independIT Integrative Technologies GmbH
Bergstraße 6
D–86529 Schrobenhausen

schedulix!Web

Benutzerhandbuch
Release 2.8

Dieter Stubler Ronald Jeninga

2. November 2017

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

1 Allgemein

1.1 Überblick

Im folgenden Kapitel wird eine Einführung in die schedulix Oberfläche gegeben. Hier werden die allgemeinen Aktionen und Verfahrensweisen beschrieben und erläutert, die unabhängig vom gerade ausgewählten Dialog sind.

Die schedulix Oberfläche ist ein browserbasiertes Dialogsystem. Es ist keine Installation von zusätzlicher Client Software nötig.

Zum Starten der schedulix Oberfläche muss ein Browser gestartet werden (z.B. Microsoft Internet Explorer). Um auf die schedulix Oberfläche zu gelangen, geben Sie bitte die URL ein, welche Sie bekommen haben. Es empfiehlt sich, diese URL in Ihrem Browser als Favorit abzuspeichern. So können Sie die schedulix Oberfläche durch einen Klick starten.

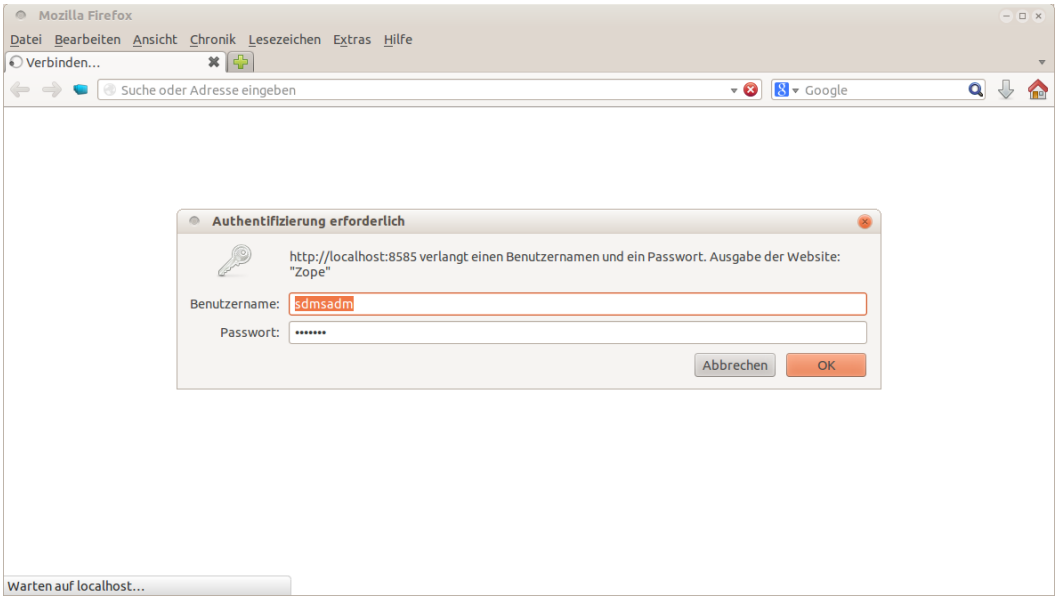


Abbildung 1.1: Login-Bildschirm

1.2 Login

Nach der Eingabe der URL öffnet sich eine Login-Maske zur Passwortüberprüfung. Jeder Benutzer muss sich mit seiner Benutzerkennung und seinem Passwort identifizieren, um mit schedulix arbeiten zu können.

Die Eingabemaske für die Passwortabfrage ist in Abbildung 1.1 dargestellt, das genaue Aussehen kann je nach Browser etwas variieren.

Hier tragen Sie Ihren persönlichen Benutzernamen und das dazugehörige Passwort ein und bestätigen Ihre Eingabe mit *OK*. Die Login-Daten erhalten Sie von Ihrem Systemadministrator.

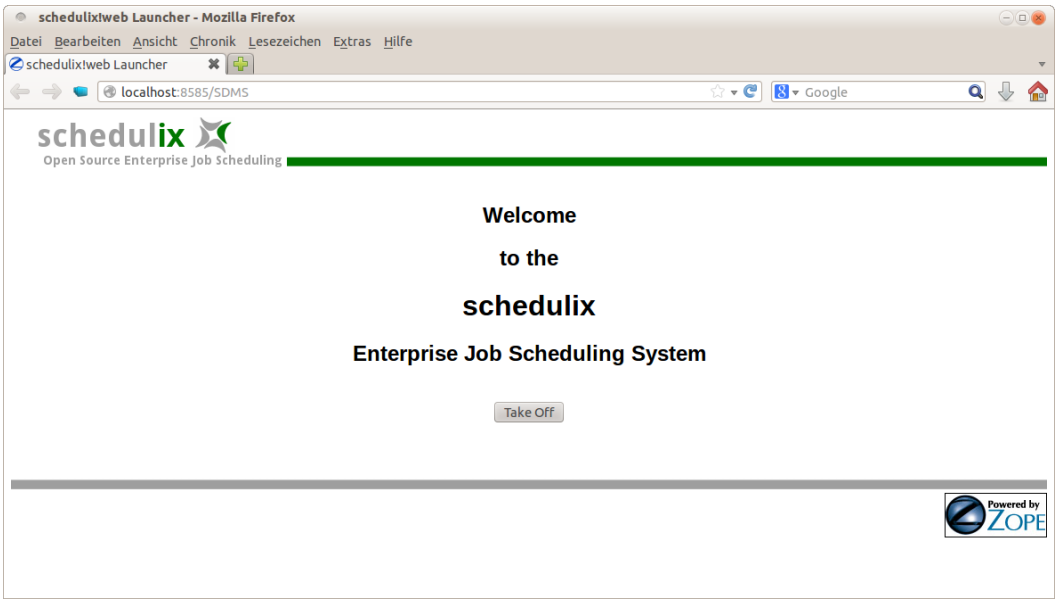


Abbildung 1.2: schedulix Launcher

Nach der korrekten Eingabe erscheint der schedulix Launcher. Starten Sie jetzt das Hauptmenü mit einem Klick auf den Button *Take Off*.

Abhängig vom Browser wird der Launcher automatisch geschlossen. Sicherheitseinstellungen des Browsers können dies verhindern oder zu einer Warnmeldung führen.

Nach dem Drücken des Button *Take Off* erscheint das Hauptmenü, der *Main Desktop*.

1.3 Main Desktop

Der *Main Desktop* ist das Hauptfenster der schedulix Oberfläche. In diesem Fenster können Sie alle Funktionsdialoge der schedulix direkt starten. Über das Optionsfeld *Connection* in der Kopfzeile wählen Sie aus, welche Serververbindung Sie für den nächsten zu öffnenden Funktionsdialog verwenden möchten.

Abbildung 1.3: Main Desktop

Der *Main Desktop* ist in Abbildung 1.3 dargestellt. Durch Anklicken der nachfolgend aufgeführten Icons (Tabelle 1.1) können Sie die Dialoge zur Pflege oder Überwachung verschiedener Objekte aufrufen:

Icon	Beschreibung
	*Bookmarks* dient zum Anlegen, Ändern und Löschen von Ansichten aktuell laufender Jobs.
	*Running Master Jobs* dient der übersichtlichen Anzeige von aktuellen Master Jobs.
	*Search Running Jobs* ermöglicht die Suche nach laufenden Jobs anhand bestimmter Kriterien.
	*Submit Jobs* startet Abläufe.
	*Batches and Jobs* dient zum Anlegen und Verwalten von Foldern, Batches, Jobs und Milestones.

Tabelle 1.1 – *Fortsetzung auf der nächsten Seite*

Tabelle 1.1 – *Fortsetzung der vorherigen Seite*

Icon	Beschreibung
	*schedulix!Web Users* dient der Verwaltung von schedulix!Web Benutzern.
	*schedulix Server Users* dient der Verwaltung von schedulix Server-Benutzern.
	*Groups* dient der Verwaltung von schedulix Benutzergruppen.
	*Jobserver and Resources* dient der Verwaltung von Jobservern, Scopes und den zugehörigen Resources.
	*Named Resources* dient zum Anlegen, Ändern und Löschen von Named Resources.
	*Environments* dient der Definition von Laufzeitumgebungen für Jobs.
	*Footprint* dient der Verwaltung von Footprints (Resources-Profilen) im System.
	In den *Exit State Definitions* können Sie Exit State-Namen definieren und verwalten.
	*Exit State Mappings* dienen der Verwaltung von Übersetzungen zwischen Rückgabewerten der Systemprozesse und logischen Exit States.
	*Exit State Profile* dient der Verwaltung von Profilen, also der Zusammenfassung und Priorisierung von Exit States.
	*Exit State Translation* dient der Verwaltung der Übersetzungen von Child Exit States nach Parent Exit States.
	*Resource State Definitions* dient der Definition und Verwaltung von Resource-Status-Namen.
	*Resource State Profile* dient der Verwaltung von Profilen, also Zusammenfassungen mehrerer Resource-States.
	*Resource State Mapping* dient der Verwaltung der Übersetzungen von Exit State nach Resource-Statusänderung.
	*Calendar* liefert eine Übersicht der anstehenden Submits.
	*SysInfo* liefert Informationen über die Konfiguration und den aktuellen Zustand des Systems.

Tabelle 1.1: Beschreibung der Icons des Hauptmenüs

1.4 Fensteraufbau Objektfenster

Nach dem Aufrufen eines Dialoges im *Main Desktop* wird der Dialog in einem neuen Fenster angezeigt. Jedes Dialogfenster ist unabhängig von anderen Dialogfenstern, es dürfen beliebig viele Dialoge zur selben Zeit geöffnet sein.

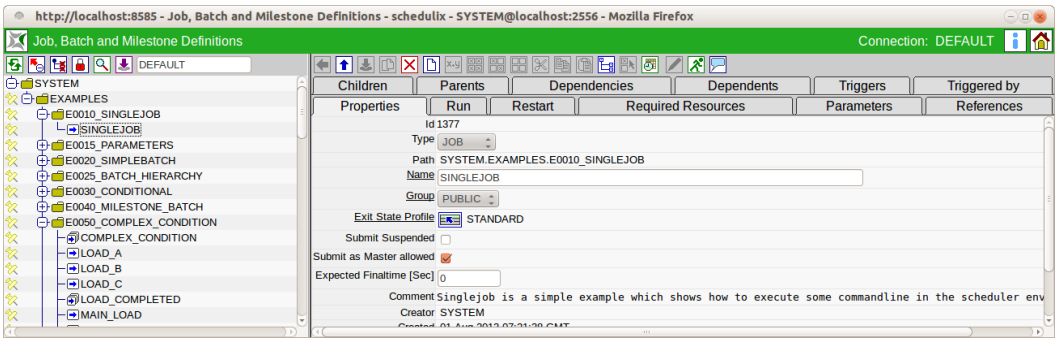


Abbildung 1.4: Beispiel eines normalen Dialogs

Ein normaler Dialog sieht aus wie im Bild 1.4.

1.4.1 Titelleiste

Jeder Dialog besitzt im oberen Teil eine Titelleiste. Die Titelleiste enthält die nachfolgend beschriebenen Elemente.

independIT-Logo, **Sanduhr**

In der linken oberen Fensterecke befindet sich das independIT-Icon. Ein Klick darauf öffnet ein neues Fenster mit der Homepage der independIT GmbH. Bei manchen Aktionen müssen neue oder geänderte Daten an den schedulix Server gesendet, bzw. Daten vom Server geladen werden. Diese Aktivität wird durch das Sanduhr-Icon angezeigt. Solange das Sanduhr-Icon aktiv ist, ist keine weitere Aktion in diesem Fenster möglich. Versuchen Sie dennoch eine Aktion durchzuführen, erscheint eine Fehlermeldung (siehe Abbildung 1.5).
In diesem Fall muss die Box mit *OK* bestätigt und das Ende der Datenübertragung vom Server abgewartet werden. Verschwindet die Sanduhr und es erscheint wieder das independIT-Icon, können Sie mit der Dateneingabe fortfahren.

Name des Dialoges
Neben dem independIT-Icon erscheint der Name des aktuellen Dialoges.

About

Mit diesem Button erhalten Sie Versions- und Systeminformationen.

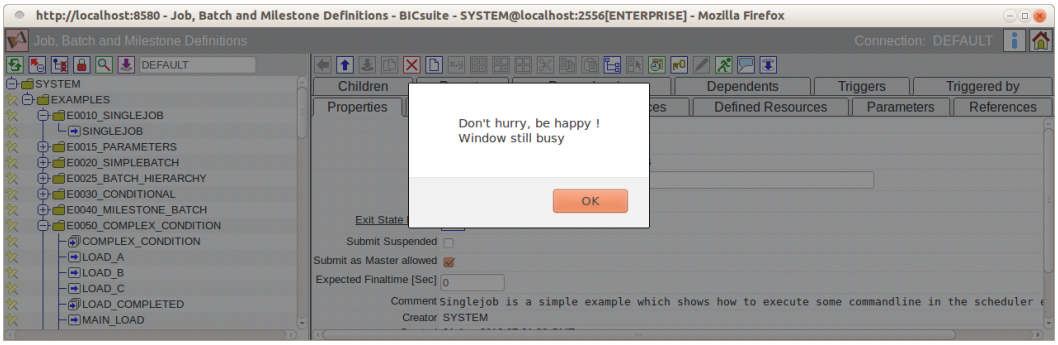


Abbildung 1.5: Busy Meldung

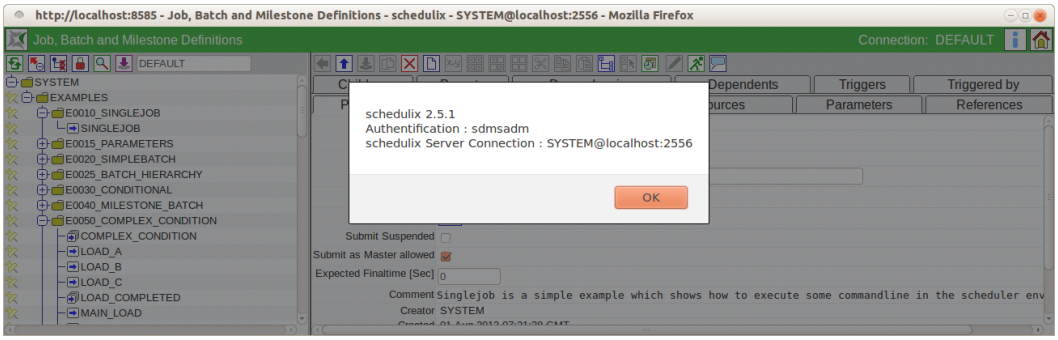


Abbildung 1.6: About Fenster

Folgende Informationen sind in dem Fenster zu sehen:
schedulix!Web n.n.n: Die aktuelle Version der schedulix Oberfläche
Authentification: Der Name des aktuell eingeloggten Benutzers
schedulix Server Connection: Die Verbindungsdaten für den schedulix Server

Help

Das Icon *Help* ist nur auf dem *Main Desktop* anstelle des Icon *Home* zu sehen. Ein Klick auf das Icon *Help* bringt Sie direkt in die Online-Dokumentation.

Home

Das Icon *Home* dient zum Aufrufen des *Main Desktop*. Ist das Fenster des *Main Desktop* minimiert oder im Hintergrund, wird es wiederhergestellt und nach vorne gehoben. Wurde es geschlossen, wird ein neues *Main Desktop*-Fenster geöffnet.

1.4.2 Navigator

Der linke Teil eines Dialogfensters ist der Navigator. Er zeigt die bereits definierten Objekte an. Dies erfolgt entweder hierarchisch (falls die Art von Objekten hierar-

chische Anordnungen zulassen, wie z.B. Named Resources) oder als einfache Liste, wenn keine hierarchische Ordnung besteht. Die hierarchische Darstellung erfolgt in einer Baumstruktur und sieht dann aus wie in Bild 1.7.

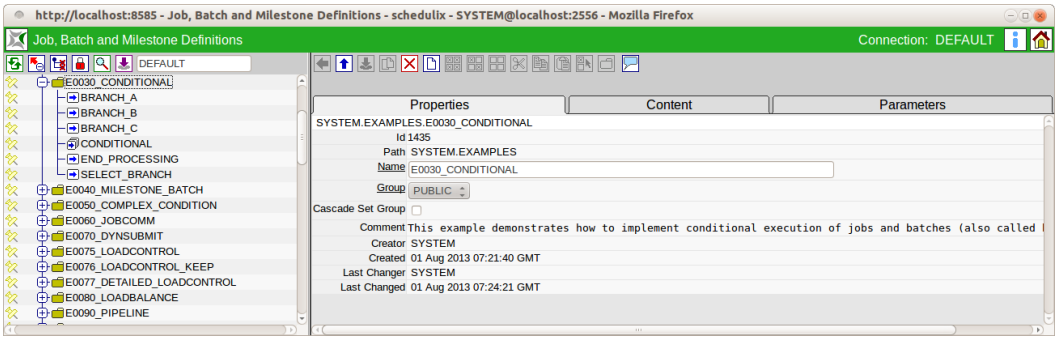


Abbildung 1.7: Navigator mit hierarchischer Darstellung

Handelt es sich bei einem in der hierarchischen Ansicht angezeigten Objekt um einen Container, der noch andere Objekte enthalten kann, so wird dieser mittels eines Ordner-Icons, z. B.

dargestellt. Je nach Farbe und Darstellung handelt es sich um unterschiedliche Objekte (Folder, Scope, Named Resource etc.). Enthält ein Container darüber hinaus noch Unterobjekte, so wird ein ⊞ oder ⊟ angezeigt. Mit einem Klick auf eines dieser Icons wird der Inhalt des Containers angezeigt ⊞ oder versteckt ⊟.

1.4.2.1 Standard-Buttons

Über die nachfolgend beschriebenen Buttons, die sich über dem Navigator befinden, steuern Sie die Anzeige und das Anzeigen bzw. Verstecken von Inhalten:

Refresh

Der Button *Refresh* frischt die Liste der Objekte in der Navigation auf. Mit einem Klick auf diesen Button werden alle verfügbaren Objekte neu vom Server gelesen.

Collapse All

Mit diesem Button wird eine aufgeklappte Hierarchie, also eine Hierarchie, in der Unterordner angezeigt werden (und der übergeordnete Ordner ein ⊟ besitzt), komplett geschlossen. Das heißt, es sind nur noch die Objekte auf der obersten Hierarchiestufe sichtbar.

 **Expand All**

Mit dem Button *Expand All* wird eine geschlossene oder teilweise geöffnete Hierarchie komplett geöffnet. Das bedeutet, alle Container der obersten Ebene werden auf Untercontainer oder Unterobjekte untersucht und diese werden angezeigt. Dem Ordnersymbol wird das  Zeichen vorangestellt. Sollten die Untercontainer weitere Unterobjekte besitzen, so werden auch diese angezeigt, bis alle Objekte in der kompletten Hierarchie auf dem Bildschirm sichtbar sind. Handelt es sich um eine große Hierarchie mit vielen Objekten und Unterobjekten, kann dies eine verhältnismäßig lange Laufzeit zur Folge haben. Aus diesem Grund ist der Button nicht bei jedem hierarchischen Objekttyp verfügbar.

 **Show Leaves**,  **Hide Leaves**

Diese beiden Buttons ermöglichen das Ein- und Ausblenden von Nicht-Containerobjekten in Hierarchien im Navigator. Es ist abwechselnd immer nur ein Button sichtbar.

 **Show Locked**,  **Hide Locked**

Diese beiden Buttons ermöglichen das Ein- und Ausblenden der Namen von Objekten, auf die keine Leserechte vorliegen. Es ist abwechselnd immer nur einer der beiden Buttons sichtbar.

 **Save View**

Mit dem Button *Save View* wird der aktuelle Zustand der Hierarchie im Navigator gespeichert.

 **Search**

Dieser Button dient der Aktivierung der Suchmaske in der Navigation. Damit lässt sich in einer komplexen Hierarchie ein namentlich bekanntes Objekt sehr leicht finden. Nach Betätigen des Buttons *Search* erscheint die Suchmaske wie im Bild 1.8.

 **Start Find**

Hiermit wird die Suche gestartet. Im abgebildeten Beispiel wird in der gesamten Hierarchie nach *Named Resources* gesucht, welche den Suchbegriff enthalten.
Treffer werden als Liste im Navigator angezeigt. Hierarchien werden nicht berücksichtigt. Das Ergebnis sieht z. B. aus wie im Bild 1.9.
Hier wurde nach dem Begriff "Host" gesucht und drei Named Resources gefunden, welche den Begriff enthalten. Die Suche ist unabhängig von Groß- und Kleinschreibung des jeweiligen Suchbegriffes.

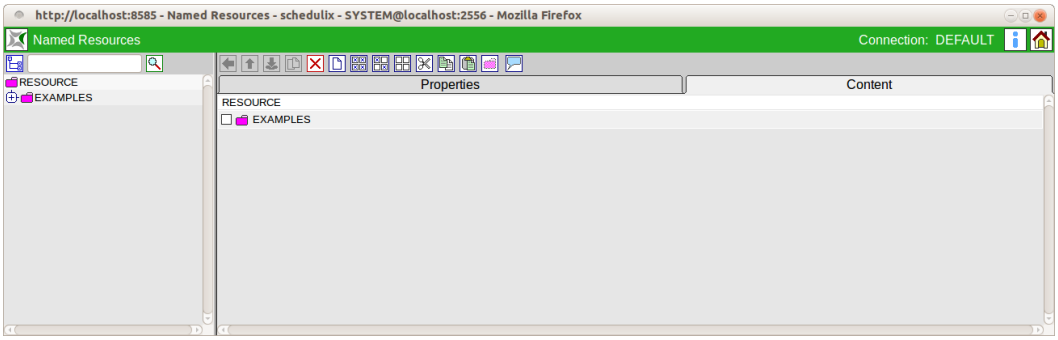


Abbildung 1.8: Suchen im Navigator

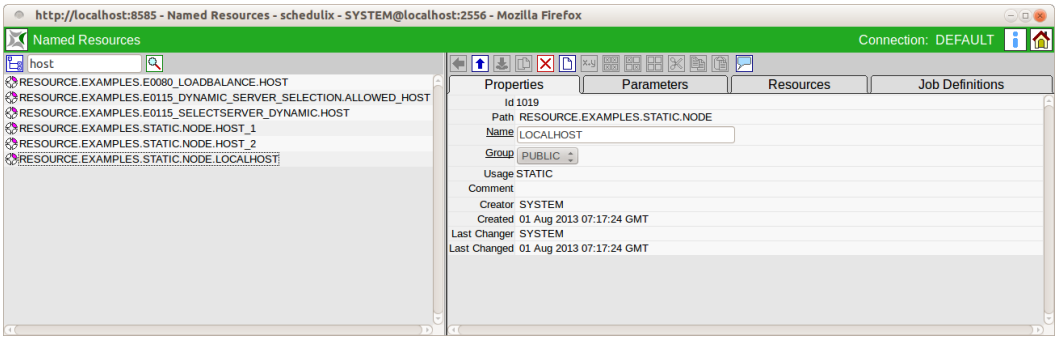


Abbildung 1.9: Suchergebnis

Exit Find

Hiermit wird die Suchmaske wieder verlassen. Nach Betätigung dieses Buttons wird wieder die normale Buttonleiste über dem Navigator angezeigt.

1.4.3 Editor

Der Editor ist der Dialog zum Ändern oder Erstellen von Objekten. Wurde ein Objekt im Navigator ausgewählt, so erscheinen seine Daten im Editor und können hier geändert werden. Ein selektiertes Objekt kann über die Buttons am oberen Rand des Editors gespeichert, gelöscht oder dupliziert werden. Mit dem *New*-Button legen Sie ein neues Objekt an.
Der Editor zeigt die Daten entweder komplett innerhalb eines Fensters an, oder die Inhalte erscheinen logisch gruppiert in unterschiedlichen Tabs.
Die Anzahl der Tab Sheets in einem Dialog ist abhängig von der Art des aktuell in der Navigation gewählten Objektes. Sie können innerhalb einer Aktion (Änderung, Neuanlage eines Objektes) beliebig oft und in beliebiger Reihenfolge zwischen den einzelnen Tabs wechseln. Eine Zwischenspeicherung ist nicht nötig, da alle Daten

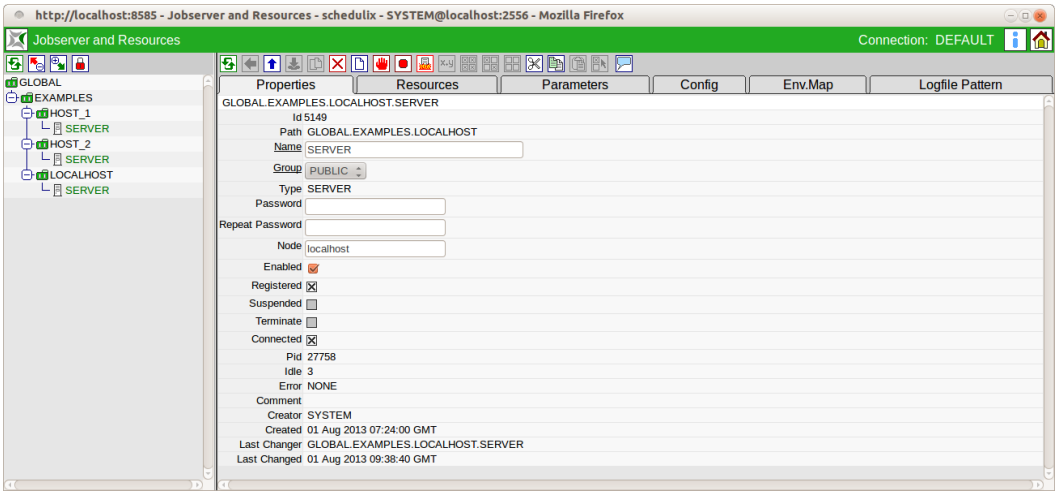


Abbildung 1.10: Editor mit Tabs

(auch die Daten der nicht sichtbaren Tabs) im Speicher vorgehalten werden. Erst beim Betätigen des Button *Save* werden die Daten aller Tabs zum schedulix Server übertragen und gespeichert. Falls Änderungen durch den *Cancel* Button rückgängig gemacht wurden, werden alle geänderten Daten verworfen, auch die auf den nicht sichtbaren Tabs.
Die Auswahl eines Tab-Sheets erfolgt einfach über das Anklicken des jeweiligen Reiters, auf dem der Name des Tab-Sheets steht. Die Daten des gewählten Tabs erscheinen anschließend auf der Maske und der gewählte Tab wird graphisch in den Vordergrund gehoben und hellblau markiert. Die nicht aktiven Tabs erscheinen dunkelblau.
Die Buttonleiste des Editors zeigt je nach Tab unterschiedliche aktive und inaktive Buttons.

1.4.3.1 Standard-Buttons

Folgende Standard-Buttons sind bei vielen Dialogen vorhanden und sind in ihrer Handhabung in jedem Dialog identisch oder vergleichbar.

Cancel

Dieser Button dient zum Widerrufen von Aktionen. Hiermit können Sie alle noch nicht gesicherten Änderungen rückgängig machen. Das bedeutet, dass alle Änderungen, die seit dem Laden oder dem letzten Speichern durchgeführt wurden, verworfen werden.
Der Button *Cancel* ist nur aktiv, wenn eine Änderung durchgeführt wurde. Änderungen werden erst erkannt, sobald das erste Feld, in dem eine Änderung durchgeführt wurde, verlassen wird. Ist nur ein Feld auf der Maske vorhanden, so muss die

Tabulatortaste gedrückt werden, um die Änderung zu akzeptieren. Anschließend schaltet sich der Button *Cancel* aktiv.
Um einen Verlust auszuschließen, folgt nach Betätigung des Buttons noch eine Sicherheitsabfrage. Sollen die Änderungen wirklich verworfen werden, muss die Abfrage mit *OK* bestätigt werden. Andernfalls kehren Sie mit einem Klick auf *Cancel* zum vorherigen Dialog zurück und die Änderungen bleiben erhalten.

Up

Mit dem Button *Up* springen Sie eine Hierarchieebene nach oben.

Save

Dieser Button dient zum Speichern einer durchgeführten Änderung. Der Button *Save* wird nur aktiv, wenn tatsächlich Daten geändert wurden. Änderungen werden erst erkannt, wenn das geänderte Eingabefeld verlassen wird.

Clone

Dieser Button dient zum Erstellen eines neuen Objektes, das alle Eigenschaften des aktuell gewählten besitzt. Um den Button zu aktivieren, muss ein neuer Name für das Objekt eingetragen werden. Durch Betätigen des Buttons wird ein neues Objekt mit diesem Namen angelegt und im Navigationsfenster angezeigt.
Achtung: Falls statt des Buttons *Clone* der Button *Save* betätigt wird, wird nur der Name des Objektes geändert, es wird kein neues Objekt erzeugt.

Drop

Dieser Button dient zum Löschen des gesamten Objektes. Vor dem Löschen erfolgt eine Sicherheitsabfrage.
Wenn Sie das Objekt wirklich löschen wollen, bestätigen Sie dies mit *OK*, falls Sie den Vorgang abbrechen wollen, drücken Sie *Cancel*.
Nach der Bestätigung mit *OK* wird das Objekt vom Server gelöscht. Es gibt nun keine Möglichkeit mehr, das Löschen rückgängig zu machen. Falls versehentlich gelöscht wurde, muss das Objekt neu angelegt werden.
Innerhalb von Listen können Selektionen ebenfalls gelöscht werden. Dafür müssen Sie zuerst die zu löschenden Zeilen einzeln oder über die *Selektion*-Buttons selektieren und danach auf den Button *Drop* klicken. Nach der Bestätigung der Sicherheitsabfrage werden alle selektierten Einträge gelöscht.

New

Dieser Button dient zum Anlegen eines neuen Objektes. Können innerhalb eines Dialoges unterschiedliche Objekte angelegt werden, so muss zuerst der Typ des neuen Objektes ausgewählt werden. Anschließend erscheinen die Felder und Tabs des entsprechenden Objekttyps. Ist keine Auswahl notwendig, erscheint der Editor

sofort nach Betätigung des Buttons. Wenn ein Editor-Fenster noch ungespeicherte Änderungen enthält, ist der Button deaktiviert, damit ein versehentliches Löschen der geänderten Werte vermieden wird.

Show Folderpath, Hide Folderpath

Bei diesen Buttons handelt es sich um Schalter. Wurde der Button *Show Folderpath* gedrückt, so werden Scopes, Submitted Entities, Scheduling Entities und Folder mit der kompletten übergeordneten Hierarchie angezeigt. Die Ebenen werden jeweils durch ein Punkt getrennt. Nach Drücken des Buttons sieht die Anzeige folgendermaßen aus:

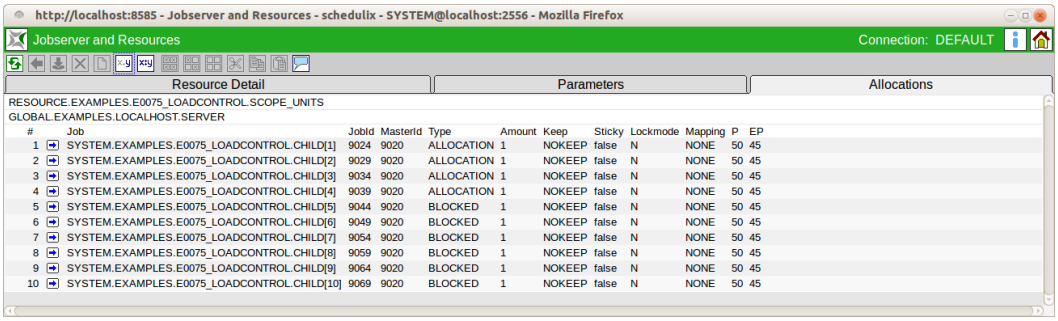


Abbildung 1.11: Show Folderpath

Wird der Button *Hide Folderpath* gedrückt, werden die bisher sichtbaren übergeordneten Ordner nicht mehr angezeigt. Die Anzeige sieht dann aus wie im Bild 1.12.

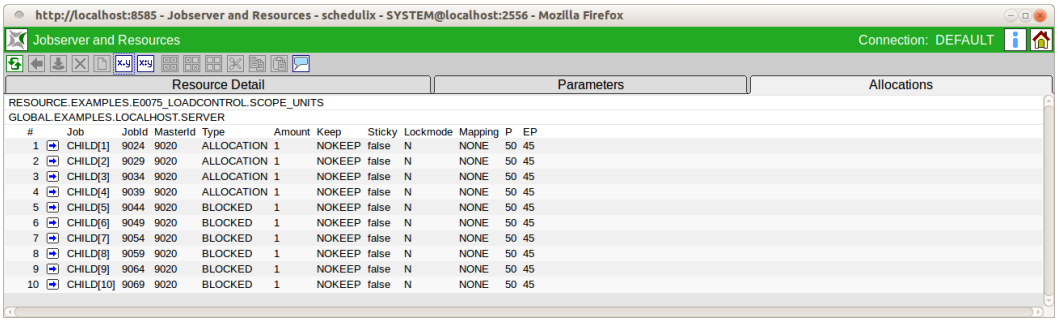


Abbildung 1.12: Hide Folderpath

Show Hierarchypath, Hide Hierarchypath

Bei diesen Buttons handelt es sich um Schalter. Wurde der Button *Show Hierarchypath* gedrückt, so wird das Ablaufobjekt mit der kompletten übergeordneten Parent-Child-Hierarchie angezeigt. Hat ein Ablaufobjekt einen Parent, wird dieser

im Namen mit angezeigt und so weiter bis die oberste Ebene erreicht ist. Die Ebenen werden jeweils durch Doppelpunkt getrennt.

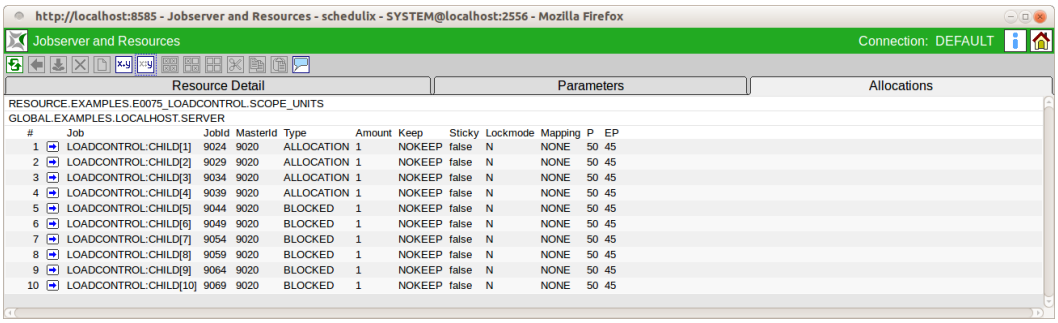


Abbildung 1.13: Show Hierarchypath

Wird der Button *Hide Hierarchypath* gedrückt, so wird die bisher sichtbare Hierarchie nicht mehr angezeigt. Nach Drücken des Buttons sieht die Anzeige folgendermaßen aus:

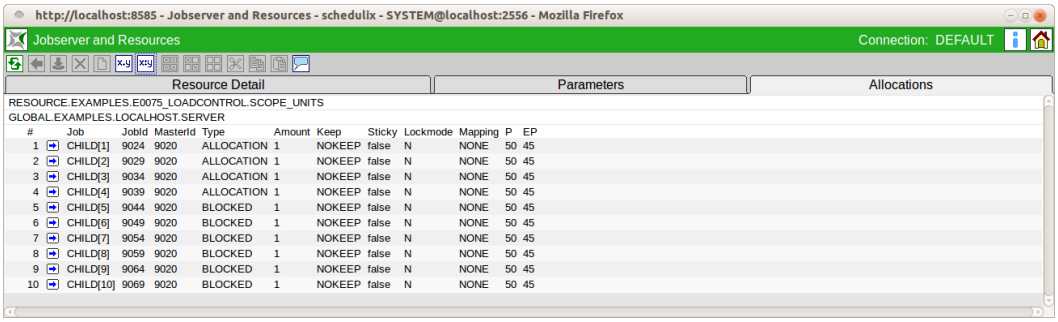


Abbildung 1.14: Hide Hierarchypath

Select All

Dieser Button dient zum Selektieren aller Objekte. Es werden alle Objekte, die aktuell im Navigator oder im Editor-Fenster angezeigt werden (je nachdem in welcher Buttonleiste sich der Button befindet), ausgewählt. Anschließend sind alle Objekte mit einem Kreuz ☒ vor dem Namen markiert.
Nachdem alle Objekte selektiert wurden, können diese nun ausgeschnitten, kopiert oder gelöscht werden.

Toggle Selection

Dieser Button dient zur Umkehrung der aktuellen Selektion. Es werden also alle selektierten Objekte deselektiert und alle noch nicht selektierten Objekte werden

selektiert. Anschließend können die selektierten Objekte gelöscht, ausgeschnitten oder kopiert werden.
Das Verhalten wird in den folgenden beiden Abbildungen verdeutlicht:

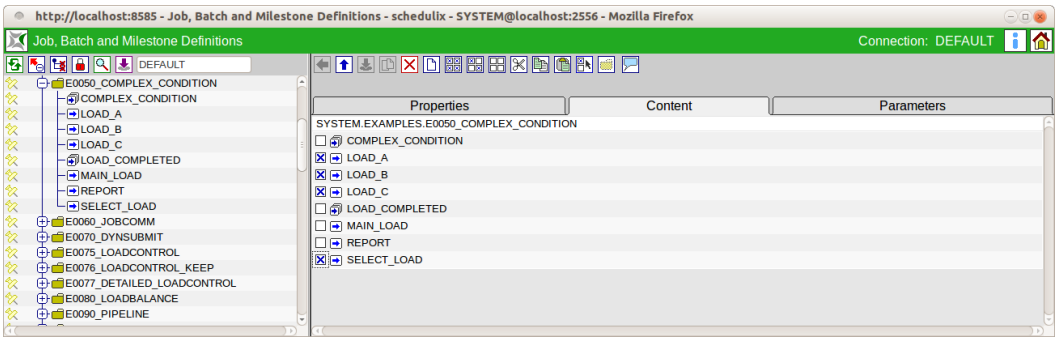


Abbildung 1.15: Selektion umkehren; Anfangszustand

Nach dem Betätigen des Toggle Selection Buttons ist die Selektion ”umgekehrt”.

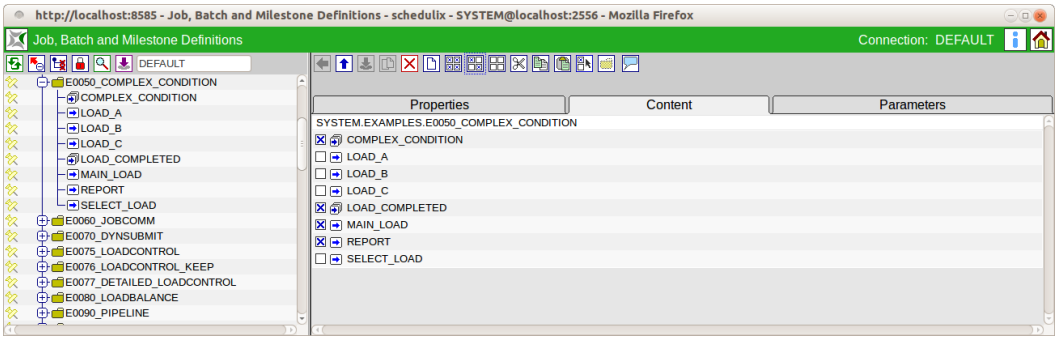


Abbildung 1.16: Selektion umkehren; Resultat

Deselect All

Dieser Button dient zum Aufheben der Selektion. Die Markierungen an den selektierten Einträgen werden entfernt. Hiermit kann eine bestehende Selektion wieder rückgängig gemacht werden.

Cut

Dieser Button dient zum Ausschneiden von Objekten aus einer Liste oder einem Baum. Die Objekte werden in die schedulix Zwischenablage kopiert. Die Objekte sind nur zum Ausschneiden vorgemerkt, sie verschwinden allerdings aus der aktuellen Ansicht. Durch Betätigen des Buttons *Paste* können die Objekte an anderer Stelle wieder eingefügt werden.

Copy

Mit dem Button *Copy* können Objekte in die schedulix Zwischenablage kopiert werden, um sie an anderer Stelle mittels des Button *Paste* wieder einzufügen.

Paste

Dieser Button dient zum Einfügen von vorher selektierten und geeigneten Objekten in einer Liste. Die Objekte müssen vorher innerhalb eines anderen Dialoges oder Pop Up-Fensters selektiert werden.

Select

Dieser Button dient zum Öffnen eines Fensters, in dem Sie für eine Liste geeignete Objekte selektieren und mit dem Button *Copy* in die Zwischenablage kopieren können. Mit dem Button *Paste* fügen Sie die Objekte aus der Zwischenablage in die Liste ein.

Time Scheduling

Dieser Button dient zum Aufrufen des Time Scheduling-Dialoges. Er wird nur angezeigt, falls im Navigationsbildschirm ein Master Submittable Objekt ausgewählt wurde.

Grants

Mit diesem Button wird der Dialog zur Verwaltung der Zugriffsrechte geöffnet.

Edit

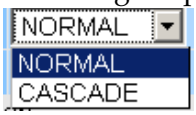
Der Button *Edit* schaltet den Editiermodus ein. Was editiert wird, ist abhängig vom jeweiligen Kontext.

1.5 Werteauswahl

1.5.1 Werteauswahl durch ”Drop Down”-Liste

Bei manchen Eingabefeldern ist die Auswahl aus einer Menge von vorgegebenen Werten erforderlich. In diesen Fällen wird ein *Drop Down*-Feld angezeigt, in dem alle möglichen Werte aufgelistet sind.

Ein *Drop Down*-Feld sieht folgendermaßen aus: 

Wird der Button neben dem Feld angeklickt oder mittels Tastaturbefehl ALT-Pfeil-Unten die ”Drop Down”-Liste ausgeklappt, ist eine Einfachauswahl in der Liste möglich.
Die ausgeklappte ”Drop Down”-Liste kann z.B. folgendermaßen aussehen:


Durch Anklicken des gewünschten Wertes wird dieser in das Feld übernommen. Alternativ ist es auch möglich, die Werte mittels der Pfeil-Unten- bzw. Pfeil-Oben-Tasten auszuwählen.

1.5.2 Werteauswahl durch Auswahl-Button (Choose)

Wird die Liste der verfügbaren Werte erst zur Laufzeit vom Server gelesen, kommt der Button *Choose* zum Einsatz. In der nachfolgenden Abbildung kann ein Resource State Profile mittels des *Choose* Buttons gewählt werden:

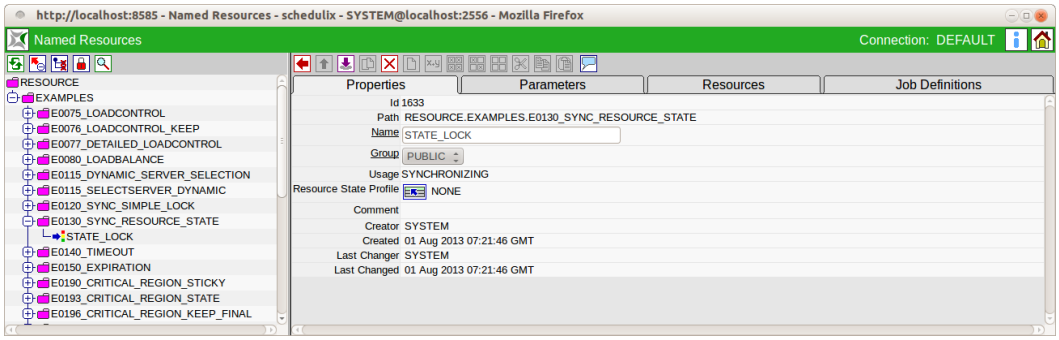


Abbildung 1.17: Auswahl eines Resource State Profiles; Anfangszustand

Nach dem Betätigen des *Choose* Buttons

erscheint im Editor eine Liste von möglichen Objekten.

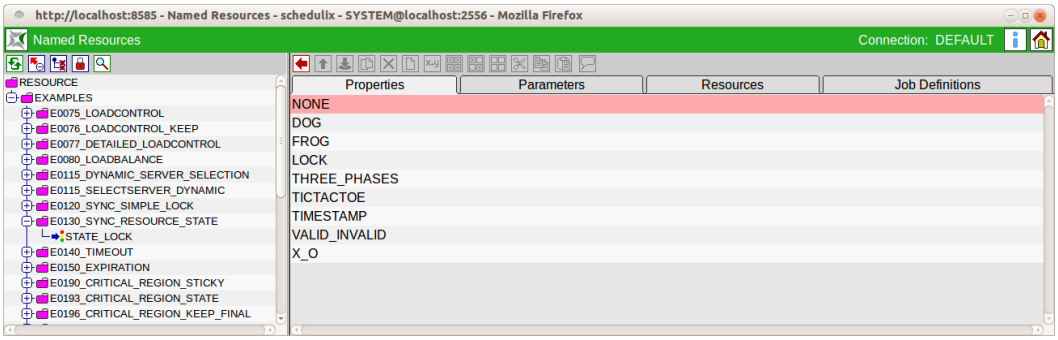


Abbildung 1.18: Auswahl eines Resource State Profiles; Auswahlliste

Durch Anklicken eines dieser Objekte wird das Element in das Feld übernommen. Mittels des *Cancel* Buttons kann von der Auswahlmaske, auch ohne eine Auswahl zu treffen, wieder in den Editorbereich zurückgesprungen werden.

