamen sowie den JDBC URL für alle in *EPworX* direkt unterstützten Datenbanksysteme importieren. Es ist aber auch darüber hinausgehend möglich andere Datenbanksysteme einzubinden. Die Verbindungs-Informationen sind dann der entsprechenden Dokumentation des Treibers zu entnehmen.

Weitere Informationen finden Sie im [Installation and Configuration](#) Handbuch, im Kapitel [Extract and Install](#).
- **JDBC URL:** Eine zu Ihrem DBMS passende Datenbankverbindungs-URL. Man kann das Feld auch über *JDBC Hilfe Seite* setzen. Importierte Wert muss man allerdings anpassen und näheres dazu findet man ebenfalls [Installation and Configuration](#) Handbuch, im Kapitel [Extract and Install](#).
- **Datenbank Userid:** Benutzer Id in der externen Datenbank.
- **Datenbank Passwort:** Benutzer Kennwort in der externen Datenbank.
- **Query Timeout:** Gibt an, wie lange eine Operation in Sekunden dauern kann, bis das Zeitlimit für den Server abgelaufen ist.
- **Beschreibung:** Freier Text bzw. die Beschreibung der Datenbankverbindung.
- **Verbindung überprüfen:** Aktivieren Sie diese Schaltfläche um zu sehen, ob die eingetragene Konfiguration richtig ist und ob die Verbindung zur externen Datenbank funktioniert. Bitte beachten Sie, dass der Eintrag zuerst gespeichert sein muss!

9.6.6 Web Dienste

Die Funktion *Web Dienste* an dieser Stelle umfasst die Funktion *Lokale Dienste*, die eine Tabelle aller gefundenen Web Dienste in der *EPworX* -Umgebung anzeigt. Hier können Sie neue Web Dienste hinzufügen, bestehende löschen oder aktivieren/deaktivieren (Deployment). Die Erstellung der Objekte *Web Dienste Client/Server* ist pro Applikation

möglich und wird bei jeder Applikation mit entsprechenden Funktionen angeboten (siehe Kapitel [Web-Dienste](#)).

Nähere Informationen über Web Dienste finden Sie im [Programmierhandbuch](#).

9.7 Entscheidungsunterstützung

In diesem Abschnitt beinhaltet Funktionen, die dem Administrator dabei helfen sollen, den Teil der Entscheidungsunterstützung im Blick zu behalten, die bei einer Prozessdefinition definiert werden kann.

9.7.1 Klassifikator-Bewertungen

Informationen zur Beurteilung von Klassifikatoren können hier eingesehen werden, wie etwa Sensitivität (Richtig-positiv-Rate) oder Genauigkeit. Auch können Klassifikatoren gebaut werden. Wird eine Prozessdefinition selektiert und anschließend die Ausführen-Aktion gewählt, so werden alle Klassifikatoren, die zu dieser Prozessdefinition gehören, ausgeführt (gebaut).

Klassifikator-Bewertungen

Name	Aktiv	Ausgabe	Klassifikator-Klasse	Zuletzt gebaut	Zuletzt ausgeführt	Bauzeit	Anzahl Erstellungen	Anzahl Ausführungen	Prozent korrekt	
Urlaub (!)										
	●	Urlaub Genehmigt	com.groiss.ml.classif				0	0		
Anfordern	●	Urlaub Kommentar	com.groiss.ml.classif	09-07-2026 12:59			5	4	0.00	
vacation	●	Urlaub MitarbeiterIn	com.groiss.ml.classif				0	0		
approved	●	Urlaub Genehmigt	com.groiss.ml.classif	09-07-2026 12:59			32	0	83.33	
substitution	●	Urlaub Vertretung	com.groiss.ml.classif				30	8	0.00	

Abbildung 9.13: Klassifikator-Bewertungen

Toolbar-Funktionen

Es werden die gleichen Funktionen angeboten wie in der Prozessdefinition-Administration (siehe Kapitel [Klassifikatorzuweisungen einer Prozessdefinition verwalten](#)). Zusätzlich stehen noch 2 weitere Funktionen zu Verfügung:



- **Anzeigen**

Diese Aktion ist nur für Klassifikatoren mit Entscheidungsbäumen verfügbar. Hier wird der gesamte Baum dargestellt.



- **Bewertung**

Durch diese Aktion werden ein paar Performance-Eigenschaften des Klassifikators dargestellt. Hierzu zählen einige Metriken bezüglich jeden Labels (Sensitivität, Genauigkeit, ...), einer Wahrheitsmatrix (stellt Klassifiziertes Tatsächlichem aus der Testmenge gegenüber) und der Anteil der korrekt klassifizierten Instanzen.



- **Bewertungen**

Zeigt alle Bewertungen an, die für den ausgewählten Klassifikator vorgenommen wurden.

9.7.2 PMML-Administration

Mithilfe dieser Funktion können PMML-Modelle importiert und exportiert werden. In diesen PMML-Modellen können Vorhersagemodelle mit zusätzlichen Informationen beschrieben werden. Tatsächlich werden diese dann im XML-Format gespeichert, wobei einem vorgegebenen Schema gefolgt werden muss.

Name	Feld	Klassifikator
Urlaub (1)		
Anfordern	Urlaub Kommentar	com.grotes.ml.classifier.impl.DecisionTree
vacation	Urlaub MitarbeiterIn	com.grotes.ml.classifier.tree.manual.ManualTreeClassifier
approved	Urlaub Genehmigt	com.grotes.ml.classifier.impl.DecisionTree
substitution	Urlaub Vertretung	com.grotes.ml.classifier.impl.DecisionTree

Abbildung 9.14: PMML-Administration

Toolbar-Funktionen

Es gibt folgende Toolbar-Funktionen, die je nach ausgewähltem Klassifikator aktiviert oder deaktiviert werden können:

- **ARFF herunterladen**

Alle Instanzen, die beim Trainieren des ausgewählten Klassifikator zum Einsatz kommen (sowohl Trainings- als auch Testmenge), können im Attribute-Relation File Format (ARFF) heruntergeladen werden. Dies ist besonders nützlich, wenn das Modell mithilfe eines externen Werkzeuges erstellt werden soll.

- **PMML hochladen**

Diese Aktion ist nur verfügbar, wenn der ausgewählte Klassifikator in der Lage ist, Modelle, die in PMML vorliegen, zu importieren. Um dies klarzustellen, muss das Interface PMMLImportable implementiert werden. Die erlaubt es dem Administrator beispielsweise, Modelle hochzuladen, die von einem anderen Programm erstellt wurden.

- **PMML herunterladen**

Trainierte Modelle können mit dieser Aktion im PMML-Format heruntergeladen werden. Exportieren und Herunterladen des Modells ist nur für Klassifikatoren möglich, die das Interface PMMLExportable implementieren.

10. Konfiguration

Dieses Kapitel beschreibt die Konfiguration des *EPworX* –Servers. Für nähere Informationen über die Menüpunkte

- Lizenz
- HTTP–Server
- Datenbank
- Verzeichnisse
- Logging
- Klassen
- Lokalisierung
- Kommunikation
- Cluster
- Workflow
- DMS
- Suche
- Tuning
- Sicherheit
- Passwort Policy
- Kalender
- Prozess-Cockpit
- Entscheidungsunterstützung
- Weitere Parameter
- Administratorpasswort ändern
- Style-Konfigurator

sehen Sie bitte im [Installationshandbuch](#), Kapitel [Configuration](#) nach.