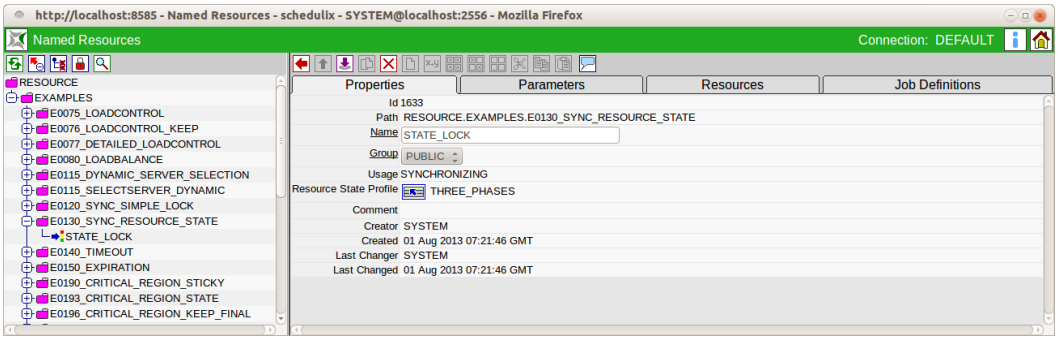


Abbildung 1.19: Auswahl eines Resource State Profiles; Resultat

1.6 Standard Listen Handling

In vielen Dialogen gibt es die Möglichkeit, nicht nur Werte in Felder einzugeben, sondern darüber hinaus noch Listen mit Verknüpfungen auf Zusatzobjekte zu erstellen. So wird z. B. innerhalb eines Resource State Profiles eine Liste von Resource States verwaltet, die zu diesem Profil gehören. Hier ist zu den normalen Feldern *Name* und *Default-Initial-State* noch eine Liste von States anzugeben. Innerhalb einer solchen Liste können nun die nachfolgenden Aktionen durchgeführt werden.

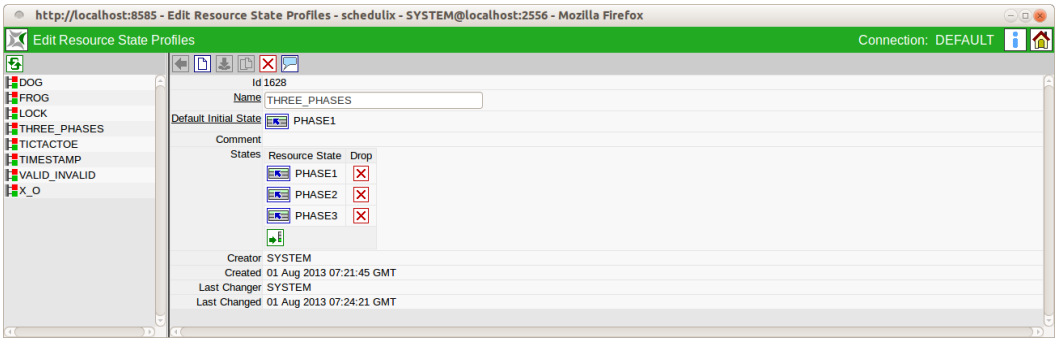


Abbildung 1.20: Beispiel Standard Listen Handling

1.6.1 Hinzufügen einer Zeile

Append (Hinzufügen)

Mit dem Button *Append* wird eine neue Zeile zur aktuellen Liste hinzugefügt. Nach dem Betätigen des Buttons erscheint eine leere Zeile. In unserem Beispiel sieht das dann aus wie in der folgenden Abbildung:

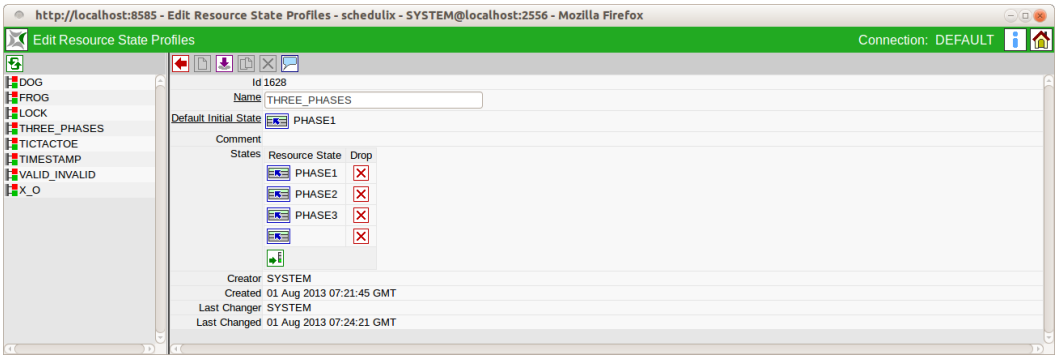


Abbildung 1.21: Listen Handling; Hinzufügen einer Zeile

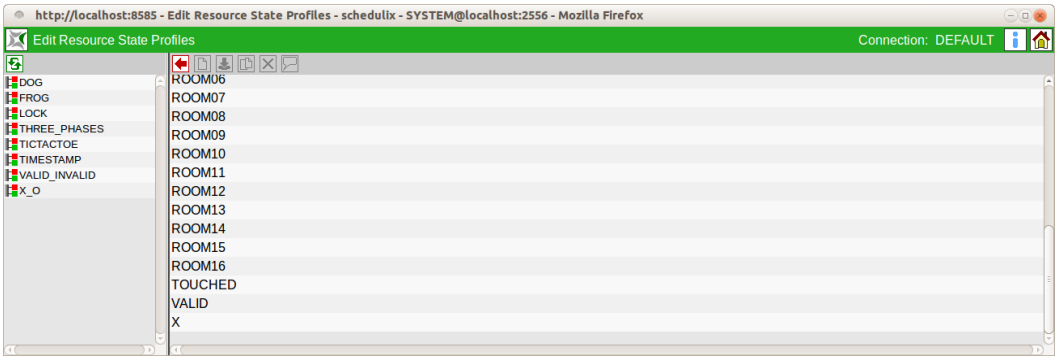


Abbildung 1.22: Listen Handling; Auswahl eines Wertes

Anschließend kann über den Auswahl-Button ein Wert ausgewählt werden.
In diesem Beispiel wurde der Wert ”Valid” ausgewählt.

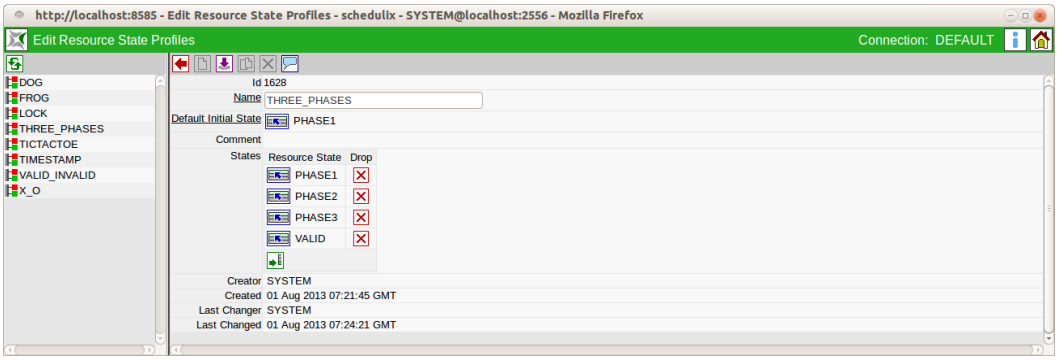


Abbildung 1.23: Listen Handling; Resultat

Die Änderung muss nun mit Hilfe des Button *Save* gespeichert werden.
Die Möglichkeit der Auswahl eines abhängigen Objektes durch den *Choose* Button kann auch für die Änderung einer bestehenden Zeile durchgeführt werden. Hier wird dann die Verknüpfung zum alten Objekt gelöst und eine neue Verknüpfung zum gewählten Objekt aufgebaut.

1.6.2 Löschen einer Zeile

Drop (Löschen)

Mittels des Buttons *Drop* wird eine Zeile entfernt und die Verknüpfung vom aktuell im Editor geladenen Objekt und dem gewählten Objekt in der Zeile nach erfolgreicher Sicherheitsabfrage gelöscht. Im Beispiel wird durch Betätigen des Buttons *Drop* die Zeile ”PHASE1” gelöscht.

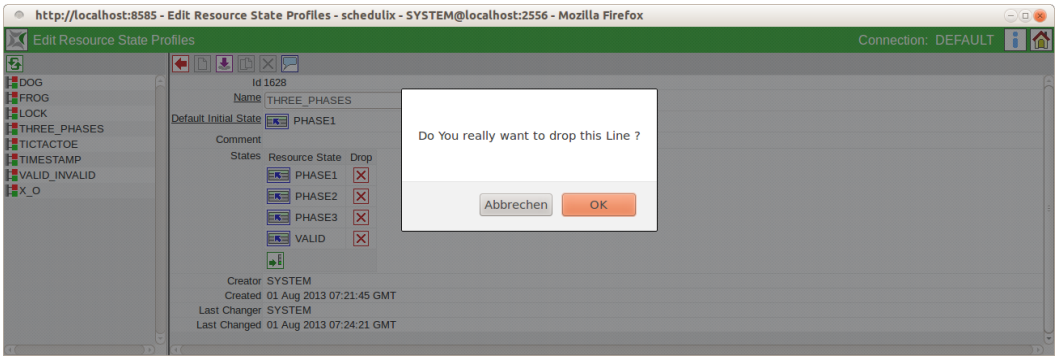


Abbildung 1.24: Listen Handling; Löschen einer Zeile

Die Zeile mit "PHASE1" ist nach einem Klick auf den Button *OK* nicht mehr vorhanden. Das Objekt der Zeile (also der Resource State "PHASE1") existiert nach wie vor, er ist nur nicht mehr diesem Profil zugeordnet.
Anschließend muss die Änderung mittels des Button *Save* auf dem Server gespeichert werden.

1.6.3 Ändern von Zeilenwerten

In vielen Dialogen besteht eine Zeile nicht nur aus Zusatzeigenschaften zu anderen Objekten, sondern darüber hinaus sind zu dieser Verknüpfung auch noch Zusatzfelder zu pflegen. Diese werden als Feldleiste im Anschluss an das verknüpfte Objekt in der Liste angezeigt. Zum Beispiel können Sie im Dialog Footprint innerhalb einer Zeile nicht nur eine *Named Resource* auswählen, sondern darüber hinaus noch die weiteren Felder *Amount* und *Keep* pflegen.

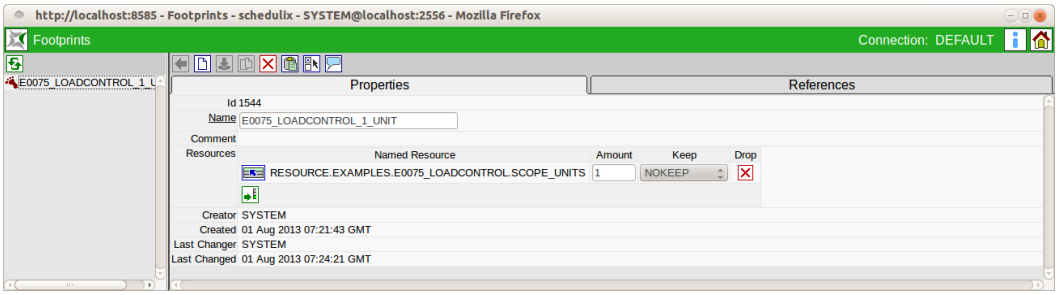


Abbildung 1.25: Listen Handling; Editieren

Das geschieht analog zum Editieren von normalen Feldern.
Anschließend müssen die Änderungen mittels des Button *Save* gespeichert werden.

1.7 Copy and Paste und Zwischenablage

Die schedulix!Web Oberfläche implementiert einen intelligenten Copy and Paste-Mechanismus und eine objektsensitive Zwischenablage. Hiermit ist es möglich, zwischen einzelnen Dialogen oder innerhalb eines Dialoges Objekte zu markieren, auszuschneiden, einzusetzen und zu löschen. Die Daten befinden sich, solange die Aktion nicht durchgeführt wurde, in der Zwischenablage des schedulix!Web Oberflächensystems und können von geeigneten anderen Dialogen benutzt werden.

Der Begriff *objektsensitiv* beschreibt die Logik, die beim Einsetzen von Objekten aus der Zwischenablage durchgeführt wird. Vor dem Einfügen wird überprüft, ob die Objekte aus der Zwischenablage vom selben Typ sind, wie sie der Dialog benötigt, in den das Objekt eingefügt werden soll. Es ist also nicht möglich, ein Objekt vom Typ A einzufügen, wenn der Dialog, in den das Objekt eingefügt werden soll, ein Objekt vom Typ B erwartet. Gibt es keine Objekte vom Typ B in der Zwischenablage, kann keine Einfügeoperation durchgeführt werden.

Dieser Mechanismus ist nicht zu verwechseln mit dem normalen Zwischenablagesystem des Windows-Betriebssystems. Die schedulix Zwischenablage ist vollkommen unabhängig von der Windows-Zwischenablage und umgekehrt. Die Daten, welche in der schedulix Zwischenablage vorgehalten werden, stehen ausschließlich schedulix Dialogen zur Verfügung und können nicht von anderen Windows-Applikationen benutzt oder ausgelesen werden.

Die Verwendung der Zwischenablage wird anhand von zwei Beispielen in den folgenden Abschnitten erläutert.

1.7.1 Verschieben von Objekten in der Hierarchie

In einer Ordnerhierarchie kann man mit der Zwischenablage Objekte in eine andere Hierarchiestufe verschieben.

Beispiel:

Im Navigationsbildschirm des Dialoges *Batches and Jobs* gibt es folgende Hierarchie: Unterhalb des ”COMPLEX_CONDITION” Ordners gibt es verschiedene Jobs und Batches. Dieser Ordner soll nun geordnet werden.
Der Batch ”COMPLEX_CONDITION” soll in den Ordner ”BATCHES” verschoben werden und anschließend sollen die Jobs ”LOAD_A”, ”LOAD_B” und ”LOAD_C” in den Ordner ”JOBS” verschoben werden.

Mittels der Zwischenablage kann dies wie folgt durchgeführt werden:

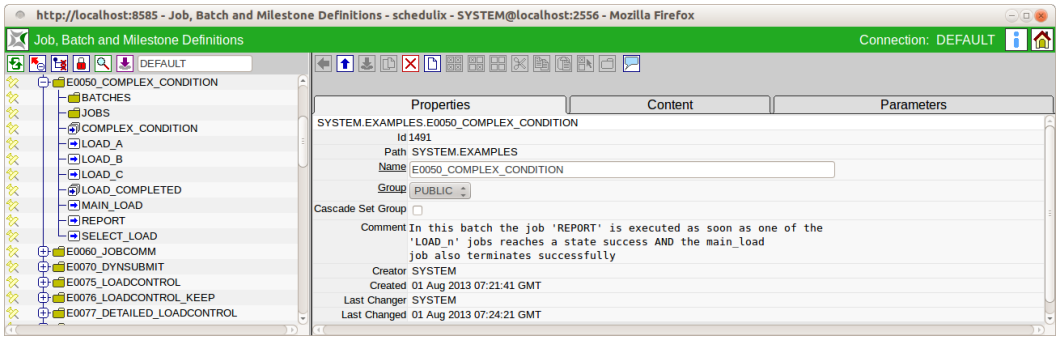


Abbildung 1.26: Objekte verschieben; Anfangszustand

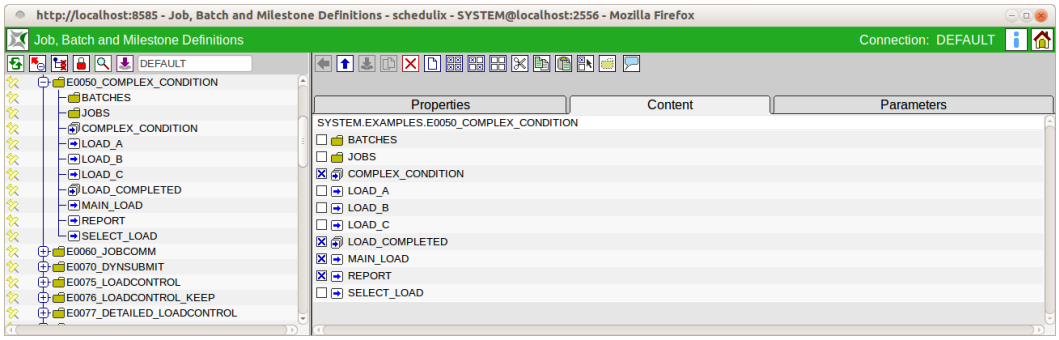


Abbildung 1.27: Objekte verschieben; Editor Content Tab

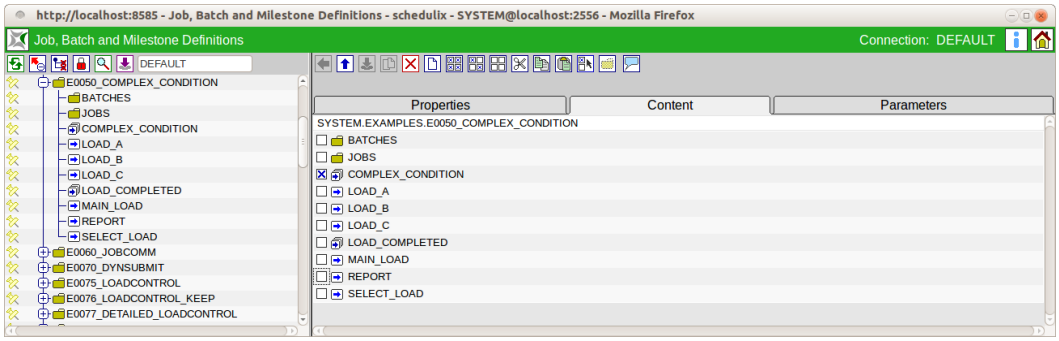


Abbildung 1.28: Objekte verschieben; Selektion

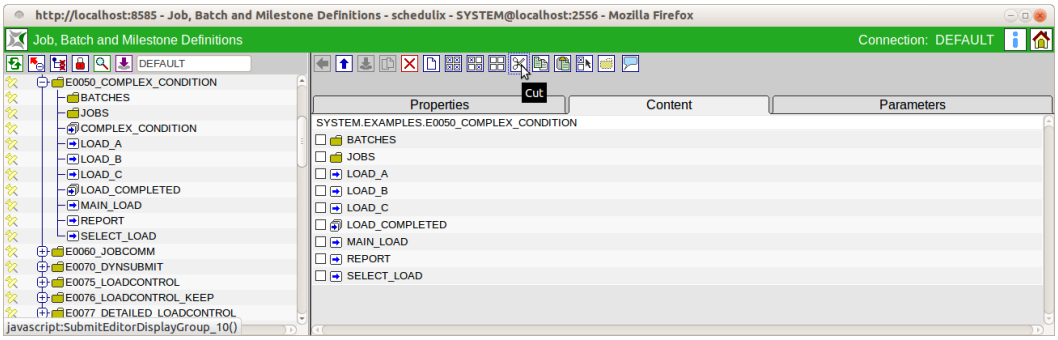


Abbildung 1.29: Objekte verschieben; Ausschneiden

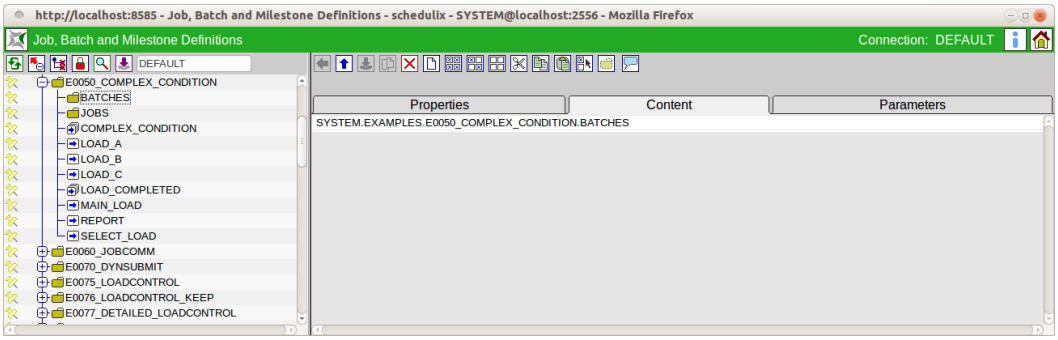


Abbildung 1.30: Objekte verschieben; Auswahl des Zielordners

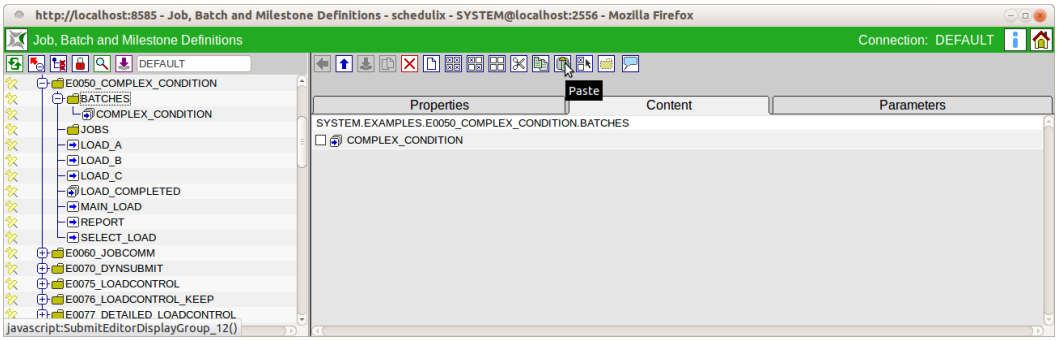


Abbildung 1.31: Objekte verschieben; Einfügen

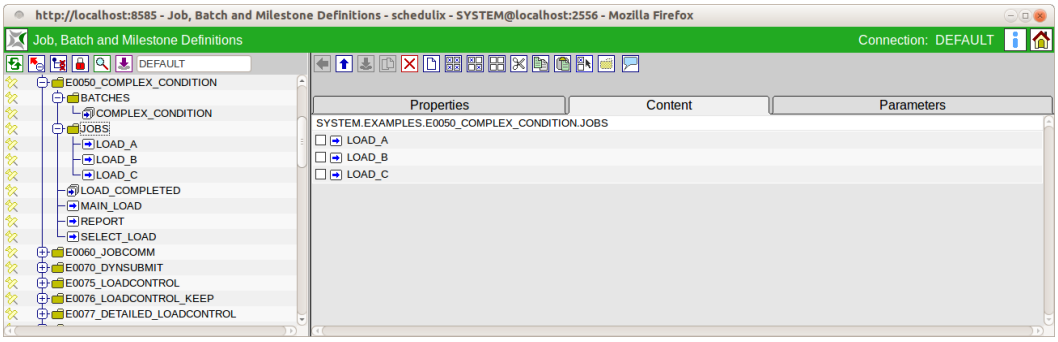


Abbildung 1.32: Objekte verschieben; Resultat

1. Selektieren der zu verschiebenden Objekte

   Zuerst wird der Batch COMPLEX_CONDITION im Navigator angewählt. Auf dem "Editor" Tab CONTENT erscheint die Liste aller Objekte, welche im Ordner COMPLEX_CONDITION liegen. (Siehe Abbildung 1.27)

   Anschließend wird der Batch durch Anklicken der Checkbox vor dem Namen und dem Icon markiert. (Siehe Abbildung 1.28)

2. Ausschneiden der selektieren Objekte

   Mit dem *Cut* Button wird der selektierte Batch ausgeschnitten und verschwindet aus der Ansicht. Das Objekt wird nun in der Zwischenablage gespeichert. Der Dialog sieht nach dem Ausschneiden aus wie in Abbildung 1.29. Es ist wichtig zu bemerken, dass das ausgeschnittene Objekt zwar nicht mehr sichtbar, jedoch immer noch in dem Folder vorhanden ist. Wird also jetzt das Fenster geschlossen, ist nichts passiert. Erst bei der späteren Paste-Operation wird die tatsächliche Verschiebung durchgeführt.

3. Auswählen des Zielordners

   Nun werden der Zieldialog und der Zielordner für die Verschiebeoperation ausgewählt. Dies kann einfach durch das Anklicken des jeweiligen Ordners im Navigationsfenster des Zieldialoges geschehen. Da die Zwischenablage fensterübergreifend funktioniert, kann sich der Zielordner jedoch auch in einem anderen Dialog befinden. In unserem Beispiel wird nun der Ordner "BATCHES" gewählt und der Content Tab erscheint. (Siehe Abbildung 1.30)

4. Einfügen der selektierten Objekte

   Um die Verschiebung abzuschließen, wird das Objekt aus der Zwischenablage eingefügt. Dies geschieht mittels des *Paste* Buttons. Das Einfügen löst nun die Verschiebeaktion auf dem Server aus. Diese ist nun abgeschlossen und kann nicht mehr widerrufen werden. Das Ergebnis sieht aus wie Abbildung 1.31.

Führt man die Aktion nun mit den oben genannten Jobs und dem Ordner ”JOBS” aus, ergibt sich das gewünschte Endresultat wie in Abbildung 1.32.

1.7.2 Verknüpfung von Objekten

In einigen Dialogen werden Objekte mit anderen Objekten oder mit Objektlisten verknüpft. Ein Beispiel ist die Definition von Abhängigkeiten. Um so eine Liste zu füllen, kann ebenfalls die Zwischenablage verwendet werden.
Um dies zu verdeutlichen und um den Ablauf einer solchen Aktion zu demonstrieren, soll die Definition von Abhängigkeiten eines Jobs durchgeführt werden.
Hier sehen Sie den Job ”END_PROCESSING”, der noch keine Abhängigkeiten hat.

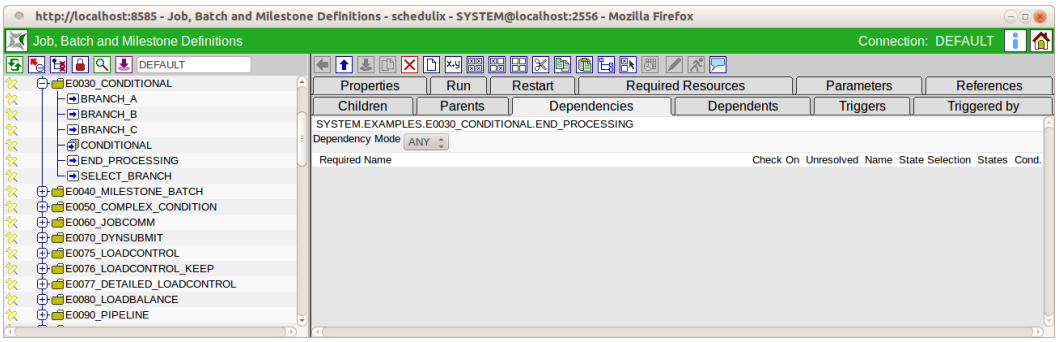


Abbildung 1.33: Verknüpfen von Objekten; Anfangssituation

Um die Ausführung dieses Jobs nun von anderen Verarbeitungsschritten abhängig zu machen, müssen an dieser Stelle andere Jobs eingefügt werden. Diese können vorher in anderen Dialogen markiert und in die Zwischenablage kopiert werden. Folgende Schritte sind durchzuführen:

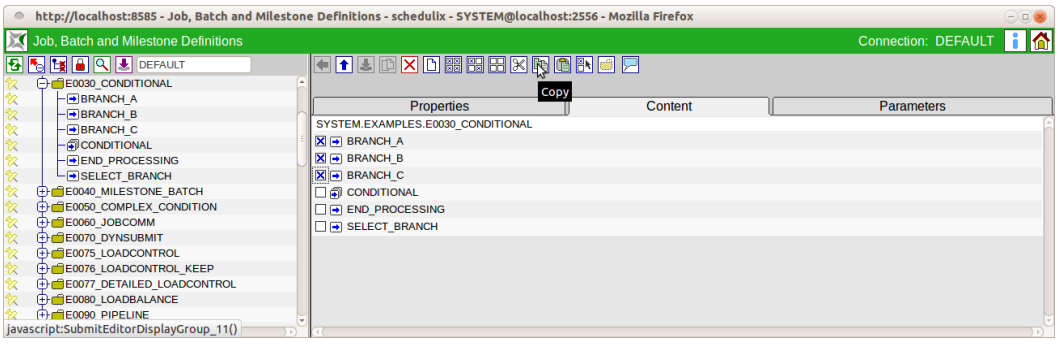


Abbildung 1.34: Verknüpfen von Objekten; Selektion

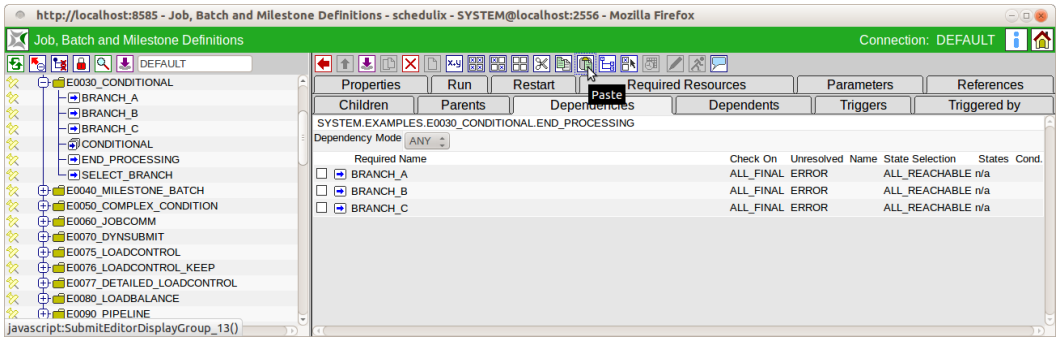


Abbildung 1.35: Verknüpfen von Objekten; Einfügen

1. Selektieren der Objekte

   Jobs können z. B. im ”Content” Tab eines Folders markiert werden. Im nachfolgenden Dialog werden die drei Verarbeitungsschritte A, B und C ausgewählt. (Siehe Abbildung 1.34)

2. Kopieren der Objekte in die Zwischenablage

   Durch Drücken des *Copy* Buttons werden die selektierten Jobs in die Zwischenablage kopiert.

3. Einfügen der Objekte

   Nun kann im Dependency Editor Fenster des Jobs ”END_PROCESSING” der *Paste* Button gedrückt werden und die zwischengespeicherten Jobs werden zur Liste hinzugefügt. Die Liste sieht anschließend aus wie in Abbildung 1.35.

4. Speichern der Änderung

   Im Gegensatz zum Verschieben von Objekten, bei dem kein zusätzliches Speichern mehr erfolgen muss, müssen Kopieraktionen in ein Editor Fenster mit dem *Save* Button gespeichert werden.

   Achtung: Der Inhalt der Zwischenablage bleibt weiterhin erhalten. Sollen die selektierten Jobs auch in anderen Dialogen hinzugefügt werden, so ist es nur nötig, das entsprechende Ziel auszuwählen und anschließend den *Paste* Button zu betätigen.

1.8 Grafik-Legende

In der folgenden Dokumentation werden verschiedene Grafiken zur Illustration von Zusammenhängen verwendet.

1.8.1 Darstellung der schedulix Objekte

In den Grafiken werden folgende Symbole und deren Bedeutungen für schedulix Objekte verwendet:

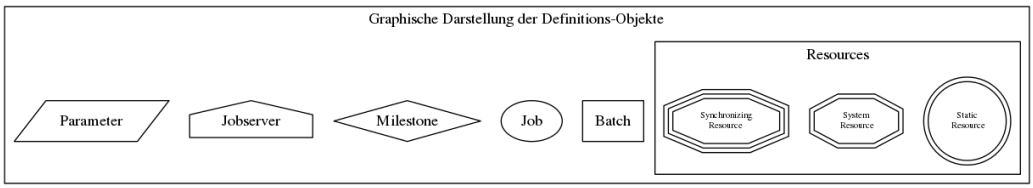


Abbildung 1.36: Legende graphische Abbildung schedulix Objekte

1.8.2 Darstellung der Parent-Child-Beziehungen in Abläufen

In den Grafiken wird folgende Darstellung der Beziehung zwischen Parent und Child verwendet:

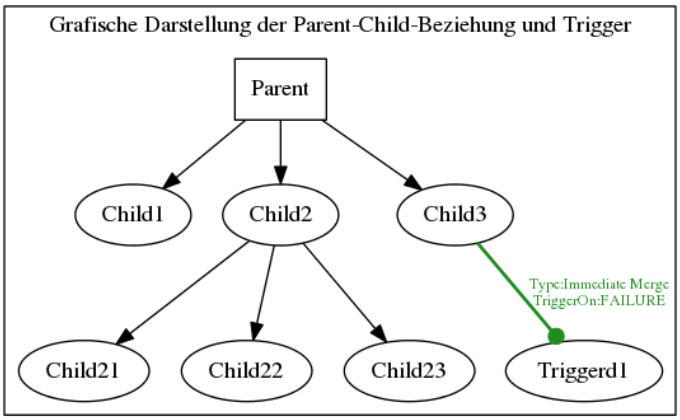


Abbildung 1.37: Legende graphische Darstellung Parent-Child-Beziehung

Die Darstellung geht normalerweise von oben nach unten, vom obersten Parent zu den jeweiligen Children. Die Richtung der Pfeile zeigt immer vom Parent zu den Children.

Mehr zu Parent-Child-Beziehungen zwischen Ablaufobjekten finden Sie im Kapitel Batches und Jobs.

1.8.3 Darstellung der Abhängigkeiten zwischen Scheduling Entities

In den Grafiken wird folgende Darstellung der Abhängigkeit zwischen einzelnen Scheduling Entities verwendet:

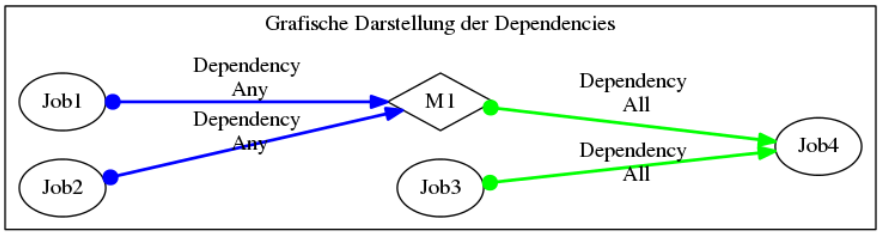


Abbildung 1.38: Legende graphische Darstellung Abhängigkeitsbeziehung

Die Darstellung der Abhängigkeiten erfolgt von rechts nach links. Das heißt, der erste Job wird ganz links angezeigt, anschließend kommt der zeitlich nächste usw. Die Richtung der Pfeile geht immer vom benötigten Scheduling Entity (required Entity) zum abhängigen Scheduling Entity (dependent Entity) an der Pfeilspitze. Sie stellt demnach die zeitliche Abfolge der Scheduling Entities dar. Also wenn A→B→C dargestellt wird, wird erst A ausgeführt anschließend darf erst B und dann C ausgeführt werden. A ist dabei das benötigte Entity von B und B ist das benötigte Entity von C. B ist demnach von A abhängig. C ist von B (und implizit dann auch von A) abhängig.
Die Farbe der Pfeile entspricht der Verknüpfung mehrerer Abhängigkeiten.
Ist die Farbe Grün, so sind alle benötigten Tasks mittels "UND" verknüpft (Dependency ALL).
Ist die Farbe Blau, so sind alle benötigten Tasks mittels "ODER" verknüpft (Dependency ANY).
Ist ein Ablaufobjekt nur von genau einem anderen Ablaufobjekt abhängig, wird der Pfeil grün dargestellt, da in diesem Fall die Verknüpfungen UND und ODER gleichwertig sind.
Mehr zu Abhängigkeitsbeziehungen zwischen Scheduling Entities finden Sie im Kapitel Batches und Jobs.

1.9 Comments

Für die meisten Objekte ist das Speichern von Comments möglich.

Comment

Mittels des *Comment*-Buttons kommt man in die Maske *Edit Comments*. Hier können beliebige Textkommentare bzgl. des Objektes eingetragen werden. Textkommentare bestehen aus einem Tag und dem Kommentartext. In der Anzeige werden Tags als Überschriften angezeigt. Wird der Tag nicht eingetragen, so wird der Kommentartext in der Anzeige an dem vorhergehenden Kommentartext angehängt. Außer dem Kommentareintrag wird keine weitere Aktion am Objekt durchgeführt.
Alternativ zu Textkommentaren kann ein Link auf eine (Intranet)Seite erfasst werden. Dazu wählt man als Typ "URL" und schreibt den URL ins Kommentarfeld. Die Links sind auch in der schedulix!Web Oberfläche aktiviert. Auf diese Weise können beliebig komplexe Dokumente mit den schedulix Objekten verknüpft werden.

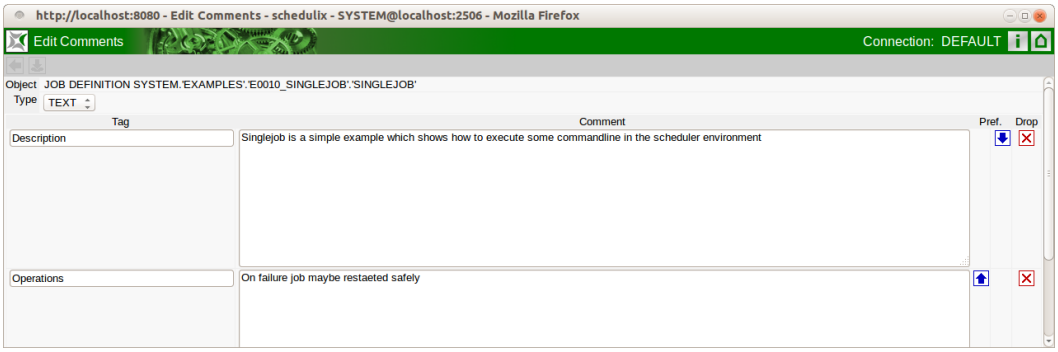


Abbildung 1.39: Erfassen von Kommentaren

1.10 Standardfelder

Für alle Objekte sind folgende Felder definiert:

Creator Der Name des Benutzers, der das Objekt angelegt hat

Created Datum und Uhrzeit, an dem dieses Objekt angelegt wurde

Last Changer Name des Benutzers, der zuletzt eine Änderung an dem Objekt vorgenommen hat

Last Changed Datum und Uhrzeit, an dem dieses Objekt zuletzt geändert wurde

2 Exit State Definitions

2.1 Bild

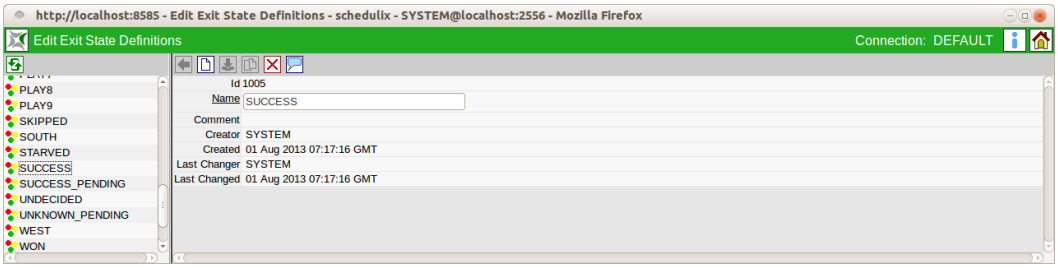


Abbildung 2.1: Exit State Definitions

2.2 Konzept

2.2.1 Kurzbeschreibung

Der Dialog Exit State Definitions dient zum Anlegen von logischen Namen für Endzustände eines Submitted Entity.

2.2.2 Ausführliche Beschreibung

Exit State Definitions sind logische Namen für Endzustände eines Submitted Entity. Diese logischen Namen besitzen keine weiteren Eigenschaften.

2.3 Editor

Dieser Tab dient der Pflege der Eigenschaften von Exit State Definitions.

Exit State Definitions dürfen nur von Benutzern, die Mitglied der Gruppe ”ADMIN” sind, editiert werden.

Für alle anderen Benutzer sind alle Eingabefelder ”read only”.

Die Felder haben folgende Bedeutung:

Id Systemeindeutige Nummer zur Identifikation des Objektes.

Name Eindeutiger Name des Exit State Definition. Dieser Name kann frei gewählt werden.

3 Exit State Mappings

3.1 Bild

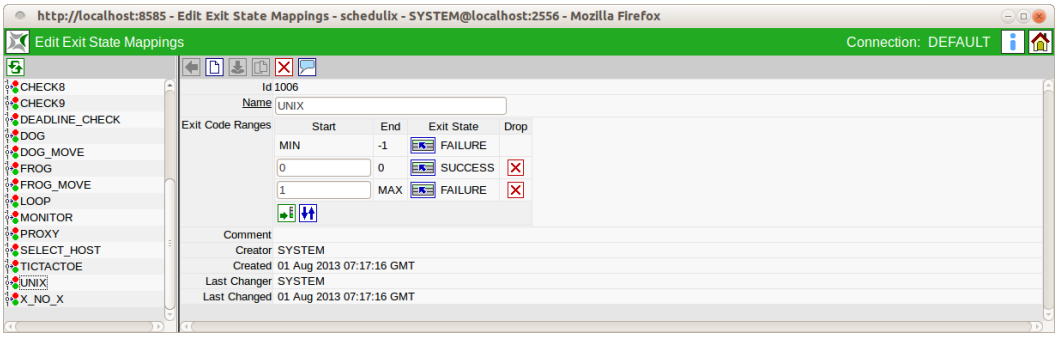


Abbildung 3.1: Exit State Mappings

3.2 Konzept

3.2.1 Kurzbeschreibung

Exit State Mappings dienen der Übersetzung von numerischen Exit Codes zu logischen Exit States.

3.2.2 Ausführliche Beschreibung

Mittels Exit State Mappings lassen sich numerische Wertebereiche von Exit States zu logischen Exit States zuordnen. Jedes Programm, das ausgeführt wird, liefert am Ende einen numerischen Wert an das aufrufende Programm zurück. Unter UNIX-Systemen bedeutet dabei ein Wert von 0 normalerweise, dass das Programm ohne Fehler beendet wurde (SUCCESS). Werte ungleich 0 weisen auf einen Fehler hin (FAILURE). Es sind jedoch auch andere Werte mit anderen Bedeutungen denkbar. Um diese logische Bedeutung innerhalb von schedulix besser zu verstehen, lassen sich mit Exit State Mappings Wertebereiche jeweils einem logischen Namen zuordnen. Die Werte reichen dabei von -2^{31} bis $2^{31} - 1$.

3.3 Editor

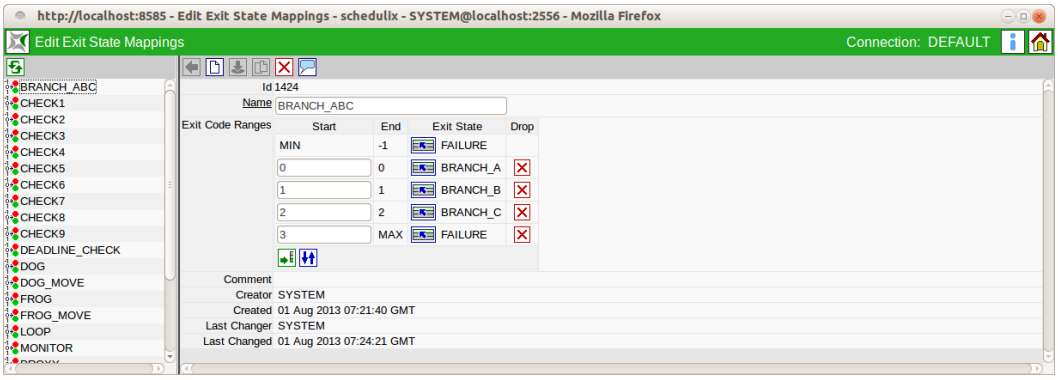


Abbildung 3.2: Exit State Mapping Editor

Im Editor werden die Zuordnungen der numerischen Werte zu den Namen vorgenommen.
Exit State Mappings dürfen nur von Benutzern, die Mitglied der Gruppe "ADMIN" sind, editiert werden. Für alle anderen Benutzer sind alle Eingabefelder "read only".
Die Beschreibung aller Standard-Buttons finden Sie im Kapitel 1.4.3.1 Standard-Buttons.
Die Buttons der obigen Liste haben folgende Bedeutung:

Neuen Wertebereich einfügen

Dieser Button erweitert die Tabelle der Exit Code Ranges um eine weitere Zeile. Die Zeile wird am Ende eingefügt und ermöglicht die Eingabe eines neuen Startwertes. Der neue Endwert wird beim Speichern oder Sortieren automatisch ermittelt.

Liste neu sortieren

Eine unsortierte Liste, die durch das Hinzufügen neuer Intervalle entsteht, kann mit diesem Button in die richtige Reihenfolge gebracht werden.

Exit State Auswahl

Über diesen Button kann ein vorher definierter Exit State aus einer Liste gewählt werden.

 **Löschen eines Wertebereiches**

Dieser Button löscht einen Wertebereich zusammen mit dem Exit State aus der Liste. Der Startwert des nachfolgenden Intervalls erhält den Startwert des gelöschten Intervalls, d. h. das nachfolgende wird automatisch vergrößert.
Die Felder haben folgende Bedeutung:

Id Systemeindeutige Nummer zur Identifikation des Objektes

Name Eindeutiger Name des Exit State Mappings. Dieser Name kann frei gewählt werden.