11. Dashboard

In der Systemadministration von *EPworX* besteht die Möglichkeit sich ein Dashboard anzulegen, das auf die Bedürfnisse des Systemadministrators abgestimmt werden kann. Nach dem Anklicken des Links *Dashboard* unter *Admin-Tasks* wird initial eine leere Seite angezeigt mit folgenden Toolbarfunktionen:

11.1 Neu

Durch das Aktivieren der Schaltfläche *Neu* wird ein neuer Dialog angezeigt, mit dem Sie neue Fenster zum Dashboard hinzufügen können. Dazu stehen Ihnen mehrere Möglichkeiten zur Verfügung:

- URL: Geben Sie hier die URL einer HTML-Seite ein, die Sie zukünftig in einem Fenster auf Ihrem Dashboard sehen möchten, und bestätigen Sie Ihre Eingabe mit der *Return*-Taste.
- Reports: Durch das Anklicken dieses Links wird eine Tabelle aller gespeicherten Anfragen angezeigt. Wählen Sie einen Eintrag aus und aktivieren Sie die Schaltfläche *Ok*, um einen gespeicherten Report dem Dashboard hinzuzufügen.
- Kalender: Durch das Anklicken dieses Links wird ein Kalender in einem Fenster auf Ihrem Dashboard angezeigt.
- Arbeitskorbübersicht: Durch das Anklicken dieses Links wird eine Übersicht über die Anzahl der Einträge im Arbeitskorb in einem Fenster auf Ihrem Dashboard angezeigt.
- Termine: Durch das Anklicken dieses Links werden die Termine des heutigen Tages in einem Fenster auf Ihrem Dashboard angezeigt.
- Neuigkeiten: Durch das Anklicken dieses Links werden die Neuigkeiten in einem Fenster auf Ihrem Dashboard angezeigt. Dafür muss im DMS ein Ordner mit dem Namen *News* unter *Allgemein* angelegt werden, wo dann Nachrichten hinterlegt werden können (z.B. eine Notiz).



Hinweis: Jedes Fenster am Dashboard kann (wie unter Windows üblich) an eine beliebige Stelle innerhalb des Dashboards verschoben bzw. in seiner Größe verändert werden.

11.2 Öffnen

Durch das Aktivieren der Schaltfläche *Öffnen* kann ein bestehendes Dashboard-Profil geladen werden. Dies muss zuvor mittels der Funktion *Speichern* persistent gemacht worden sein. Es wird dabei zwischen persönlichen Dashboards (*Meine Dashboards*) und anderen Dashboards unterschieden. Persönliche Dashboards sind jene, die der aktuelle Benutzer abgespeichert hat. Andere Dashboards sind jene, die von anderen Benutzer über den Reiter *Zugriff* des zu speichernden Dashboards freigegeben werden (mittels *Freigabe*-Recht).

11.3 Speichern

Durch das Aktivieren der Schaltfläche *Speichern* werden die aktuellen Einstellungen für das geöffnete Dashboard gespeichert. Dabei wird ein neuer Dialog mit folgenden Attributen geöffnet:

- **Name:** Der eindeutige Name des Dashboards muss hier eingegeben werden.
- *Speichern unter:* Mit dieser Funktion können Sie das aktuelle Dashboard unter einem neuen Namen speichern. Ändert ein Benutzer durch das Aktivieren der Schaltfläche *Neu* und *Speichern* das Dashboard, wird es zu seinem persönlichen Dashboard und das Standard-Dashboard bleibt unverändert. Der Benutzer kann aber auch ein bereits bestehendes Dashboard-Profil mittels der Funktion *Öffnen* laden. Die Identifikation erfolgt über den URL Parameter *id*.
- *Beschreibung:* Frei wählbarer Text
- *Default:* Hier kann aus 2 Einträgen ausgewählt werden:
 - Dashboard <dashboard-id> für <user>: Dashboard-Einstellungen werden für den aktuellen Benutzer als Default gespeichert. Dieses Dashboard wird für den Benutzer standardmäßig angezeigt, falls dieser noch kein anderes besitzt.
 - Dashboard <dashboard-id> für alle Benutzer: Dashboard-Einstellungen werden für alle Benutzer als Default gespeichert. Wenn ein Benutzer kein eigenes Dashboard besitzt, wird dieses standardmäßig angezeigt.
- *Spalten:* Hier können Sie definieren, wie viele Fenster im Dashboard nebeneinander dargestellt werden können. Die Standardeinstellung dabei sind 2 Spalten.
- *Erzeugt von:* Dieses Feld kann nicht verändert werden und zeigt den Ersteller des Dashboards an.
- *Dashboard-Id:* Dieses Feld kann ebenso nicht verändert werden und zeigt an, ob das Dashboard im Arbeitskorb (user) oder in der Administration (admin) gespeichert wird/wurde. Über diese Maske können nur Dashboards mit Id *admin* gespeichert werden.