Exit Code Ranges Die Zuordnung der jeweiligen Wertebereiche wird in einer Tabelle dargestellt. In jeder Zeile kann einem Bereich ein Exit State zugeordnet werden. Die erste Zeile enthält den Startwert -2^{31}, die letzte Zeile hat als Endwert $2^{31} - 1$.

4 Exit State Profiles

4.1 Bild

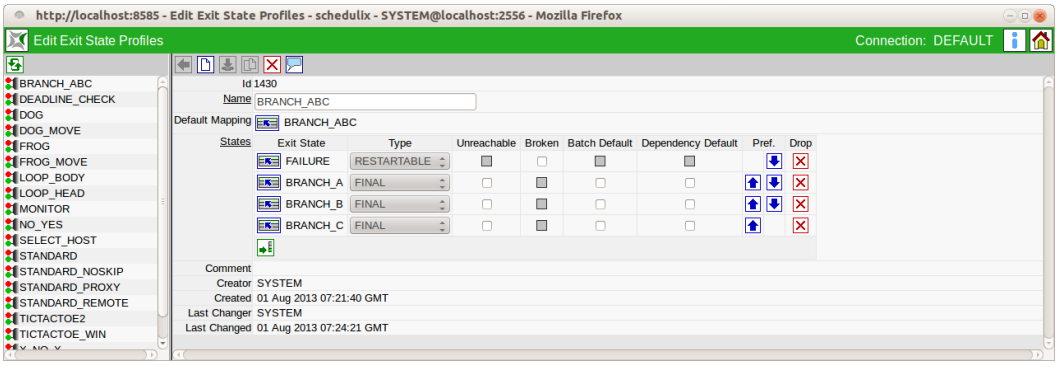


Abbildung 4.1: Exit State Profiles

4.2 Konzept

4.2.1 Kurzbeschreibung

Ein Exit State Profile beschreibt, welche Exit States von einem Submitted Entity erreicht werden können. Außerdem gibt das Exit State Profile an, ob ein Submitted Entity seinen Exit State noch wechseln kann oder ob der Exit State endgültig (FINAL) ist.

4.2.2 Ausführliche Beschreibung

Jedes Submitted Entity kann sich mit verschiedenen Exit States beenden. Die Menge aller gültigen Exit States eines Submitted Entities wird mit einem Exit State Profile beschrieben. In dieser Beschreibung wird auch die Präferenz eines Exit States festgelegt, die innerhalb einer hierarchischen Ablaufstruktur wichtig ist, um für ein übergeordnetes Element den korrekten Exit State zu bestimmen. Dabei gewinnt der Exit State mit der höchsten Präferenz.

4.3 Editor

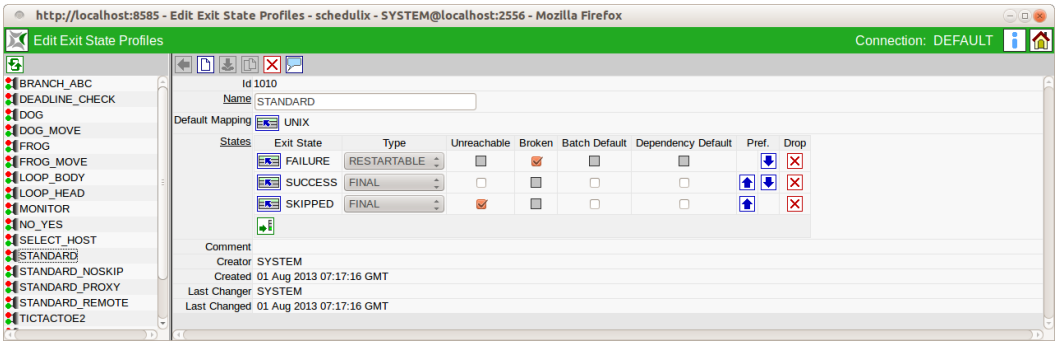


Abbildung 4.2: Exit State Profile Editor

Im Editor lassen sich die zu einem Profile gehörenden Exit States definieren und mit Eigenschaften (*Type*, *Unreachable*, *Broken*, *Preference*) versehen.
Exit State Profile dürfen nur von Benutzern, welche Mitglied der Gruppe "ADMIN" sind, editiert werden. Für alle anderen Benutzer sind alle Eingabefelder "read only".
Die obigen Felder haben folgende Bedeutung:

Name Über das Feld *Name* wird einem Exit State Profile ein eindeutiger Name zugewiesen.

Default Mapping Mit dem *Chooser* Button lässt sich aus einer Liste ein Exit State Mapping auswählen, das standardmäßig in Zusammenhang mit dem Exit State Profile verwendet wird, solange kein anderes Mapping in der Job Definition angegeben wird. Ein Exit State Mapping dient der Übersetzung von numerischen Exit Codes zu Exit States.
Für jeden dieser Exit States muss im Profile ein Eintrag vorhanden sein. Jedoch nicht alle Exit States eines Exit State Profiles müssen über das Mapping erreichbar sein.

Type Jedem Exit State kann einer von drei Typen zugewiesen werden: FINAL, RESTARTABLE und PENDING.
Hat ein Ablaufobjekt einen Final State erreicht, kann der Exit State nicht mehr geändert werden. Das Ergebnis ist endgültig. Abhängigkeiten zwischen Jobs und Batches können nur auf solchen FINAL Exit States aufbauen.
Soll ein Ablaufobjekt nach seiner Ausführung nochmals gestartet werden können, z. B. weil ein Fehler aufgetreten war, so muss der Exit State vom Typ RESTARTABLE sein.

PENDING beschreibt einen State-Typ, der es ermöglicht, den endgültigen State von außerhalb über das API zu setzen. Ein Ablaufobjekt, das in einem PENDING State steht, kann weder neu gestartet werden, noch werden dadurch irgendwelche Abhängigkeiten erfüllt.
Ein Beispiel für die Anwendung von PENDING:
Ein Job sendet einem Mitarbeiter eine E-Mail und bittet um Freigabe eines Ergebnisses. Nach dem Versenden der E-Mail beendet sich der Job mit einem PENDING State. Der Mitarbeiter kann nun den State manuell auf FINAL setzen. Alternativ kann ein Prozess, der die Antwort-E-Mail auswertet, die Statusänderung vornehmen. Erst nachdem der State FINAL ist, können abhängige Ablaufobjekte anlaufen.

Unreachable Maximal ein Exit State in der Liste darf als Unreachable markiert sein. Dieser Exit State wird gesetzt, wenn ein Job innerhalb eines Ablaufs nicht mehr zur Ausführung kommen kann, weil die Abhängigkeiten nicht mehr erfüllt werden können.
Der Unreachable State wird nicht angenommen, wenn ein oder mehrere Vorgänger gecancelt wurden. Gerade weil hier ein manueller Eingriff vorgenommen wurde, müssen auch die Folgen des Eingriffs manuell behandelt werden.
Der Unreachable State muss vom Typ "FINAL" sein.

Broken Maximal ein Exit State in der Liste darf als Broken State markiert sein. Dieser State wird gesetzt, wenn ein Job aufgrund eines Fehlers in ein Error State versetzt wurde. Ein solcher Fehler tritt z. B. dann auf, wenn das Run Program nicht gestartet werden kann. Der Error State muss ein RESTARTABLE State sein. Durch Verwendung des Broken Flags kann mittels eines Triggers automatisch auf solche Fehlersituationen reagiert werden.

Batch Default Ein finaler Exit State in der Liste darf als Batch Default gekennzeichnet werden. Dieser State wird gesetzt, wenn ein Batch oder Milestone mit diesem Profil keine Children hat, also auch keinen definierten Exit State hat. Ist kein Batch Default gesetzt, so wird der State mit der niedrigsten Präferenz verwendet, der (falls vorhanden) nicht als UNREACHABLE State gesetzt wurde.

Dependency Default Ein oder mehrere finale Exit States können als Dependency Default gekennzeichnet werden. Wird eine Abhängigkeit zwischen Batches bzw. Jobs oder Milestones mit State Selection DEFAULT angelegt, so erfüllen die als Dependency Default markierten Exit States des Profiles die Abhängigkeitsbedingung bzgl. des Exit States.

Preference  Über die *Preference* Buttons lassen sich die Präferenzen der einzelnen Exit States festlegen. Dazu werden die Zeilen nach Betätigen der Buttons

nach oben oder unten verschoben. Eine höhere Position bedeutet eine höhere Präferenz. Bei der Ermittlung des resultierenden Exit State eines Submitted Entities, wird aus der Liste der Exit States der Children und des eigenen Exit States den Exit State mit höchster Präferenz gewählt. Dabei wird ein Exit State nur dann berücksichtigt, wenn der Exit State im Exit State Profile des Parents vorhanden ist oder der State mittels einer Exit State Translation nach einem Exit State des Parents übersetzt wird.
Beispiel: Ein Batch hat drei untergeordnete Child Jobs. Der Batch soll genau dann einen Fehler anzeigen, wenn mindestens einer seiner Children einen Fehler meldet. Das bedeutet, der State FAILURE muss in der Reihenfolge vor SUCCESS kommen. In der Regel wird der SUCCESS State die niedrigste Priorität haben, Warnungen haben höhere Priorität und Fehler-States die höchste Priorität.
Wird ein leerer Batch (ein Batch ohne Children) FINAL, so gibt es keinen Exit State, aus dem der FINAL Exit State des Batches ermittelt werden kann. In diesem Fall wird der Exit des Exit State Profiles verwendet, welcher die niedrigste Präferenz hat. Dies ist im allgemeinen SUCCESS. Dabei wird versucht einen FINAL State, welcher nicht der UNREACHABLE State ist zu verwenden, falls ein solcher existiert.

5 Exit State Translations

5.1 Bild

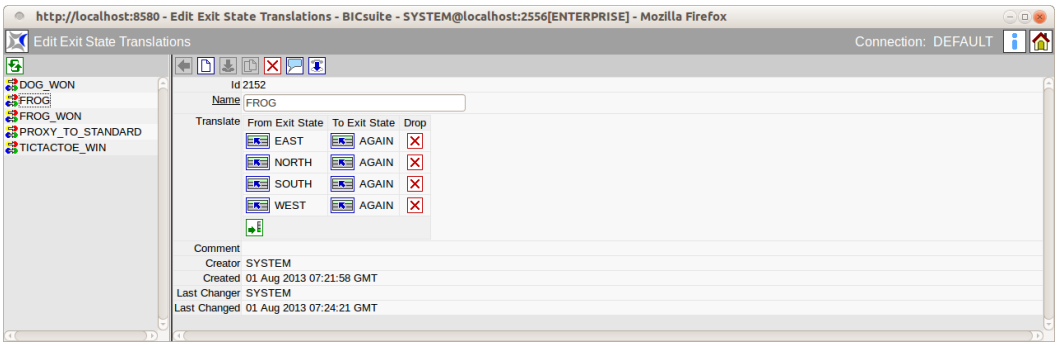


Abbildung 5.1: Exit State Translations

5.2 Konzept

5.2.1 Kurzbeschreibung

Exit State Translations werden benötigt, um den Exit State eines Childs zu einem Exit State des Parents zu übersetzen.

5.2.2 Ausführliche Beschreibung

Exit State Translations dienen der Übersetzung von Exit States eines Childs zu Exit States des Parents. Wenn keine Übersetzung vorhanden ist, wird die Identität genutzt oder der Exit State des Childs vom Parent ignoriert. Ist die Übersetzung vorhanden, müssen alle Exit States implizit oder explizit übersetzt werden können.

Verwendet zum Beispiel ein Child ein Exit State Profile mit den Exit State SUCCESS, WARNING, und FAILURE, und das Exit State Profile des Parents kennt nur SUCCESS und FAILURE, können die finalen Exit States SUCCESS und WARNING nach dem Exit State SUCCESS des Parents übersetzt werden. Der RESTARTABLE Exit State FAILURE wird nach Exit State FAILURE des Parents übersetzt.

5.3 Editor

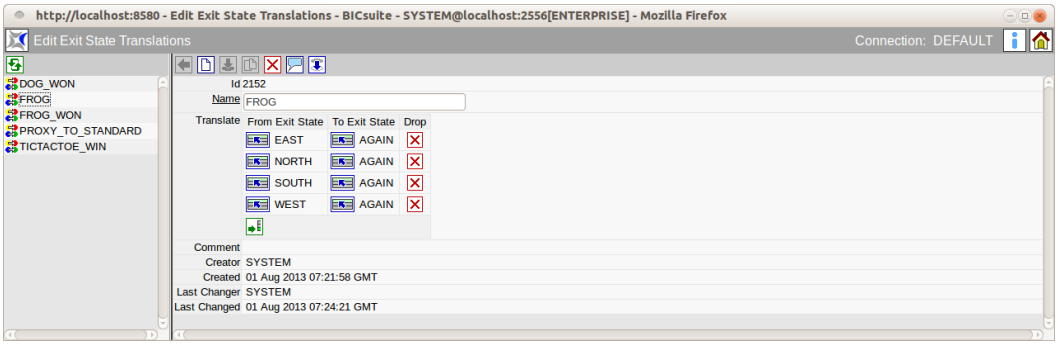


Abbildung 5.2: Exit State Translation Editor

Im Editor lassen sich die Exit State Translations definieren und anpassen.
Exit State Translations dürfen nur von Benutzern, welche Mitglied der Gruppe "ADMIN" oder einer anderen Gruppe mit dem "manage exit state translation" Privileg sind, editiert werden. Für alle anderen Benutzer sind alle Eingabefelder "read only".
Die obigen Felder haben folgende Bedeutung:

Id Die Id enthält eine systemweit eindeutige Nummer.

Name Über das Feld *Name* wird der Exit State Translation ein eindeutiger Name zugewiesen.

Translate From/To Exit State In jeder Zeile wird angegeben, welcher State eines Childs nach welchem State des Parents übersetzt wird.

6 Resource State Definitions

6.1 Bild

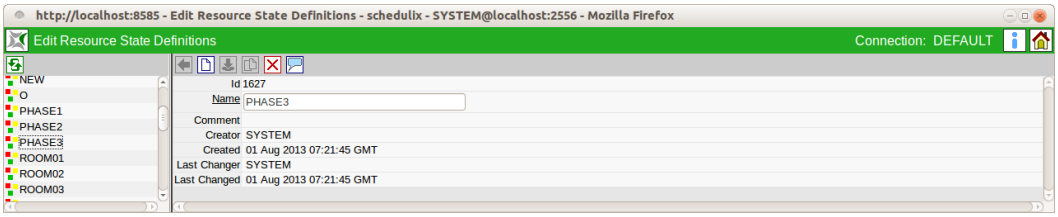


Abbildung 6.1: Resource State Definitions

6.2 Konzept

6.2.1 Kurzbeschreibung

Die Resource State Definitions dienen dem Anlegen von Namen für Resource States.

6.2.2 Ausführliche Beschreibung

Resource State Definitions sind logische Namen für Resource States. Jede Synchronizing Resource kann verschiedene States annehmen, die beim Beenden eines Jobs gesetzt werden können.

6.3 Editor

Dieser Tab dient der Pflege der Eigenschaften von Resource State Definitions.
Resource State Definitions dürfen nur von Benutzern, welche Mitglied der Gruppe ”ADMIN” sind, editiert werden. Für alle anderen Benutzer sind alle Eingabefelder ”read only”.
Die Felder haben folgende Bedeutung:

Id Systemeindeutige Nummer zur Identifikation des Objektes.

Name Eindeutiger Name der Resource State Definition. Dieser Name kann frei gewählt werden.

7 Resource State Profiles

7.1 Bild

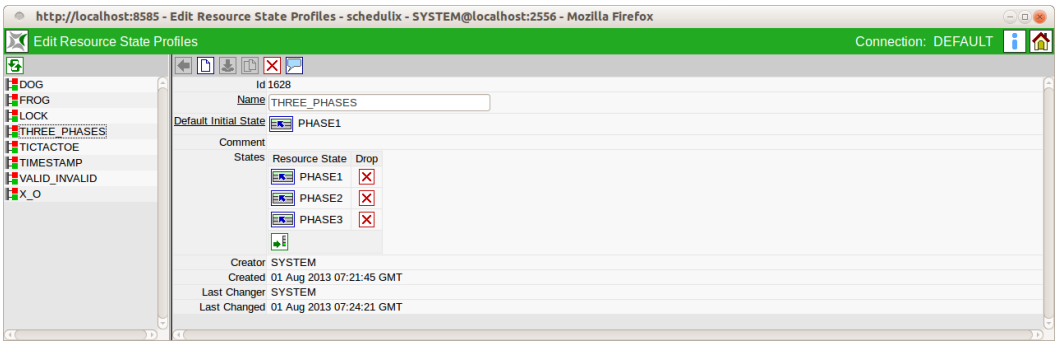


Abbildung 7.1: Resource State Profiles

7.2 Konzept

7.2.1 Kurzbeschreibung

Ein Resource State Profile beschreibt, welchen Status eine Resource annehmen kann.

7.2.2 Ausführliche Beschreibung

Jede Synchronizing Resource kann verschiedene Resource States annehmen. Die Menge aller gültigen Resource States einer Resource wird mit einem Resource State Profile beschrieben. In dieser Beschreibung wird auch der initiale Status einer Resource festgelegt. Der initiale Status muss dabei nicht unbedingt in der Liste der States vorkommen.

7.3 Editor

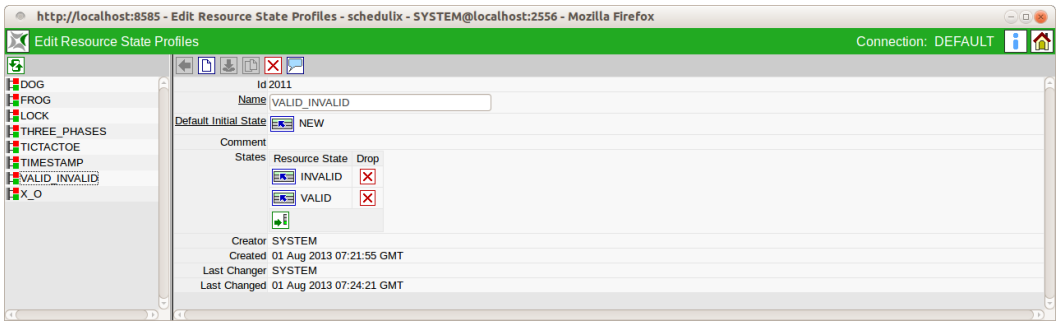


Abbildung 7.2: Resource State Profile Editor

Im Editor lassen sich die zu einem Resource State Profile gehörenden Resource States definieren.
Resource State Profile dürfen nur von Benutzern, welche Mitglied der Gruppe "ADMIN" sind, editiert werden. Für alle anderen Benutzer sind alle Eingabefelder "read only".
Die obigen Felder haben folgende Bedeutung:

Name Über das Feld *Name* wird dem Resource State Profile ein eindeutiger Name zugewiesen.

Default Initial State Das Feld *Default Initial State* definiert den initialen State der Resource. Dieser Resource State muss nicht in der Liste gültiger Resource States vorhanden sein.

Resource State In der Tabelle *States* stehen in der Spalte *Resource State* die gültigen Resource States.

7.3.1 Beispiel

Eine Datenbanktabelle eines Datamarts wird als Synchronizing Resource mit State Modell im Scheduling System dargestellt. Das Zustandsdiagramm sieht wie im Bild 7.3 aus. Die Definition eines geeigneten Resource State Profiles wird im obigen Screenshot (Abbildung 7.2) dargestellt.

Editor

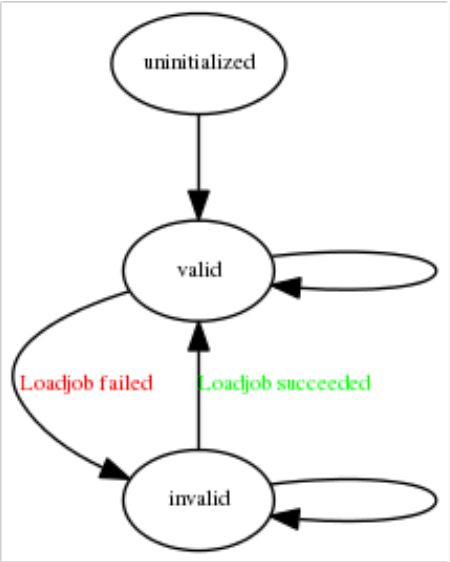


Abbildung 7.3: Zustandsdiagramm einer Resource

8 Resource State Mappings

8.1 Bild

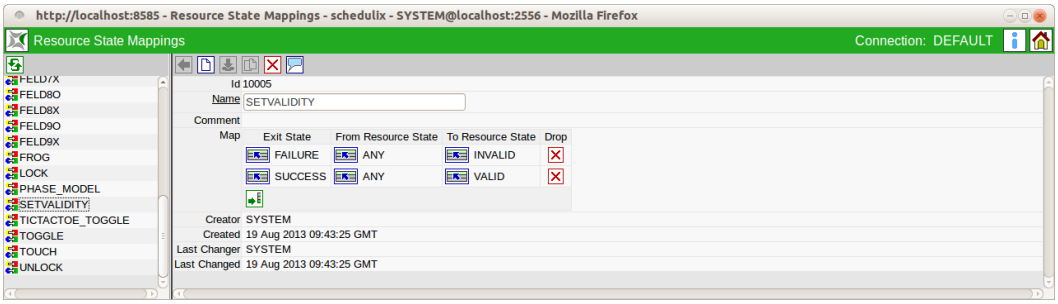


Abbildung 8.1: Resource State Mappings

8.2 Konzept

8.2.1 Kurzbeschreibung

Resource State Mappings beschreiben den Statuswechsel einer Resource bei bestimmten Exit States.

8.2.2 Ausführliche Beschreibung

Resources können ihren Resource State nach der Beendigung eines Jobs in Abhängigkeit dessen Exit State verändern. Resource State Mappings beschreiben diese Statusübergänge. Jeweils ein Exit State und ein Resource State bestimmen den neuen State der Resource.
Beispiel: eine Resource ”TABELLE” kann den State ”VALID” oder ”INVALID” annehmen. Wird der zur Tabelle gehörende Ladeprozess erfolgreich beendet (SUCCESS), soll der Resource State ”VALID” gesetzt werden. Bei einer fehlerhaften Ausführung (FAILURE) soll die Tabelle als ”INVALID” gekennzeichnet werden. Die entsprechende Definition des Resource State Mappings würde wie im Bild 8.2 aussehen.
Existiert für die Kombination ’Exit State’ und ’From Resource State’ kein Mapping, wird der Status der Resource auch nicht verändert. So ist es nicht notwendig, ein Mapping von SUCCESS/VALID nach VALID zu definieren.

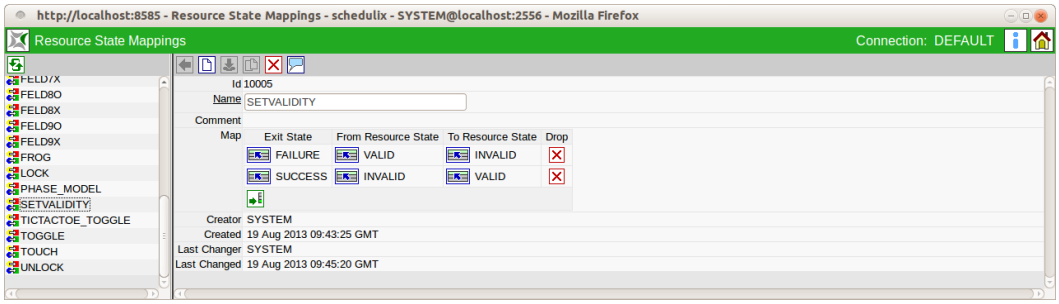


Abbildung 8.2: Resource State Mapping Beispiel

Soll der Statusübergang unabhängig von einem Resource State erfolgen, soll in der Zeile ”From Resource State”, der Wert ”ANY” gewählt werden.
Beim Umsetzen des Resource States wird der Zeitpunkt des Umsetzens gespeichert. Bei Resource Anforderungen kann auf diese Zeit Bezug genommen werden, sodass auch das Setzen von ”VALID” nach ”VALID” sinnvoll sein kann.

8.3 Editor

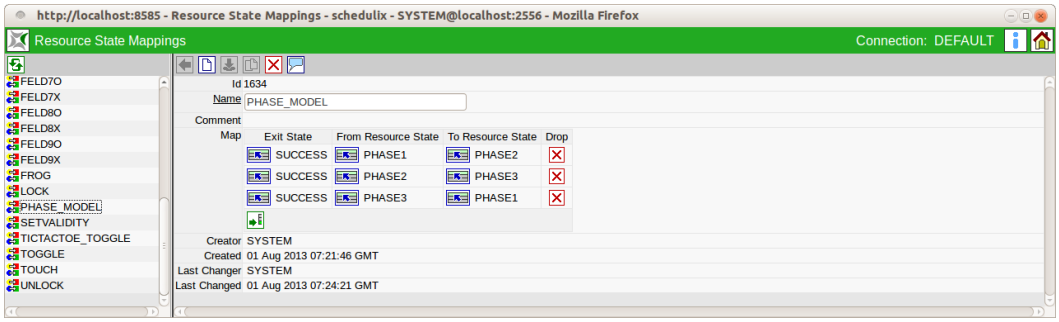


Abbildung 8.3: Resource State Mapping Editor

Im Editor werden die Zuordnungen der Exit States zu den Resource-Statusübergängen vorgenommen.
Resource State Mappings dürfen nur von Benutzern, welche Mitglied der Gruppe ”ADMIN” sind, editiert werden. Für alle anderen Benutzer sind alle Eingabefelder ”read only”.
Die Felder haben folgende Bedeutung:

Id Systemeindeutige Nummer zur Identifikation des Objektes.

Name Eindeutiger Name des Resource State Mappings. Dieser Name kann frei gewählt werden.

Exit State Der Exit State des Jobs steht im Feld *Exit State*.

From Resource State Im Feld *From Resource State* steht entweder ein bestimmter Resource State oder ANY. Falls der aktuelle State in Kombination mit dem Exit State explizit in der Tabelle genannt wird, wird beim Umsetzen des State diese Regel für die Bestimmung des neuen State herangezogen. Ansonsten wird die ANY-Regel, falls vorhanden, benutzt.

To Resource State Das Feld *To Resource State* ist der resultierende State der Resource.

9 Named Resources

9.1 Bild

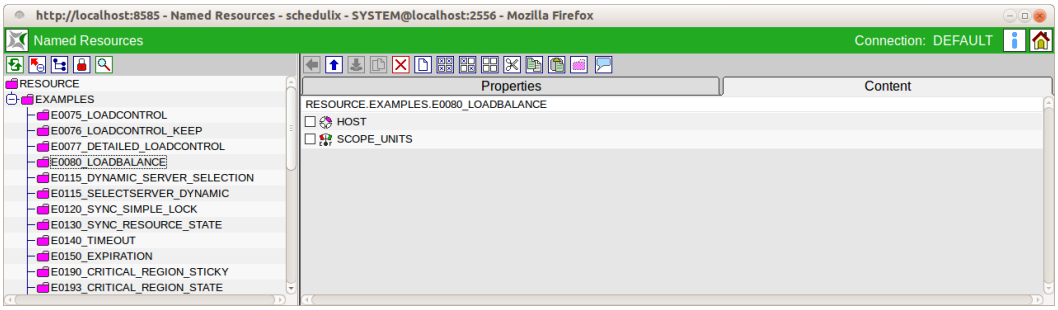


Abbildung 9.1: Named Resources

9.2 Konzept

9.2.1 Kurzbeschreibung

Dieser Bildschirm dient zum Erstellen und Verwalten von Named Resources. Eine Named Resource ist die Definition einer Klasse von Resources.

9.2.2 Ausführliche Beschreibung

Eine Named Resource definiert eine Klasse von Resources. Named Resources können innerhalb eines Scopes oder Folders bzw. eines Submitted Entities als Resources instanziiert werden.
Named Resources gehören einer Gruppe an. Die Benutzung (d.h. Instanziierung und Anforderung), sowie das Editieren oder Löschen der Named Resource ist in erster Linie Benutzern der Gruppe vorbehalten. Diese Rechte können anderen Gruppen zugeteilt werden.
Named Resources werden innerhalb einer Hierarchie von Kategorien gespeichert. Kategorien sind beliebig ineinander schachtelbar. Named Resources sind nicht schachtelbar. Die Kategorisierung dient ausschließlich der Verbesserung der Übersichtlichkeit bei der Verwaltung von Named Resources und hat darüber hinaus keinen Einfluss auf die Funktion des Systems.

9.3 Editor

Im Editor Menü werden Named Resources und Kategorien gepflegt. Wird im Navigator ein Eintrag gewählt, so erscheinen hier die Details dieser Named Resource oder Kategorie. Des Weiteren können neue Named Resources und Kategorien erstellt werden.

9.3.1 Tab Properties bei Kategorien

Dieser Tab dient der Pflege von Kategorieeigenschaften. Er sieht folgendermaßen aus:

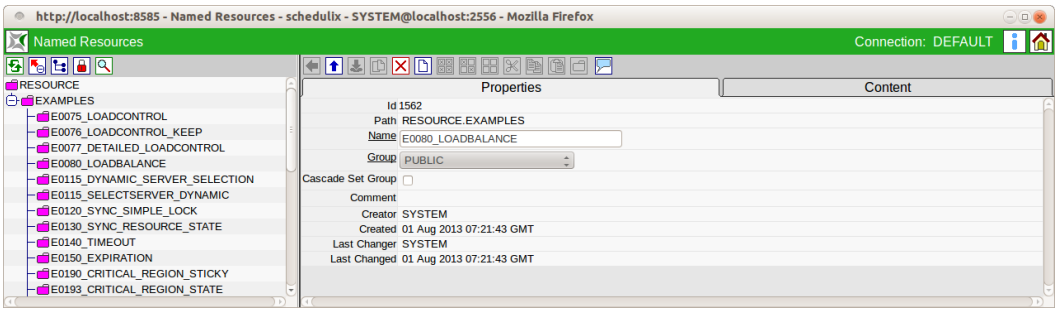


Abbildung 9.2: Kategorien; Properties Tab

Der ”Properties” Tab für eine Kategorie und Named Resources hat folgende Felder:

ID Die Id dient zum eindeutigen Identifizieren der Kategorie oder Named Resource Definition. Die Id wird automatisch vom System vergeben und ist systemweit eindeutig.

Path (Pfad) Der Path stellt den kompletten Pfad (Hierarchie) des aktuellen Eintrages dar. Die einzelnen Hierarchiestufen werden mittels eines Punktes getrennt.

Name Der Name der Kategorie oder der Named Resource kann frei gewählt werden. Er muss jedoch innerhalb der übergeordneten Kategorie eindeutig sein.

Group Die Gruppe, der die Kategorie zugeordnet ist, kann aus der ”Drop Down” Liste ausgewählt werden.

Cascade Set Group Wird dieses Feld markiert, wird die Gruppe in allen Named Resources und Kategorien unterhalb dieser Kategorie ebenfalls gesetzt.

Comment Dient einer näheren Erläuterung des Objektes

9.3.2 Tab Content

Der Tab ”Content” wird nur bei Kategorien angezeigt und sieht folgendermaßen aus:

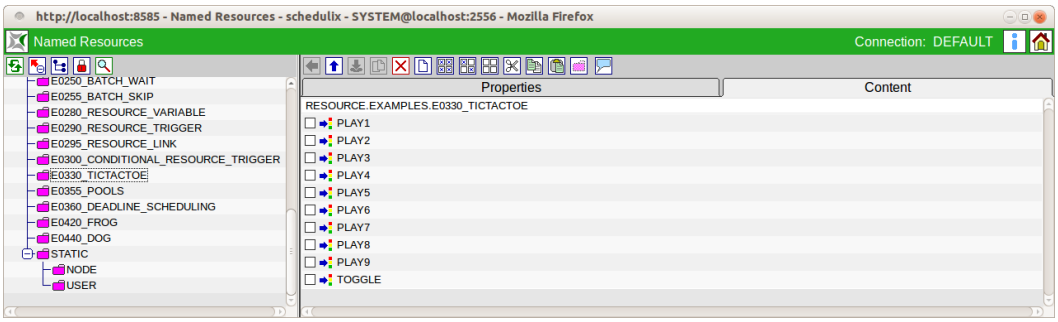


Abbildung 9.3: Kategorien; Content Tab

Der Tab enthält eine Liste von allen Named Resources, welche sich in der gewählten Kategorie befinden. Folgende Werte werden innerhalb der Liste angezeigt:

Name der Named Resource Hier erscheint der Name der Named Resource. In diesem Dialog kann der Name der Named Resource angewählt werden, was direkt zur Selektion und zur Anzeige der ”Detail” Daten dieser Named Resource führt. In diesem Tab ist es möglich, mittels der normalen Cut-, Copy- and Paste-Methoden, Resources zu verschieben.

9.3.3 Tab Properties bei Named Resource Definitionen

Handelt es sich um eine Named Resource Definition sieht der Tab ”Properties” folgendermaßen aus:

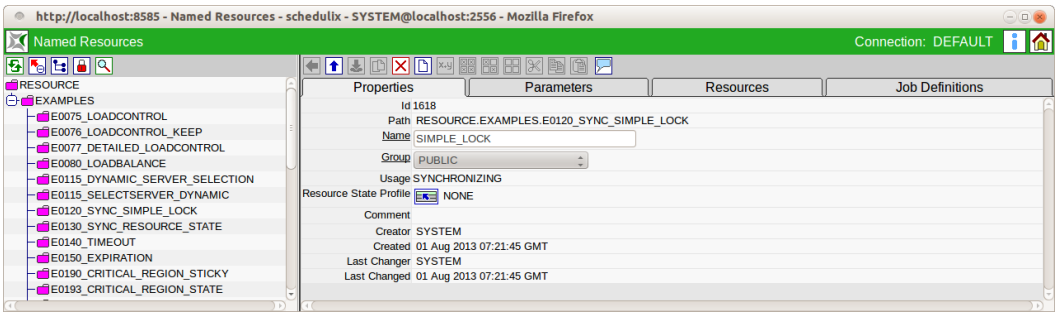


Abbildung 9.4: Named Resources; Properties Tab

Zusätzlich zu den o.g. Feldern müssen bei der Definition einer Named Resource weitere Felder ausgefüllt werden.

Usage Die *Usage* gibt an, um welche Art einer Named Resource es sich handelt. Liste der Möglichkeiten für Usage Parameter:

1. *Category*

   Kategorien verhalten sich wie Folder und können benutzt werden, um die Named Resources in eine übersichtliche Hierarchie einzuordnen.

2. *Static*

   Static Resources beschreiben Ablaufumgebungen, die etwa durch installierte Software-Pakete, Benutzerumgebungen geboten werden. Eine Instanz dieser Named Resource muss vorhanden und verfügbar sein, damit ein Submitted Entity, welches diese Resource als Voraussetzung benötigt, starten kann.

3. *System*

   Die Named Resource stellt eine Resource dar, welche einen Systemparameter abbildet. Dies kann zum Beispiel eine CPU Einheit, eine Hauptspeichereinheit oder eine Einheit von Plattenplatz sein. Diese Resource muss beim Gebrauch zusätzlich quantifiziert werden, das heißt der Ersteller eines Jobs muss die Anforderungen, die dieses Objekt an System Resources stellt, qualifizieren und quantifizieren. Bei der Definition eines Jobs muss also zum Beispiel angegeben werden, dass 3 CPU Einheiten und 3 Einheiten à 512 MB Hauptspeicher und 10 Einheiten à 1 GB Plattenplatz benötigt werden.

4. *Synchronizing*

   Synchronizing Resources sind spezielle Resources, die dazu dienen parallel laufende Jobs zu synchronisieren. Ob eine Resource-Anforderung erfüllbar und die Resource belegbar ist, kann dabei vom gegenwärtigen Status der Resource, der letzten Status-Update-Zeit und der konkurrierenden Allocation der Resource durch andere Ablaufobjekte abhängig gemacht werden. Wie bei System Resources ist eine Quantifizierung der Anforderung möglich.

Resource State Profile Einer Synchronizing Resource kann ein Resource State Profile zugeordnet werden, welcher den Status, den die Synchronizing Resource annehmen kann, definiert. Wenn kein Resource State Profile angegeben ist, können Status- und Zeitstempelbezogene Anforderungen nicht definiert werden. Weitere Informationen finden Sie in den Kapiteln 6 und 7.

9.3.4 Tab Parameters

Im Tab ”Parameters” können zusätzliche Informationen, die von Jobs oder Resource Trigger ausgewertet werden können, zu einer Resource gespeichert werden.
Der Tab ”Parameters” sieht aus wie folgt:
Durch Anklicken des Parameternamens kommt man in den Tab ”Parameter Details”.

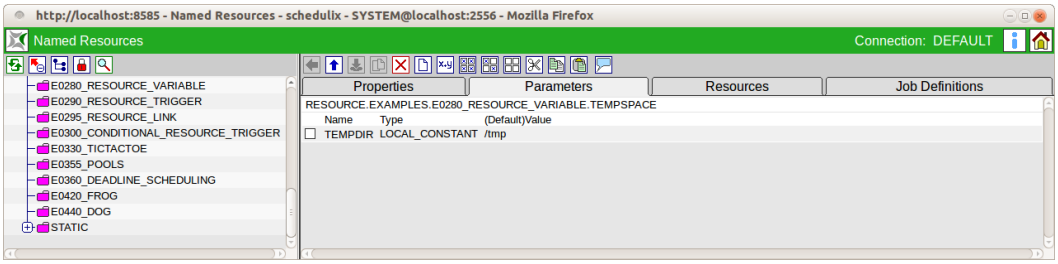


Abbildung 9.5: Named Resources; Parameter Tab

9.3.4.1 Tab Parameter Details

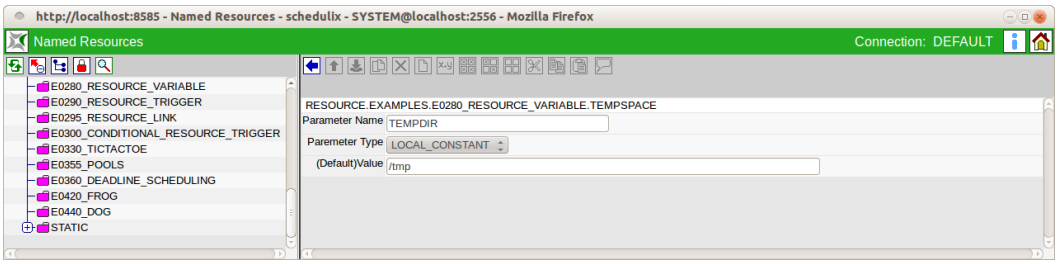


Abbildung 9.6: Named Resources; Parameter Details

Die Felder des Tabs "Parameter Details" haben folgende Bedeutung:

Parameter Name Der Name des Parameters

Parameter Type Beim Type handelt es sich um die Art des Parameters.
Es gibt folgende Möglichkeiten:

- Constant: Der Constant hat einen festen Wert und gilt für alle Resources.
- Local Constant: Der Local Constant hat einen festen Wert, der von Resource zu Resource abweichen kann.

Default Value Beim Default Value unterscheiden wir zwischen Constants und Local Constants. Bei Constants ist er der Wert des Parameters und bei Local Constants der Default-Wert.

9.3.4.2 Standard Parameters

Ebenso wie es Standard Parameters für Jobs und Batches gibt, stehen auch für Resources Standard Parameters zur Verfügung.

Um sie referenzieren zu können, muss, analog zu der Situation bei Batches und Jobs, ein kleiner Trick verwendet werden. Soll etwa der Inhalt der Standard Variable ”STATE” für eine Verarbeitung genutzt werden, legt man eine weitere Variable, eine Konstante, mit als Wert ”$STATE” an. Durch die rekursive Parameterauflösung erhält die Konstante den Wert des Parameters ”STATE”.
Folgende Standard Parameters stehen zur Verfügung:

STATE Der Status der Resource. Für nicht Synchronizing Resources ist der Inhalt leer.

AMOUNT Die insgesamt zur Verfügung stehende Menge der Resource. Für Static Resources ist der Wert leer.

FREE_AMOUNT Die zur Verfügung stehende freie Menge der Resource. Für Static Resources ist der Wert leer.

REQUESTABLE_AMOUNT Die maximale Menge der Resource, die beantragt werden darf. Für Static Resources ist der Wert leer.

9.3.5 Tab Resources

Wurde eine Named Resource in der Navigation selektiert, erscheinen Instanzen dieser in dem Tab ”Resources”. Es zeigt alle Scopes, Folder, Submitted Entities, Scheduling Entities bzw. Jobserver, die Instanzen dieser Named Resource anbieten, an. Der Tab ”Resources” sieht aus wie folgt:

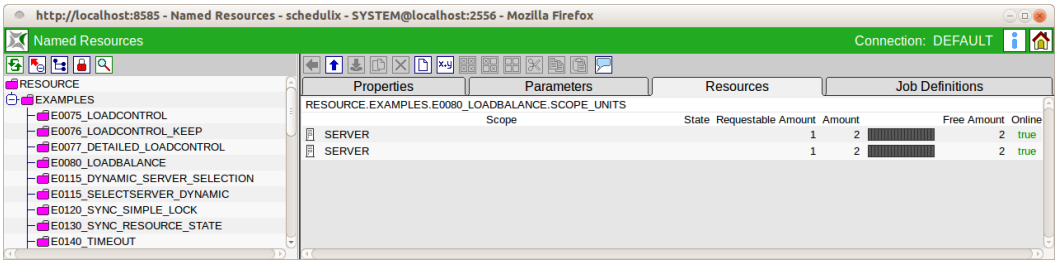


Abbildung 9.7: Named Resources; Instanzen

Die Instanziierungen der Named Resource werden durch folgende Felder beschrieben:

Scope Hier erscheinen die Namen der Scopes, Submitted Entities, Scheduling Entities oder Folder, welche Instanzen (Resources) der jeweilige Named Resource anbieten.

State Besitzt die gewählte Named Resource ein Resource State Profile, wird hier der aktuelle State der Resource angezeigt. Dies ist nur bei Synchronizing Resources möglich.

Requestable Amount Die Menge der Resources, die maximal von einem Job angefordert werden darf

Free Amount Hier wird die Anzahl der freien Instanzen dieser Resource angezeigt. Ein Balken gibt den momentanen Belegungsgrad graphisch wieder.

Amount Die Amount ist die Anzahl der zur Verfügung stehenden Einheiten der Resource.

Online Der Verfügbarkeitsstatus der Resource

9.3.6 Tab Job Definitions

Der Tab "Job Definitions" zeigt alle Job Definitions an, welche diese Named Resource anfordern. Dieser Tab dient lediglich zur Information, es können hier keine Änderungen vorgenommen werden.

Der Tab sieht folgendermaßen aus:

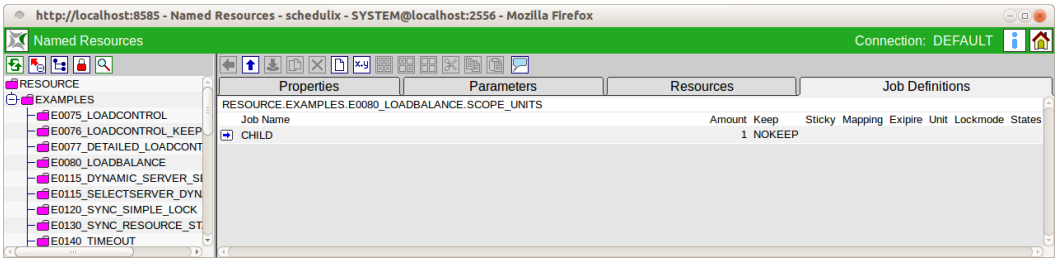


Abbildung 9.8: Named Resources; Job Definitions Tab

Die Liste der Jobs wird durch folgende Felder beschrieben:

Job Name Hier erscheinen die Namen der Submitted Entities, die die Named Resource benötigen.
Durch Anklicken des Namens wird ein Job Editor-Fenster geöffnet.

Amount Die Menge der Resource, die von dem Job benötigt wird

Keep Der Wert des Keep Parameters für die Resource-Anforderung des Jobs

Sticky Der Wert des Sticky Flags für die Resource-Anforderung des Jobs

Mapping Wurde bei der Resource-Anforderung ein Resource State Mapping angegeben, wird dies hier angezeigt.

Expire Der Expire-Wert gibt die maximale Zeit an, die seit dem letzten Statuswechsel der Resource vergangen sein darf, um die Resource als belegbar zu betrachten. Nun hat eine negative Expiration zur Folge, dass eine Resource mindestens so ”alt” wie angegeben sein muss.

Lockmode Der Lockmode beschreibt den Modus des Zugriffs auf diese Resource (exclusive, shared etc.).

States Zeigt alle Resource States an, die von diesem Job akzeptiert werden. Falls mehrere States für diesen Job akzeptabel sind, werden die einzelnen States durch Komma getrennt.

9.4 Selector Named Resource

Dieser Navigator dient zur Ein- oder Mehrfachauswahl von Named Resource Definitionen und wird von anderen Dialogen z. B. Footprint als Such- und Selektionsmaske aufgerufen. Die angezeigten Resources sind abhängig vom aufrufenden Dialog. Wird der Selector aus dem Footprint Editor aufgerufen, werden nur System Resources angezeigt. Wird der Selector aus dem Environment aufgerufen, werden nur statische Resources angezeigt.

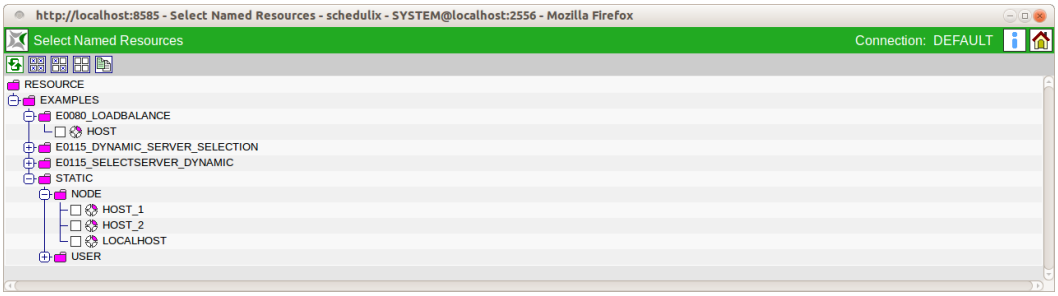


Abbildung 9.9: Named Resources; Selector

10 Environments

10.1 Bild

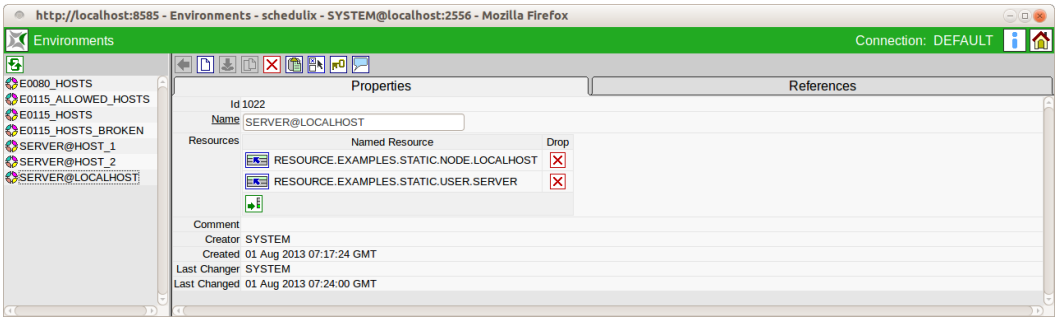


Abbildung 10.1: Environments

10.2 Konzept

10.2.1 Kurzbeschreibung

Der Dialog "Environment" dient zum Verwalten von Umgebungsdefinitionen. Ein Environment fasst mehrere Anforderungen statischer Resources unter einem Namen zusammen.

10.2.2 Ausführliche Beschreibung

Die Environments haben zwei Zwecke. Einerseits dienen sie zur vereinfachten Pflege von Resource-Anforderungen, da sie nicht mehr einzeln innerhalb der Job Definitions hinzugefügt werden müssen. Es kann dafür das jeweilige Environment angegeben werden. Andererseits wird mittels Environments festgelegt, welche Benutzer bzw. welche Benutzergruppe bestimmte Ablaufumgebungen benutzen dürfen.
Die Environment Definition gibt also an, welche Laufzeitumgebung ein Job benötigt. Zum Beispiel kann über ein Environment gesteuert werden, auf welchem Host ein Job laufen soll oder welche Benutzer und Programme vorhanden sein müssen. Das Environment ist ein Pflichtparameter in einer Job Definition.

Environments können auch als Folder Environment bei einem Folder eingetragen werden. Alle unter einem solchen Folder liegenden Job Definitions ”erben” dadurch die Resource-Anforderungen des Folders Environments.
Gibt es zum Beispiel jeweils einen Jobserver für Entwicklung und Produktion, kann am Job ein Environment verwendet werden, welches beide Jobserver als Ausführungsort zulässt. Werden nun für Entwicklung und Test zwei Folder angelegt, welchen jeweils ein Environment ”Entwicklung” bzw. ”Produktion” zugeordnet wird, deren Resource-Anforderungen nur von Entwicklungs- bzw. Produktions-Jobservern erfüllt werden kann, wird damit eine leicht bedienbare Trennung von Entwicklungs- und Produktionsumgebung erreicht. Befindet sich ein solcher Job irgendwo unter dem Folder ”Entwicklung”, führt die Kombination der Resource-Anforderungen von Job Environment und Folder Environment zur Auswahl des richtigen Entwicklungs-Jobservers. Wird der Job unter den Folder ”Produktion” verschoben oder kopiert und dann ausgeführt, so wird automatisch der Produktions-Jobserver ausgewählt. Eine Änderung der Job Definition ist nicht notwendig.

10.3 Navigator

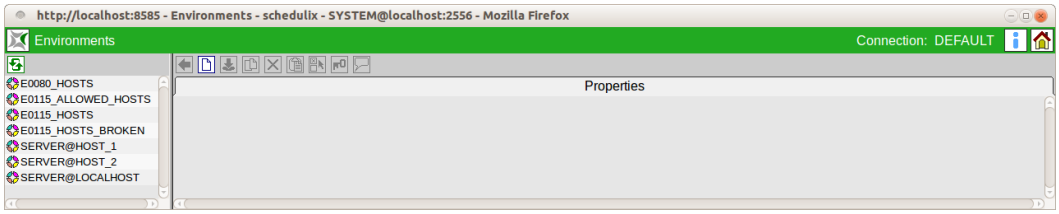


Abbildung 10.2: Environments Navigator

Der Navigationsbildschirm des ”Environment” Dialoges zeigt vorhandenen Environments an.
Ist der angemeldete Benutzer Mitglied der Gruppe ”ADMIN”, so werden alle Environments angezeigt. Ist dies nicht der Fall, so werden nur Environments angezeigt, die vom aktuell angemeldeten Benutzer genutzt werden dürfen. Das bedeutet, der angemeldete Benutzer muss Mitglied einer Gruppe sein welche das ”use” Privileg auf das jeweilige Environment besitzt.

10.4 Editor

10.4.1 Tab Properties

Dieser Tab dient der Pflege der Environment-Eigenschaften. Environments dürfen nur von Benutzern, welche Mitglied der Gruppe "ADMIN" sind, editiert werden. Für alle anderen Benutzer sind alle Eingabefelder "read only".

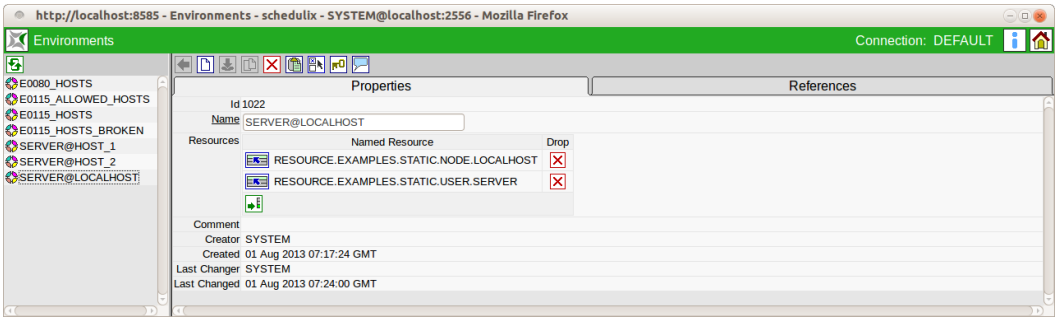


Abbildung 10.3: Environment Properties

Der "Properties" Tab für Environments hat folgende Felder:

ID Die Id wird automatisch vergeben und dient zur eindeutigen systemweiten Identifikation des Objektes.

Name Der Name des Environments. Dieser ist frei wählbar und muss innerhalb des Environments eindeutig sein.

Die Liste "Resources" zeigt alle zum Environment gehörenden Resources an. Es handelt sich hier ausschließlich um statische Resources. Das Feld *Named Resource* hat folgende Bedeutung:

Named Resource Hier wird der Name der Named Resource angezeigt.

Condition Im Feld *Condition* wird eine Bedingung eingetragen, die erfüllt sein muss, damit die Resource als gültig erkannt wird. Die Condition wird im Kontext des anfordernden Jobs ausgewertet.

Drop Mit dem *Drop*-Button kann die betreffende Zeile gelöscht werden.

10.4.2 Tab References

Dieser Tab dient der Anzeige welche Referenzen auf das Environment bestehen.

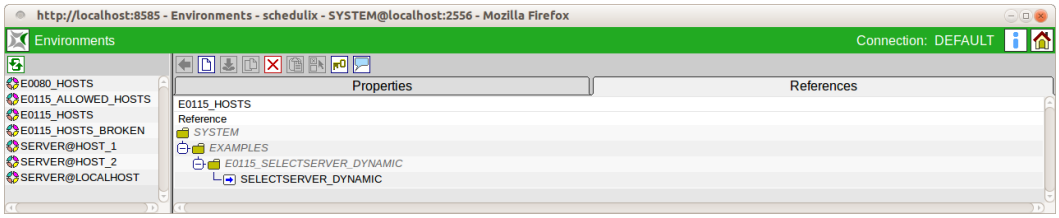


Abbildung 10.4: Environment References

Der ”References” Tab zeigt eine Baumansicht, welche nur die Folder enthält unter denen sich Objekte befinden, die das Environment referenzieren.
Folder werden normalerweise kursiv dargestellt, es sei denn, der Folder referiert das Environment als Folder Environment.
Ein Mausklick auf den Namen eines nicht rekursiv dargestellten Folders oder eines Jobs, öffnet ein Editor-Fenster für den Folder bzw. Job.

11 Footprints

11.1 Bild

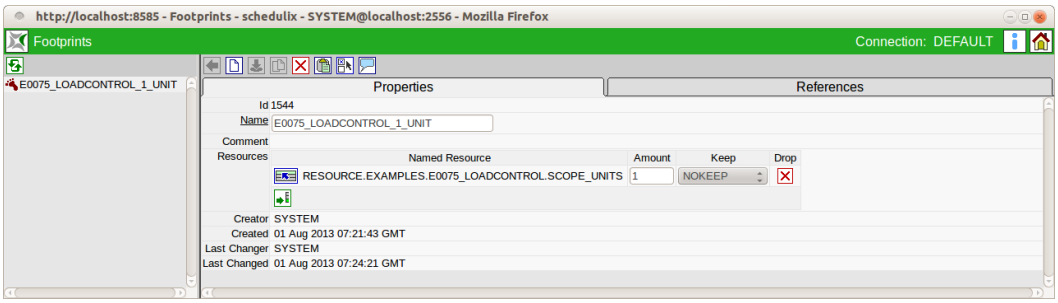


Abbildung 11.1: Footprints

11.2 Konzept

11.2.1 Kurzbeschreibung

Dieser Dialog dient zum Erstellen einer Footprint Definition. Ein Footprint fasst eine Gruppe Anforderungen von System Resources unter einem Namen zusammen.

11.2.2 Ausführliche Beschreibung

Die Zusammenfassung von Resource-Anforderungen zu Footprints, dient der leichteren Pflege der Anforderungen von Jobs an die Laufzeitumgebung, da sie nicht einzeln spezifiziert werden müssen.
Der Footprint kann für jeden Job übersteuert werden. Die Übersteuerung ist durch eine explizite Anforderung einer (im Footprint enthaltenen) Resource mit einer abweichenden Menge oder unterschiedlicher Einstellung des Keep Parameters möglich. Eine Resource-Anforderung kann nicht entfernt werden, d.h. alle im Footprint definierten Resources werden angefordert. Die Menge kann bis zum Wert 0 vermindert werden, allerdings wird trotzdem das Vorhandensein der Resource gefordert.
Beispiel:
Im Footprint wird eine System Resource CPU_UNIT mit der Menge 4 (CPU Einheiten) definiert. Durch Hinzufügen einer Anforderung der selben System Resource

(CPU_UNIT) innerhalb der Job Definition (Tab Resources) mit einer Menge von 2 (CPU Einheiten), kann der Footprint übersteuert werden. Es ist aber nicht möglich die Resource CPU_UNIT ganz zu entfernen.

11.3 Editor

11.3.1 Tab Properties

Dieser Tab dient der Pflege der Footprint-Eigenschaften.
Footprints dürfen nur von Benutzern, welche Mitglied der Gruppe "ADMIN" editiert werden. Für alle anderen Benutzer sind alle Eingabefelder "read only".
Er sieht folgendermaßen aus:

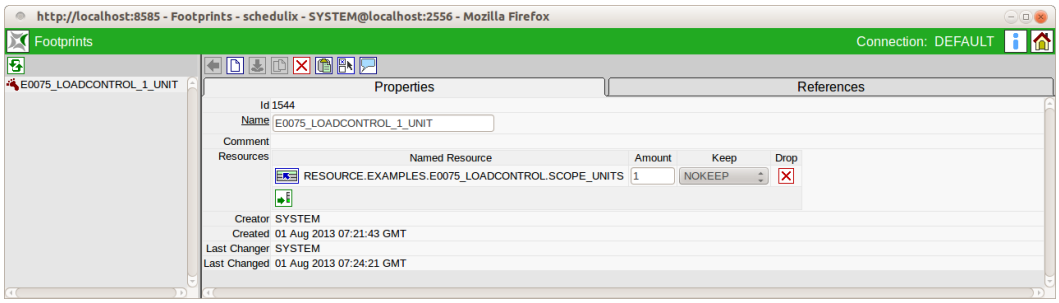


Abbildung 11.2: Footprint Properties

Der "Properties" Tab für Footprints hat folgende Felder:

ID Die ID wird automatisch vergeben und dient der eindeutigen systemweiten Identifikation des Footprints.

Name Der Name des Footprints. Dieser ist frei wählbar.
In der Liste "Resources" kann eine Liste für die enthaltenen Resources des Footprints angegeben werden.
Die Felder haben folgende Bedeutung:

Named Resource Hier wird der Name der Named Resource angezeigt. Die Auswahl erfolgt über einen *Chooser* Button.

Amount Das ist die Menge der Resource die ein Job benötigt.

Keep Mittels des Keep Parameters wird bestimmt, ob die Named Resource nach Ablauf eines Jobs gehalten wird oder wieder freigegeben werden kann. Es gibt folgende Auswahlmöglichkeiten:

1. *No_Keep*

   Die Named Resource wird nach Beendigung des Jobs freigegeben. Ob der Job erfolgreich beendet oder wegen eines Fehlers abgebrochen wurde, spielt für die Freigabe keine Rolle.

2. *Keep*

   Die Resource wird erst freigegeben, wenn der Job einen finalen Status erreicht hat. Im Fehlerfall (Restartable State) wird die Resource gehalten.

3. *Keep_Final*

   Die Resource wird erst freigegeben, wenn der Job und alle seine Children einen finalen Status erreicht haben.

11.3.2 Tab References

Dieser Tab dient der Anzeige, welche Referenzen auf den Footprint bestehen. Er sieht folgendermaßen aus:

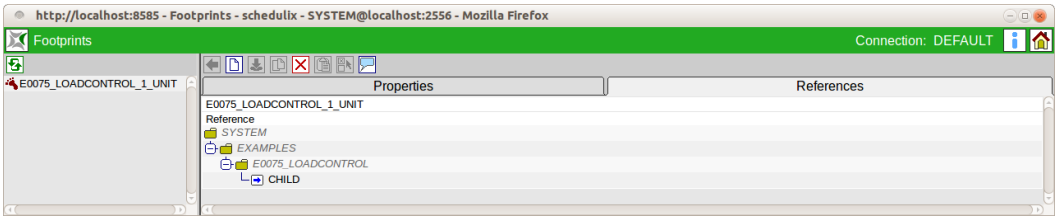


Abbildung 11.3: Footprint References

Der "References" Tab zeigt eine Baumansicht, welche nur die Folder enthält unter welchen sich Jobs befinden, welche den Footprint referenzieren.
Folder werden kursiv dargestellt.
Ein Mausklick auf den Namen eines Jobs öffnet ein Editor-Fenster für den Job.

12 Jobservers and Resources

12.1 Bild

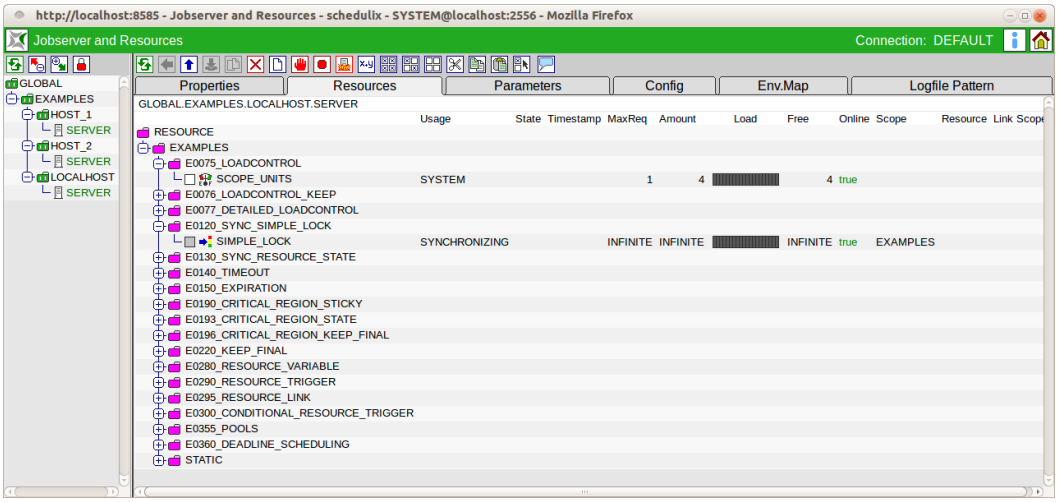


Abbildung 12.1: Jobservers and Resources

12.2 Konzept

12.2.1 Kurzbeschreibung

Ein Jobserver ist ein schedulix Systemprozess, der auf allen Rechnern, die als ausführende Einheiten des schedulix Scheduling Systems definiert wurden, laufen muss. Wird einem Jobserver durch den schedulix Server ein Ablaufobjekt zum Ausführen zugeteilt, führt der Jobserver das in der Job Definition angegebene Skript oder Programm in der notwendigen Umgebung mit den notwendigen Parametern aus. Der Jobserver überwacht das Programm und gibt den Rückkehrstatus des Skripts oder Programms an den schedulix Server zurück.

12.2.2 Ausführliche Beschreibung

Im Dialog "Jobserver and Resources" kann die physikalische Aufteilung definiert und gepflegt werden, d. h. welche Jobserver auf welchen Maschinen laufen und

welche Resources von diesen Jobservern zur Verfügung gestellt werden.
Die Einrichtung und der Start dieser Jobserver muss auf Systemebene durchgeführt werden.
Im Dialog werden zwei Arten von Objekten unterschieden: Einerseits gibt es sogenannte Scopes, die als Container für die jeweiligen Jobserver oder untergeordnete Scopes dienen können.
Andererseits werden in diesem Dialog die Jobserver verwaltet, die die ausführenden Systemprozesse widerspiegeln.
Die Unterteilung der Hierarchie ist frei durchführbar und hängt größtenteils von der Komplexität der unterlagernden Systemumgebung und Verwaltungsstruktur (z. B. im Rechenzentrum) ab.
Beispielsweise kann jeder physikalische Hostrechner als Scope, und alle auf diesem Rechner benötigten Ablaufumgebungen als Jobserver unterhalb dieses Scopes definiert werden. Falls die Systemlandschaft komplexer ist, sind weitere Ebenen für Abteilungen, Systemarchitekturen (Unix, Windows) oder ähnliches hilfreich.
Auf jeder Hierarchieebene können den Scopes und Jobservern Resources zugeordnet werden. Diese Resources stehen dann allen untergeordneten Jobservern zur Verfügung.

Beispiel:
Auf der Ebene eines Hosts wird die System Resource CPU_UNITS mit einem Wert von 5 eingetragen. Werden auf dem Rechner 2 Jobserver angelegt, stehen diesen Jobservern diese 5 CPU Einheiten gemeinsam zur Verfügung. Belegt ein Jobserver durch die Ausführung eines Jobs 3 CPU Einheiten, stehen dem anderen Jobserver nur noch 2 CPU Einheiten zur Verfügung. Dieses Verhalten ist jedoch nicht additiv. Wird auf einem der oben genannten Jobserver ebenfalls die Resource CPU_UNITS angelegt, etwa mit einem Wert von 3, stehen diesem Jobserver ausschließlich diese 3 CPU_UNITS zur Verfügung. Dem anderen Jobserver stehen demnach dann 5 CPU_UNITS vom übergeordneten Scope zur Verfügung.

12.3 Navigator

Der Navigator bietet eine hierarchische Darstellung, in der Scopes als Ordner definiert werden. Innerhalb der Scopes befinden sich die Jobserver.

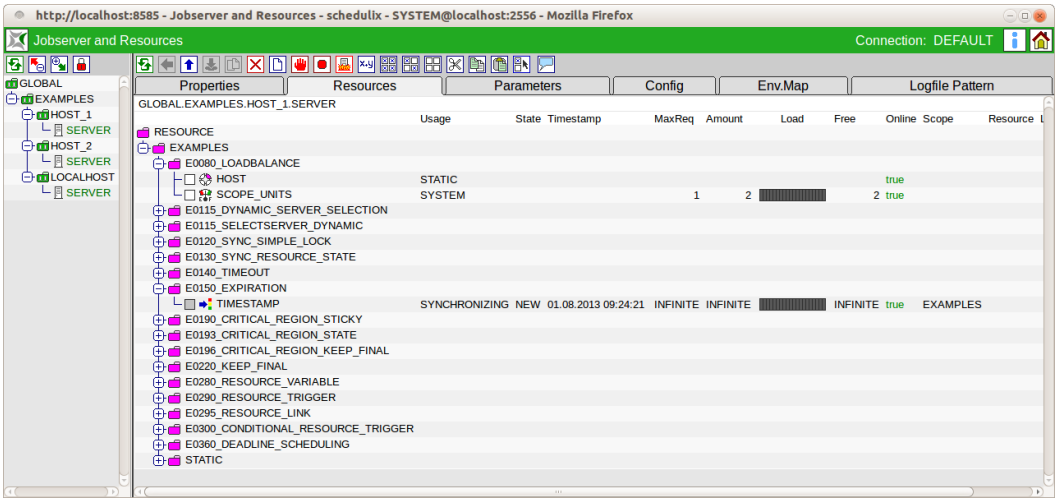


Abbildung 12.2: Jobservers und Scopes Navigator

12.4 Editor

12.4.1 Tab Properties

Der Tab ”Properties” dient zur Pflege der Eigenschaften von Scopes und Jobservern.

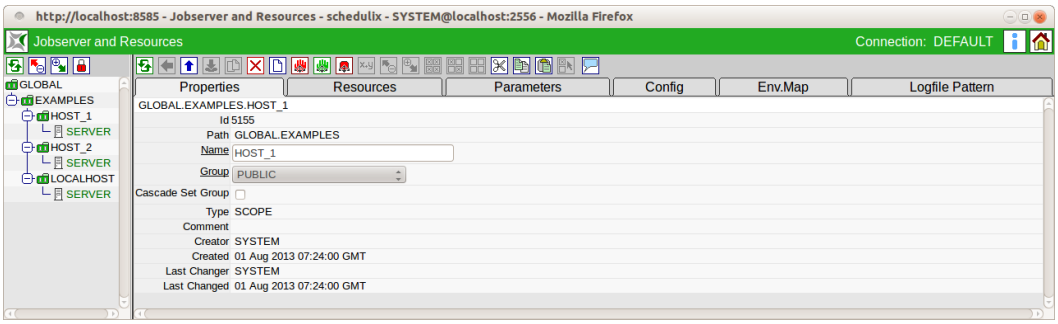


Abbildung 12.3: Scope Properties

Tab ”Properties” für Scopes ist dargestellt in Bild 12.3.
Tab ”Properties” für Jobserver ist dargestellt in Bild 12.4.

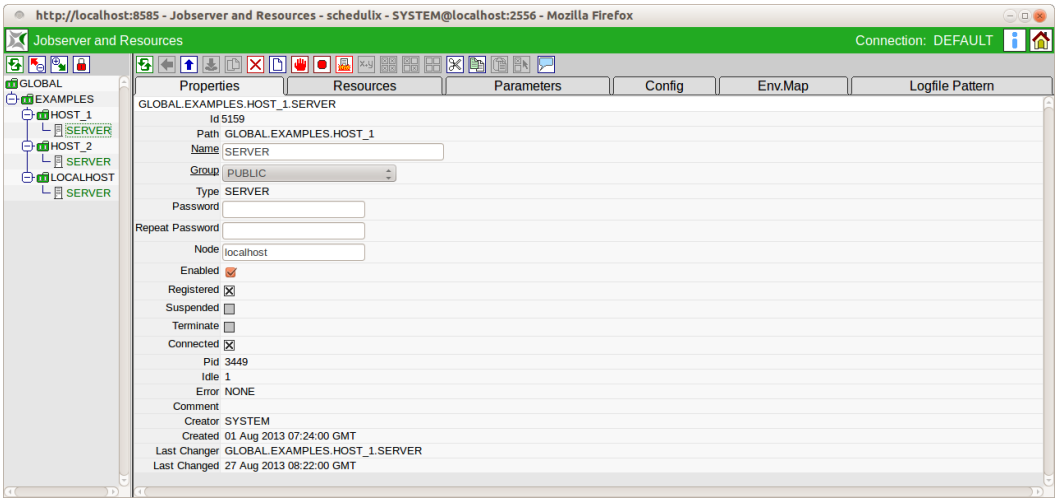


Abbildung 12.4: Jobserver Properties

Die Felder des Tabs "Properties" haben folgende Bedeutung:

ID Die ID dient zur eindeutigen systemweiten Identifikation des Objektes.

Path (Pfad) Der Path stellt den kompletten Pfad des Objektes innerhalb der Hierarchie dar.

Name Das ist der Name des Scopes oder des Jobservers. Der Name kann frei gewählt werden, muss aber innerhalb der Hierarchiestufe eindeutig sein.

Group Die Gruppe kann aus einer "Drop Down" Liste ausgewählt werden. Sie bezeichnet die Owner-Gruppe des Scopes.

Type Hierbei handelt es sich um den Typ des selektierten Objektes. Der Typ wird bei der Neuanlage eines Servers oder Scopes gewählt und kann dann nicht mehr verändert werden.
OPTIONEN FÜR TYPE

1. *Scope*

   Ein Scope ist eine Kategorie, unter der Jobserver gruppiert und hierarchisch geordnet werden können. Dies könnte zum Beispiel ein Hostrechner sein, oder auf noch höherer Ebene alle Rechner der Abteilung oder alle Windows-basierten Rechner. Der Aufbau und die Tiefe der Hierarchie kann je nach Systemlandschaft angepasst werden und sollte diese widerspiegeln.

2. *Server*

   Bei dem Objekt handelt es sich um einen physikalischen Jobserver. Der physikalische Prozess muss sich unter dem gewählten Namen beim schedulix System anmelden und damit die Verbindung zu dieser Jobserver-Definition schaffen.

Password Das *Password* dient zum Anmelden des Jobservers am schedulix Server. Jeder Jobserver benötigt seinen Namen und das Passwort, um sich am schedulix Server anmelden zu können. Nur mit gültiger Anmeldung kann ein Job an den Jobserver zur Ausführung vergeben werden.
Das Passwort wird in der Anzeige verdeckt und muss mit dem Feld *Repeat Password* identisch sein.
Da das Passwort im Dialog nicht angezeigt wird, ist es nicht ersichtlich, ob ein Passwort bereits eingegeben wurde oder dies noch notwendig ist. Bei Änderung von anderen Eigenschaften muss das Passwort nicht erneut spezifiziert werden. Falls der Jobserver angemeldet ist, wird er automatisch von eventuellen Änderungen eines Passwortes in Kenntnis gesetzt.

Repeat Password Das *Repeat Password* muss bei der Eingabe des Passwortes identisch zum Feld *Password* sein. Die doppelte Eingabe ist nötig, um versehentliche Tippfehler festzustellen.

Node Der *Node* gibt an, auf welchem Rechner der Jobserver läuft. Dieses Feld hat einen rein dokumentativen Charakter.

Enabled Falls der Jobserver das Enabled Flag gesetzt hat, ist eine Anmeldung des Jobserver-Prozesses am schedulix Server möglich. Ist das Flag nicht gesetzt, schlägt die Anmeldung fehl.

Suspended Ist das suspended Flag gesetzt, kann sich der Jobserver-Prozess zwar anmelden, es werden ihm aber keine Jobs zur Ausführung zugeteilt. Statusänderungen von bereits zugeteilten Jobs werden vom schedulix Server entgegengenommen.

Terminate Das Terminate Flag zeigt an, ob der Jobserver sich terminieren soll. Wenn das Flag gesetzt ist, wird der Jobserver diese Nachricht beim nächsten Kommunikationsschritt erhalten und sich entsprechend verhalten.
Die Terminierung des Jobservers kann einige Sekunden dauern. Die Beendigung ist mittels des *Refresh* Buttons überprüfbar. Wurde der Jobserver beendet, so wechselt die Anzeige des Feldes *Connected* auf den Wert "FALSE".

Connected Das Feld *Connected* gibt Auskunft über den Verbindungsstatus eines externen Jobserver-Prozesses. Wurde die Anmeldung des Jobserver-Prozesses am schedulix System erfolgreich abgeschlossen, steht dieser Wert auf "TRUE".

PID Bei der PID handelt es sich um die Prozess-Identifikationsnummer des Jobserver-Prozesses auf dem jeweiligen Hostsystem. Sie wird während der Registrierung dem schedulix System mitgeteilt. Über die PID und das *Node*-Feld ist es möglich, den Prozess auf dem Betriebssystem des jeweiligen Servers zu lokalisieren.

Error Ist bei der Ausführung des Jobserver-Prozesses ein Fehler aufgetreten, wird hier eine Fehlermeldung angezeigt.

12.4.2 Tab Resources

Der Tab "Resources" sieht folgendermaßen aus:

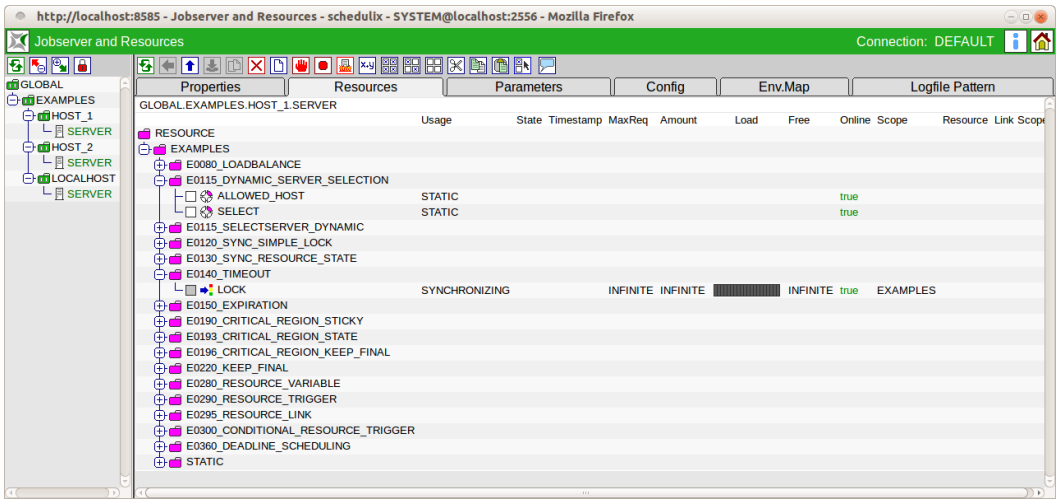


Abbildung 12.5: Jobserver und Scope Resources

Im Tab "Resources" werden alle Resources angezeigt, die der aktuelle Scope oder Jobserver anbietet. Sollen Resources, insbesondere vom Typ Synchronizing, systemweit vergeben werden, kann dies im Scope "GLOBAL" geschehen.
Werden Resources mehrmals innerhalb von unterschiedlichen Scopes oder Jobservern eingetragen, handelt es sich hierbei um eine jeweils eigene Instanz der Resource.
Beispiel:
Wird eine Synchronizing Resource A in Scope X und Scope Y eingetragen, sind beide Instanzen der Resource A (in X und Y) voneinander unabhängig. Das heißt, eine

mögliche Synchronisation findet nur innerhalb aller Jobs in Scope X und innerhalb aller Jobs in Scope Y statt. Soll hier eine globale Synchronisation stattfinden, muss die Resource A im Scope GLOBAL definiert werden.
Die Anzeige der Resources findet als Abbildung der Hierarchie aus der Navigation des Dialoges Named Resources statt, wobei zusätzliche Informationen angezeigt werden.
Das erste Feld der Liste gibt den Namen der Named Resource oder der Kategorie an. Handelt es sich um eine Kategorie, erscheint zusätzlich noch ein Ordner-Icon. Durch Anklicken dieses Icons kann die Kategorie geöffnet und geschlossen werden. Handelt es sich um eine Named Resource, kann der Name angeklickt werden und führt zum Tab "Resource Detail".

Die restlichen Felder werden im nächsten Abschnitt erläutert.

12.4.2.1 Tab Resource Detail

Der Tab "Resource Details" wird angezeigt, sobald eine Resource ausgewählt wurde. Der Tab dient zur Eingabe und Pflege der Instanzinformationen für diese Named Resource. Mit Instanz ist hier die Ausprägung der Named Resource im gewählten Scope bzw. Jobserver gemeint. Alle Instanzparameter können hier eingegeben und gepflegt werden.
Der Tab "Resource Detail" sieht folgendermaßen aus:

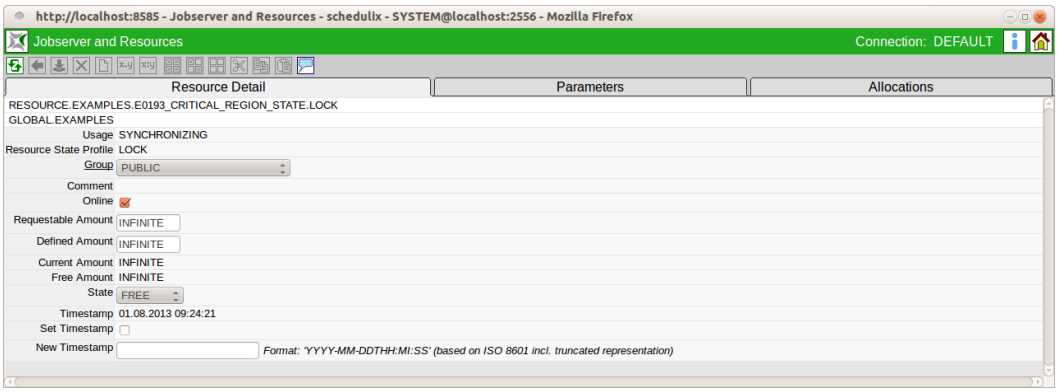


Abbildung 12.6: Resource Details

Die Felder des Tabs "Resource Detail" haben folgende Bedeutung:

Usage Die *Usage* gibt an, um welchen Typ "Resource" es sich handelt. Mehr zu Typen von Resources finden Sie im Dialog Named Resource.

Resource State Profile Dieses Feld ist nur zu sehen, wenn die Usage der Resource "Synchronizing" ist.

Es handelt sich hier um das zur Resource zugeordnete Resource State Profile.

Online Mittels *Online* ist es möglich eine Resource in diesem Scope oder Jobserver on- bzw. offline zu schalten. Ist die Resource offline, so steht diese Resource vorübergehend nicht zur Verfügung.
Beispiel:
Eine geplante Downtime eines Rechners oder einer Datenbank steht an. Wurde der Rechner oder die Datenbank als Resource im System abgebildet, kann die Resource, sobald sie physikalisch nicht mehr verfügbar ist, offline gesetzt werden. Alle Jobs die diese Resource benötigen, werden automatisch nicht mehr ausgeführt. Sollte die Resource in einem anderen Scope ebenfalls definiert sein (als eigene Instanz), so würden alle Jobs, die die Resource benötigen auf diesen Scope oder Jobserver ausweichen.
Ist eine Resource offline, führt dies *nicht* zu einer "Job cannot run in any scope because of resource shortage" Fehlermeldung.

Requestable Amount Der *Requestable Amount* ist die Menge, die maximal angefordert werden kann.

Defined Amount Dieses Feld ist nur zu sehen, wenn die Usage der Resource "Synchronizing" oder "System" ist.
Der *Defined Amount* gibt die aktuelle Anzahl von Instanzen der Named Resource für diesen Scope oder Jobserver an. Sollten sich Änderungen in der Anzahl für diesen Scope ergeben, so können sie mit diesem Feld angepasst werden.
Beispiel:
Dies kann bei Hardware-Änderungen unter Umständen notwendig werden. Stellt die Resource eine CPU Einheit dar und sind im aktuellen Host 2 CPU Einheiten verbaut, so ist der Amount 2. Werden nun 2 weitere CPU Einheiten eingebaut, muss der Wert auf 4 erhöht werden.

Current Amount Der *Current Amount* und der *Defined Amount* sind identisch.

Free Amount Dieses Feld ist nur zu sehen, wenn die Usage der Resource "Synchronizing" oder "System" ist.
Der *Free Amount* bezeichnet die Anzahl aller noch nicht von Jobs belegten Instanzen einer Resource, innerhalb des gewählten Scopes oder Jobservers.

Status/State Das Feld *State* ist nur zu sehen, wenn die Usage der Resource "Synchronizing" ist und ein Resource State Profile zugeordnet wurde.
Hierbei handelt es sich um den aktuellen Status der Resource in diesem Scope oder Jobserver. Der Status kann mittels dieses Wertes gesetzt bzw. geändert werden. In der "Drop Down" Liste sind alle gültigen Resource States des Resource State Profiles enthalten und können ausgewählt werden.

Dies kann bei einer manuellen Fehlerbehebung notwendig sein, um eine Resource wieder in einen Status zu bringen, in dem eine Weiterverarbeitung durch Folgejobs möglich ist.
Falls hier der Wert manuell geändert wird, wird der Wert des Feldes *State* im Tab "Resources" nicht automatisch aktualisiert. Falls dies gewünscht und notwendig ist, kann dies mittels des *Refresh* Buttons durchgeführt werden.

Timestamp Dieses Feld ist nur zu sehen, wenn die Usage der Resource "Synchronizing" ist und ein Resource State Profile zugeordnet wurde.
Der *Timestamp* gibt die Zeit des letzten Statuswechsels einer Resource an.
Timestamps spielen eine Rolle, falls in einem Ablaufobjekt ein Expiration-Intervall angegeben wurde. Ist dies geschehen, darf der Timestamp nicht älter sein als das angegebene Intervall. Mehr zu Expiration-Zeiten finden Sie im Kapitel 13.5.8.1.
Wird durch die Beendigung eines Jobs eine Änderung des Resource State hervorgerufen, wird der Timestamp automatisch vom schedulix Server aktualisiert.

Set Timestamp Dieses Feld ist nur zu sehen, wenn die Usage der Resource "Synchronizing" ist und ein Resource State Profile zugeordnet wurde. Das *Set Timestamp* Flag ist ein sog. Action Flag. Wird das Flag gesetzt und anschließend der *Speichern*-Button gedrückt, so wird die Zeit des *Timestamp*-Feldes auf die aktuelle Zeit gesetzt.
Beispiel:
Dies kann notwendig werden, wenn ein manueller Eingriff bei einer Fehlerbehebung notwendig geworden ist und dabei die Statusänderung manuell über das Feld *State* durchgeführt wurde.

12.4.2.2 Tab Parameters

Im Tab ”Parameters” können zusätzliche Informationen, die von Jobs oder Resource Trigger ausgewertet werden können, zu einer Resource gespeichert werden. Die Definition der Parameters erfolgt bei den Named Resources.

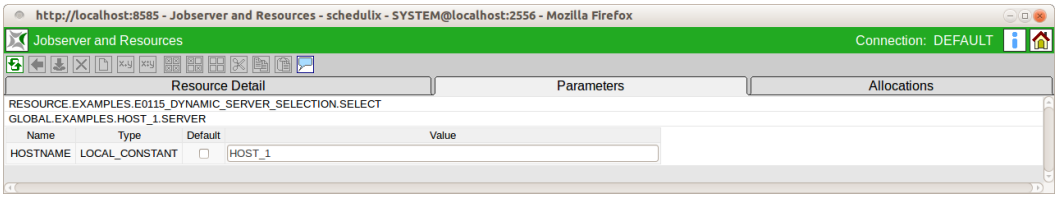


Abbildung 12.7: Resource Parameters

Die Spalten der obigen Liste haben folgende Bedeutung:

Name In der Spalte *Name* wird der Name des Parameters angezeigt.

Type Beim *Type* handelt es sich um die Art des Parameters. Es gibt folgende Möglichkeiten:

- Constant: Der Constant hat einen festen Wert und gilt für alle Resources.
- Local Constant: Der Local Constant hat einen festen Wert, der von Resource zu Resource abweichen kann.

Die Werte der Local Constants sind an dieser Stelle änderbar.

Default Das Feld *Default* gibt an, ob der angezeigte Wert der Default-Wert ist. Das Flag muss gesetzt werden, damit der Wert wieder auf den Default-Wert zurück gesetzt wird.

Value Der *Value* ist der Wert des Parameters.

12.4.2.3 Tab Allocations

Der Tab "Allocations" liefert Informationen, welche Tasks aktuell die Resource belegen und welche Tasks Resources benötigen, sie aber etwa aufgrund eines falschen Status oder wegen eines Locks nicht bekommen können.
Der Tab Sheet sieht folgendermaßen aus:

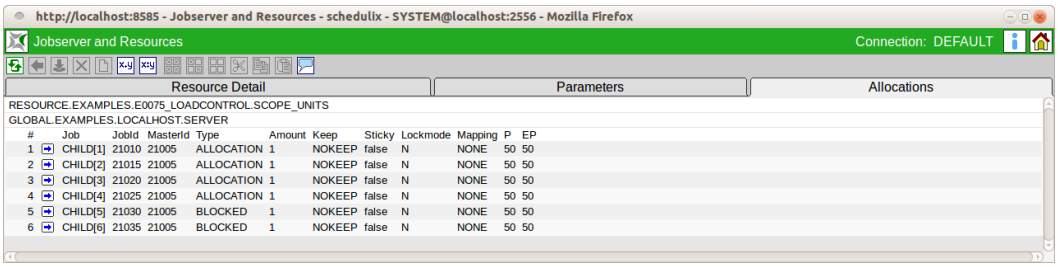


Abbildung 12.8: Resource Allocations

Die Spalten der obigen Liste haben folgende Bedeutung:

Job Je nach Einstellung wird hier entweder der Name des Jobs oder der gesamte Pfad der Ordnerhierarchie für diesen Job angezeigt.

Job Id Hierbei handelt es sich um die ID der Job-Instanz, welche durch ein direktes Submit des Jobs im Dialog Submit Jobs oder durch ein Submit des Master Batches oder Jobs gestartet wurde.

Master Id Hierbei handelt es sich um die ID der Job- oder Batch-Instanz, welche als Master Job im Dialog Submit Jobs gestartet wurde und den aktuellen Job als Child beinhaltet. Handelt es sich bei dem Job selber um einen Master Job, wird die eigene Job ID angezeigt.

Type Beim *Type* handelt es sich um die Art, wie die aktuelle Job-Instanz auf die Resource zugreift bzw. zugreifen will. Es gibt folgende Optionen:

1. Requested

   Der Job fragt beim Server an, ob die Resource verfügbar ist.

2. Reserved

   Die Resource ist vom Job reserviert. Dies hat die gleiche Auswirkung wie Allocated, nur können reservierte Resources beim Job Start wieder freigegeben werden. Belegte (allocated) Resources werden frühestens beim Job End wieder freigegeben.

3. Allocated

   Der Typ "Allocated" gibt an, dass der Job aktuell läuft und die Resource im Zugriff hat. Andere Jobs, welche auf diese Resource zugreifen wollen, können abhängig vom Typ der Zugriffssperre diese ebenfalls belegen oder werden blockiert.

4. Blocked

   Ist ein Job mit dem Typ "Blocked" gekennzeichnet, kann dieser im Moment nicht ausgeführt werden, da er auf den Zugriff der gewählten Resource wartet, diese aber wegen eines nicht passenden Lockmode oder anderer Kriterien nicht bekommen kann.

   Der Grund weshalb die Resource nicht zur Verfügung steht wird durch rote Schrift gekennzeichnet.

   Sobald die Resource wieder verfügbar ist, wechselt der Typ von Blocked in Requested.

5. Available

   Die Resource steht zur Verfügung.

6. Ignored

   Der Anwender hat das System beauftragt die Resource-Anforderung zu ignorieren.

7. Master Reservation

   Eine Master Reservation ist eine Reservierung von Sticky Resources.

Amount Beim *Amount* handelt es sich um die Anzahl der vom aktuellen Job belegten oder angeforderten Resource-Instanzen. Übersteigt der Wert des Amounts die aktuell verfügbare Anzahl, die im Feld *Amount* im Tab "Resource Detail" gepflegt werden kann und ist kein alternativer Scope mit einer ausreichenden Anzahl vorhanden, kann der Job nicht ausgeführt werden.

Keep Der *Keep*-Parameter gibt an, mit welchem Keep-Status die aktuelle Resource gehalten wird. Weitere Informationen zum Keep-Status finden Sie im Kapitel 11.3.1.

Sticky Der *Sticky*-Parameter gibt an, ob der Job die aktuelle Resource "Sticky" hält oder nicht. Weitere Informationen zum Sticky Flag finden Sie im Kapitel 13.5.8.1.