Weiters besteht die Möglichkeit über den Reiter *Zugriff* das *Freigabe*-Recht für andere Benutzer zu vergeben. Diese können dann bei positiver Vergabe des Rechts das aktuelle Dashboard öffnen (siehe Funktion *Öffnen*).

11.3.1 Löschen

Mit Hilfe dieser Schaltfläche werden die aktuellen Dashboard-Einstellungen bzw. das Dashboard-Profil (aus der Datenbank) gelöscht. Nach dem Löschen eines Dashboards, wird das Standard-Dashboard angezeigt, wenn keines vorhanden ist dann eine leere Seite.

12. Administration via Kommandozeile

Dieses Kapitel beschreibt die Administration von *EPworX* über die Kommandozeile. Dadurch ergeben sich folgende Vorteile:

- Administrationstätigkeiten können in einem Skript zusammengefasst und auf verschiedenen Servern ausgeführt werden
- Synchronisierung zwischen Entwicklungs- und Produktionsumgebung
- Übermitteln eines Skripts an den Systemoperator
- Aktionen können dokumentiert werden

12.1 Architektur und Anwendung

Das Kommandozeilen-Interface hat eine Client- und eine Serverkomponente. Die Serverkomponente ist im *EPworX* Server integriert. Die Clientkomponente ist in einer separaten Datei *ep-adminshell-11.0.xxx.jar* integriert, die im *bin*-Verzeichnis von *EPworX* liegt. Der Client kann sich über HTTP oder HTTPS mit dem Server verbinden. Die Konfiguration dazu ist auf der Konfigurationsmaske *Kommunikation* zu finden. Aktiviert wird das Kommandozeilen-Interface mit dem versteckten Parameter *ep.adminshell.enable*. Nähere Details dazu finden Sie im [Installations- und Konfigurationshandbuch](#).

Hinweis: Der ausführende Benutzer muss das Recht *Ausführen* auf alle Objekte besitzen! Weiters benötigt er die entsprechenden Rechte, um Server Kommandos korrekt auszuführen.

Um die Kommandozeile zu aktivieren, müssen Sie folgenden Befehl absetzen:

```
java -jar ep-adminshell-11.0.xxx.jar url user [password]  
[-log logfile | -append logfile] [-passwdfile file] [-execute scriptfile]
```

Parameter:

- **url:** Die URL zum Server, z.B. *http://localhost:8083/wf/*. Wenn kein Context-Root angegeben wurde, wird automatisch *wf* als Standard verwendet.

- **user:** Der Benutzername des ausführenden Benutzers
- **password:** Das Passwort des Benutzers (falls vorhanden). Wenn Sie kein Passwort angeben, müssen Sie die Option `-passwdfile` verwenden oder Sie werden beim Login zur Eingabe des Passworts aufgefordert.

Optionen:

- **-log logfile:** Das Logfile, wo alle Interaktionen der Kommandozeile hingelogg werden.
- **-append logfile:** Analog zu `-log`, jedoch werden die Interaktionen dem bestehenden Logfile angehängt.
- **-passwdfile file:** Diese Datei beinhaltet das Passwort (plain) für den ausführenden Benutzer ohne führende bzw. nachstehende Zeichen.
- **-execute scriptfile:** Führt das Skript in der angegebenen Datei aus.

Hint: Wenn Sie die `adminshell` unter Linux aufrufen, dann können Sie anstelle der vollständigen Versionsnummer der `jar`-Datei auch einen Wildcard verwenden, also z.B.:

```
java -jar adminshell*.jar ...
```

12.2 Kommandos

Bei den Kommandos wird in 2 Gruppen unterschieden:

1. **Client Kommandos** werden auf dem Client ausgeführt und definieren das Verhalten des Skript-Clients.
2. **Server Kommandos** werden auf dem Server ausgeführt und beinhalten Funktionen zur Administration von *EPworX*.