Lockmode Der *Lockmode* gibt an, mit welchem Zugriffsmodus die Resource vom aktuellen Job belegt wird. Es gibt folgende Zugriffsmodus:

1. Exclusive (X)

   Kein anderer Job hat Zugriff auf diese Resource.

   Beispiel: Laden einer Datenbanktabelle.

2. Shared Exclusive (SX)

   Diese Zugriffe sind untereinander verträglich, jedoch unverträglich mit Shared-Zugriffen.

   Beispiel: Eine Anwendung führt viele kleine schreibende Transaktionen auf einer Datenbanktabelle durch.

3. Shared (S)

   Shared-Zugriffe sind untereinander verträglich.

   Beispiel: Auswertungen über komplette oder große Teile von Datenbanktabellen.

4. Shared Compatible (SC)

   Diese Zugriffe sind sowohl untereinander als auch mit S- und SX-Zugriffen verträglich.

   Beispiel: Eine Anwendung führt viele kurzlaufende, lesende Transaktionen auf einer Datenbanktabelle durch. Solche Anwendungen behindern sich nicht gegenseitig. Auch kleine schreibende Transaktionen, sowie große Auswertungen stellen keine Behinderung dar.

5. NoLock (N)

   Der Lockmode NoLock gibt an, dass weder eine Sperrung der Resource durchgeführt wird, noch beachtet werden soll. Ein Job, der eine Resource mit NoLock sperrt, kann immer auf diese zugreifen, unabhängig von anderen Sperroptionen.

Hier noch einmal zusammenfassend die Kompatibilitätsmatrix aller Sperroptionen untereinander.
(+) bedeutet, die Lockmodes sind kompatibel zueinander. Zwei Jobs, welche die beiden Lockmodes anfordern, können zur selben Zeit laufen.
(-) bedeutet, die Lockmodes schließen sich aus. Ein Job wird blockiert, bis der andere Job die Resource wieder freigibt.

	Exclusive (X)	Shared (S)	Shared Exclusive (SX)	Shared Compatible (SC)	NoLock (N)
Exclusive	-	-	-	-	+
Shared	-	+	-	+	+
Shared Exclusive	-	-	+	+	+
Shared Compatible	-	+	+	+	+
NoLock	+	+	+	+	+

Tabelle 12.1: Lockmode-Matrix

Mapping Das *Mapping* gibt das aktuelle Resource State Mapping an, welches die Resource verwendet.

P (Priorität) Die *Priorität* gibt an, welche Startpriorität der Prozess beim Submit im Dialog Submit Jobs hatte. Der Wertebereich der Priorität beginnt bei 100 (niedrigste Priorität) und endet bei 0 (höchste Priorität)

EP (Effektive Priorität) Je länger ein Job auf die Ausführung im System wartet, desto höher wird die effektive Priorität der Ausführung im schedulix System. Früher gestartete Jobs sollen damit vor den später gestarteten Jobs berücksichtigt werden (je nach Einstellungen im Feld Priority im Batches and Jobs-Dialog und der Startzeit).
In der Serverkonfiguration ist einstellbar, wie die Priorität über die Zeit steigt. Standardmäßig ist eingestellt, dass mit jeder halben Stunde, die seit dem Submit des Jobs vergangen ist, die Priorität um einen Punkt steigt.
Als Beispiel: Wurde ein Job um 12 Uhr mit der Priorität 50 gestartet, so ist die effektive Priorität um 12:30 Uhr 49, um 13:00 Uhr 48 usw. bis die höchste Stufe erreicht wurde.

Blockierte Prozesse Ist der aktuelle Job in der Liste als blockiert markiert, erscheint der Grund warum der Job nicht ausgeführt werden kann in rot.
Beispiel: Wird die Resource gerade exklusiv von einem anderen Job belegt und der aktuelle Job möchte die Resource ebenfalls exklusiv belegen, ist das Feld *Lockmode* rot markiert.

12.4.3 Tab Parameters

Der Tab "Parameters" verwaltet alle im Jobserver bzw. Scope definierten Parameter. Diese Parameter stehen den Jobs und den Programmen, welche innerhalb der Jobs ausgeführt werden, zur Verfügung. Innerhalb von Scope-Hierarchien findet eine Vererbung der Parameter statt. Es kann ein Parameter auf höchster Ebene definiert werden und jeder Scope und jeder Jobserver, welcher unterhalb dieser angesiedelt ist, hat automatisch diesen Parameter zur Verfügung.
Der Tab "Parameters" sieht folgendermaßen aus:

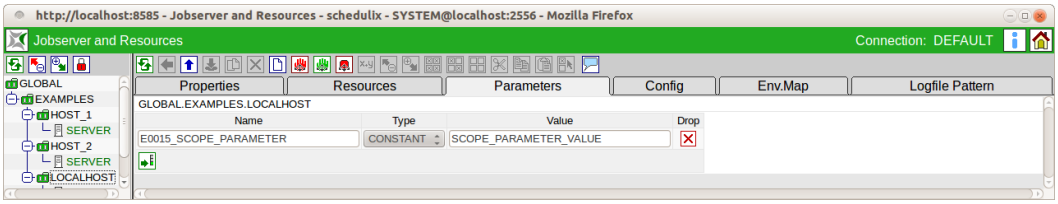


Abbildung 12.9: Jobserver und Scope Parameters

Hier erscheint die Liste aller Parameter. Wurde in der Navigation ein Jobserver angewählt, sind über die aktuellen Parameter hinaus, alle aus der übergeordneten Scope-Hierarchie vererbten Parameter ebenfalls sichtbar. Sie können aber auf dieser Ebene nicht verändert werden. Soll ein vererbter Parameter auf dieser Ebene abgeändert werden, so muss er mit identischem Namen neu angelegt werden. Dieser Wert überschreibt den Wert aus dem vererbten Teil. Die Anzeige des Wertes ist nun änderbar.
Die Spalten der obigen Liste haben folgende Bedeutung:

Name Hier wird der Name des Parameters angezeigt.

Type Der Typ des Parameters gibt an, um welche Art von Parametern es sich handelt. Es gibt folgende Optionen:

1. Dynamic

   Es handelt sich um einen dynamischen Parameter. Der Wert des Parameters ergibt sich aus dem Wert einer Umgebungsvariablen, die auf dem System gepflegt sein muss auf dem der aktuelle Scope bzw. Jobserver läuft.

   Dieser Wert wird erst zur Startzeit des Jobservers ermittelt. Läuft der Jobserver aktuell auf der Maschine, wird im Feld *Value* der aktuelle Wert der gleichlautenden Umgebungsvariablen auf der Maschine und der Umgebung in der der Jobserver läuft, angezeigt.

   Achtung: Die Umgebungsvariable muss auf der Maschine und in der Startumgebung des Jobservers angelegt worden sein.

2. Constant

   Ist der Typ eines Parameters ”Constant”, so handelt es sich um einen konstanten Ausdruck, der im Feld *Value* eingetragen werden muss. Verwendet ein Job bzw. dessen Programm den Wert, wird der Parameter während der Ausführung durch diesen konstanten Ausdruck ersetzt.

   Beispiel:

   Wenn ein Skript unabhängig von der aktuellen Datenbank laufen soll, kann die Datenbankverbindung als Parameter abgebildet werden. Wird der Datenbankverbindungsparameter im Scope 1 (zum Beispiel die Test-Datenbank) und gleichzeitig im Scope 2 (zum Beispiel die Produktivdatenbank) ebenfalls mit jeweils unterschiedlichen Werten definiert, kann das Skript ohne Änderungen in beiden Umgebungen laufen, da es seine Datenbankverbindung über den Parameter mitgeteilt bekommt.

Value Hierbei handelt es sich um den Wert des Parameters. Ist der Parameter vom Typ ”Constant”, muss hier der Parameterwert eingetragen werden. Handelt es sich um einen dynamischen Parameter, wird hier der aktuelle Wert der Umgebungsvariablen auf dem System, auf dem der Jobserver läuft, angezeigt.

12.4.4 Tab Config

Im Tab ”Config” steht die Konfiguration des Jobservers beschrieben. Die Konfiguration erfolgt über eine Tabelle mit Key/Value Pairs. Für die Bedeutung der einzelnen Werte wird auf die Jobserver-Dokumentation verwiesen.

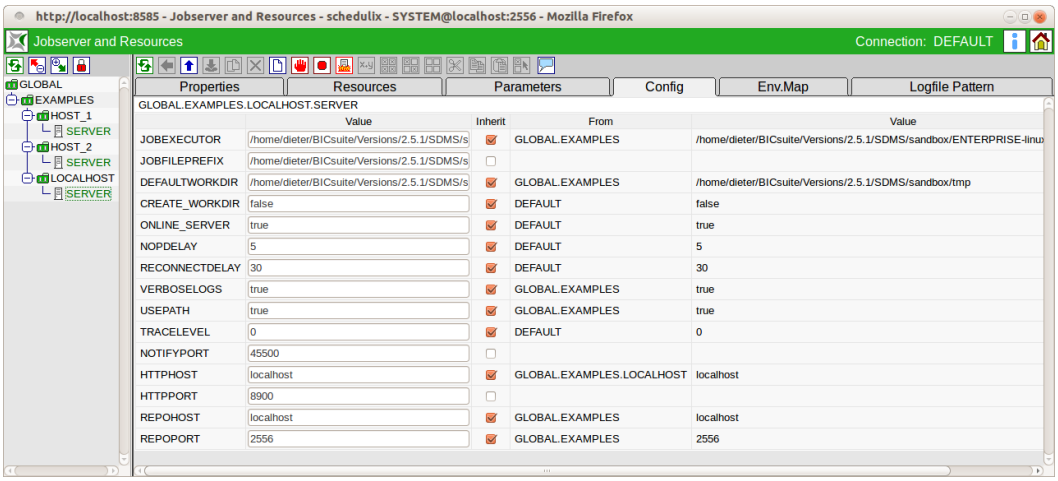


Abbildung 12.10: Jobserver und Scope Konfiguration

Die Felder des Tabs ”Config” haben folgende Bedeutung:

Value Die *Value* zeigt den jeweiligen Key/Value Pair an.

Inherit Das Feld *Inherit* zeigt an, ob der Key/Value Pair-Wert von einem übergeordneten Scope geerbt wird. Durch Setzen des Flags im *Inherit*-Feld wird der aktuelle Eintrag gelöscht. Der Wert des Key/Value Pairs wird in dem Fall über den Vererbungsmechanismus ermittelt. Der Eintrag ”DEFAULT” bedeutet, dass es sich um eine Voreinstellung des Systems handelt.

From Das Feld *From* zeigt an, von welchem Scope geerbt wird. Wird als Scope ”DEFAULT” angezeigt, handelt es sich um Standard Konfigurationswerte.

Value Der geerbte Wert der Key/Value Pairs.

12.4.4.1 Standard Konfigurationsparameter

Die folgende Tabelle zeigt die Namen der Konfigurationsparameter eines Jobservers, sowie ihre Bedeutung:

Name	**B**edeutung
JOBEXECUTOR	Der vollqualifizierte Pfad des Jobexecutors
JOBFILEPREFIX	Pfad und Fileprefix der Taskfiles
DEFAULTWORKDIR	Working Directory des Jobservers und default working Directory der über diesen Jobserver ausgeführten Jobs
ONLINE_SERVER	Boolean Wert. Wenn "False", meldet der Jobserver sich ab, wenn gerade keine Arbeit vorliegt
NOPDELAY	Zeit zwischen zwei Anfragen nach Jobs beim Scheduling Server
RECONNECTDELAY	Zeit zwischen zwei Versuchen, sich beim Scheduling Server an zu melden
VERBOSELOGS	Wenn "True", werden zusätzlich zu den Prozessausgaben auch Start- und Endzeit der Jobs in ihren Logfiles geschrieben.
USEPATH	Dieser Parameter definiert, ob der Jobserver beim Starten von Prozessen die Pfadeinstellung nutzt. Steht "USEPATH" auf "False", müssen alle Programmaufrufe vollqualifiziert sein.
TRACELEVEL	Definiert, wie "gesprächig" der Jobserver ist • 0: Fehler werden protokolliert • 1: Fehler und Warnings werden protokolliert • 2: Fehler, Warnings und Info werden protokolliert • 3: Debug Level; alle Meldungen werden protokolliert
NOTIFYPORT	Der Notifyport ist der Port, dem ein UDP-Paket geschickt wird, wenn ein Job für den Jobserver zur Ausführung bereit steht.
HTTPHOST	Der Jobserver kann über das HTTP Protokoll Logfiles übermitteln. Dazu müssen beide Parameter "HTTPHOST" und "HTTPPORT" gesetzt sein.
HTTPPORT	Port für die Übermittlung von Logfiles
REPOHOST	Hostname oder IP-Adresse des Scheduling Servers
REPOPORT	Port des Scheduling Servers

Tabelle 12.2: Beschreibung der Jobserver

12.4.5 Tab Env.Map

Beim Starten eines Jobs werden eine Anzahl Standard Parameter des Jobs dem Jobserver als Key/Value Pairs übergeben. Diese Parameter können vor dem Start des Prozesses als Umgebungsvariablen gesetzt werden. Ob und unter welchem Namen diese Variablen sichtbar werden, ist über die Maske "Env.Mapping" konfigurierbar.

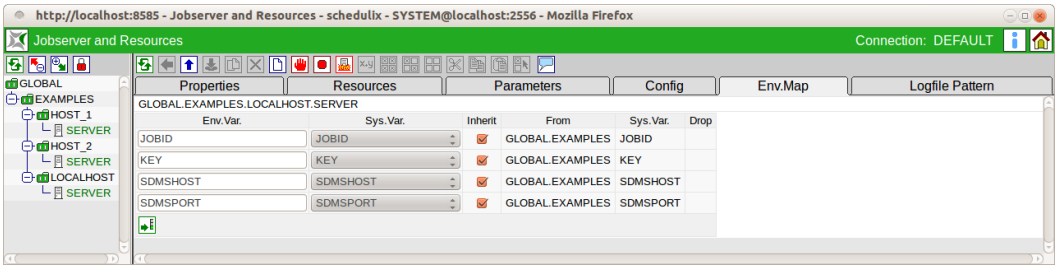


Abbildung 12.11: Jobserver Environment Mapping

Die Spalten der obigen Liste haben folgende Bedeutung:

Env.Var. Im Feld *Env.Var.* steht der Name der Umgebungsvariablen.

Sys.Var. Im Feld *Sys.Var.* steht der Name des Parameters.

Inherit Das Feld *Inherit* zeigt an, ob der Wert des Key/Value Paares von einem übergeordneten Scope geerbt wird. Durch Setzen des Flags im *Inherit*-Feld wird der aktuelle Eintrag gelöscht. Der Wert des Key/Value Pairs wird in diesem Fall über den Vererbungsmechanismus ermittelt.

From Das Feld *From* zeigt an, von welchem Scope geerbt wird. Wird als Scope DEFAULT angezeigt, handelt es sich um Standard Konfigurationswerte.

12.4.6 Tab Logfile Pattern

Die Jobserver sind in der Lage, die Inhalte von Logfiles über das HTTP-Protokoll zu übermitteln. Um zu verhindern, dass beliebige Dateien unkontrolliert gelesen werden, dürfen nur Dateien, deren Name bestimmte Muster aufweisen, angefordert werden.
Zulässige Muster müssen vom Administrator eingetragen werden. Wird dies unterlassen, können keine Dateien angefordert werden.

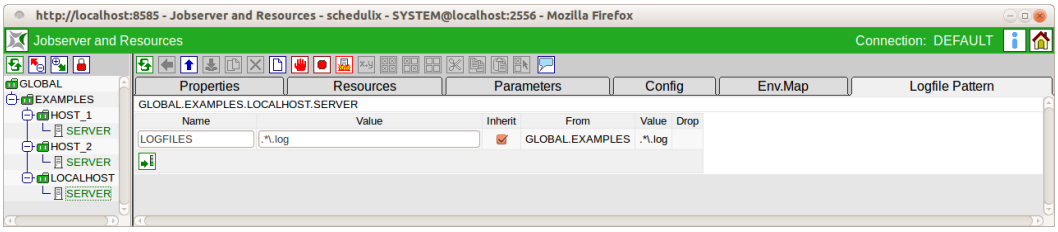


Abbildung 12.12: Jobserver Logfile Name Patterns

Die Logfile Patterns sind Regular Expressions mit einer Besonderheit. Die Zeichenfolge "`/../`" bzw. "`\..\`" (Directory up) darf generell nicht im Namen einer angeforderten Datei auftreten. Diese Regel vereinfacht es dem Administrator, die Logfile Patterns zu definieren.
Generell wird empfohlen, den gesamten Namen zu testen. Ein Pattern wie "`.*\.log`" erlaubt es alle Dateien zu lesen, in deren Namen irgendwo die Zeichenfolge "`.log`" auftritt. Dagegen führt das Pattern "`.*\.log$`" bereits zu der wichtigen Einschränkung, dass der Name auf "`.log`" enden muss.
Im Screenshot wird das Muster "`^/tmp/.*\.log$`" benutzt. Alle Dateien unterhalb des Verzeichnisses `/tmp` die auf `.log` enden, dürfen angefordert werden.
Das Anfordern von Logfiles, die keinem der Muster entsprechen, werden als Fehler im Logfile des Jobservers protokolliert. Die Meldungen sehen etwa folgendermaßen aus:

```
ERROR [Jobserver]  01-12-2009 16:43:43 CET [HttpThread] ERROR: Illegal file request : /etc/passwd
```

mit zusätzlicher Information (die bei jedem File-Request protokolliert wird):

```
...[HttpThread] Got Request from 1.2.3.4 : GET /?FNAME=/etc/passwd HTTP/1.1
...[HttpThread] Got Request from 1.2.3.4 : Host: localhost:8881
...[HttpThread] Got Request from 1.2.3.4 : User-Agent: Mozilla/5.0 (X11; U; Linux x86_64; en-US; rv:1.9.0.15)
     Gecko/2009102815 Ubuntu/9.04 (jaunty) Firefox/3.0.15
...[HttpThread] Got Request from 1.2.3.4 : Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
...[HttpThread] Got Request from 1.2.3.4 : Accept-Language: en-us,en;q=0.5
...[HttpThread] Got Request from 1.2.3.4 : Accept-Encoding: gzip,deflate
...[HttpThread] Got Request from 1.2.3.4 : Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
...[HttpThread] Got Request from 1.2.3.4 : Keep-Alive: 300
...[HttpThread] Got Request from 1.2.3.4 : Connection: keep-alive
...[HttpThread] Got Request from 1.2.3.4 : Cookie: _ZopeId="55722729A4JSGHnrljM"; tree-s="..."
...[HttpThread] Got Request from 1.2.3.4 : Cache-Control: max-age=0
```

Der Anwender bekommt eine Fehlermeldung angezeigt:

```
ERROR: The requested filename doesn't match any of the configured patterns
```

12.5 Resource Links

Resource Links erlauben die Allocation von Resources in einem anderen Scope als dem, in welchem ein Job ausgeführt wird.
Werden zum Beispiel in einer Client/Server Umgebung Operationen auf einem Datenbankserver ausgeführt, so werden die Resources, welche die Datenbankoperation benötigt, nicht auf dem Client Rechner, auf dem der Job ausgeführt wird belegt, sondern auf dem Rechner, auf dem der Datenbankserver liegt.
Resource Links erlauben die Abbildung einer Resource eines Scopes in einem anderen Scope.
Ein Resource Link wird erzeugt, in dem eine Resource mit Usage STATIC, SYSTEM oder SYNCHRONIZING aus einem Scope bzw. Server ins Clipboard kopiert und in einem anderen Scope eingefÃijgt wird.
Ist beim Paste eine Resource und/oder ein Resource Link im Clipboard wird folgende Maske angezeigt, auf der ausgewählt werden kann, wie die Paste Operation ausgeführt werden soll.

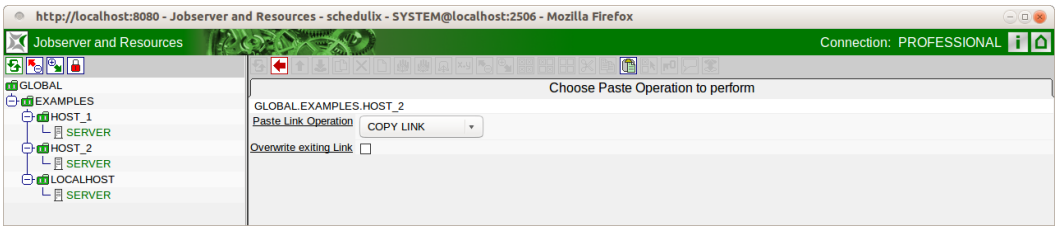


Abbildung 12.13: Resource Links kopieren und erstellen

Ist beim Paste eine Resource im Clipboard so wird das Feld *Paste Resource Operation* angezeigt.
Folgende Auswahlen sind möglich:

- COPY RESOURCE: Die Resource wird kopiert.
- LINK TO RESOURCE: Ein Resource Link wird angelegt.

Action Flag
Ist beim Paste ein Resource Link im Clipboard so wird das Feld *Paste Link Operation* angezeigt.
Folgende Auswahlen sind möglich:

- COPY RESOURCE: Die Resource auf die der Link verweist wird kopiert.
- COPY LINK: Der Link wird kopiert.
- LINK TO LINK: Ein Resource Link der auf den Link verweist wird angelegt.

Ist beim Paste ein Resource Link im Clipboard wird zusätzlich das Feld *Overwite existing Link* angezeigt. Ist diese Checkbox gesetzt, so wird, falls ein Link gleichen Namens bereits existiert, keine Fehlermeldung ausgegeben, sondern der Link überschrieben bzw. geändert.
Resource Links werden in der Resources-Liste mit einem Pfeil im Icon angezeigt wird.

12.5.1 Tab Resource Detail

Der Tab "Resource Details" eines Resource Links zeigt die Details der über den Resource Link angesprochenen tatsächlichen Resource. Wird der Resource Link editiert, so werden die Änderungen auf die tatsächlich angesprochene Resource angewendet.

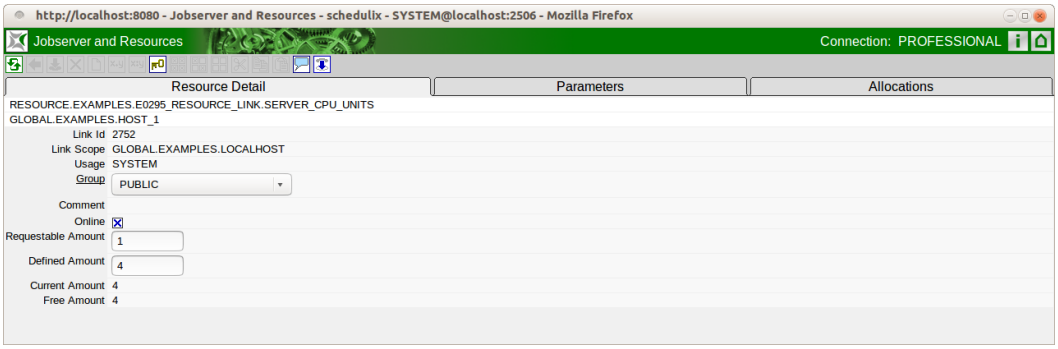


Abbildung 12.14: Resource Link Information

Neben den schon erklärten Attributen der zugrunde liegenden Resource werden folgende 'read only'-Felder angezeigt:

Link Id Die ID der Resource, auf die der Link verweist

Link Scope Der Scope, in dem die verlinkte Resource liegt

13 Batches und Jobs

13.1 Bild

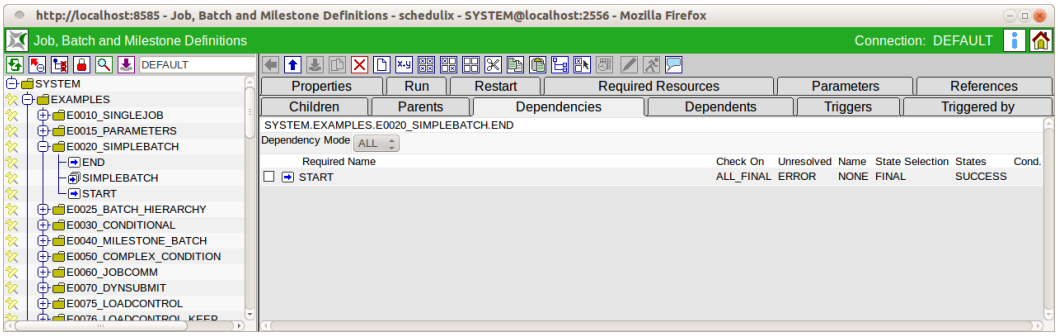


Abbildung 13.1: Batches und Jobs

13.2 Konzept

13.2.1 Kurzbeschreibung

Der Dialog Batches and Jobs dient der Verwaltung aller Job Definitions im schedulix Scheduling System.
Die Objekte ”Batches”, ”Jobs” und ”Milestones” dienen der Definition der auszuführenden Einheiten des Scheduling Systems.
Ein Job ist die Hülle für ein auszuführendes Programm oder Skript im Scheduling System. Wird ein Job submitted und gestartet, so führt der Jobserver das angegebene Programm oder Skript aus und gibt eine Rückmeldung nach Beendigung des Prozesses über den Erfolg oder Misserfolg (Exit State).
Ein Batch ist ein Container für andere Objekte, er enthält sogenannte Children. Durch Starten des Batches werden automatisch alle enthaltenen Children ebenfalls gestartet. Ein Batch hat kein eigenes Programm oder Skript, welches ausgeführt wird.
Der Batch (oder Job), welcher im Dialog Submit Batches and Jobs als erstes Objekt gestartet wird, wird Master Job genannt.
Ein Milestone ist ein Objekt, welches zur Benachrichtigung bei Erreichen (oder Nichterreichen) einer gewissen Anzahl von abgeschlossenen Jobs oder zur Ab-

bildung von komplexen Abhängigkeiten dienen kann. Ein Milestone hat ebenfalls kein lauffähiges Programm oder Skript.

13.2.2 Ausführliche Beschreibung

Jedes Scheduling Entity (Batch, Milestone oder Job) kann in einer Parent-Child-Hierarchie in einem Batch-Lauf stehen. Ein Job wird gestartet, wenn sein Parent gestartet wurde. Das bedeutet, beim Submit einer solchen Ausführungsobjekt-Hierarchie, werden alle Jobs von oben nach unten submitted. Dies geschieht jedoch innerhalb einer Transaktion, sodass alle Aktionen, von außen gesehen, gleichzeitig erfolgen.
Jedes Ablaufobjekt (Batch, Milestone oder Job) kann Abhängigkeiten besitzen. Eine Abhängigkeit beschreibt, welches Ausführungsobjekt (required Job) vorher gelaufen sein muss. Soll nur ein bestimmter Exit State beachtet werden, kann der abhängige (dependent) Job erst starten, wenn der benötigte (required) Job vollständig gelaufen ist und den richtigen Exit State zurückgemeldet hat.
Hiermit sind Abhängigkeitsmodelle möglich, die sicherstellen, dass wirklich alle Vorarbeiten erfolgreich durchgeführt werden konnten, um einen nachfolgenden Prozess (z. B. einen Report) zu starten.
Jedes Ablaufobjekt kann verschiedene Resources benötigen. Diese können durch die Eingabe des Environments, eines Footprints und durch die Benennung der benötigten Resources festgelegt werden.
Dies stellt sicher, dass die zur Ausführung des Jobs verwendete Laufzeitumgebung, alle benötigten Resources (z. B. genügend Hauptspeicher, genügend CPU Einheiten, richtige Datenbank, benötigte Systemprogramme etc.) zur Verfügung hat.
Darüber hinaus kann mittels Synchronizing Resources sichergestellt werden, dass Prozesse sich nicht gegenseitig behindern oder das Resources sich in bestimmten States befinden und zeitlich aktuell sind. Über die Resources kann darüber hinaus eine Lastverteilung erreicht werden.
Jedes Scheduling Entity kann bestimmte Parameter definieren oder verwenden. Hiermit ist eine Datenübergabe zwischen einzelnen Jobs bzw. aus der Laufzeitumgebung möglich. Die Parameterübergabe kann innerhalb der Parent-Child-Beziehung in beide Richtungen (von Parent zu Child, von Child zu Parent) stattfinden.
Es ist aber ebenfalls möglich, auch auf Parameter von Scheduling Entities zuzugreifen, welche sich außerhalb der Parent-Child-Beziehung befinden. Hiermit ist eine übergreifende Datenübermittlung zwischen allen Scheduling Entities eines Master Jobs möglich. Weitere Informationen zu Parametern finden Sie im Kapitel 13.5.10.
Für ein Ablaufobjekt können beim Eintritt eines bestimmten Ereignisses (zum Beispiel Beendigung des Jobs) und beim Erreichen bestimmter Exit States, Aktionen definiert werden. Diese Aktionen werden Trigger genannt. Ein Trigger definiert, welche Aktion beim Eintreten eines bestimmten Ereignisses ausgeführt werden

soll. Die Aktion, die ausgelöst wird, ist wieder ein Ausführungsobjekt.
Tritt der Fall ein, dass dieser Trigger ausgelöst wird (er feuert), so wird dieser Job (oder Batch) submitted. Damit kann zum Beispiel eine Benachrichtigung an einen Systemadministrator erfolgen.

13.2.3 Empfohlene Konvention für Batch-Objekte

Um die Übersichtlichkeit zu verbessern und das Arbeiten mit Batch-Objekten zu vereinfachen, empfehlen wir unseren Anwendern, sich an folgende Konvention zu halten.
Ein Batch-Objekt sollte in einem Ordner mit gleichem Namen liegen. Dieser Ordner sollte den Batch und dessen Children (Sub Batches und Jobs) enthalten. Sub Batches sollten wiederum in einem eigenen Unterordner liegen. Job-Objekte, welche Children haben, sollten wie Batches behandelt werden, also auch in eigenen Unterordnern liegen.
Diese Konvention muss nicht zwingend eingehalten werden. Der Aufwand bei der Vergabe zu Zugriffsrechten, dem Deployment, Kopieren und Verschieben von Teilabläufen etc. kann so jedoch deutlich reduziert und die Orientierung erheblich verbessert werden.
Um die Umsetzung dieser Konvention zu erleichtern, wurden folgende Funktionserweiterungen in der schedulix!Web Oberfläche umgesetzt:

- Beim Anlegen eines Ordners kann auch gleich ein Batch-Objekt in diesem Ordner mit angelegt werden.
- Beim Anlegen eines Batches kann auch gleich ein übergeordneter Ordner für den Batch angelegt werden.
- Beim Umbenennen von Batches oder Ordnern kann auch der gleichnamige Ordner bzw. Batch mit umbenannt werden.
- Beim Erzeugen von Objekten in einem Ordner mit gleichnamigem Batch oder Job kann das neue Objekt gleich als Child in den Batch bzw. Job aufgenommen werden.
- Beim Duplizieren (Clonen) von Jobs kann der neue Job als Child in die gleichnamigen Batches bzw. Jobs des Ordners aufgenommen werden.
- Beim Duplizieren (Clonen) von Foldern kann der gleichnamige Batch bzw. Job des neuen Folders automatisch umbenannt werden.
- Beim Duplizieren (Clonen) von Foldern kann der gleichnamige Batch bzw. Job des neuen Folders als Child in die gleichnamigen Batches bzw. Jobs des übergeordneten Ordners aufgenommen werden.

13.3 Navigator

Im Navigator werden hierarchisch alle Objekte vom Typ ”Batch”, ”Job” und ”Milestone” innerhalb von Containern sog. Folders verwaltet. Die Folder dienen zur besseren Übersichtlichkeit der Objekte.
Darüber hinaus können in ihnen Parameter und ein Default Environment definiert werden. Allen Jobs, welche sich innerhalb eines Folders befinden, stehen alle Parameter des Folders zur Verfügung. Das Environment des Folders verhält sich additiv zum Environment eines Jobs.

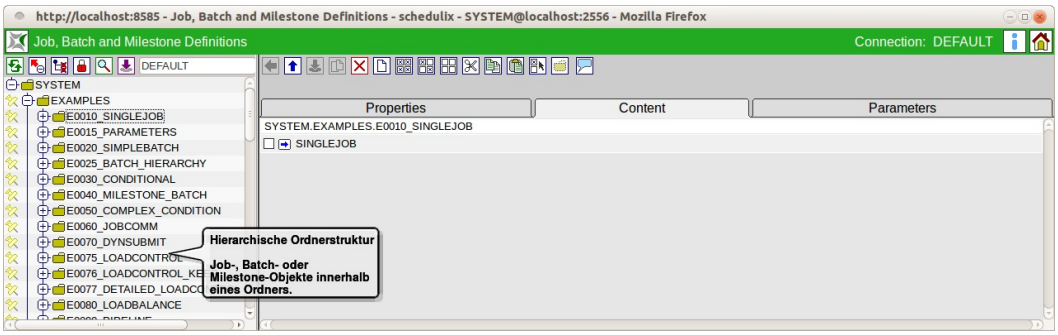


Abbildung 13.2: Batches und Jobs Navigator

13.3.1 Pinning

Um im Folder Navigator eine bessere Übersichtlichkeit zu erreichen, können Objekte durch einen Mausklick auf das *Pin* Icon  ”gepinnt” werden.

Ein Pin kann durch einen Mausklick auf das *Unpin* Icon  wieder entfernt werden.

Zeilen mit einem ”Pin” werden immer angezeigt, unabhängig davon, ob die übergeordneten Folder expandiert sind oder nicht. Der Anwender muss damit im Navigator wesentlich weniger blättern.

Die ”Expand” und ”Collapse” Icons der Baumansicht werden gelb eingefärbt, wenn sich unter einem Folder ein ”gepinntes” Objekt befindet. Damit wird verdeutlicht, dass ein ”Collapse” nicht zum kompletten Schließen des Ordners führt, sondern der Pfad zum ”gepinnten” Objekt immer sichtbar bleibt.

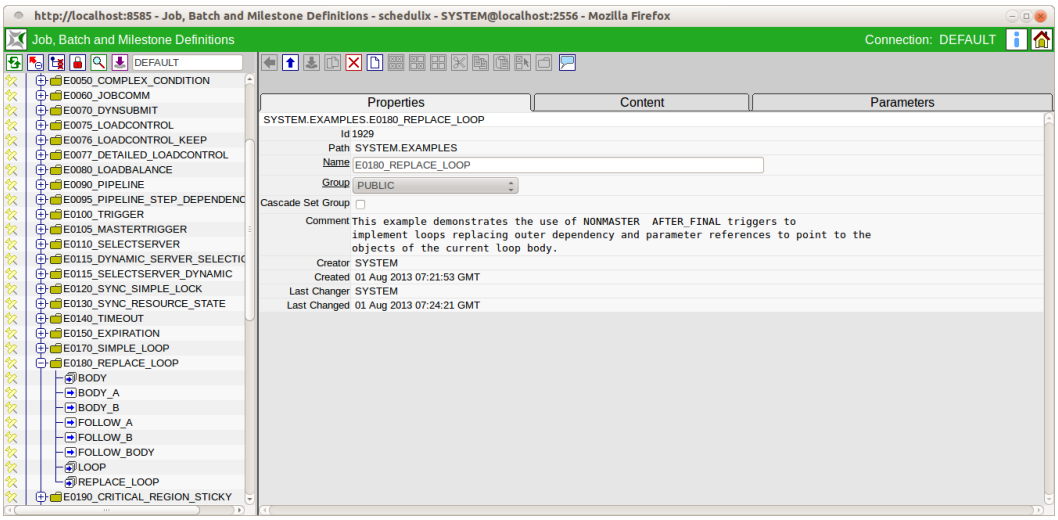


Abbildung 13.3: Batches und Jobs; Ansicht ohne Pin

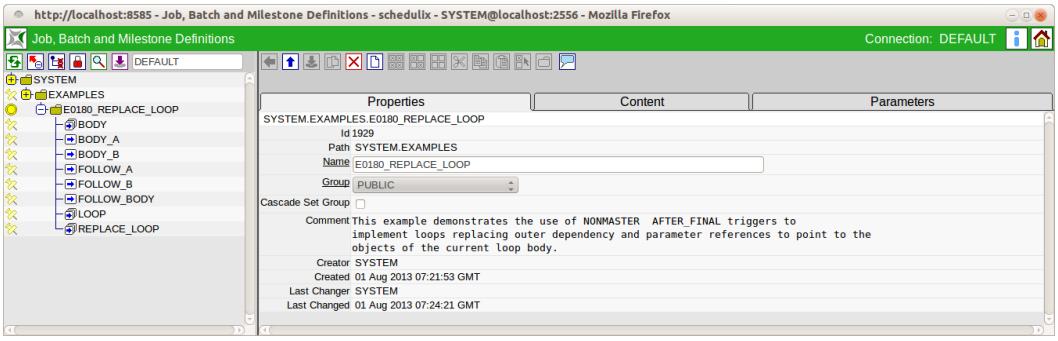


Abbildung 13.4: Batches und Jobs; Ansicht mit Pin

13.3.2 Folder Bookmarks

 **Store Bookmark**
Betätigt man den *Store Bookmark*-Button, so wird ein Bookmark für die aktuellen Einstellung des Navigators angelegt.

Die Navigationseinstellungen enthalten die Informationen über die Anzeige von Jobs und Batches sowie gesperrten Objekten, den aktuellen Expansions-Status der Baumansicht und "gepinnte" Objekte.

Der Name, unter dem der Bookmark gespeichert werden soll, kann im Eingabefeld rechts vom *Store Bookmark* Button eingegeben werden.

Der Bookmark "DEFAULT" wird beim Aufruf von "Batches and Jobs" aus dem schedulix!Web Desktop verwendet.

Andere Bookmarks können über Bookmark (Folder) aufgerufen werden.

Falls der aktuelle User ein "Web GUI ADMIN" ist, so wird neben dem Namen des Bookmarks noch ein Optionsfeld eingeblendet, über welches eingestellt werden kann, ob der Bookmark als System Bookmark (für alle User sichtbar) oder als User Bookmark (nur für den aktuellen User sichtbar) gespeichert werden soll.

13.3.3 Folder Search

 **Find**
Betätigt man den *Find* Button, wechselt der Navigator in den Suchmodus.

Im Eingabefeld kann ein Suchmuster eingegeben werden, welches der, im Optionsfeld rechts vom Eingabefeld, gewählten Syntax entspricht. Unterstützt werden dabei die SQL LIKE Syntax sowie "Regular Expressions".

Über ein weiteres Optionsfeld kann bestimmt werden, ob das Muster auf den gesamten Pfad (FULL) oder nur auf das Pfadende (TAIL) angewendet werden soll.
Betätigt man im Suchmodus den *Find* Button, so werden im Navigator die Treffer angezeigt.

Ein Mausklick auf einen Treffer öffnet das Objekt im Editorbereich, das Objekt wird "gepinnt" und der Suchmodus wird wieder verlassen.

13.4 Editor für Folder

13.4.1 Tab Properties

Der ”Properties” Tab dient zum Erfassen aller Informationen, welche die allgemeinen Eigenschaften des Folders abbilden.

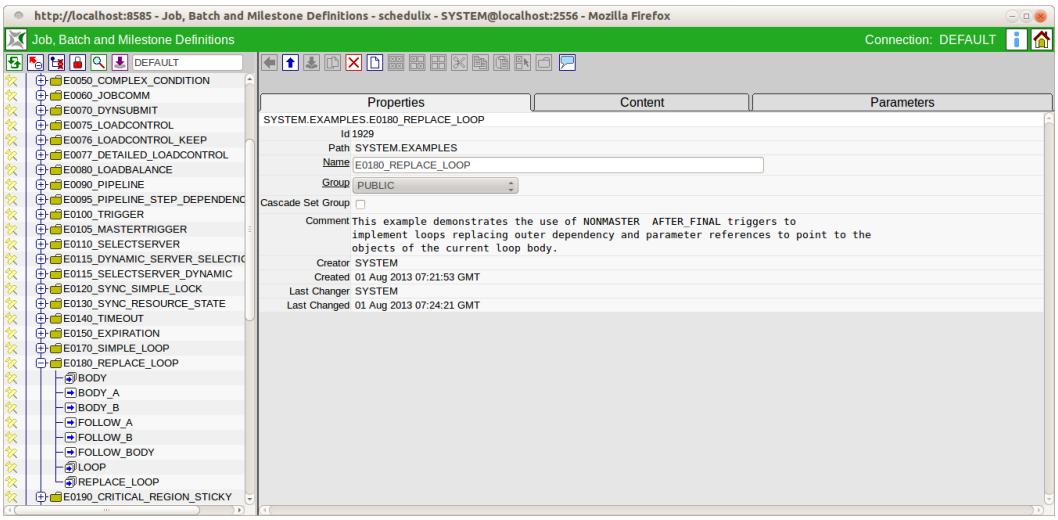


Abbildung 13.5: Folder Properties

Id Die *Id* beschreibt die eindeutige Identifikationsnummer des Objektes.

Path Der *Path* beschreibt die darüber liegende Ordnerhierarchie. Alle übergeordneten Ordner werden punktgetrennt angezeigt.

Name Durch den Namen soll eine verständliche und eindeutige Identifizierung des Objektes im aktuellen Kontext möglich sein. Er muss innerhalb seines Ordners eindeutig sein. Gleichlautende Namen in unterschiedlichen Ordnern sind zulässig.

Rename in Content Wird der Name eines Folders geändert und folgt der Folder der ”Batch in Folder Konvention” (enthält ein gleichnamiges Batch oder Job-Objekt), so wird die Checkbox *Rename in Content* sichtbar. Wird diese gesetzt, so wird beim Umbenennen und Duplizieren (Clonen) des Folders auch das gleichnamige Batch oder Job-Objekt im Folder mit umbenannt.

Group Das Feld *Group* gibt die schedulix Benutzergruppe an, die Eigentümer des Objektes ist.

Cascade Set Group Mit dem *Cascade Set Group* setzt man die Benutzergruppe des Folders und allen darunter liegenden Objekte.

Create Batch in Folder Die Checkbox *Create Batch in Folder* wird bei der Neuanlage eines Folders angezeigt und dient der Unterstützung der empfohlenen Konvention für Folder, Batch und Job-Objekte. Ist das Feld angekreuzt, so wird mit dem Folder auch gleich ein Batch-Objekt gleichen Namens in dem neu angelegten Folder angelegt.

Batch Exit State Profile Das Feld *Batch Exit State Profile* wird nur angezeigt, wenn die Checkbox *Create Batch in Folder* gesetzt ist und dient der Auswahl des Exit State Profiles für den neu anzulegenden Batch.

Add as Child Die Checkbox *Add as Child* wird in folgenden Fällen angezeigt.
Bei der Neuanlage ist die Checkbox *Create Batch in Folder* gesetzt und der Parent Folder des neu anzulegenden Ordners folgt der Batch in Folder Konvention (enthält ein gleichnamiges Batch oder Job-Objekt).
Bei einem bestehenden Folder wird der Name geändert und sowohl der Folder selbst, als auch sein Parent Folder folgen der Batch in Folder Konvention (enthalten ein gleichnamiges Batch oder Job-Objekt). In diesem Fall hat die Checkbox *Add as Child* nur eine Auswirkung beim Duplizieren (Clonen) des Folders. Beim Speichern (Umbenennen) des Folders wird die Checkbox *Add as Child* ignoriert.
Mit der Checkbox *Add as Child* kann gewählt werden, ob der neu angelegte Batch (bei Neuanlage) bzw. kopierte Batch bzw. Job (beim Duplizieren bzw. Clonen) auch als Child des übergeordneten Folder-Batch bzw. -Job eingehängt werden soll.

Environment Das Feld *Environment* ist eine Auswahlbox, in der ein gültiges Environment-Objekt ausgewählt werden kann. Diese Auswahl definiert die zusätzlichen Anforderungen die alle Jobs aus dem Folder an ihre Ablaufumgebung stellen.

Comment Das Feld *Comment* dient als Möglichkeit einer näheren Erläuterung (Kommentar) des Objektes.

13.4.2 Tab Content

Im Tab ”Content” wird der Inhalt des Folders angezeigt.

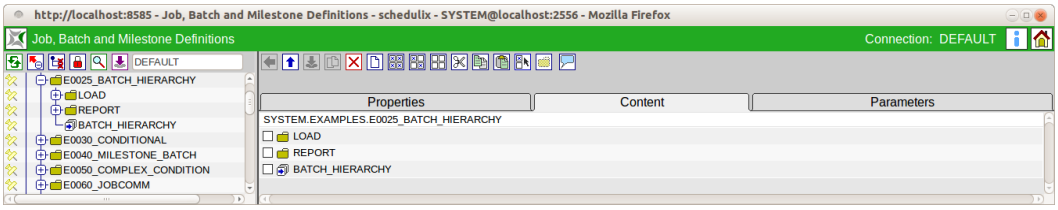


Abbildung 13.6: Folder Content

Beim Copy/Cut und nachfolgendem Paste in Content Tabs von Foldern, wird je nachdem, ob Quelle und Ziel der Operation der Konvention (siehe auch ntspricht, eine Liste eingeblendet, welche die Operationen auflistet, welche an den Parent-Child-Beziehungen durchgeführt werden, um die Konvention einzuhalten. Diese Operationen können dann vor der Ausführung der Paste Operation gezielt ausgewählt werden.