12.2.1 Client Kommandos

Folgende Client Kommandos sind verfügbar:

- **exit:** Beendet den Client.
- **help** oder **?**: Gibt eine Übersicht aller Kommandos aus.
- **log <file>:** Schreibt Kommandos in das angegebene Logfile.
- **log off:** Kommandos werden ab diesem Zeitpunkt nicht mehr im Logfile mitprotokolliert.
- **append <file>:** Schreibt Kommandos in das angegebene Logfile. Wenn die Datei bereits existiert, werden die Kommandos der Datei hinten angehängt.
- **execute <file>:** Führt das angegebene Skript aus der Datei aus.

Alle Kommandos, die nicht in der oben genannten Liste vorhanden sind, werden als Serverkommandos interpretiert.

12.2.2 Server Kommandos

Diese Kommandos werden als Groovy Expressions interpretiert. Groovy ist eine Skriptsprache basierend auf Java. Kommentare haben dieselbe Syntax wie in Java (Inline- und Blockkommentare). Ein Kommando wird nur durch eine Zeile mit einem Punkt-Zeichen abgeschlossen und auf Loglevel 1 oder höher im Serverlog mitprotokolliert.

Die folgenden Variablen sind im initialen Kontext enthalten (varname und Instanz von):

- **admin**: com.groiss.server.Admin
- **store**: com.groiss.store.Store
- **engine**: com.groiss.wf.WfEngine
- **dms**: com.groiss.dms.DMS
- **orgdata**: com.groiss.org.OrgData
- **config**: com.groiss.component.Configuration (die Systemkonfiguration)
- **user**: com.groiss.org.User (der aktuelle Benutzer)
- **session**: jakarta.servlet.http.HttpSession (die HttpSession)

Nähere Informationen über die einzelnen Objektmethoden sind im API vorhanden. Jedes Kommando wird in einer eigenen Transaktion ausgeführt. Nachdem ein Kommando ausgeführt wurde, wird entweder ein Commit oder - falls ein Fehler auftritt - ein Rollback durchgeführt.

Wenn Sie (eigene) Variablen im Skript verwenden wollen, können Sie diese mit folgendem Kommando definieren:

```
set(varname,value);
```

Um den Wert wieder auszulesen, müssen Sie folgendes Kommando absetzen:

```
get(varname);
```

Selbst deklarierte Variablen haben den Vorteil, dass sie über eine Transaktion hinaus verfügbar sind, da sie in die Session geschrieben werden.

12.3 Beispiele

12.3.1 Einen Konfigurationsparameter setzen

```
config.setProperty("database.connections",5);  
config.store();  
.
```

Alternative Version mit eigenen Variablen:

```
set("connections",5);
config.setProperty("database.connections",get("connections"));
config.store();
.
```

12.3.2 Serverneustart

```
admin.restartServer(); //restarts the server - no login necessary for current user
.
```

12.3.3 Rolle einem Benutzer hinzufügen bzw. entziehen

```
u = orgdata.getByld(com.groiss.org.User.class,'my_user'); //replace by existing user
role = orgdata.getByld(com.groiss.org.Role.class,'sys'); //get SYS role
checkuserrole = store.get(com.groiss.org.UserRole.class,"role = ? AND userid = ?",
    role.getOid(), u.getOid()); //with prepared statements - new Object[] {args}
//If User has no sys-role, add it
if(checkuserrole == null) {
    userrole = orgdata.createUserRole();
    userrole.setRole(role);
    userrole.setUser(u);
    userrole.setActive(true);
    orgdata.insert(userrole);
}
//If User has sys-role, remove it
else {
    orgdata.delete(checkuserrole);
}
.
```

12.3.4 Zeitintervall eines Timers setzen

```
t = orgdata.getByld(com.groiss.timer.TimerEntry.class,'Suspension');
t.setPattern("360");
store.update(t);
.
```