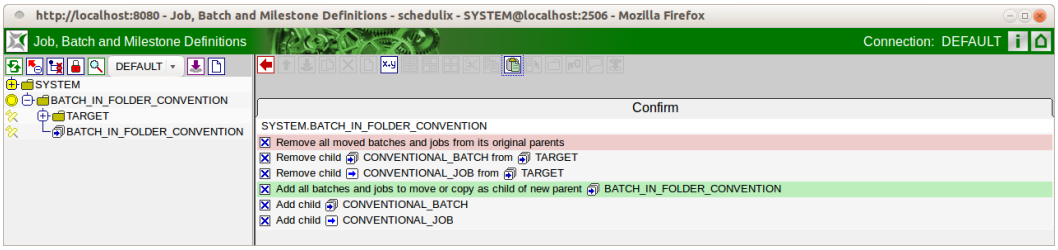


Abbildung 13.7: Unterstützung der Konvention bei Paste

13.4.3 Tab Parameters

Im Tab ”Parameters” können Parameter definiert werden.

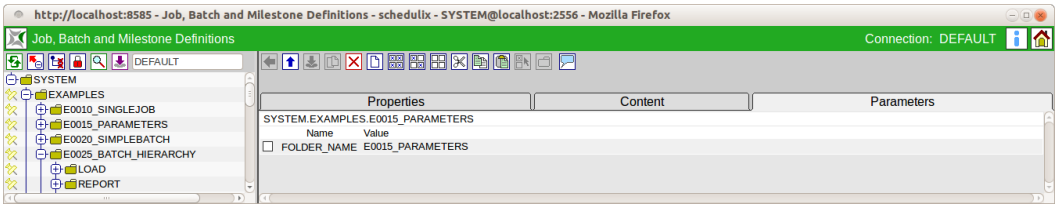


Abbildung 13.8: Folder Parameters

Name Hierbei handelt es sich um den Namen des Parameters. Durch den Namen erfolgt die komplette Werteübergabe der Parameter zwischen einzelnen Jobs.

Value Der *Value* zeigt den Wert des Parameters an.

13.4.4 Tab Resources

Im Tab "Resources" werden die Resources angezeigt. Durch Anklicken des Namens der Resource kommt man in den Tab Resource Details.

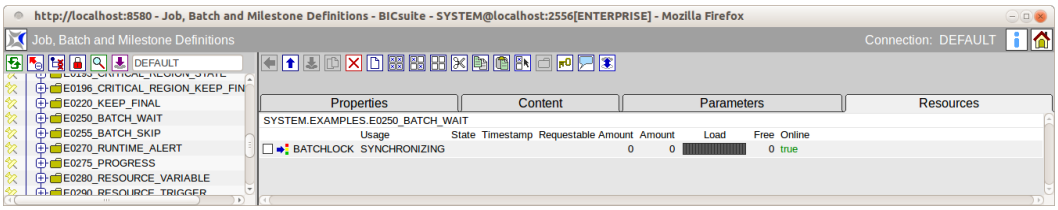


Abbildung 13.9: Folder Resources

Usage Die *Usage* gibt an, um welchen Typ "Resource" es sich handelt. Mehr zu Typen von Resources finden Sie im Dialog Named Resource.

State Das Feld *State* ist nur zu sehen, wenn die Usage der Resource "Synchronizing" ist und ein Resource State Profile zugeordnet wurde.
Hierbei handelt es sich um den aktuellen Status der Resource in diesem Scope oder Jobserver. Der Status kann mittels dieses Wertes gesetzt bzw. geändert werden. In der "Drop Down" Liste sind alle gültigen Resource States des Resource State Profiles enthalten und können ausgewählt werden.
Dies kann bei einer manuellen Fehlerbehebung notwendig sein, um eine Resource wieder in einen Status zu bringen, in dem eine Weiterverarbeitung durch Folgejobs möglich ist.
Falls hier der Wert manuell geändert wird, wird der Wert des Feldes *State* im Tab "Resources" nicht automatisch aktualisiert. Falls dies gewünscht und notwendig ist, kann dies mittels des *Refresh* Buttons durchgeführt werden.

Timestamp Dieses Feld ist nur zu sehen, wenn die Usage der Resource "Synchronizing" ist und ein Resource State Profile zugeordnet wurde. Der *Timestamp* gibt die Zeit des letzten Statuswechsels einer Resource an.
Timestamps spielen eine Rolle, falls in einem Ablaufobjekt ein Expiration-Intervall angegeben wurde. Ist dies geschehen, darf der Timestamp nicht älter sein als das angegebene Intervall. Mehr zu Expiration-Zeiten finden Sie im Kapitel 13.5.8.1.
Wird durch die Beendigung eines Jobs eine Änderung des Resource State hervorgerufen, wird der Timestamp automatisch vom schedulix Server aktualisiert.

Requestable Amount Die Menge der Resources, die maximal von einem Job angefordert werden darf.

Amount Beim *Amount* handelt es sich um die Anzahl der vom aktuellen Job belegten oder angeforderten Resource-Instanzen. Übersteigt der Wert des Amounts die aktuell verfügbare Anzahl, die im Feld *Amount* im Tab "Resource Detail" gepflegt werden können und ist kein alternativer Scope mit einer ausreichenden Anzahl vorhanden, kann der Job nicht ausgeführt werden.

Load Der *Load* zeigt die graphische Auslastung der Resource.

Free Dieses Feld ist nur zu sehen, wenn die Usage der Resource "Synchronizing" oder "System" ist.
Der *Free Amount* bezeichnet die Anzahl aller noch nicht von Jobs belegten Instanzen einer Resource, innerhalb des gewählten Scopes oder Jobservers.

Online Mittels *Online* ist es möglich, eine Resource in diesem Scope oder Jobserver on- bzw. offline zu schalten. Ist die Resource offline, so steht diese Resource vorübergehend nicht zur Verfügung.

13.5 Editor für Job Definitions

13.5.1 Tab Properties

Der Tab "Proberties" dient zum Erfassen aller Informationen, welche die allgemeinen Eigenschaften des Job-, Batch-, Milestone- oder Ordnerobjektes abbilden. Je nach Art des in der Navigation oder während der Neuanlage gewählten Objekttyps, werden im "Properties" Tab nur die jeweils notwendigen Informationen angezeigt.

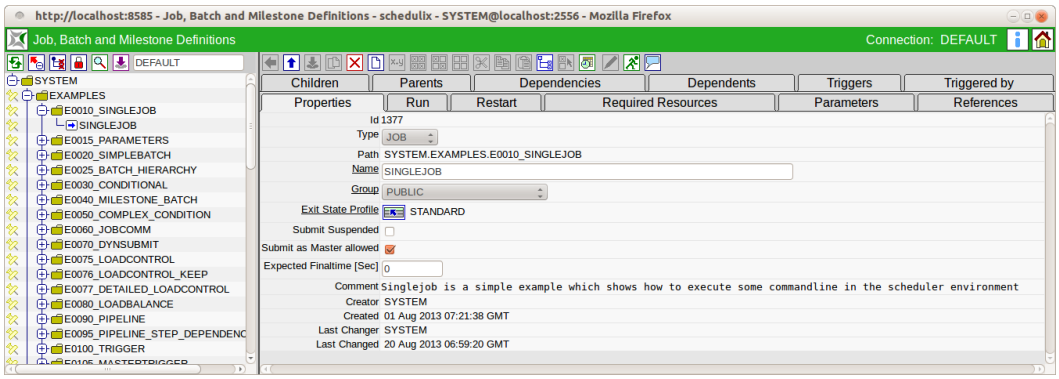


Abbildung 13.10: Job Definition Properties

Die Felder des Tabs "Properties" haben folgende Bedeutung:

Id Die *Id* beschreibt die eindeutige Identifikationsnummer des Objektes.

Type Der *Type* gibt die Art eines Objektes an. Hier kann aus den folgenden Arten ausgewählt werden:

1. Batch

   Der Typ des Objektes ist ein Batch. Ein Batch dient als startbarer Container für eine Anzahl von Jobs, Batches oder Milestones. Er hat selber kein Run-Programm, dient also nur als Startobjekt seiner Child-Objekte.

   Als Beispiel für eine Modellierung als Batch kann eine Anzahl von Jobs dienen, welche täglich laufen müssen. Diese können nun alle unter einem Batch "Täglich" zusammengefasst werden. Alle wöchentlichen Jobs werden ebenfalls unter einem Batch "Wöchentlich" zusammengefasst, darunter auch der Batch täglich, da dieser ja auch zum selben Zeitpunkt als Batch laufen muss. Des Weiteren kann ein Batch "Monatlich" erstellt werden, der alle monatlich laufenden Jobs enthält. Somit müssen in diesem System nur diese drei Batches im Time Scheduling angelegt werden und alle Child Jobs, -Batches bzw. -Milestones werden automatisch zu diesen Zeiten gestartet.

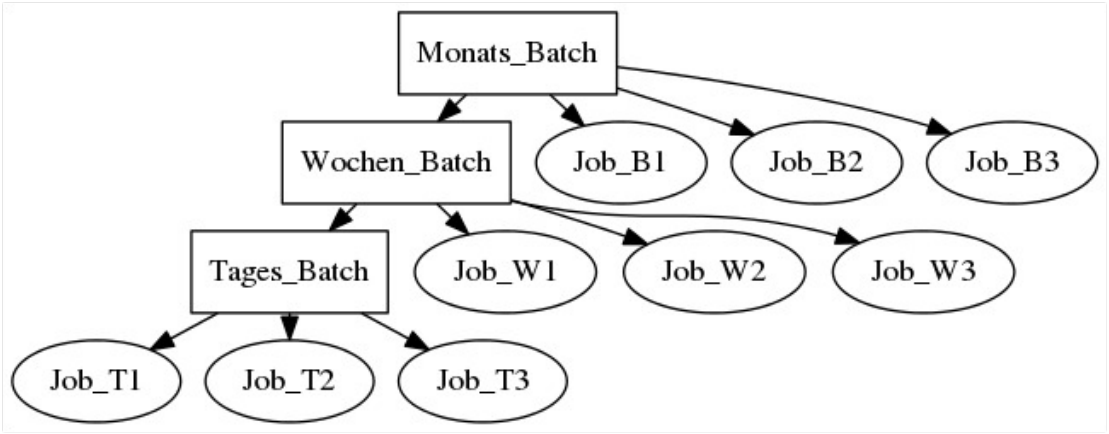


Abbildung 13.11: Batchbeispiele

Die Legende für die Grafik finden Sie im Kapitel 1.8.

2. Milestone

   Ein Milestone wird aus mehreren Gründen benötigt. Erstens dient er als Platzhalter für einen bestimmten Zeitpunkt im Ablauf eines komplexen Batch-Laufes. Wird dieser Milestone erreicht, so kann mittels eines Triggers eine Aktion durchgeführt werden (zum Beispiel eine SMS oder E-Mail gesendet werden).

   Als zweiten Grund werden Milestones benötigt um komplexe Abhängigkeiten zwischen einzelnen Jobs zu definieren. Da es für einen Job immer nur möglich ist von allen benötigten Objekten oder von mindestens einem benötigten Objekt abhängig zu sein, ist es nicht möglich komplexere Abhängigkeiten (von Job1 und Job2 oder Job3 etc.) direkt zu definieren.

   Diese Möglichkeiten können nun mittels eines Milestones definiert werden. Durch die Modellierung eines Milestone1, der erreicht wird, wenn Job1 und Job2 erreicht wurden, kann die Abhängigkeit unseres Jobs von Job3 oder Milestone1 modelliert werden und die Bedingung ist nun abbildbar.

   Die Legende für die Grafik finden Sie im Kapitel 1.8.

   Milestones haben außerdem eine einzigartige Eigenschaft. Es können für Milestones dynamische Children definiert werden. Jedes Mal, wenn ein Job, der auch Child eines Milestones ist, jetzt dynamisch submitted wird, bekommt der Milestone diesen Job ebenfalls als Child.

   Diese Funktionalität ermöglicht eine ”Pipeline”-Verarbeitung. Nehmen wir zum Beispiel an, dass Daten einer (partitionierten) Tabelle verarbeitet wer-

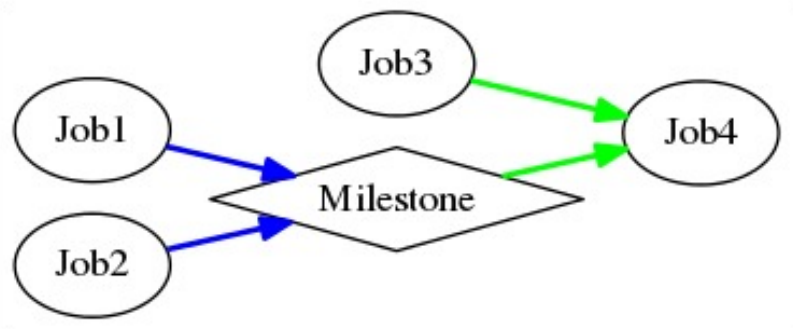


Abbildung 13.12: Beispiel für Milestone

den sollen. Aufgrund der Partitionierung wird beschlossen, die Arbeit zu parallelisieren. Für jede Partition werden die relevante Daten extrahiert und anschließend in irgendeiner Form parallel weiter verarbeitet. Sobald alle Extraktions-Jobs fertig sind, soll eine weitere Verarbeitung laufen. Wird nun der Extraktions-Job als dynamisches Child eines Milestones definiert, dann wird der Milestone genau dann den finalen Status erreichen, wenn alle Extraktions-Jobs erfolgreich beendet sind. Die weitere Verarbeitung der Tabelle kann abhängig gemacht werden vom Milestone, und damit wiederum parallel zu der Verarbeitung der extrahierten Partitionsdaten ablaufen.

3. Job

   Alle ausführbaren Objekte (Programme, Skripte etc.), welche im Scheduling System auf einem Jobserver ausgeführt werden sollen, müssen als Job definiert werden.

4. Folder

   Ein Folder stellt einen Ordner innerhalb der Ordnerhierarchie für die Ausführungsobjekte dar. Er dient zur Ordnung der anderen Objekte, da in einem normalen Scheduling System viele verschiedene Jobs, Batches und Milestones verwaltet werden und bei einer flachen Hierarchie schnell die Übersicht verloren gehen kann. Der Aufbau der Hierarchie ist dem jeweiligen Benutzer selbst überlassen. Es könnte sich eine logische Ordnung (alle Objekte zum selben Thema etc. in einem Ordner) oder eine Abbildung der Abteilungs- oder Personalstruktur (alle Objekte vom Benutzer/Abteilung X in einem Ordner) anbieten. Es sollte nur möglich sein, sich in der Hierarchie schnell zurecht zu finden und die benötigten Objekte einfach zu finden.

   Folder können nicht ausgeführt werden und können keine Abhängigkeiten besitzen. Es ist möglich ein Environment für einen Ordner zu definieren. Darüber hinaus ist es möglich Parameter zu definieren, welche alle Jobs, die sich

innerhalb des Ordners befinden, verwenden können.

Path (Pfad) Der *Path* beschreibt die darüber liegende Ordnerhierarchie. Alle übergeordneten Ordner werden punktgetrennt angezeigt.

Name Durch den Namen soll eine verständliche und eindeutige Identifizierung des Objektes im aktuellen Kontext möglich sein. Er muss innerhalb seines Ordners eindeutig sein. Gleichlautende Namen in unterschiedlichen Ordnern sind zulässig.

Decomposite Die Checkbox *Decomposite* wird angezeigt, wenn der Typ eines Objektes von Job nach Batch geändert wird. Ist diese gesetzt, so wird eine "Decomposition" ausgeführt. Dabei wird nicht nur der Objekttyp nach Batch geändert, sondern es wird auch ein Job als Child dieses Batches angelegt, welcher die Job Properties des ursprünglichen Job-Objektes übernimmt. Das nach einer Decomposition entstandene Batch-Objekt (mit einem Job als Child) verhält sich im Kontext des Scheduling System wie das zuvor existierende Batch-Objekt. Alle Time Schedules, Abhängigkeiten, Parameter References. etc. bleiben erhalten. Es entsteht eine optimale Ausgangssituation für die Zerlegung des Jobs in kleinere Einzelschritte (Process Decomposition).

Job Name Das *Job Name* wird nur bei gesetzter *Decomposite* Checkbox angezeigt. Hier kann der Name des bei der Decomposition anzulegenden Job-Objektes bestimmt werden.

Rename Parent Folder Wird der Name eines Batches geändert und folgt der Batch der "Batch in Folder Konvention" (liegt in einem gleichnamigen Folder), so wird die Checkbox *Rename Parent Folder* sichtbar. Wird diese gesetzt, so wird beim Speichern (Umbenennen) des Batches auch der gleichnamige Parent Folder mit umbenannt. Bei Duplizieren (Clonen) eines Batches wird die Checkbox *Rename Parent Folder* ignoriert.

Create Batch Folder Die Checkbox *Create Batch Folder* wird angezeigt, wenn ein Batch neu angelegt wird oder der Typ eines Objektes nach Batch geändert wird. Sie dient der Unterstützung der empfohlenen Konvention für Folder, Batch und Job-Objekte. Ist das Feld angekreuzt, so wird für den Batch ein Folder mit dem Namen des Batches erzeugt und der neue Batch in diesem Folder angelegt. Die Checkbox *Create Batch Folder* wird auch angezeigt, wenn der Typ eines Objektes von Job auf Batch geändert wird (Decomposition).

Add as Child Die Checkbox *Add as Child* wird in folgenden Fällen angezeigt:
Bei der Neuanlage eines Batch-, Job- oder Milestone-Objektes, wenn der Parent Folder des anzulegenden Objektes der Batch in Folder Konvention (enthält ein gleichnamiges Batch oder Job-Objekt) folgt.
Bei einem bestehenden Job oder Milestone wird der Name geändert und der Parent Folder folgt der Batch in Folder Konvention. In diesem Fall hat die Checkbox *Add as Child* nur eine Auswirkung beim Duplizieren (Clonen) des Jobs oder Milestones. Beim Speichern (Umbenennen) wird die Checkbox *Add as Child* ignoriert.
Mit der Checkbox *Add as Child* kann gewählt werden, ob das neu angelegte Objekt (bei Neuanlage) bzw. kopierte Job bzw. Milestone (beim Duplizieren bzw. Clonen) auch als Child des übergeordneten Folder-Batch bzw. -Job eingehängt werden soll.

Group Das Feld *Group* gibt die schedulix Benutzergruppe an, die Eigentümer des Objektes ist.

Exit State Profile In diesem Auswahlfeld kann ein Exit State Profile ausgewählt werden.

Submit Suspended Der Schalter *Submit Suspended* gibt die Möglichkeit, den tatsächlichen Start eines Ablaufes zu verzögern. Zum Beispiel könnte für ein Job eine externe Genehmigung durch eine dritte Person nötig sein. Erst, wenn diese erfolgt ist, darf der Job starten.

Resume Wird nur angezeigt, falls *Submit Suspended* gesetzt wurde.
Hier kann ausgewählt werden, ob ein automatischer Resume stattfinden soll.
Es gibt folgende Möglichkeiten:

- NO: Wählt diese Funktionalität ab
- AT: Wählt einen automatischen Resume zu einem festen Zeitpunkt. Das Eingabefeld *Resume Time* wird angezeigt.
- IN: Wählt einen automatischen Resume nach Ablauf einer Zeit. Die Eingabefelder *Resume In* und *Unit* werden angezeigt.

Resume Time Wird nur angezeigt, falls im *Resume* "AT" gewählt wurde. Hier wird der gewünschte Resume Zeitpunkt im Format "YYYY-MM-DDTHH:MI:SS" eingegeben.
Das Format orientiert sich an der ISO Norm 8601 und erlaubt auch unvollständige Angaben. Die Eingabe von 'T16:00' wird den Job um 16:00 Uhr resumen (ausgehend von der aktuellen Zeit).

Resume In Wird nur angezeigt, falls im *Resume* "IN" gewählt wurde.
Hier wird angegeben, wie viele Zeiteinheiten (siehe *Unit*) bis zum Resume gewartet werden soll.

Unit Wird nur angezeigt, falls im *Resume* "IN" gewählt wurde.
Hier wird eingegeben, ob es sich bei der Eingabe im *Resume In* um Minuten (MINUTE), Stunden (HOUR), oder Tage (DAY) handeln soll.

Nice Value Das Feld ist nur sichtbar, falls es sich bei dem Objekt um einen Batch handelt.
Der *Nice Value* gibt an, mit welcher Priorität Children dieses Objektes laufen sollen. Der Nice Value gibt einen Offset an, der auf die Priorität der Children addiert wird.
Beispiel:
Besitzt der Prozess einen Nice Value von -50 und das Child hat eine Priorität von 100, dann ist die Priorität des Child-Prozesses des Batches 50. Würde der Child-Prozess alleine gestartet wäre die Priorität 100.

Submit as Master Allowed Dieses Feld ist nur sichtbar, falls es sich bei dem Objekt um einen Batch oder einen Job handelt.
Dieser Schalter gibt an, ob es möglich ist, diesen Job als Master Job im Dialog Submit Batches and Jobs zu submitten. Ist der Schalter nicht gesetzt, so ist dies nicht möglich und der Job oder Batch kann nicht eigenständig, sondern nur als Child von irgendeinem übergeordneten Job ausgeführt werden.

Comment Das Feld *Comment* dient als Möglichkeit einer näheren Erläuterung (Kommentar) des Objektes.

13.5.2 Tab Run

Der Tab ”Run” erscheint nur, falls in der Navigation ein Objekt vom Typ ”Job” handelt oder im Falle einer Neuanlage der Typ ”Job” in der Auswahlbox gewählt wurde.
Im Tab ”Run” werden alle Informationen über die auszuführenden Kommandozeilen eines Job-Objektes verwaltet.

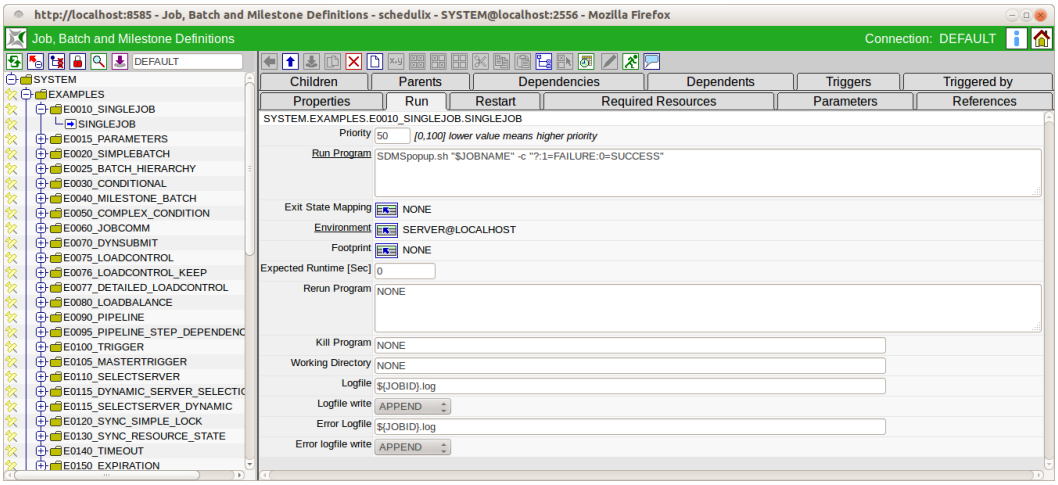


Abbildung 13.13: Job Run Tab

Die Felder des Tabs ”Run” haben folgende Bedeutung:

Priority Das Feld *Priority* gibt an, mit welcher Dringlichkeit der Prozess, falls er gestartet werden soll, vom Scheduling System berücksichtigt wird. Der Wertebereich der Priorität erstreckt sich von 100 (sehr niedrig) bis 0 (sehr hoch). Im System werden alle startbaren Jobs mit ihren Prioritäten berücksichtigt und dann der Job mit der höchsten Priorität gestartet. Haben zwei Jobs eine identische Priorität wird der Job mit niedrigster ID gestartet.
Mit der Dauer der Wartezeit eines Jobs, der darauf wartet im Scheduling System gestartet zu werden, erhöht sich seine Priorität. Das heißt, wird ein Prozess mit einer Priorität von 50 gestartet, so steigt seine Priorität pro Zeiteinheit. Dies ist im Scheduling System in der Konfiguration einstellbar. Zum Beispiel kann ein Job pro halbe Stunde um einen Prioritätspunkt steigen. Das heißt, nach einer halben Stunde ist die Priorität bei 49, nach einer Stunde bei 48 usw. Damit steigen ”ältere” Jobs in der Priorität und werden vor neu submitteten Jobs berücksichtigt.

Run Program Im Feld *Run Program* wird eine Kommandozeile angegeben, die von einem geeigneten Jobserver ausgeführt werden soll. Die Kommandozeile wird zuerst vom Scheduling System interpretiert und in Programmaufruf und einzelne

Parameter zerlegt. Dabei werden die Regeln der Bourne Shell in Bezug aufs Quoting befolgt.
Sollen Environment-Variablen oder Parameter angegeben werden, so müssen diese konform der jeweiligen Ausführungsumgebung (je nach Shell oder Kommandozeilen-Interpreter der Umgebung, siehe Environment) gequoted werden. Eine Auflistung aller verwendbaren Standardparameter finden Sie in Paragraph 13.5.10.2.
Für weitergehende Informationen über das zu verwendende Quoting, lesen Sie bitte die Dokumentation Ihres verwendeten Kommandozeilen-Interpreters (Shell).
Falls es sich beim Zielsystem um Windows handelt, muss mit einigen weiteren Schwierigkeiten gerechnet werden. Wenn unter Windows eine Tabelle aus Executable und Parameter zur Ausführung angeboten wird, konstruiert Windows zuerst eine Kommandozeile aus der Tabelle, um diese Kommandozeile anschließend wieder zu interpretieren. Problematisch ist auch die Verwendung von Quoting unter Windows, da die aufgerufenen Programme die Quotes entfernen oder eben nicht entfernen, je nach Typ des Executables (`.exe` oder `.bat`).
Ist das auszuführenden Programm (erstes Element der Kommandozeile) eine gültige Ganzzahl, so wird die Kommandozeile nicht vom Jobserver ausgeführt, sondern der Job wird so behandelt, als hätte er sich mit der Ganzzahl als Exit Code beendet. Dummy Jobs mit 'true' oder 'false' als Programm können nun als '0' statt 'true' bzw. '1' statt 'false' implementiert und so vom System wesentlich effizienter und schneller verarbeitet werden.
Sollte es tatsächlich einmal nötig sein, ein Executable mit einer Zahl als Namen auszuführen, so kann dies durch einen Pfad Prefix ('./42' statt '42') erreicht werden.

Exit State Mapping Im Feld *Exit State Mapping* kann ein bestehendes Exit State Mapping eingegeben werden. Dieses Mapping definiert die Abbildung des Exit Codes eines Jobs nach einem Exit State. Wird kein Mapping angegeben, wird das Default Mapping aus dem Exit State Profile genutzt. Das Mapping muss mit dem Profile verträglich sein.

Environment Im Feld *Environment* muss ein bestehendes Environment ausgewählt werden. Alle Named Resources dieses Environments müssen auf einem Jobserver vorhanden sein, damit der Job auf diesem Jobserver ausgeführt werden kann.

Footprint Im Feld *Footprint* kann ein bestehender Footprint gewählt werden. Alle Named Resources für diesen Footprint müssen auf einem Jobserver in ausreichender Anzahl vorhanden sein, damit der Job auf diesem Jobserver ausgeführt werden kann.

Expected Runtime Die *Expected Runtime* beschreibt die zu erwartete Zeit, die ein Job für seine Ausführung benötigt. Sie ist manuell einzutragen und kann z. B. für Laufzeitüberwachung genutzt werden.

Expected Finaltime Die *Expected Finaltime* beschreibt die zu erwartete Zeit, die ein Job für seine Ausführung benötigt (von Submit bis zum Erreichen eines finalen Status). Sie ist manuell einzutragen und wird für die Darstellung im Kalender verwendet. Natürlich gibt es auch einen Standard Parameter dazu, sodass der zu erwartete Wert dem Job zur Verfügung steht.

Rerun Program Das Feld *Rerun Program* gibt das Kommando an, welches bei einer wiederholten Ausführung des Jobs nach einem Fehlerzustand (Rerun) ausgeführt werden soll. Dabei kann es sich um ein anderes Kommando handeln, als das Run Program, da es vielleicht nötig ist, bei einem wiederholten Start Aufräumarbeiten auszuführen, bevor das ursprüngliche Kommando wieder aufgesetzt werden kann.
Wurde im Rerun Program kein Wert eingetragen, so wird bei einem wiederholten Start des Jobs das Run Program aufgerufen.
Zu Details bzgl. der Verarbeitung (Quoting, Behandlung von Integern als Program Name) siehe 'Run Program' weiter oben.

Kill Program Das *Kill Program* bestimmt, welches Programm ausgeführt werden soll, um einen aktuell laufenden Job zu beenden.
Ist in diesem Feld kein Wert eingetragen, so ist es nicht möglich den Job über die schedulix Oberfläche vorzeitig zu terminieren.
Ein Beispiel für ein Kill Program könnte in einer Unix-Umgebung das **"kill"** Kommando sein. Im Kill Program kann auch über die vordefinierte Systemvariable "PID" auf die Process Id des laufenden Jobs zugegriffen werden (Beispiel: "kill -9 $PID").

Working Directory Das *Working Directory* gibt an, in welchem Verzeichnis das Run Program gestartet werden soll. Der Jobserver wechselt vor dem Starten des Run Programs in das angegebene Verzeichnis.
Wurde in diesem Feld kein Wert eingegeben, so wird als Working Directory das Default Directory des Jobservers verwendet.
Es ist auch möglich, als Working Directory einen Parameter, Folder-, Scope- oder Jobserver-Parameter anzugeben.

Logfile Das Feld *Logfile* gibt an, in welche Datei alle normalen Ausgaben des Run Programs ausgegeben werden sollen. Unter normalen Ausgaben sind alle Ausgaben gemeint, die den normalen Ausgabekanal (STDOUT unter UNIX) benutzen.

Auch im Feld *Logfile* sind Parameterersetzungen möglich. So kann zum Beispiel der Parameter JOBID zur eindeutigen Kennzeichnung eines Jobs oder ein TIMESTAMP zur Kennzeichnung des Datums verwendet werden.
Wurde kein Wert eingetragen, so werden alle Ausgaben verworfen.
Wurde im Logfile keine Pfadangabe angegeben, so wird das Logfile im Working Directory geschrieben.

Logfile Write Mit dem Feld *Logfile Write* wird dem System mitgeteilt wie ein Logfile anzulegen ist. Es gibt folgende Optionen:

- Append

  Mit Append werden alle Ausgaben des Logfiles an ein bestehendes Logfile angehängt. Falls noch kein Logfile existiert, wird ein neues erzeugt.

- Truncate

  Mit Truncate wird bei jedem neuen Ablauf das Logfile gelöscht und neu angelegt.

Error Logfile Das Feld *Error Logfile* gibt an, in welcher Datei alle Fehlerausgaben des Run Program ausgegeben werden sollen. Unter Fehlerausgaben sind alle Ausgaben gemeint, die den Fehlerausgabekanal (stderr unter UNIX) benutzen.
Auch im Feld *Error Logfile* sind Parameterersetzungen möglich. So kann zum Beispiel der Parameter JOBID zur eindeutigen Kennzeichnung eines Jobs oder ein TIMESTAMP zur Kennzeichnung des Datums verwendet werden.
Wurde kein Wert eingetragen, so werden alle Ausgaben verworfen.
Wurde im Logfile keine Pfadangabe angegeben, so wird das Logfile im Working Directory geschrieben.
Wird derselbe Name, wie im Feld *Logfile* angegeben, so werden diese Ausgaben ebenfalls in diese Datei geschrieben. Die Reihenfolge der Ausgaben (Error oder Normal) sind allerdings durch die Pufferung zufällig. Es werden jedoch unabhängig vom Inhalt des Feldes Error Logfile Write keine Ausgaben des normalen Ausgabekanals (stdout) überschrieben.

Error Logfile Write Mit dem Feld *Error Logfile Write* wird dem System mitgeteilt, wie das Error Logfile anzulegen ist. Es gibt folgende Optionen:

- Append

  Mit Append werden alle Ausgaben des Error Logfiles an ein bestehendes Error Logfile angehängt. Falls noch kein Error Logfile existiert, so wird ein neues erzeugt.

- Truncate

  Mit Truncate wird bei jedem neuen Ablauf das Error Logfile gelöscht und neu angelegt.

13.5.3 Tab Restart

Im Tab ”Restart” kann das Restart-Verhalten eines Jobs definiert werden.
Ein Job wird nur neu gestartet, wenn er sich im Job State ”FINISHED” befindet.
Erhält ein Job einen ”RESTARTABLE” Exit State aufgrund eines Job State ”ERROR”, so wird KEIN Restart durchgeführt!

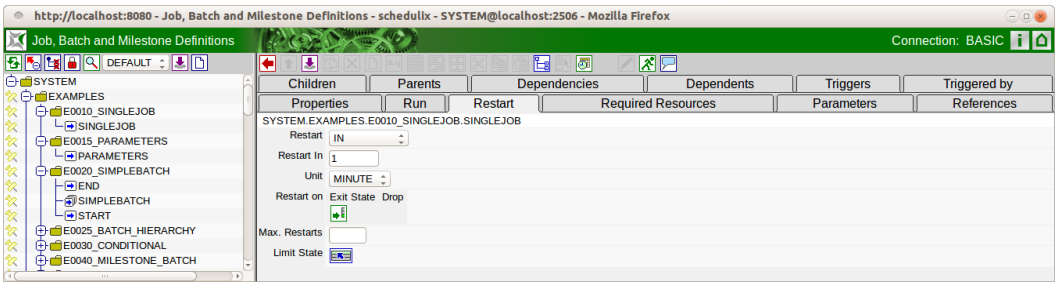


Abbildung 13.14: Job Tab Restart

Die Felder des Tabs ”Restart” haben folgende Bedeutung:

Restart Bestimmt, ob der Job automatisch neu gestartet werden soll, falls er einen ”RESTARTABLE” Exit State erreicht.
Folgende Auswahlen sind möglich:

- NO: Kein Restart. Auf dem Tab ”Restart” werden keine weiteren Felder angezeigt.
- IMMEDIATE: Sofortiger Neustart
- AT: Neustart zu einem gegebenen Zeitpunkt (z.B. ”T16:00” um 16:00 Uhr)
- IN: Neustart nach einer Wartezeit (z.B. 10 Minuten)

Restart Time Wird nur angezeigt, wenn im Feld *Restart* ”AT” gewählt wurde.
Die Zeit, zu der der Job neu gestartet werden soll, wird im Format ”YYYY-MM-DDTHH:MI:SS” eingetragen.
Das Format orientiert sich an der ISO Norm 8601 und erlaubt auch unvollständige Angaben. Die Eingabe von ’T16:00’ wird den Job ”suspended” neu starten und um 16:00 Uhr resumen (ausgehend vom Submit Zeitpunktes des Jobs).

Restart In Wird nur angezeigt, wenn im Feld *Restart* "IN" gewählt wurde.
Anzahl der Einheiten (siehe *Unit*), die gewartet werden soll, bis der Job wieder anläuft

Unit Wird nur angezeigt, wenn im Feld *Restart* "IN" gewählt wurde.
Einheiten für die Wartezeit (siehe *Restart In*)

Restart On Tabelle der Exit States, welche zu einem Restart führen sollen.
Bleibt die Tabelle leer, so wird bei allen "RESTARTABLE" Exit States neu gestartet.

Condition Optionale Bedingung (ab schedulix PROFESSIONAL) für das Auslösen einen Restarts

Max. Restarts Maximale Anzahl, wie oft der Job neu gestartet wird.
Wird nichts eingegeben gilt der Systemparameter TriggerSoftLimit. Die Eingabe von 0 startet den Job beliebig oft wieder.

Limit State Hier kann eine Exit State gesetzt werden, den der Job annehmen soll, falls die maximale Anzahl der Restarts erreicht wurde.

13.5.4 Tab Children

Im Tab "Children" werden alle Ausführungsobjekte als Liste angezeigt, welche als Child des aktuellen Jobs definiert worden sind. Mittels Copy and Paste können weitere Objekte als Children hinzugefügt werden.
Eine Reihenfolge der Ausführung der Children ist nicht festgelegt, sie kann aber über die Dependencies festgelegt werden.
Der Tab "Children" sieht folgendermaßen aus:

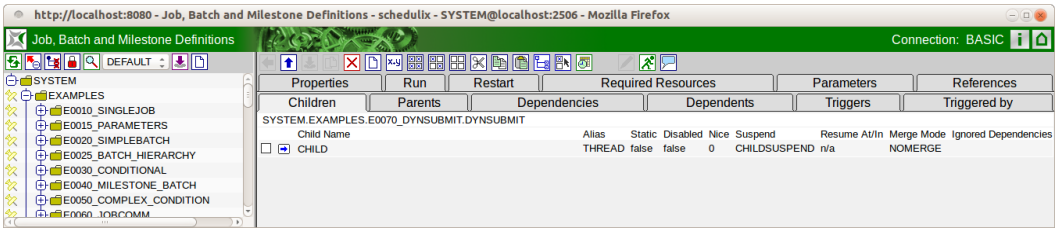


Abbildung 13.15: Batches und Jobs; Children Tab

Hier erscheint eine Liste von allen Job, Milestone und Batch-Objekten, welche dem gewählten Job als Children untergeordnet sind. Die Spalten der obigen Liste haben folgende Bedeutung:

Child Name Das ist der Name des Childs. Durch Anklicken des Namens erscheint der Tab "Child Details".

Die restlichen Spalten werden im nächsten Abschnitt erläutert.

13.5.4.1 Tab Child Details

Der Tab "Child Details" erscheint, wenn im Tab "Children" der Name eines Child-Objektes angewählt wurde. Hier werden alle Daten, welche schon in der Tabelle als Elemente angezeigt wurden, noch einmal gelistet und können geändert werden. Der Tab sieht folgendermaßen aus:

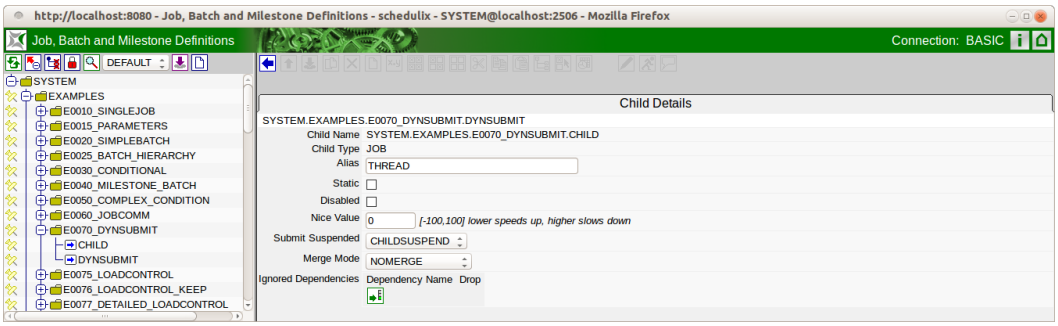


Abbildung 13.16: Batches und Jobs; Child Details

Die Felder des Tabs "Child Details" haben folgende Bedeutung:

Child Name Der *Child Name* wird immer mit dem gesamten Pfad der darüberliegenden Ordnerhierarchie angezeigt.

Child Type Der *Child Type* gibt die Art des Childs (Batch, Job oder Milestone) an.

Alias Mit dem *Alias* kann einem Child eine neue logische Benennung zugeordnet werden. Diese Benennung muss innerhalb aller Children des gewählten Jobs eindeutig sein. Der Alias ist nur bei Dynamic Children von Bedeutung. Der Parent Job kann beim Submitten durch Angabe des Alias weitere Instanzen der Children erzeugen. Damit ist im Parent kein Wissen über die Speicherstelle des Childs notwendig. Durch diese Abstraktion werden Umstrukturierungen der Folderstruktur erheblich vereinfacht.

Static Der Schalter *Static* gibt an, ob ein Child statisch oder dynamisch submitted wird. Handelt es sich um ein statisches Child, so wird die Child-Instanz schon beim Submit des Parents instanziiert. Handelt es sich um ein dynamisches Child (der

Static-Schalter ist nicht gesetzt), so können eine oder mehrere Instanzen des Childs zur Laufzeit dynamisch erzeugt werden.
Dynamische Children werden häufig bei Parallelisierungsaufgaben verwendet. Hier wird erst zur Laufzeit vom Run Program entschieden, ob und wie viele dynamische Children angelegt werden sollen und diese werden durch die schedulix API angelegt. Hiermit ist eine beliebige Aufteilung eines Prozesses in viele parallel arbeitende Teilprozesse möglich.

Disabled Ist der Schalter *Disabled* gesetzt, so wird nicht der tatsächliche Batch oder Job als Child submittet, sondern es wird ein Dummy Job mit gleichem Namen submitted, der sich wie ein leerer Batch verhält. Abhängigkeiten des disabled Jobs oder Batches werden jedoch korrekt abgebildet.
Damit ist es möglich, einzelne Jobs oder Batches einer Verarbeitungskette von der Verarbeitung auszuschließen und dabei die Abhängigkeitsbeziehungen und die Verarbeitungsreihenfolge der Verarbeitungskette zu erhalten.

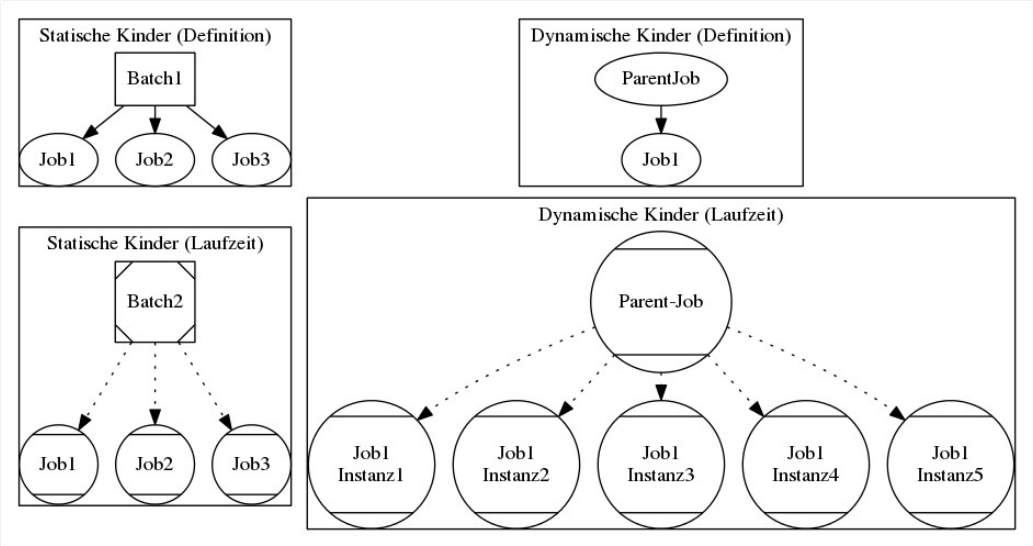


Abbildung 13.17: Statische und dynamische Children

Die Legende für die Grafik finden Sie im Kapitel 1.8.

Nice Value Der *Nice Value* gibt an, mit welcher Priorität das gewählte Child laufen soll. Der Nice Value gibt einen Offset an, der auf die Priorität des Childs addiert wird.
Beispiel: Hat der Prozess einen Nice Value von -50 und das Child hat eine Priorität von 100, dann hat die als Child von diesem Job gestartete Job-Instanz eine Priorität von 50. Wird der Child-Prozess alleine gestartet, hat er eine Priorität von 100.

Handelt es sich beim Parent um einen Batch, wird der Nice Value zum Nice Value des Parents gezählt.

Submit Suspended Der Parameter *Submit Suspended* gibt an, in welcher Form das Child-Objekt beim Start verzögert wird, oder ob er sofort gestartet werden kann. Es gibt folgende Optionen:

1. YES

   Das Child wird beim Submit als Suspended angelegt und muss durch eine explizite Freigabe gestartet werden.

2. NO

   Das Child wird beim Submit nicht Suspended und kann sofort starten.

3. CHILDSUSPEND

   Ob eine Verzögerung stattfindet oder nicht, hängt vom Feld *Suspend* des Child Jobs ab. Das heißt, je nachdem, wie die Einstellung im Child Job definiert wurde, wird diese übernommen.

Resume Wird nur angezeigt, falls *Submit Suspended* auf "YES" gesetzt wurde.
Hier kann ausgewählt werden, ob ein automatischer Resume stattfinden soll.
Es gibt folgende Möglichkeiten:

- NO: Wählt diese Funktionalität ab
- AT: Wählt einen automatischen Resume zu einem festen Zeitpunkt. Das Eingabefeld *Resume Time* wird angezeigt.
- IN: Wählt einen automatischen Resume nach Ablauf einer Zeit. Die Eingabefelder *Resume In* und *Unit* werden angezeigt.

Resume Time Wird nur angezeigt, falls im *Resume* "AT" gewählt wurde.
Hier wird der gewünschte Resume Zeitpunkt im Format "YYYY-MM-DDTHH:MI:SS" eingegeben.
Das Format orientiert sich an der ISO Norm 8601 und erlaubt auch unvollständige Angaben. Die Eingabe von 'T16:00' wird den Job um 16:00 Uhr resumen (ausgehend von der Zeit des Job Submits).

Resume In Wird nur angezeigt, falls im *Resume* "IN" gewählt wurde.
Hier wird angegeben, wie viele Zeiteinheiten (siehe *Unit*) bis zum Resume gewartet werden soll.

Unit Wird nur angezeigt, falls im *Resume* "IN" gewählt wurde.
Hier wird eingegeben, ob es sich bei der Eingabe im *Resume In* um Minuten (MINUTE), Stunden (HOUR), oder Tage (DAY) handeln soll.

Merge Mode Der "Merge Mode" gibt an, ob ein Child-Objekt mehrfach innerhalb eines "Master Job"-Laufes gestartet wird oder nicht.

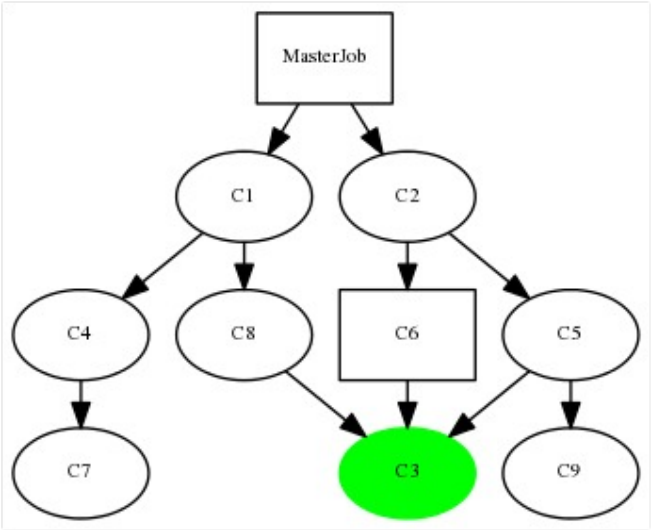


Abbildung 13.18: Beispiel für Merge Mode

Die Legende für die Grafik finden Sie im Kapitel 1.8.
Es gibt folgende Optionen:

1. No Merge

   Es erfolgt keine Zusammenfassung von Jobs. Der Child Job wird gestartet ohne Berücksichtigung, ob dieser Job im aktuellen Master Batch bereits gelaufen ist.

   In dem in der Abbildung 13.18 dargestellten Beispiel wird Job C3 dreimal gestartet, wenn in jeder Parent-Child-Beziehung ein No Merge steht.

2. Failure

   Zur Submit-Zeit des Jobs wird geprüft, ob der Child Job bereits gestartet wurde und, falls dies der Fall ist, wird eine Fehlermeldung ausgegeben und der Start des Master Jobs wird abgebrochen.

   In dem in der Abbildung 13.18 dargestellten Beispiel kann der Job nicht submitted werden.

3. Merge Local

   In der Child-Hierarchie des Parent-Objektes wird ein Merge durchgeführt. Das heißt, wenn der Job in der Child-Hierarchie schon einmal gestartet worden ist, wird er nicht noch einmal gestartet. Wurde der Job innerhalb des Master Jobs, aber außerhalb der Child-Hierarchie des Parents schon einmal gestartet, so wird dies nicht berücksichtigt, sondern der Job wird neu submitted.

   Die Abhängigkeiten von diesem Job werden dabei genauso berücksichtigt, wie wenn dieser Job in dieser Child-Beziehung gestartet worden wäre.

   Falls im oberen Beispiel bei der Child-Beziehung C2→C6 und C2→C5 der Merge Mode Local gesetzt ist, wird der Job C3 nur zweimal gestartet, da in der Child-Hierarchie von C2 der Merge durchgeführt würde. Da die Beziehung C8→C3 nicht in dieser Child-Hierarchie liegt, wird dort C3 ausgeführt, da hier kein Merge durchgeführt wird.

4. Merge Global

   Es wird innerhalb des gesamten Master Jobs geprüft, ob dieser Job schon einmal gestartet wurde und, falls dies der Fall ist, wird der Job nicht neu gestartet.

   Dies kann nötig sein, falls ein Job nur einmal laufen soll, aber von mehreren Jobs als Voraussetzung gebraucht wird. Er kann von mehreren Parent-Prozessen als Child definiert werden. Der Parent-Prozess, der als erster ausgeführt wird, startet den Prozess. Die anderen Parent-Prozesse überprüfen durch das "Merge Global", ob dieser Job bereits gelaufen ist und starten ihn nicht noch einmal.

   Die Abhängigkeiten von diesem Job werden dabei genauso berücksichtigt, wie wenn dieser Job in dieser Child-Beziehung gestartet worden wäre.

   Im oberen Beispiel wird der Job C3 nur einmal gestartet, da ein globaler Merge alle Beziehungen im Master Job berücksichtigt und deshalb bei den Beziehungen (C8→C3, C6→C3 und C5→C3) den Merge durchführt.

   Durch welche Beziehung der Job nun wirklich gestartet wird, ist unerheblich.

Exit State Translation In dieser Auswahlmaske kann die Exit State Translation ausgewählt werden, die verwendet wird, um dem Parent-Objekt den Exit State des Child-Objektes mitzuteilen. Mehr zu Exit State Translations finden Sie im Kapitel 5.

Liste Ignored Dependencies Hier kann eine Liste von Abhängigkeiten (Dependencies) hinzugefügt werden, welche innerhalb dieser Parent-Child-Beziehung vom Child ignoriert werden soll. All diese Abhängigkeiten werden nun für das

Child nicht mehr berücksichtigt und das Child kann gestartet werden, ohne dass die Abhängigkeit des Parents erfüllt ist.

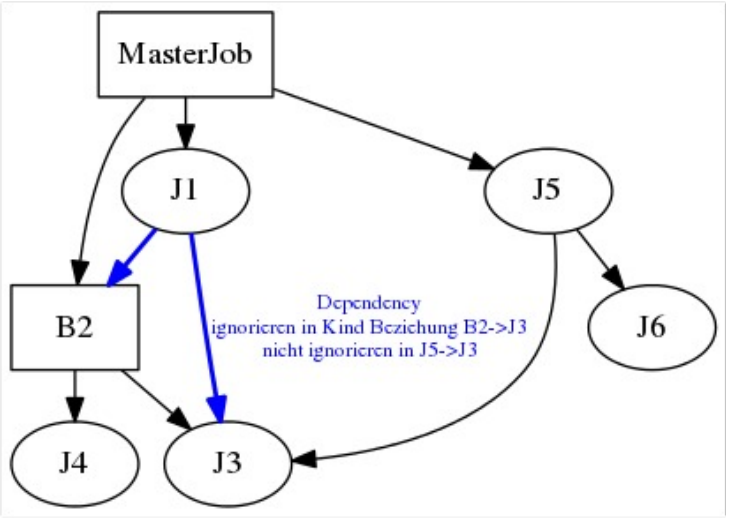


Abbildung 13.19: Beispiel Ignored Dependencies

Ein Beispiel für Ignored Dependencies ist im Bild 13.19 abgebildet. Die Legende für die Grafik finden Sie im Kapitel 1.8.
Es können nur die Abhängigkeiten ignoriert werden, welche im Feld *Dependency Name* einen Wert eingetragen haben. Nicht benannte Abhängigkeiten können nicht in der Auswahlbox ausgewählt und dadurch nicht ignoriert werden.

13.5.5 Tab Parents

Der Tab ”Parents” zeigt eine Liste von übergeordneten Parent-Objekten an, welche das gewählte Objekt als Child verwenden. Dies stellt also bildlich die Sicht nach oben dar, während der Children die Sicht nach unten darstellt, da dort alle Children des aktuell gewählten Objektes angezeigt werden.

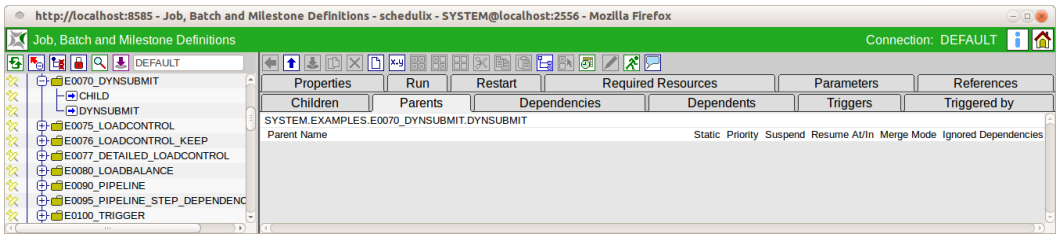


Abbildung 13.20: Batches und Jobs; Parents Tab

Der Tab dient nur zur Information, es können keine Werte geändert oder neue Listeneinträge hinzugefügt werden. Dies ist nur in der Parent Child-Richtung möglich.
Die Spalten der obigen Liste werden im Tab ”Children” erläutert.

13.5.6 Tab Dependencies

Der Tab ”Dependencies” zeigt eine Liste von allen Objekten an, von denen das aktuelle Objekt abhängig ist. Eine Abhängigkeit ist dann vorhanden, wenn das aktuelle Objekt nicht starten darf, bevor nicht alle Objekte, von denen der Job abhängig ist, vorher ausgeführt und mit einem gewissen Exit State beendet wurden.
Mittels Copy and Paste können weitere Objekte als benötigte Scheduling Entities (required Scheduling Entities) hinzugefügt werden.

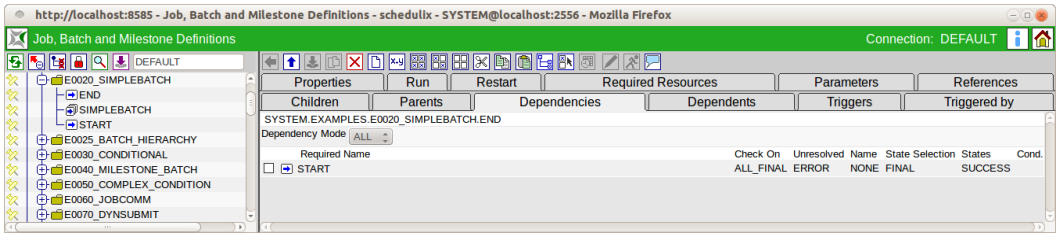


Abbildung 13.21: Batches und Jobs; Dependencies Tab

Das Feld der obigen Liste hat folgende Bedeutung:

Dependency Mode Der ”Dependency Mode” gibt an, in welchem Zusammenhang die Liste der Dependencies gesehen werden muss. Es gibt folgende Optionen:

1. ALL

   Es müssen alle Abhängigkeiten gemeinsam erfüllt sein, damit das Objekt starten kann. Dies entspricht einer Verknüpfung aller Bedingungen mit **"UND"**.

   Beispiel: Objekt C ist von Job A und Job B abhängig (das heißt, er soll nur starten, wenn A und B fertig sind) so muss der Dependency Mode ALL gewählt werden.

2. ANY

   Es muss mindestens eine der Abhängigkeiten erfüllt sein, damit das Objekt starten kann. Dies entspricht einer Verknüpfung aller Bedingungen mit einem **"ODER"**.

   Beispiel: Objekt C ist von Job A oder Job B abhängig (das heißt, er soll starten wenn A oder B fertig ist), so muss der Dependency Mode ANY gewählt werden.

Innerhalb der Dependencies eines Jobs kann es entweder die Verknüpfung mit ANY oder mit ALL geben. Sollen komplexere Abhängigkeitsverknüpfungen abgebildet werden (zum Beispiel abhängig von A oder B und C oder D) so wird

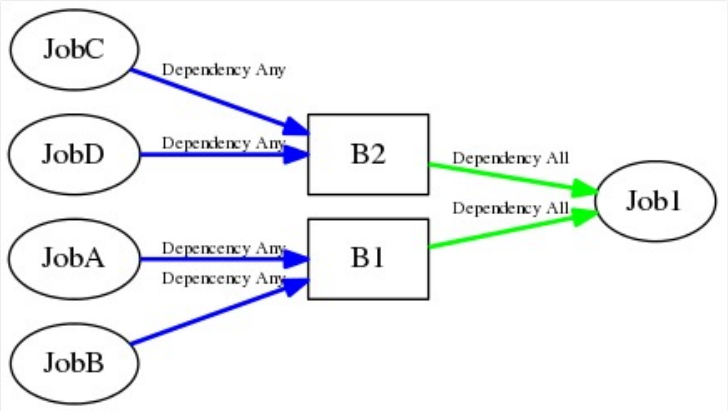


Abbildung 13.22: Beispiel für Dependency Modes

dies, wie im Bild 13.22 dargestellt, mittels der Verwendung eines Milestone oder Batch-Objektes durchgeführt.
Die Legende für die Grafik finden Sie im Kapitel 1.8.
Im oben genannten Beispiel wurden Batch B1 und B2 generiert. B1 hat die Abhängigkeiten A und B mittels ANY verknüpft. B2 analog C und D ebenfalls mittels ANY. Anschließend kann unser Objekt (Job1) die beiden Batches mittels ALL als Abhängigkeit verknüpfen und hat die komplexe Bedingung (abhängig von A oder B und C oder D) hiermit abgebildet.

Die Liste ”Dependencies” zeigt alle Objekte an, von denen das aktuell gewählte Objekt abhängig ist. Alle Elemente in der Liste werden mittels des Dependency Modes verknüpft.

Die Spalten der Liste werden im nächsten Abschnitt erläutert.

13.5.6.1 Tab Dependency Details

Der Tab ”Dependency Details” erscheint, wenn im Tab ”Dependencies” ein Eintrag durch Anklicken des Required Name ausgewählt wurde.
In den Dependency Details sind alle Aspekte einer Abhängigkeit zwischen den zwei Jobs einstellbar. Der Tab sieht folgendermaßen aus:

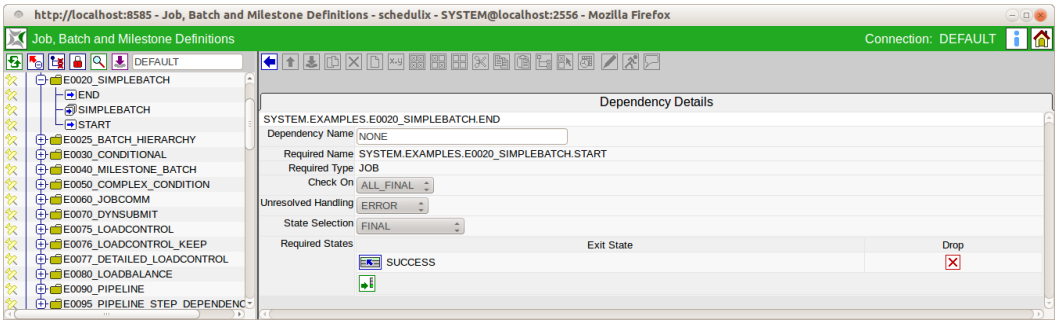


Abbildung 13.23: Batches und Jobs; Dependency Details

Die Felder des Tabs ”Dependency Details” haben folgende Bedeutung:

Dependency Name Der *Dependency Name* ist optional und ist die Voraussetzung zum Ignorieren der Dependencies.

Required Name Hier steht der Name des Objektes, von dem der aktuell gewählte Job abhängig ist. Er entspricht dem vollen Namen, mit übergeordneter Ordnerhierarchie.

Required Type Im *Required Type* wird die Art des benötigten Objektes angezeigt.

Check On Im Feld *Check On* wird die Art der Vollständigkeitsprüfung für das benötigte Objekt beschrieben. Es gibt folgende Optionen:

1. ALL_FINAL

   Wird diese Option ausgewählt, so muss der benötigte Job einen Final State erreicht haben und alle seine Children ebenfalls.

2. JOB_FINAL

   Wird diese Option ausgewählt, so wird nur geprüft, ob der Job einen Final State erreicht hat. Der State der Child-Objekte wird nicht geprüft.

Condition Die *Condition* ist eine Bedingung, die zusätzlich erfüllt sein muss damit die Abhängigkeit als erfüllt gilt. Parameter vom Typ "Resource Reference" können nicht benutzt werden, denn zum Zeitpunkt der Auswertung der Condition sind noch keine Resources belegt.

Unresolved Handling Im Auswahlfeld *Unresolved Handling* wird beschrieben was zu tun ist, wenn eine abhängige Objektinstanz im aktuellen Master Batch nicht vorhanden ist.

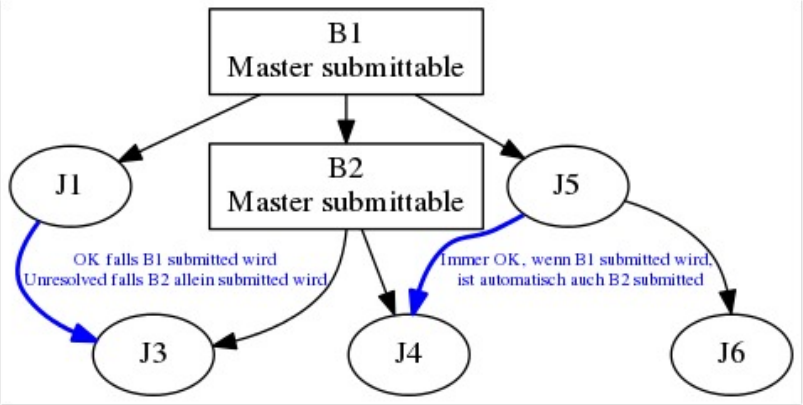


Abbildung 13.24: Beispiel für Unresolved Handling

Die Legende für die Grafik finden Sie im Kapitel 1.8.
Es gibt folgende Optionen:

1. IGNORE

   Die Abhängigkeit wird ignoriert, falls der benötigte Job nicht vorhanden ist. Ist der Job vorhanden, wird sie ganz normal beachtet.

   Ein IGNORE kann dann nötig sein, wenn kleinere Batches zu größeren zusammengefasst werden. Läuft der kleine Batch alleine, so kann er alle Abhängigkeiten, die nur beim größeren Lauf zu beachten sind, ignorieren. Falls er als Teil des großen Batches läuft, sind die benötigten Jobs dann vorhanden und die Abhängigkeiten müssen beachtet werden.

   Im oberen Beispiel muss in der Abhängigkeit J1→J3 ein Ignore eingetragen werden, wenn der Batch B2 auch allein gestartet werden darf.

2. ERROR

   Ist der benötigte Job nicht vorhanden, so wird der Submit-Vorgang mit einem Fehler beendet.

3. SUSPEND

   Der aktuelle Job geht in den Suspend State. Das heißt, er muss durch ein externes Programm oder manuellen Eingriff gestartet werden, sobald der benötigte Job vorhanden ist.

   Im oberen Beispiel würde Job J3 in den Suspend State gehen, falls J1 ein dynamisches Child von B1 wäre und zeitlich nicht klar ist, wann der Job submitted wird.

   Die Idee hinter SUSPEND ist, dass weder ein unbewachtes Starten, noch ein "ERROR" auftritt.

4. DEFER

   Die Abhängigkeit wird, falls der benötigte Job im Ablauf durch einen Trigger oder dynamischen Submit erzeugt wird, zu einem späteren Zeitpunkt aufgelöst.

   Falls ein Ablauf inaktiv wird und nur noch Jobs im Status DEPENDENCY_WAIT enthält, werden alle Jobs, welche zu diesem Zeitpunkt nur auf DEFER Abhängigkeiten warten UNREACHABLE.

5. DEFER_IGNORE

   Die Abhängigkeit wird, falls der Benötigte Job im Ablauf durch einen Trigger oder dynamischen Submit erzeugt wird zu einem späteren Zeitpunkt aufgelöst.

   Falls eine Job nur noch auf DEFER_IGNORE Abhängigkeiten wartet, werden diese ignoriert.

State Selection Im Feld *State Selection* wird definiert, welche Exit States des benötigten Jobs die Abhängigkeit erfüllen und der abhängige Job starten kann. Es gibt folgende Optionen:

1. FINAL

   Wird diese Option ausgewählt, so muss der benötigte Job einen Final State erreicht haben. Diese Option ermöglicht die Auswahl der benötigten Exit States in der nachfolgenden Tabelle *Required States*.

2. ALL_REACHABLE

   Wird diese Option ausgewählt, sind alle Final Exit States gültig, welche im Exit State Profile des benötigten Jobs nicht als Unreachable gekennzeichnet wurden.

3. UNREACHABLE

   Wird diese Option ausgewählt, ist nur der Exit State gültig, welcher im Exit State Profile des benötigten Jobs als Unreachable gekennzeichnet wurde.

4. DEFAULT

   Wird diese Option ausgewählt, sind alle Final Exit States gültig, welche im Exit State Profile des benötigten Jobs als Dependency Default gekennzeichnet wurden.

Required States Hier steht die Liste von allen gültigen Exit States, welche das benötigte Objekt haben muss, damit die Abhängigkeit erfüllt ist und der abhängige Job starten kann. Diese Liste ist nur sichtbar, wenn im Feld *State Selection* die Option FINAL gewählt wurde.
Beispiel:
Ein vom Job B benötigter Job A kann zwei Exit States (SUCCESS oder WARNING) annehmen. Da der Job B nur starten soll, wenn Job A mit einem Exit State SUCCESS beendet wurde (also wenn er erfolgreich durchgelaufen ist), darf in der Liste der Required States nur ein Eintrag mit SUCCESS stehen. Ist Job A durchgelaufen und hat den Exit State SUCCESS erhalten, so kann Job B starten. Beendet sich Job A mit WARNING, kann Job B nicht starten.
Die Liste ist notwendig, da es möglich ist, dass ein abhängiger Job mehrere Exit States als Voraussetzung akzeptiert. In unserem Beispiel heißt das, Job B hätte beide Exit States (SUCCESS und WARNING) als Required States eingetragen, wenn er auf jeden Fall starten soll, egal ob Job A vollständig erfolgreich war oder mit einer Warnung beendet wurde.

13.5.7 Tab Dependents

Der Tab ”Dependents” zeigt alle Scheduling Entities an, welche von dem gewählten Scheduling Entity abhängig (dependent) sind. Dies ist die umgekehrte Sicht des Tabs ”Dependencies”, da jetzt die abhängigen (dependent) Jobs angezeigt werden und nicht mehr die benötigten (required) Jobs.
Der Tab sieht folgendermaßen aus:
Die Liste zeigt alle Objekte (Jobs, Milestones, Batches) an, welche das gewählte Scheduling Entity benötigen. Die Liste ist rein informativ und kann nicht geändert werden. Auch die Feldwerte können in dieser Ansicht nicht geändert werden. Um eine Änderung durchzuführen, muss in das in der Liste angezeigte Objekt navigiert und dort der Tab ”Dependencies” aufgerufen werden. Hier ist die Änderung möglich.
Die Spalten der Liste werden im Tab ”Dependency Details” erläutert.

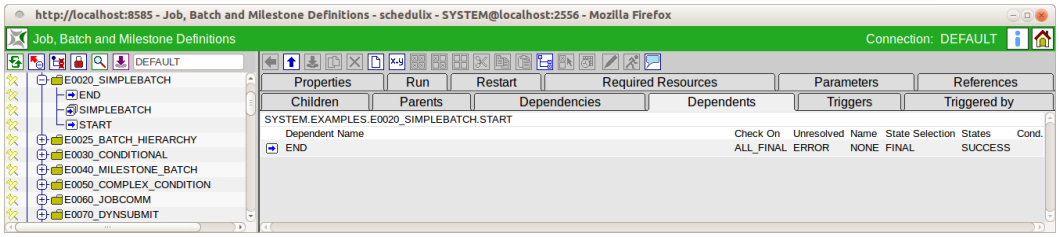


Abbildung 13.25: Batches und Jobs; Dependents Tab

13.5.8 Tab Required Resources

Der Tab "Required Resources" zeigt alle für die Ausführung dieses Objektes notwendigen Resources an. Weitere notwendige Resources können hier hinzugefügt werden.
Es werden alle, bereits durch das Environment oder einen angegebenen Footprint definierten, Resources angezeigt. Diese können nur durch Änderung des entsprechenden Environments bzw. Footprints entfernt werden.
Der Tab "Required Resources" sieht folgendermaßen aus:

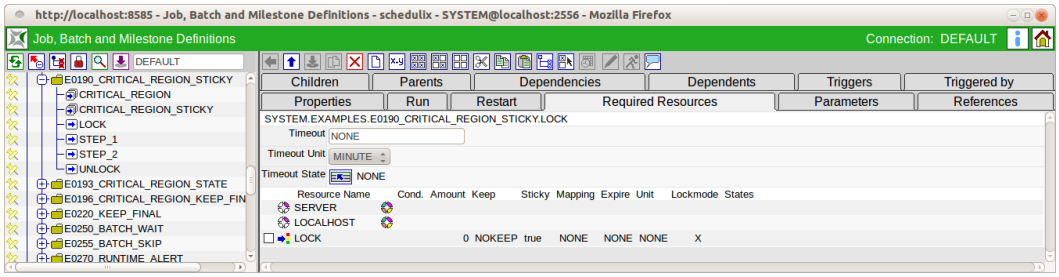


Abbildung 13.26: Jobs; Required Resources Tab

Die Felder und Spalten des Tabs "Required Resources" haben folgende Bedeutung:

Timeout Der *Timeout* gibt an wie lange nach dem Submit des Jobs auf alle Resources gewartet werden soll. Ist hier kein Wert angegeben, so wird unendlich lange auf die Resources gewartet.
Der Wert in diesem Feld muss im Zusammenhang mit dem Feld *Timeout Unit* gesehen werden. Im Feld *Timeout* wird der Wert des Timeouts eingegeben. Im Feld *Timeout Unit* die Einheit.
Beispiel: Steht im Feld *Timeout* der Wert 5 und die Timeout Unit ist **MINUTES**, so wartet ein Job 5 Minuten auf eine Resource und, falls diese in dieser Zeit nicht verfügbar wird, kann dieser Job nicht gestartet werden und beendet sich mit dem Exit State, welcher im Feld *Timeout State* definiert wurde.

Timeout Unit Die *Timeout Unit* gibt die Einheit an, welche für den gewählten Wert aus dem Feld *Timeout* anzuwenden ist. Ist im Feld *Timeout* kein Wert eingetragen, so spielt der Wert des Feldes *Timeout Unit* keine Rolle.

Timeout State In diesem Auswahlfeld kann gewählt werden, welche Exit State der Job annehmen soll, falls ein Timeout eingetroffen ist. Wird der Auswahlknopf gedrückt, so kann aus der Liste die Exit States ausgewählt werden, die vom *Exit State Profile* zur Verfügung gestellt wurden.
Es könnte hier zum Beispiel ein eigener Timeout State definiert werden oder es wird der Failure State verwendet.
Ist im Feld *Timeout* kein Wert eingetragen, so spielt der Wert des Feldes *Timeout State* keine Rolle.
In der Liste "Resources" werden alle Resources angezeigt, die aktuell vom gewählten Job benötigt werden um ausgeführt werden zu können. Die Liste enthält folgende Spalten:

Resource Name Das ist der Name der Resource, welche für den Start des aktuellen Jobs benötigt wird.
Nach dem Namen des Feldes ist ein Icon angegeben, welches angibt, aus welchem Zusammenhang die Resource stammt. Folgende Icons werden angezeigt:

- Footprint

  Die Resource stammt aus dem Feld *Footprint*. Für jede Resource kann eine individuelle Anforderung angelegt werden. Diese überschreibt die ursprüngliche Anforderung.

- Environment

  Die Resource stammt aus dem Feld *Environment*. Er ist nur zu Informationszwecken vorhanden und kann in diesem Tab nicht geändert werden.

- Kein Icon

  Ist nach dem Namen der Resource kein Icon vorhanden, so handelt es sich um eine in diesem Tab hinzugefügte Resource, welche auch hier änderbar ist. Durch das Anklicken des Namens kann in den Tab "Resource Details" gesprungen und die Werte in der Liste geändert werden. Darüber hinaus ist es möglich durch Anklicken des *Selections* Buttons vor dem Namen diese Zeile zu markieren und zu löschen, oder zu verschieben.

Cond. Im Feld *Cond.* kann eine Bedingung stehen, die erfüllt sein muss, damit die Resource als gültig anerkannt wird. Die restlichen Spalten werden im nächsten Abschnitt erläutert.

13.5.8.1 Tab Resource Details

Der Tab ”Resource Detail” erscheint, wenn im Tab Required Resources ein Listeneintrag angewählt wurde.
Im Tab ”Resource Detail” können alle Detailinformationen zu einer zugeordneten Resource eingetragen und geändert werden.
Der Tab sieht folgendermaßen aus:

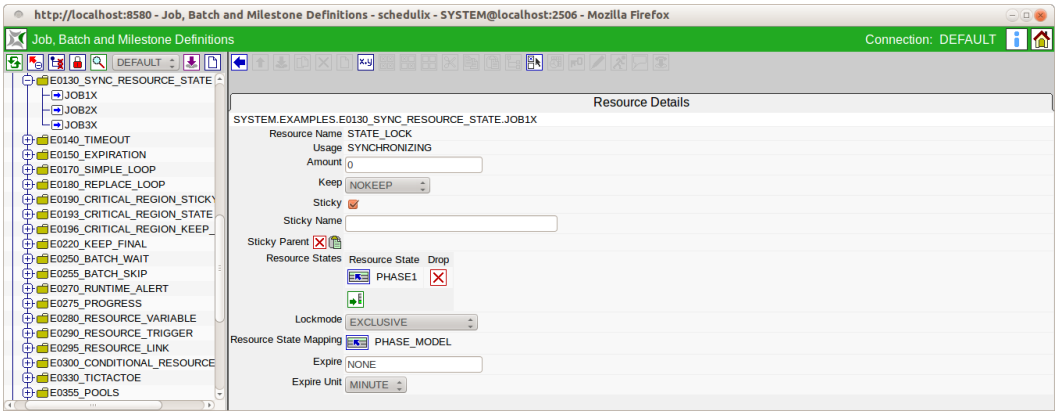


Abbildung 13.27: Batches und Jobs; Resource Requirement Details

Die Felder des Tabs ”Resource Details” haben folgende Bedeutung:

Resource Name Hierbei handelt es sich um den Namen der benötigten Resource. Es wird jeweils der volle Name mit der darüber liegenden Ordnerhierarchie angezeigt.

Usage Hierbei handelt es sich um die Usage der Resource. Weitere Informationen zur Usage finden Sie im Kapitel 9.3.3.

Amount Dieses Feld wird nur angezeigt, falls die Usage vom Typ ”SYSTEM” oder ”SYNCHRONIZING” ist.
Das ist die Menge der vom Job benötigten Resource. Im Jobserver (oder dem darüber liegenden Scopes, siehe Jobserver) werden maximale Mengen der Resource zur Verfügung gestellt. Jeder Job, der von dieser Resource eine Menge benötigt, reduziert die Anzahl der freien Mengen.
Beispiel:
Ein Jobserver stellt von der Resource A die Menge 5 zur Verfügung (zum Beispiel 5 CPU Einheiten auf diesem Server).
Job A startet nun und benötigt eine Menge von 3. Der Jobserver hat nun 3 CPU Einheiten belegt und 2 noch verfügbar.

Job B möchte starten, benötigt aber ebenfalls einen Amount von 3. Der Job kann nicht an den Jobserver zum Starten übergeben werden, da dieser im Moment nur noch 2 CPU Einheiten zur Verfügung hat.
Job B kann erst starten, wenn Job A beendet ist und mindestens drei CPU Einheiten zur Verfügung stehen.

Keep Hier wird der Wert des Keep Parameters eingestellt. Weitere Informationen zum Keep Parameter finden Sie im Kapitel 11.3.1.

Sticky Mittels dieses Schalters kann die Resource-Rückgabe gesteuert werden. Ist dieser Schalter gesetzt, so wird die Resource innerhalb eines Master Batch solange behalten, bis der letzte Job, der diese Resource mit eingeschaltetem Sticky Flag benötigt, beendet wurde.
Dies dient dazu, um einmal reservierte Resources innerhalb eines Master Batch zu behalten, ohne das die Resource von anderen Jobs in der Zwischenzeit verwendet werden darf.
Als Beispiel:

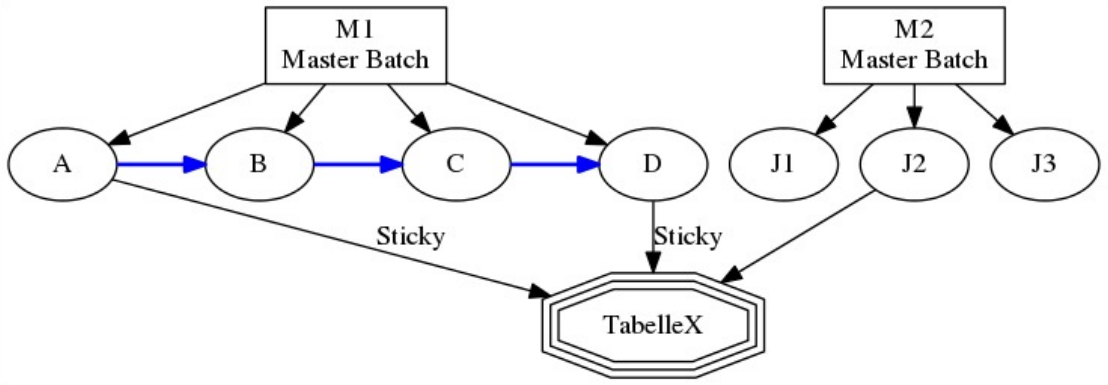


Abbildung 13.28: Beispiel für Sticky Handling

Master Batch M1 besteht aus den Jobs A, B, C und D, die nacheinander ausgeführt werden. Job A generiert eine Tabelle X, welche von Job D benötigt wird. Die Jobs B und C benötigen diese Tabelle X nicht. Andere Master Batches (M2) möchten auch auf die Tabelle zugreifen und diese verändern. Solange Job D aber noch nicht gelaufen ist, soll dies verhindert werden, da diese Aktionen das Ergebnis des Jobs A verfälschen und unbrauchbar machen würden. Die Tabelle X wird nun als Synchronizing Resource abgebildet.
Ohne gesetztes Sticky Flag würde die Resource nach dem Ende des Jobs A zurückgegeben werden und die Tabelle könnte von J2 verändert werden. Mit gesetztem

Sticky Flag (welches in Job A und in Job D gesetzt werden muss) wird die Resource nach der Beendigung des Jobs A nicht zurückgegeben, sondern im Master Batch als belegt gehalten.
Andere Jobs können diese nun nicht ändern. Erst nach der Beendigung des Jobs D (des letzten Jobs in diesem Master Job, der die Resource mit gesetztem Sticky Flag benötigt), wird die Resource freigegeben und andere Jobs können diese verwenden.
Mit dem Sticky Flag kann man über eine Sticky Resource alle notwendigen Prozesse auf dem selben Scope bzw. Jobserver ausführen lassen.
Hier ein weiteres Beispiel zur Erläuterung:

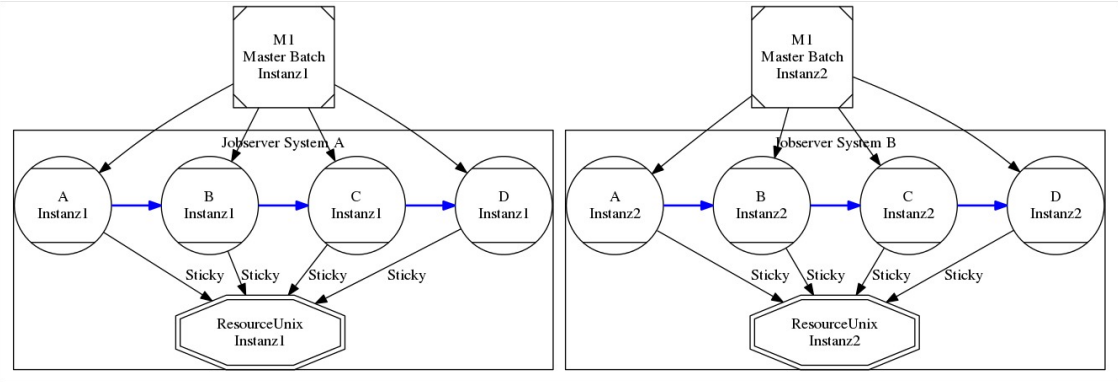


Abbildung 13.29: Beispiel für Lastverteilung mit Sticky Handling

Es gibt in einer Systemumgebung 2 Unix-Rechner mit identischer Konfiguration und Umgebung, die zur Lastverteilung eingesetzt werden. Nun kann ein Master Batch entweder auf der einen oder auf der anderen Maschine laufen. Da aber Zwischenergebnisse als Files zwischen den einzelnen Jobs des Master Batch übertragen werden sollen, muss der gesamte Master Batch, nachdem die Maschine ausgewählt wurde, zwingend auf dieser laufen.
Dies kann mittels des Sticky Flags erreicht werden. Es werden auf diesen Maschinen jeweils Jobserver (Jobserver System A und System B) installiert und die Umgebung wird als eine Synchronizing Resource Definition (Resource Unix) abgebildet. Innerhalb der beiden Jobserver wird nun diese Resource Definition zugeordnet (2 mal, in jeden Jobserver 1 mal). Nun müssen alle Children des Master Batch so definiert werden, dass sie die Resource Definition mit dem Sticky Flag benötigen.
Der erste Job (Job A) des Master Batch belegt nun eine der beiden verfügbaren Resources (auf dem einen oder dem anderen Jobserver). Da diese sticky belegt wurde, wird sie nach Beendigung des ersten Jobs nicht mehr zurückgegeben. Alle weiteren Jobs des Master Batches sind über diese Sticky Resource auf den gewählten Jobserver und damit auf diese Maschine festgelegt.
In der Grafik läuft nun ein Master Batch (Instanz 1) vollständig auf dem Jobserver

System A und ein zweiter Master Batch (Instanz 2) vollständig auf dem Jobserver System B.

Sticky Name Ist das oben beschriebene Sticky Flag gesetzt, so wird auch das Feld *Sticky Name* sichtbar. Ist der *Sticky Name* gesetzt, so steht die Resource nicht allen Jobs des selben Master Runs zur Verfügung, sondern nur den Jobs, welche die Resource mit dem selben *Sticky Name* allokieren wollen.

Sticky Parent Ist das oben beschriebene Sticky Flag gesetzt, so wird auch das Feld *Sticky Parent* sichtbar. Ist der *Sticky Parent* gesetzt, so steht die Resource nicht allen Jobs des selben Master Runs zur Verfügung, sondern nur den Jobs, welche die Resource mit dem selben *Sticky Parent* allokieren wollen.

Liste Resource States Die *Liste Resource States* wird nur angezeigt, wenn die Resource vom Typ "Synchronizing" ist und ein Resource State Profile zugeordnet wurde.
Hier kann eine Liste von benötigten Resource States eingetragen werden. Der Job kann nur dann starten, wenn die benötigte Resource in einem dieser States ist.

Lockmode Der *Lockmode* gibt an, mit welchem Zugriffsmodus auf die benötigte Resource zugegriffen werden soll. Weitere Informationen zum *Lockmode* finden Sie im Kapitel 12.4.2.3.

Resource State Mapping Dieses Feld wird nur angezeigt, wenn die Resource vom Typ "Synchronizing" ist und ein Resource State Profile zugeordnet wurde.
Das *Resource State Mapping* gibt an, wie und ob der State der Resource nach Beendigung des Jobs geändert werden soll. Weitere Informationen zum Resource State Mapping finden Sie im Kapitel 8.

Expire Gibt den Wert des Expirations-Intervalls an. Weitere Informationen zu Expire finden Sie im Kapitel 9.3.6.

Expire Unit Gibt die Einheit des Expirations-Intervalls an. Weitere Informationen zu Expire Unit finden Sie im Kapitel 9.3.6.

Ignore Expiration On Rerun Wird dieses Flag gesetzt, so wird die Expire Bedingung bei einem Rerun nicht erneut geprüft. Damit lassen sich Jobs, die schon einmal gelaufen sind, also die Expiration Bedingung erfüllt war, auch nach einer Zeit restarten, selbst wenn die Expire Bedingung dann nicht mehr erfüllt ist.

13.5.9 Tab Defined Resources

Die ”Defined Resources” werden in dem Moment instanziiert, in dem der Job startet. Diese Resources sind nur für Jobs sichtbar die im gleichen Ablauf sind.

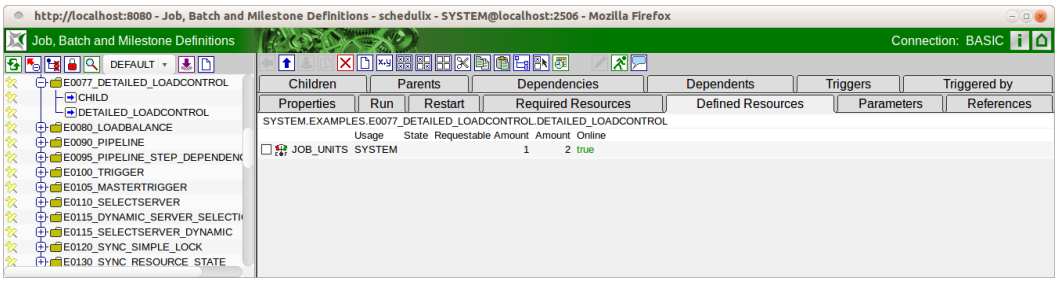


Abbildung 13.30: Batches und Jobs; Defined Resources

Die Felder der obigen Liste haben folgende Bedeutung:

Usage Die *Usage* gibt an, um welchen Typ ”Resource” es sich handelt. Mehr zu Typen von Resources finden Sie im Dialog Named Resource.

State Das Feld *State* ist nur zu sehen, wenn die Usage der Resource ”Synchronizing” ist und ein Resource State Profile zugeordnet wurde. Hierbei handelt es sich um den aktuellen Status der Resource in diesem Scope oder Jobserver. Der Status kann mittels dieses Wertes gesetzt bzw. geändert werden. In der ”Drop Down”-Liste sind alle gültigen Resource States des Resource State Profiles enthalten und können ausgewählt werden.
Dies kann bei einer manuellen Fehlerbehebung notwendig sein, um eine Resource wieder in einen Status zu bringen, in dem eine Weiterverarbeitung durch Folgejobs möglich ist.
Falls hier der Wert manuell geändert wird, wird der Wert des Feldes *State* im Tab ”Resources” nicht automatisch aktualisiert. Falls dies gewünscht und notwendig ist, kann dies mittels des *Refresh* Buttons durchgeführt werden.

Requestable Amount Die Menge der Resources die maximal von einem Job angefordert werden darf.

Amount Beim *Amount* handelt es sich um die Anzahl der vom aktuellen Job belegten oder angeforderten Resource-Instanzen. Übersteigt der Wert des Amounts die aktuell verfügbare Anzahl, die im Feld *Amount* im Tab ”Resource Detail” gepflegt werden können und ist kein alternativer Scope mit einer ausreichenden Anzahl vorhanden, kann der Job nicht ausgeführt werden.

Online Mittels *Online* ist es möglich, eine Resource in diesem Scope oder Jobserver on- bzw. offline zu schalten. Ist die Resource offline, so steht diese Resource vorübergehend nicht zur Verfügung.

13.5.10 Tab Parameters

Im Tab ”Parameters” können Parameter definiert werden, welche zur Kommunikation und Datenübermittlung zwischen Jobs verwendet werden können. Auch Parameter, welche von der Umgebung oder aus dem Scope bzw. Ordnerumfeld der Jobserver und Job-Umgebung kommen, können hier für die Verwendung innerhalb des aktuellen Jobs definiert werden.
Der Tab ”Parameters” sieht folgendermaßen aus:

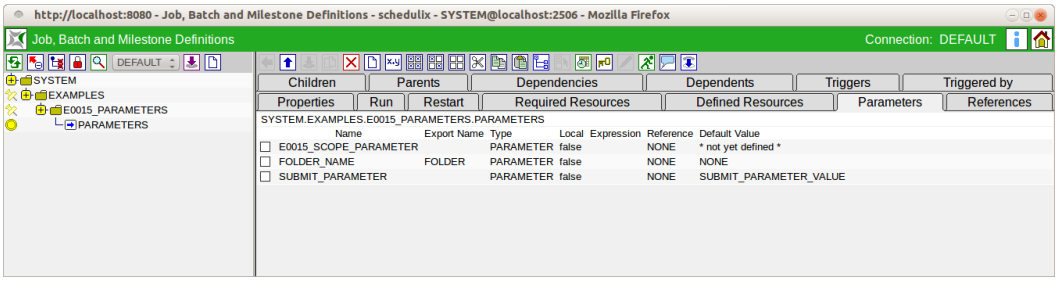


Abbildung 13.31: Batches und Jobs; Parameters Tab

In der obigen Liste sind alle Parameter aufgelistet, welche aktuell in diesem Scheduling Entity definiert wurden. Mittels des *New* Buttons kann ein neuer Parameter angelegt werden. Nach Eingabe der Daten und Rückkehr in die Liste, erscheint dieser auch hier. Mittels des *Delete* Buttons können ausgewählte Parameter auch gelöscht werden. Durch Anklicken des Parameternamens kommt man in den Tab ”Parameter Details”.

Die Spalten des Tabs ”Parameters” werden im nächsten Abschnitt erläutert.