12.3.5 Arbeitskorb Handling

Dieses Beispiel durchsucht den Arbeitskorb der Applikation *default* und beendet bereits abgelaufene Tasks, sofern diese beendet werden können:

```
appl = orgdata.getByld(com.groiss.org.Application.class, "default");
worklist = engine.getWorklist(appl,true);
for(com.groiss.wf.ActivityInstance ai:worklist) {
    duedate = ai.getDuedate();
    //if ai's duedate is expired, finish task
}
```

```
if(duedate != null && duedate.getTime() < new java.util.Date().getTime()) {  
    try {  
        engine.finish(ai);  
    }  
    catch(ex) { /*Do nothing, but continue with finishing other ai's*/};  
}  
}  
.
```

12.3.6 Session Handling

Dieses Beispiel prüft die derzeitige Session und invalidiert sie, falls der letzte Zugriff (lastAccessed) nicht innerhalb der Toleranzzeit liegt. Die Informationen der Session werden ins Serverlog von *EPworX* ab Loglevel DEBUG geschrieben:

```
attrbnames = session.getAttributeNames();  
invalidate = false;  
log = " \n";  
log = log + "Session-Parameter:\n";  
for(String attrname:attrbnames) {  
    attrvalue = session.getAttribute(attrname);  
    if(attrname.equalsIgnoreCase("lastAccessed")) {  
        if(attrvalue instanceof java.util.Date) {  
            onehour = 60*60*1000; //tolerance time  
            //invalid, if not in tolerance time  
            if((attrvalue.getTime()+onehour) >= new java.util.Date().getTime() ||  
                (attrvalue.getTime()-onehour) <= new java.util.Date().getTime()) {  
                invalidate = true;  
            }  
        }  
    }  
    log = log + "Attribute-Name: " + attrname + "/Attribute-Value: " + attrvalue + "\n";  
}  
log = log + " ";  
com.groiss.util.Settings.log(log,2); //write all session parameter to Server-Log on Level 2  
if(invalidate == true) {  
    session.invalidate();  
}  
.
```

13. Eingeschränkte Administration

EPworX bietet neben der vollständigen Administration auch eine eingeschränkte Version an, mit der Benutzer und Organisationseinheiten administriert werden können. Zu diesem Zwecke gibt es die Rolle *Benutzer-Administrator*, die aus folgenden Rechten besteht:

- **Administration**
- **Objekt ausführen** auf Organisationseinheiten (hierarchisch in der Rollen-OE)
- **Objekt ausführen** auf die Rolle *Home*
- **Objekte erzeugen** auf Organisationseinheiten und Benutzer (hierarchisch in der Rollen-OE)
- **Objekte bearbeiten** auf Organisationseinheiten und Benutzer (hierarchisch in der Rollen-OE)
- **Objekte bearbeiten** auf OE-Bäume
- **Objekte löschen** auf Organisationseinheiten und Benutzer (hierarchisch in der Rollen-OE)

Mit dieser Rechtekonstellation ist es möglich Benutzer und Organisationseinheiten in jenen Organisationseinheiten zu bearbeiten, in der die home-Rolle des aktuell eingeloggtten Benutzers vergeben wurde bzw. in allen im OE-Baum hierarchisch darunterliegenden Organisationseinheiten. Diese Organisationseinheiten müssen jedoch Teil vom OE-Baum *default* sein!

Hat ein Benutzer die home- und useradmin-Rolle in der gleichen Organisationseinheit zugeordnet, so wird die eingeschränkte Administration nach dem Einloggen angezeigt, d.h. die GUI-Konfiguration mit der ID *useradmin* wird geladen. Voraussetzung dafür ist jedoch, dass keine weiteren Administrationsrechte vergeben wurden (z.B. SYS-Rolle)!

13.1 Benutzer

Die Verwaltung der Benutzer erfolgt analog wie im Abschnitt [Benutzer](#) beschrieben, jedoch gibt es einige Einschränkungen/Abweichungen:

13. Eingeschränkte Administration

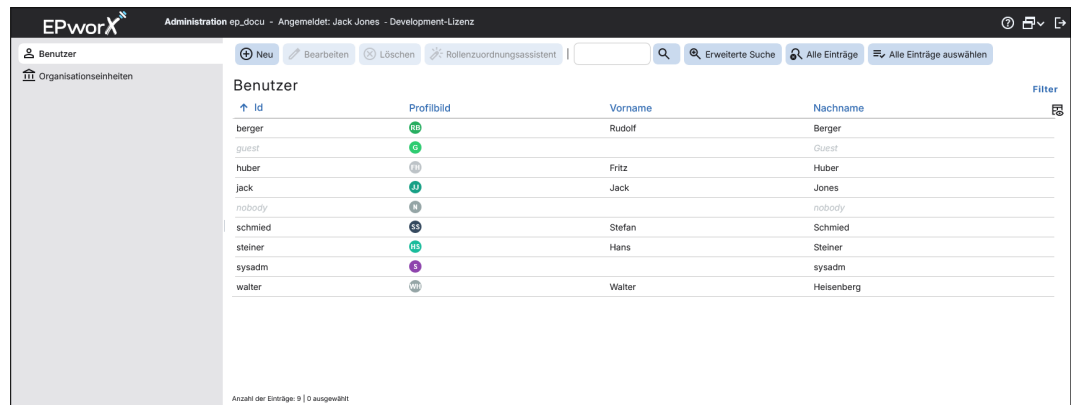


Abbildung 13.1: **Eingeschränkte Administration**

- Beim Anlegen von Benutzern über die Toolbarfunktion *Neu* wird zuerst eine vereinfachte Maske angezeigt. Nachdem die Felder ausgefüllt und die Schaltfläche *Erstellen* ausgeführt wurde, wird ein Benutzer-Objekt angelegt mit der Rollenzuordnung *Home in der Organisationseinheit des aktuell eingeloggten Benutzers*.
- Bei der Rollenzuordnung kann nur die Organisationseinheit verändert werden, d.h. es werden nur Organisationseinheiten angeboten, die im OE-Baum hierarchisch unterhalb der aktuellen Organisationseinheit liegen. Wird die Rollenzuordnung *home* gelöscht, kann der Administrator der eingeschränkten Administration das Benutzer-Objekt nicht mehr bearbeiten bzw. löschen!
- Eine Definition von Vertreter bzw. anderen Berechtigungen ist nicht möglich.

13.2 Organisationseinheiten

Die Verwaltung der Organisationseinheiten erfolgt analog wie im Abschnitt [Organisationseinheiten](#) beschrieben, jedoch gibt es auch hier Einschränkungen/Abweichungen:

- Beim Anlegen von Organisationseinheiten über die Toolbarfunktion *Neu* wird zuerst eine vereinfachte Maske angezeigt. Nachdem die Felder ausgefüllt und die Schaltfläche *Erstellen* ausgeführt wurde, wird ein OE-Objekt angelegt. Die neue Organisationseinheit wird dabei im OE-Baum hierarchisch unterhalb der Organisationseinheit des aktuell eingeloggten Benutzers eingefügt.
- Im Reiter *übergeordnete OEs* kann die Organisationseinheit verändert werden. Dabei werden nur jene Organisationseinheiten angeboten, die hierarchisch unterhalb der Organisationseinheit im OE-Baum positioniert sind bzw. die Organisationseinheit wie bei der Rollenzuordnung des aktuell eingeloggten Benutzers definiert.

13. Eingeschränkte Administration

Allgemein	Übergeordnete Org.-Einheiten	Rollen	Historie	Zugriff	Referenzen
OE-Baum	Organisationseinheit				
Default	▼ Department 1 Department 1a Department 1b Department 2 Department 2a Finance Human Resources Management				
Löschen		Speichern und Schließen		Speichern	Abbrechen

Abbildung 13.2: **Eingeschränkte Administration - Reiter übergeordnete OEs**