13.5.10.1 Tab Parameter Details

Der Tab ”Parameter Details” erscheint, falls im Tab ”Parameters” ein Parameter ausgewählt wurde oder mittels des *New* Buttons ein neuer Parameter angelegt werden soll. Hier können alle Detailinformationen eines Parameters eingetragen und geändert werden.
Der Tab ”Parameter Details” sieht folgendermaßen aus:

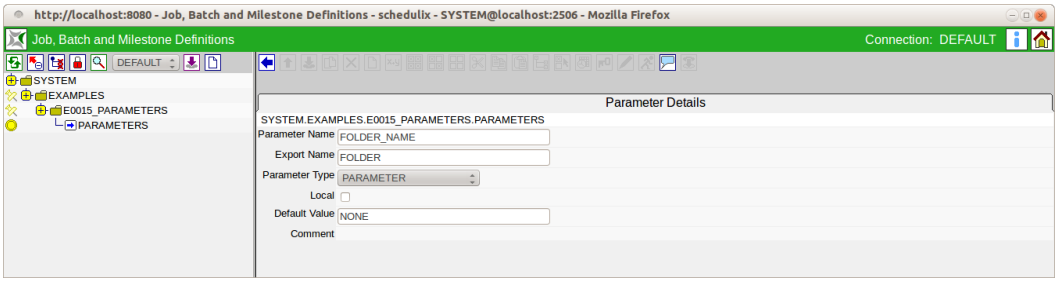


Abbildung 13.32: Batches und Jobs; Parameter Details

Die Felder des Tabs ”Parameter Details” haben folgende Bedeutung:

Parameter Name Hierbei handelt es sich um den Namen des Parameters. Durch den Namen erfolgt die komplette Werteübergabe der Parameter zwischen einzelnen Jobs. Der Wert des Parameters wird vom ersten Job gesetzt und kann im zweiten über den identischen Namen verwendet werden.

Export Name Wird hier ein, für eine OS Umgebungsvariable gültiger Name eingegeben, so wird der Wert des Parameters zur Laufzeit dem Job unter diesem Namen zur Verfügung stehen. Ist das Feld leer oder NONE so wird dieser Parameter nicht als Umgebungsvariable für den Job verfügbar gemacht.

Parameter Type Beim Type handelt es sich um die Art des Parameters und wie er verwendet wird.
Optionen für ”Parameter Type”:

1. PARAMETER

   Parameter vom Typ ”Parameter” können zum Zeitpunkt des Submits spezifiziert werden, wenn ein Default-Wert spezifiziert wurde oder müssen, wenn kein Default spezifiziert wurde. Dies gilt allerdings nur für die Parameter des Master Batches. Alle anderen Parameter verhalten sich, als wären sie vom Typ ”Import”.

2. IMPORT

Der Wert muss erst zur Ausführungszeit auflösbar sein. Die Verwendung der Variable wird explizit dokumentiert.

3. RESULT

   Das Result stellt das Ergebnis eines Jobs dar, welches an einem Nachfolgejob exportiert werden kann.

4. CONSTANT

   Der Parameter ist eine Konstante, die nicht mehr verändert werden kann. Der Wert der Konstante wird mittels des Feldes *Default Value* eingetragen. Dieser Wert ist im Falle des Typs ”Constant” zwingend erforderlich.

5. EXPRESSION

   Ein Expression Parameter stellt eine Sonderform eines Parameters für parallele Verarbeitung mit dynamischen Child Jobs dar. Um in diesem Fall ein Aggregat über alle Children zu bekommen, kann im zusätzlich erscheinenden Feld ”Expression” eine Aggregatsfunktion ausgewählt und im Feld ”Expression Parameter Name” den zu aggregierenden Child Parameter angegeben werden.

   Wird der Expression Parameter zur Laufzeit benutzt, wird in allen Instanzen der Child Jobs nach dem Wert dieses Parameters gesucht und die dementsprechende Aggregatsfunktion ausgeführt.

   Ein Beispiel zur Verdeutlichung:

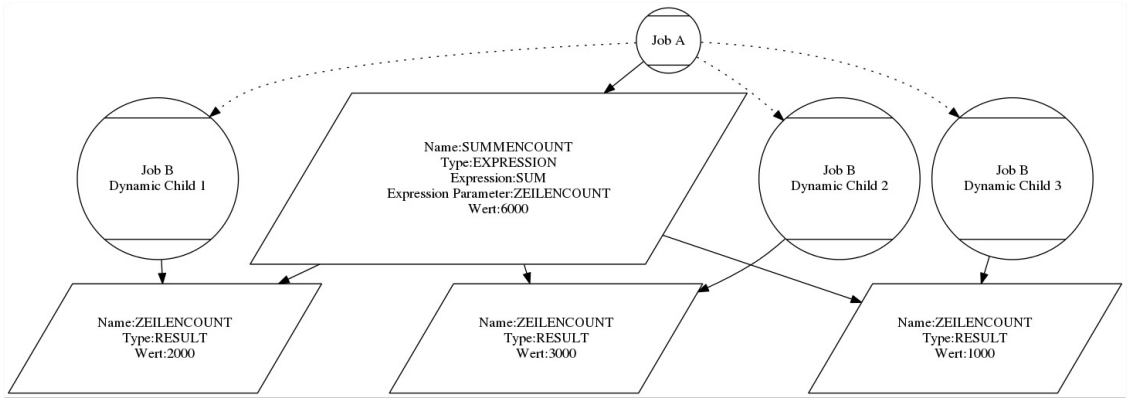


Abbildung 13.33: Beispiel für Expression

Job A hat einen dynamischen Child Job B, welcher zur Laufzeit parallel submitted wird. Die Child Jobs B schreiben alle x-Zeilen einer Tabelle parallel. Nun möchte A gerne wissen, wie viele Zeilen alle Child Jobs in Summe geschrieben haben.

Um dies zu erreichen, muss Job B einen Parameter (Type Result) ZEILENCOUNT definiert haben, der beim Abschluss des Job-Laufes die Anzahl der bearbeiteten Zeilen enthält.

In Job A wird nun ein Parameter SUMMENCOUNT angelegt, mit dem Typ "Expression", mit der Expression Option "SUM" und dem Parameternamen "ZEILENCOUNT".

Nun startet Job A und submitted drei Child Jobs B (B1, B2, B3). B1 schreibt 2000 Zeilen, B2 schreibt 3000 Zeilen, B3 schreibt 1000 Zeilen. Nach Beendigung der Children fragt Job A (oder ein Nachfolgejob) den Parameter SUMMENCOUNT ab und erhält als Ergebnis: 6000 Zeilen, was der Summe aller Children entspricht.

6. REFERENCE

   Durch den Typ "Reference" kann ein Parameter als Referenz eines anderen Parameters vergeben werden. Dieser Parameter wird durch eine Quersuche in allen Jobs, welche mit dem aktuellen Job auf einer Ebene liegen, durchgeführt. Hiermit ist es möglich, Parameter von Jobs zu bekommen, welche nicht unmittelbar ein Parent des aktuellen Jobs sind. Der Parameter, der referenziert werden soll, muss im Feld *Reference* mittels *Paste* eingefügt werden (*Copy* aus dem "Parameter" Tab einer anderen Job Definition).

7. CHILDREFERENCE

   Durch den Typ "Childreference" kann ein Parameter als Referenz eines anderen Parameters vergeben werden. Dieser referenzierte Parameter wird durch eine Suche in allen Children des aktuellen Jobs ermittelt. Hiermit ist es möglich, Parameter von den eigenen Children zu ermitteln.

8. RESOURCEREFERENCE

   Durch den Typ "Resourcereference" kann ein Parameter als Referenz eines Resource Parameters vergeben werden. Der Parameter, der referenziert werden soll, muss im Feld *Reference* mittels *Paste* eingefügt werden (*Copy* aus dem "Parameter" Tab einer Named Resource).

Expression Dieses Feld erscheint nur, falls im Feld *Parameter Type* der Wert "Expression" gewählt wurde.
Hier kann die Art der Aggregatsfunktion ausgewählt werden.
Bei allen Aggregatsfunktionen gilt: Sollte in manchen Child-Objekten der Parameter oder der Parameterwert nicht vorhanden oder vom falschen Typ sein (String statt Zahl), dann werden diese Werte ignoriert und nur die zur Verfügung stehenden und korrekten Werte aggregiert.
Es stehen folgende Aggregate zur Verfügung:

1. SUM

   Die Summe aller Werte des referenzierten Parameters aller Child-Elemente

2. AVG

   Der Durchschnitt aller Werte des referenzierten Parameters aller Child-Elemente, welche gültige und vorhandene Werte enthalten

3. MIN

   Das Minimum aller Werte des referenzierten Parameters

4. MAX

   Das Maximum aller Werte des referenzierten Parameters

5. COUNT

   Die Anzahl der Child-Instanzen, welche diesen Parameter definiert haben

6. NONE

   Es wurde noch keine Aggregatsfunktion gewählt. None stellt keinen gültigen Wert dar.

Expression Parameter Name Dieses Feld erscheint nur, falls im Feld *Parameter Type* der Wert "Expression" gewählt wurde.
Das ist der Name des Child-Parameters, auf dem das Aggregat gebildet werden soll. Weitere Informationen zu Aggregaten finden Sie im Feld *Expression*.

Local Wenn das Häkchen gesetzt ist, ist dieser Parameter nur vom Job selbst sichtbar.

Default Value Der *Default Value* gibt an, welcher Wert der Standardwert ist. Dieser Wert wird bei der Parameterauflösung zurückgegeben, wenn bei der Suche kein explizit gesetzter Wert gefunden wird. Das Feld ist optional. Im Falle einer Konstante muss das Feld einen Wert besitzen.

Variablen werden rekursiv ausgewertet. Das bedeutet, wenn innerhalb einer Variable etwas der Form $NAME bzw. $NAME auftritt, wird die Variable NAME aufgelöst und der Wert an der Stelle eingesetzt. Auf diese Art und Weise ist es nun auch möglich geworden - indirekt - auf Systemvariablen anderer Jobs zuzugreifen. Ein $-Zeichen wird durch \$ dargestellt. Ein Backslash durch \\.

Comment Das Feld *Comment* dient als Möglichkeit einer näheren Erläuterung (Kommentar) des Objektes.

Reference Dieses Feld erscheint nur, falls im Feld *Parameter Type* der Wert "Reference", "Childreference" oder "Resourcereference"gewählt wurde.
Hier kann mittels *Copy and Paste* ein Parameter eingetragen werden, welcher über den gewählten Namen referenziert wird.

13.5.10.2 Predefined Standardparameter des Laufzeitsystems

Predefined Standardparameter im Laufzeitsystem können von allen Submitted Entities im Feld *Run Program* als Übergabeparameter oder im Run Program selbst verwendet werden. Die Predefined Standardparameter enthalten Informationen über das Submitted Entity und dessen Umgebung.
Es werden folgende Standardparameter vom Laufzeitsystem angeboten:

Job-Umgebungsparameter Hier handelt es sich um alle Parameter, welche mit dem Submitted Entity selber zu tun haben. Job-Umgebungsparameter-Namen:

1. JOBNAME

   Das ist der Name des Jobs, wie er im Feld *Name* angegeben wurde.

2. JOBID

   Hierbei handelt es sich um die Id des Submitted Entities. Hiermit ist eine eindeutige Identifikation der laufenden Instanz des Entities möglich.

3. KEY

   Beim *Key* handelt es sich um das Passwort für den jeweiligen Job. Hiermit ist ein Anmelden an den Server möglich.

4. MASTERID

   Die *MASTERID* ist die Id des Master Jobs, der den aktuellen Job ausführt.

5. PID

   Die *PID* ist die Prozess-ID auf Betriebssystemebene des Submitted Entities. Dies kann zum Beispiel für das Kill Program verwendet werden.

6. LOGFILE

   Der Parameter LOGFILE gibt das verwendete Logfile an, welches im Feld *Logfile* im Batches and Jobs-Dialog definiert wurde.

7. ERRORLOG

   Der Parameter ERRORLOG gibt das verwendete Error Logfile an, welches im Feld *Error Logfile* im Batches and Jobs-Dialog definiert wurde.

8. SDMSHOST

   Der Parameter SDMSHOST gibt den aktuellen Host des SDMS Servers an.

9. SDMSPORT

   Der Parameter SDMSPORT gibt den aktuellen Port des SDMS Servers auf dem Host an.

10. JOBTAG

    Das JOBTAG gibt das Child Tag des Submitted Entities an. Hiermit ist eine eindeutige Identifikation eines dynamisch erzeugten Child Entities möglich.

11. SUBMITTIME

    Die SUBMITTIME gibt den 'Submitted' Zeitpunkt' des Master Jobs an.

12. STARTTIME

    Die STARTTIME gibt den Zeitpunkt des tatsächlichen Starts des Submitted Entities auf dem Jobserver (falls es sich um einen Job handelt) an.

13. SEID

    Die SEID ist die ID des Scheduling Entities.

14. WORKDIR

    DER WORKDIR ist der Pfad des Working Directory.

15. JOBSTATE

    Der JOBSTATE ist der Exit State des Jobs, Batches oder Milestones.

16. MERGEDSTATE

    Der MERGEDSTATE ist der resultierende Exit State eines Jobs und seiner Children.

17. EXPRUNTIME

    Die EXPRUNTIME ist die Expected Runtime, die in der Job Definition hinterlegt wurde.

18. EXPFINALTIME

    Die EXPFINALTIME ist die Expected Finaltime, die in der Job Definition hinterlegt wurde.

19. PARENTID

    Die PARENTID ist die ID des Parent Jobs, Milestones oder Batches.

20. STATE

    Der STATE ist der aktuelle Status des Jobs.

21. ISRESTARTABLE

    ISRESTARTABLE sagt über den Job aus, ob er Restartable ist oder nicht.

22. SYNCTIME

    Die SYNCTIME ist die Zeit, zu der der Job in den Status SYNCHRONIZE_WAIT übergegangen ist.

23. RESOURCETIME

    Die RESOURCETIME ist die Zeit, zu der der Job in den Status RESOURCE_WAIT übergegangen ist.

24. RUNNABLETIME

    RUNNABLETIME ist die Zeit, zu der der Job in den Status RUNNABLE gewechselt ist.

25. FINISHTIME

    FINISHTIME ist der Endzeitpunkt des Prozesses.

26. SYSDATE

    Die Variable SYSDATE ist das aktuelle Datum und Uhrzeit.

27. LAST_WARNING

    Diese Variable enthält den Text des letzten Audit-Eintrags, der das Hochsetzen des Warncounts zur Folge hatte.

28. RERUNSEQ

    Diese Variable gibt an, wie oft der Job bereits restartet wurde.

29. SCOPENAME

    Liefert den vollen Pfadnamen des Jobservers, der den Job ausführt bzw. ausgeführt hat.

Trigger Parameter Für die Verwendung in Triggern gibt es eine weitere Anzahl von vordefinierten Parametern, welche in Skripten von Trigger Jobs verwendet werden können. Diese Parameter stehen nur zur Verfügung, falls das Submitted Entity im Rahmen eines Triggers gestartet wurde. Wurde das Entity normal submitted, sind diese Parameter nicht gefüllt.
Trigger-Umgebungs-Namen:

1. TRIGGERNAME

   Hierbei handelt es sich um den Namen des Triggers.

2. TRIGGERTYPE

   Hiermit kann aufgrund des Trigger-Typs ein Job unterschiedliche Handlungen vornehmen.

3. TRIGGERBASE

   Der Parameter TRIGGERBASE enthält den Namen des "Triggering Objects", also desjenigen Jobs, welcher zum Beispiel durch eine Exit State Translation den Trigger nach oben übermittelt.

4. TRIGGERORIGIN

   Der Parameter TRIGGERORIGIN gibt den Namen des Entities an, welcher den Trigger enthält. Das ist der Job der den Trigger in seinem Tab Trigger angezeigt bekommt.

5. TRIGGERREASON

   Der Parameter TRIGGERREASON gibt den Namen des Objektes aus, welcher den Trigger auslöst.

6. TRIGGERBASEID, TRIGGERORIGINID, TRIGGERREASONID

   Für jeden der drei oben genannten Objekte (TRIGGERBASE, TRIGGERREASON, TRIGGERORIGIN) wird in diesem Parameter die ID des Submitted Entities gespeichert.

7. TRIGGERBASEJOBID, TRIGGERORIGINJOBID, TRIGGERREASONJOBID

   Für jeden der drei oben genannten Objekte wird in diesem Parameter die ID der Job Definition gespeichert.

8. TRIGGEROLDSTATE

   Der Parameter gibt den Job State vor dem auslösenden Moment, welches für den Trigger relevant ist, an.

9. TRIGGERNEWSTATE

   Der Parameter gibt den Job State nach dem auslösenden Moment, welches für den Trigger relevant ist, an.

10. TRIGGERSEQNO

    TRIGGERSEQNO ist die Anzahl der Auslösungen des Triggers.

13.5.11 Tab References

Im Tab ”References” werden alle Referenzen (Parameter vom Type REFERENCE und CHILDREFERENCE) anderer Job- bzw. Batch-Objekte auf Parameter des aktuellen Objektes angezeigt.
Der Tab ”References” sieht folgendermaßen aus:

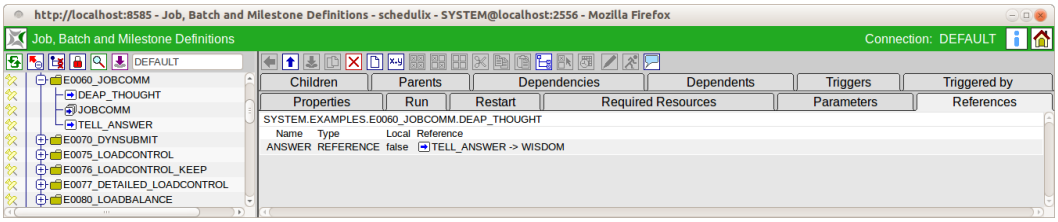


Abbildung 13.34: Batches und Jobs; References Tab

13.5.12 Tab Trigger

Der Tab ”Trigger” beschreibt alle Trigger, die für dieses Scheduling Entity definiert sind. Ein ”Trigger” dient zum Starten eines weiteren Ausführungsobjektes, falls beim aktuell gewählten Job (oder eines seiner Children, siehe unten) ein einstellbares Ereignis erreicht wurde. Als Beispiel können Benachrichtigungen verschickt werden, wenn ein Job beendet worden ist. Auch können Nachfolgeverarbeitungen oder eine Fehlerbehandlung bei Problemen automatisch gestartet werden.
Der Tab ”Trigger” sieht folgendermaßen aus:

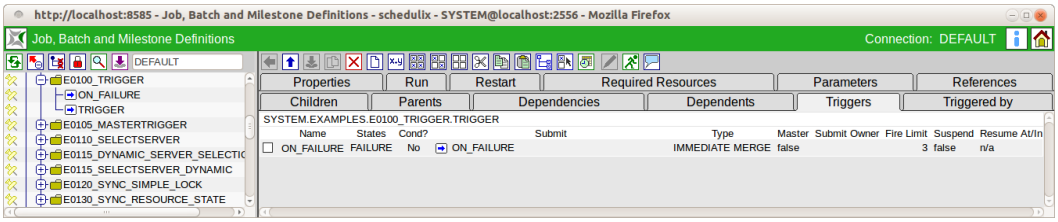


Abbildung 13.35: Batches und Jobs; Trigger Tab

Im Dialog ”Trigger” wird eine Liste mit allen, für den gewählten Task, zugeordneten Triggern angezeigt. Durch Selektieren eines Jobs kann der Job, der durch den Trigger gestartet werden soll, mittels Copy and Paste in die Liste eingefügt werden. Der Name des Triggers wird beim Kopieren vom Namen des Jobs übernommen. Anschließend können durch Anklicken des Trigger-Namens die Trigger Details angezeigt und geändert werden. Mittels des *Drop* Buttons werden Trigger gelöscht. Die Spalten der Liste haben folgende Bedeutung:

Name Das ist der Name des Triggers. Beim Einfügen eines neuen Jobs in die Trigger-Liste, wird der Trigger-Name auf dem Namen des Jobs gesetzt. Klickt man den Namen des Triggers an, kommt man in den Tab "Trigger Details". Hier können alle Daten (einschließlich des Trigger-Namens) eingesehen und geändert werden.

States Hierbei handelt es sich um eine kommagetrennte Liste aller States, bei denen der Trigger beachtet werden soll.

Cond? Das Feld *Cond?* gibt an, ob eine Bedingung spezifiziert wurde.

Submit Das ist der Name des Jobs oder Batches, welcher durch den Trigger submitted werden soll.
Achtung: Dieses Feld ist ebenfalls durch einen Mausklick anwählbar. Wird dies gemacht, kommt man in die Editiermaske des jeweiligen Jobs. Hat man Daten hinzugefügt oder geändert, und bestätigt die Sicherheitsabfrage, sind alle geänderten Daten verloren.
Die restlichen Spalten werden im nächsten Abschnitt erläutert.

13.5.12.1 Tab Trigger Details

Der Tab "Trigger Details" erscheint, nachdem ein Trigger im Tab "Trigger" gewählt wurde. Im "Trigger Details" Tab können alle Detailinformationen bzgl. des Triggers geändert, bzw. eingetragen werden.

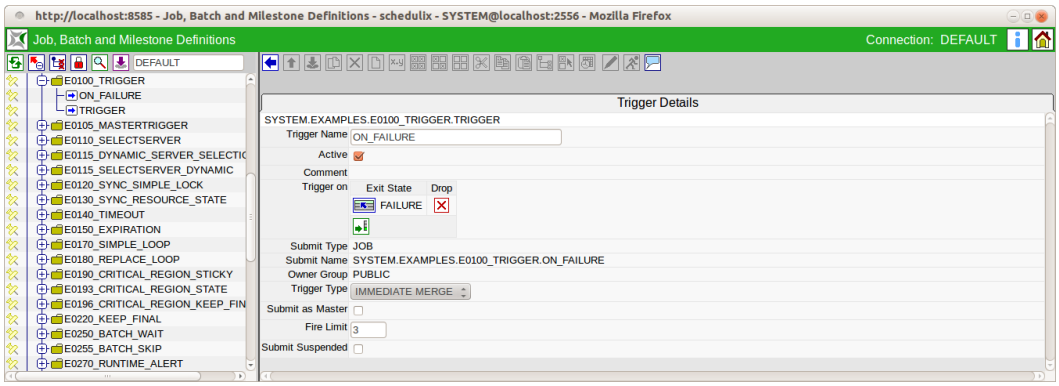


Abbildung 13.36: Batches und Jobs; Trigger Details

Folgende Felder sind auf dem Tab zu sehen:

Trigger Name Hierbei handelt es sich um den Namen des Triggers. Bei der Neuanlage eines Triggers, das heißt beim Einfügen eines Jobs in die Trigger-Liste, wird automatisch der Job Name übernommen. Anschließend kann in den "Trigger Details" der Name beliebig geändert werden.

Active Gibt an, ob der Trigger aktiv ist. Ist dieser Schalter nicht gesetzt, so wird der Trigger beim Eintreten des Trigger-Ereignisses nicht ausgelöst. In der Liste der Trigger wird ein inaktiver Trigger mit einer schwarzen Hand (Suspend Symbol) angezeigt.

Liste Trigger On Hier kann eine Liste mit Exit States des gewählten Prozesses (nicht des zu triggernden Jobs) angegeben werden, bei denen der Trigger feuern soll. Wird keine Liste angegeben, ist der Exit State des Jobs egal, er feuert immer, wenn das zu erreichende Ereignis eingetreten ist.
Beispiel:
Ein Trigger soll starten, wenn ein Job mit einem Fehler (EXIT_STATE: FAILURE) beendet wird. Hier muss die Liste 'Trigger On' den State FAILURE gewählt haben. Andere States (SUCCESS, etc.) werden nicht eingetragen, da diese beim Feuern des Triggers nicht beachtet werden sollen.
Soll ein Trigger immer gestartet werden, wenn der Prozess endet (unabhängig vom State, der vielleicht nur im Job, den der Trigger startet, unterschieden wird), so kann die Liste "Trigger On" leer gelassen oder mit allen möglichen States, die der Job annehmen kann, gefüllt werden.

Condition Im Feld *Condition* wird die Bedingung, die erfüllt werden muss, eingetragen.

Submit Type Das ist der Typ des Objektes (Batch oder Job), welches gestartet wird, wenn der Trigger feuert.

Submit Name Das ist der Pfad und der Name des Objektes welches gestartet wird, wenn der Trigger feuert. Hier handelt es sich um den kompletten Namen des Objektes mit allen übergeordneten Ordnerhierarchien

Owner Group Im Feld *Owner Group* ist der Eigentümer des Triggers definiert.

Trigger Type Hier handelt es sich um die Art des Ereignisses, welches den Trigger auslösen soll. Es gibt folgende Ereignisse:

1. Immediate Merge

   Der Job selber oder eines seiner Children hat einen State in der 'Trigger On'-Liste angenommen. Ist kein Wert eingetragen, wird der Trigger bei jedem Statuswechsel gefeuert.

   Beispiel:

   Falls der Job oder eines seiner Children einen Fehler hat, soll eine SMS gesendet werden.

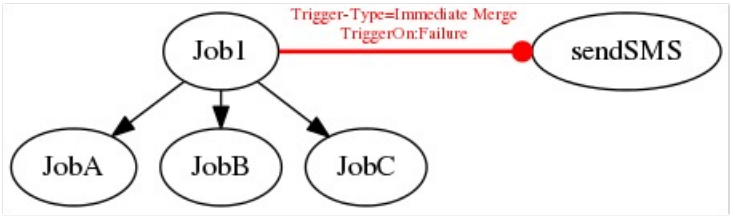


Abbildung 13.37: Beispiel für Immediate Merge Trigger

2. Immediate Local

   Der Job selber hat einen State in der 'Trigger On'-Liste angenommen. Ist kein Wert eingetragen, wird der Trigger bei Beenden des Jobs gefeuert. Der State der Children wird nicht berücksichtigt.

   Beispiel:

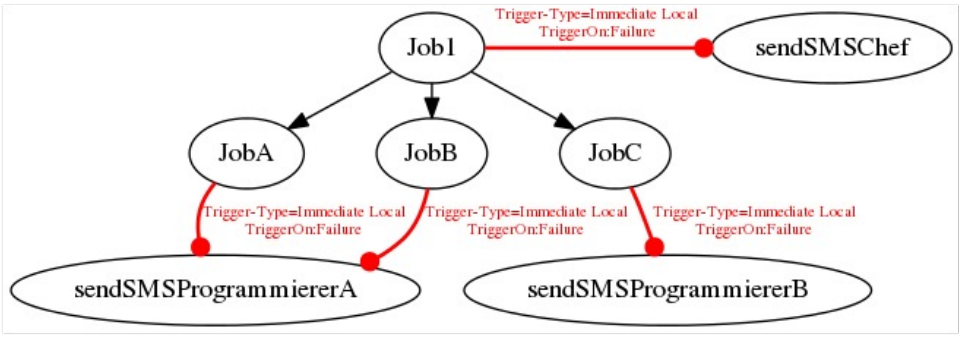


Abbildung 13.38: Beispiel für Immediate Local Trigger

   Falls der Job einen Fehler hat, soll eine SMS an den Projektverantwortlichen gesendet werden. Für den Fall, dass einer der Children einen Fehler hat, wird im Child ein zweiter Trigger implementiert, der eine SMS an einen der verantwortlichen Programmierer schickt.

3. Before Final

   Die Trigger-Bedingung wird, kurz bevor das Scheduling Entity einen Final State erreichen würde, evaluiert. Die Trigger On-Liste muss hierbei nur aus Final State (oder keinen, dann wird auf jeden Final State getriggert) bestehen, da der Trigger sonst nicht feuert. Wird das zu triggernde Scheduling Entity nicht als Master submitted, kann das Scheduling Entity erst wieder Final werden, nachdem das durch den Trigger gestartete Scheduling Entity ebenfalls Final geworden ist.

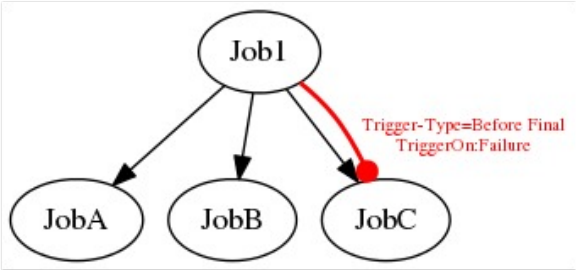


Abbildung 13.39: Beispiel für Before Final Trigger

Im oberen Beispiel, wird JobC erneut aufgerufen, falls der Job fehlschlägt. Da dies ein Before Final Trigger ist, kann damit Job 1 nicht Final werden, solange JobC ohne FAILURE beendet wurde.

4. After Final

   Der Trigger wird kurz nachdem der Job einen Final State erreicht hat ausgeführt. Die Trigger On-Liste sollte hierbei nur aus Final States bestehen (oder keinen, dann wird auf jeden Final State getriggert), da der Trigger sonst nicht feuert.

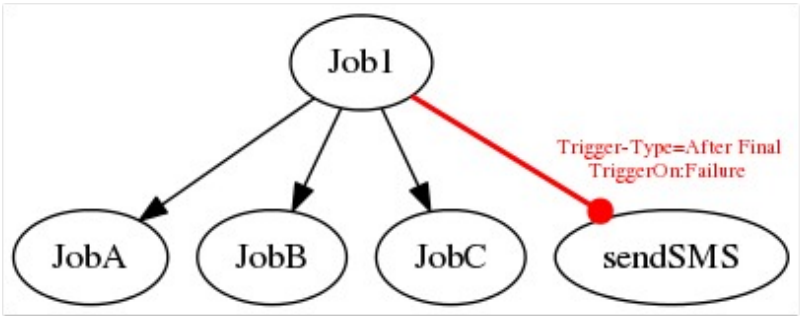


Abbildung 13.40: Beispiel für After Final Trigger

Die Legende für die Grafik finden Sie im Kapitel 1.8.

5. Finish Child

   Der Trigger wird ausgeführt, wenn ein direktes oder indirektes Child des triggernden Jobs einen State in der Trigger On-Liste erreicht. Hiermit ist es möglich, eine Überwachung an zentraler Stelle zu implementieren.

   Im oberen Beispiel würde eine SMS mit einer Fehlermeldung gesendet, falls einer der Child Jobs einen Failure State annimmt. Falls der eigene Job (Job 1)

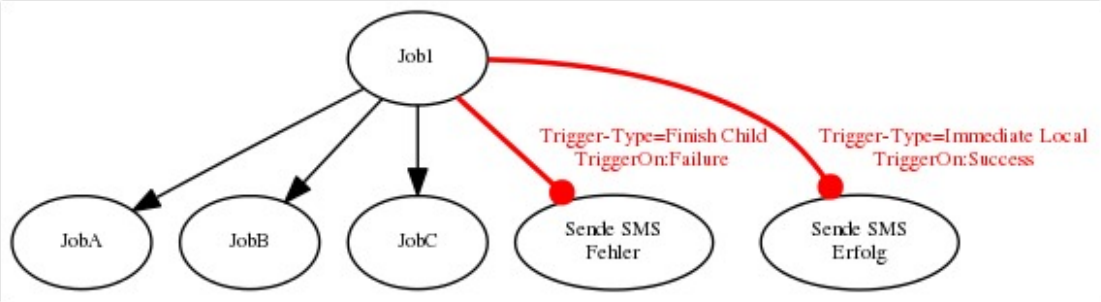


Abbildung 13.41: Beispiel für Finish Child Trigger

erfolgreich ist, wird die Erfolgsmeldung per SMS geschickt.

6. Until Finished

   Der Until Finished Trigger überprüft periodisch die Condition. Wenn die Condition ”True” ist, wird solange getriggert, bis der State ”Finished” erreicht wird. Anschließend wird die Condition zum letzten Mal ausgewertet, und wenn sie ”True” ist, wird getriggert.

7. Until Final

   Der Until Final Trigger überprüft periodisch die Condition. Wenn die Condition ”True” ist, wird solange getriggert, bis der State ”Final” erreicht wird. Anschließend wird die Condition zum letzten Mal ausgewertet, und wenn sie ”True” ist, wird getriggert.

Submit as Master Wurde das Flag ”Submit as Master” gesetzt, so wird der Job, der durch den Trigger gestartet wird, als eigener Master Job submitted und hat keinen Einfluss auf den aktuellen Master Job-Lauf des triggernden Jobs. Wird das Flag nicht gesetzt, wird der getriggerte Job als Child des zu triggernden Jobs gestartet. Das bedeutet auch, dass der Job auf den getriggert wird, nicht beendet ist (einen Final State bekommen kann), solange der getriggerte Job noch läuft. Als einzige Ausnahme gilt der Trigger Type After Final, bei dem der Trigger erst ausgelöst wird, nachdem der getriggerte Job Final geworden ist. Ist dies der Fall, wird der getriggerte Job als Child des Parents des triggernden Jobs gestartet. Wenn es sich bei dem getriggerten Job um das gleiche Scheduling Entity handelt wie der des triggernden Jobs, nimmt der getriggerte Job den Platz des triggernden Jobs ein.

Fire Limit Hiermit wird die Maximalanzahl der Auslösung des Triggers innerhalb eines Ablaufes angegeben. Ist die Maximalanzahl der Trigger-Auslösungen

erreicht, so wird kein weiterer Trigger mehr ausgelöst und kein neuer Job mehr gestartet.

Submit Suspended Gibt an, ob das getriggerte Scheduling Entity als "Suspended" gestartet werden soll.

Resume Wird nur angezeigt, falls *Submit Suspended* gesetzt wurde.
Hier kann ausgewählt werden, ob ein automatischer Resume stattfinden soll.
Es gibt folgende Möglichkeiten:

- NO: Wählt diese Funktionalität ab
- AT: Wählt einen automatischen Resume zu einem festen Zeitpunkt. Das Eingabefeld *Resume Time* wird angezeigt.
- IN: Wählt einen automatischen Resume nach Ablauf einer Zeit. Die Eingabefelder *Resume In* und *Unit* werden angezeigt.

Resume Time Wird nur angezeigt, falls im *Resume* "AT" gewählt wurde.
Hier wird der gewünschte Resume Zeitpunkt im Format "YYYY-MM-DDTHH:MI:SS" eingegeben.
Das Format orientiert sich an der ISO Norm 8601 und erlaubt auch unvollständige Angaben. Die Eingabe von 'T16:00' wird den Job um 16:00 Uhr resumen (ausgehend von der Zeit der Triggerauslösung).

Resume In Wird nur angezeigt falls im *Resume* "IN" gewählt wurde.
Hier wird angegeben, wie viele Zeiteinheiten (siehe *Unit*) bis zum Resume gewartet werden soll.

Unit Wird nur angezeigt, falls im *Resume* "IN" gewählt wurde.
Hier wird angegeben, ob es sich bei der Eingabe im Feld *Resume In* um Minuten (MINUTE), Stunden (HOUR), oder Tage (DAY) handeln soll.

13.5.13 Tab Triggered by

Der Tab ”Triggered by” listed alle Trigger, welche dieses Scheduling Entity submitten. Er sieht folgendermaßen aus:

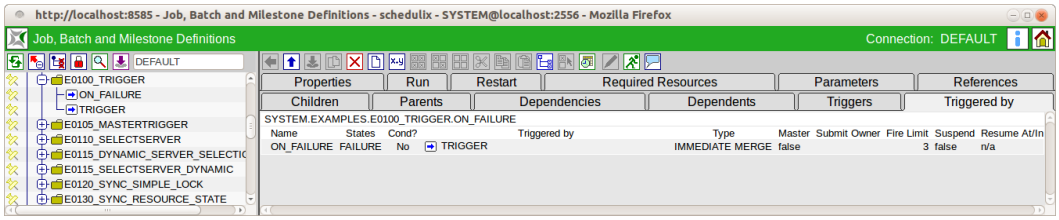


Abbildung 13.42: Jobs und Batches; Triggered By Tab

Die Spalten der Liste haben folgende Bedeutung:

Name Das ist der Name des Triggers.

States Hierbei handelt es sich um eine kommagetrennte Liste aller States, bei denen der Trigger beachtet werden soll.

Cond? Das Feld *Cond?* gibt an, ob eine Bedingung spezifiziert wurde.

Triggered by Das ist der Name des Jobs, Batches oder Milestones, welcher den Trigger definiert.
Achtung: Dieses Feld ist ebenfalls durch einen Mausklick anwählbar. Wird dies gemacht, kommt man in die Editiermaske des jeweiligen Jobs. Hat man Daten hinzugefügt oder geändert, und bestätigt die Sicherheitsabfrage, sind alle geänderten Daten verloren.
Die Bedeutung der restlichen Spalten sind dem Kapitel ”Tab Trigger Details” zu entnehmen.

13.6 Job Hierarchy Navigation

Das Navigationsfenster Job Hierarchy kann im Dialog "Batches and Jobs" zu jedem Batch und Job angezeigt werden. Der Navigationsbildschirm sieht folgendermaßen aus:

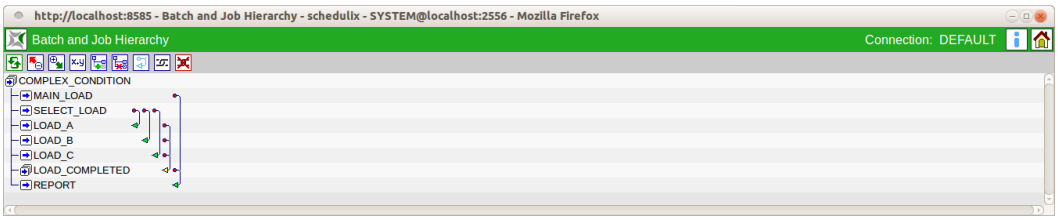


Abbildung 13.43: Batches und Jobs; Hierarchie Ansicht

Im Navigationsfenster werden alle Children des aktuellen Entities angezeigt. Haben die Children auch eine Child-Hierarchie, so kann diese ebenfalls expandiert werden, bis die komplette Hierarchie des Objektes sichtbar ist.

Zusätzlich zu den Standard-Buttons sind folgende Buttons sichtbar:

Add Children

Mit dem *Add Children* Button kann man Children direkt in das Job Hierarchy Navigationsfenster einfügen. Dazu wird der Button gedrückt und anschließend der Job oder Batch, der die Children bekommen soll, angeklickt. Es erscheint die Maske, in der ein oder mehrere Children ausgewählt werden können. Nachdem Sie ausgewählt haben, können Sie die Maske wieder schließen und als Letztes wird der Parent nochmals angeklickt und die Children werden automatisch hinzugefügt.

Remove Child

Mit dem *Remove Child* Button, können die Children gelöscht werden. Dazu wird der Button gedrückt und anschließend der Parent. Nachdem Sie dies mit "OK" bestätigt haben, werden die Children automatisch gelöscht.

Show Dependencies

Der *Show Dependencies* Button, ist ein Schalter mit zwei Werten. Standardmäßig werden die Abhängigkeiten im Dialog nicht angezeigt und es erscheint der *Show Dependencies* Button in der Buttonleiste.
Mit dem *Show Dependencies* Button werden die Abhängigkeiten der Scheduling Entities untereinander graphisch angezeigt. Jede Abhängigkeit wird mittels eines Pfeils angezeigt. Das benötigte Scheduling Entity (required Scheduling Entity) stellt den Start des Pfeils dar (der runde Teil des Pfeils) und das abhängige Scheduling Entity (dependent Scheduling Entity) ist die Pfeilspitze.

Beispiel:

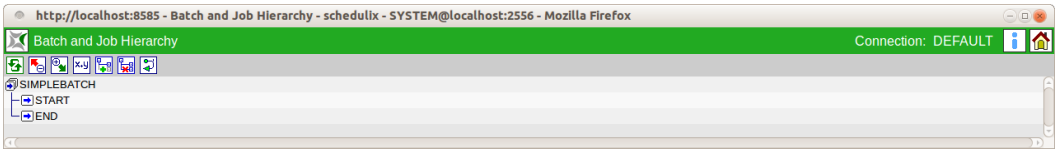


Abbildung 13.44: Hierarchie Ansicht mit Dependencies

Im oberen Beispiel ist der Job END vom Job START abhängig.
Ist ein Job von mehreren Objekten abhängig, so erscheint ein Pfeil mit mehreren Anfängen (die runden Teile des Pfeils), welche alle benötigten Scheduling Entities (required Scheduling Entities) darstellen. Die Pfeilspitze bleibt weiterhin das abhängige Scheduling Entity.
Beispiel:

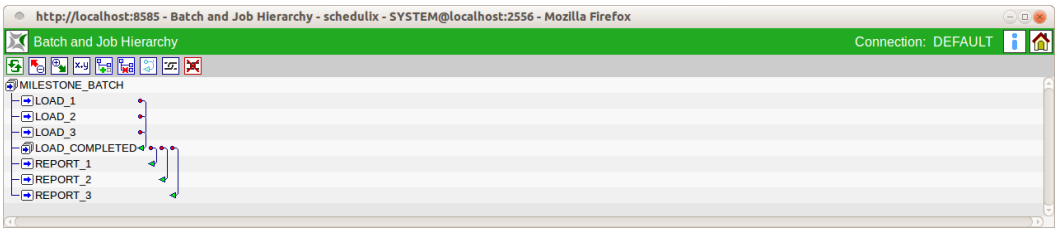


Abbildung 13.45: Hierarchie Ansicht mit mehrfachen Dependencies

In diesem Beispiel ist der Batch LOAD_COMPLETED von den Jobs LOAD_1, LOAD_2 und LOAD_3 abhängig.

Hide Dependencies

Der *Hide Dependencies* Button ist der zweite Zustand des Schalters. Er erscheint, falls der *Show Dependencies* Button gedrückt wurde und dadurch die Anzeige der Abhängigkeiten aktiviert ist. Mittels des *Hide Dependencies* Buttons ist es nun möglich, die Anzeige der Abhängigkeiten wieder abzuschalten.

Chain

Mit dem *Chain* Button kommt man in den Chain Modus. Hier kann man die Abhängigkeiten von Jobs direkt im Job Hierarchy Navigationsbildschirm einfügen. Dazu wird zuerst der *Chain* Button angeklickt und anschließend der Dependent Job und Required Job. Jetzt besteht eine Abhängigkeitsbeziehung zwischen den

beiden Jobs.
Mit dem *Cancel* Button verlässt man den Chain Modus.

Unchain

Mit dem *Unchain* Button werden Abhängigkeitsbeziehungen gelöst. Dazu wird zuerst der *Unchain* angeklickt und anschließend die Abhängigkeit, die gelöst werden soll.
Mit den *Chain/Unchain* Buttons ist es einfach, zyklische Abhängigkeiten zu erzeugen. Sie werden vom Frontend festgestellt und gemeldet. (Abbildung 13.46)

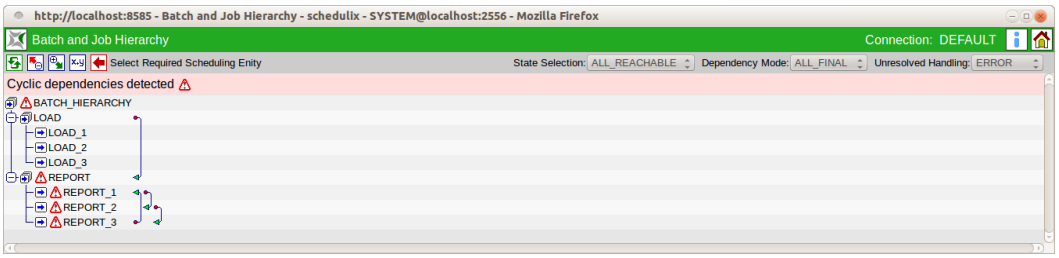


Abbildung 13.46: Zyklische Abhängigkeiten

13.7 Konvention für Batches, Jobs und Folder

Um das Arbeiten mit dem System und die Übersicht zu verbessern, empfehlen wir die Einhaltung folgender Konvention:
Für jeden Batch wird ein Folder mit gleichem Namen angelegt, welcher den Batch und seine direkten Children enthält. Hat ein Child selbst Children (z.B. ein Batch), so wird für dieses Child entsprechend dieser Konvention ein Subfolder erstellt, welches das Child und seine direkten Children enthält. Abgewichen wird von dieser Konvention nur, wenn ein Job oder Batch in mehreren Batches als Child verwendet wird.
Das schedulix!Web GUI unterstützt diese Konvention, wenn in den schedulix!Web User Einstellungen die Checkbox *Enable Batch In Folder Convention Support* gesetzt ist, wir folgt:

1. Beim Anlegen einen Folders kann über die Checkbox *Create Batch in Folder* ein Batch mit gleichem Namen im Folder erzeugt werden.

2. Beim Anlegen eines Batches kann über die Checkbox *Create Batch Folder* ein Folder mit gleichem Namen angelegt werden, der dann den Batch enthält.

3. Beim Anlegen von Objekten in einem Folder, welcher dieser Konvention folgt, also einen Batch oder Job gleichen Namens enthält, kann dieses über

die Checkbox *Add as Child* automatisch als Child dieses Batches bzw. Jobs eingehängt werden.

4. Wird der Name eines Folders geändert, kann über die Checkbox *Rename in Content* auch der Name des gleichnamige Batches oder Jobs im Folder mit verändert werden.

5. Wird der Name eines Batchs oder Jobs geändert, welcher in einem gleichnamigen Folder liegt, kann mit über die Checkbox *Rename Parent Folder* auch der Name des übergeordneten Folders mit verändert werden.

6. Wird der Name eines Objektes geändert, kann mit dem Button *Clone* eine Kopie des Objektes mit neuem Namen erzeugt werden. Deshalb werden auch in diesem Fall die für diese Konvention relevanten Checkboxes eingeblendet.

7. Beim Copy/Cut und nachfolgendem Paste in "Content" Tabs von Foldern, wird je nachdem, ob Quelle und Ziel der Operation der Konvention entspricht, eine Liste eingeblendet, welche die Operationen auflistet, welche an den Parent/Child-Beziehungen durchgeführt werden sollten um die Konvention einzuhalten. Diese Operationen können dann vor der Ausführung der Paste-Operation gezielt ausgewählt werden.

14 schedulix!Web Users

14.1 Bild

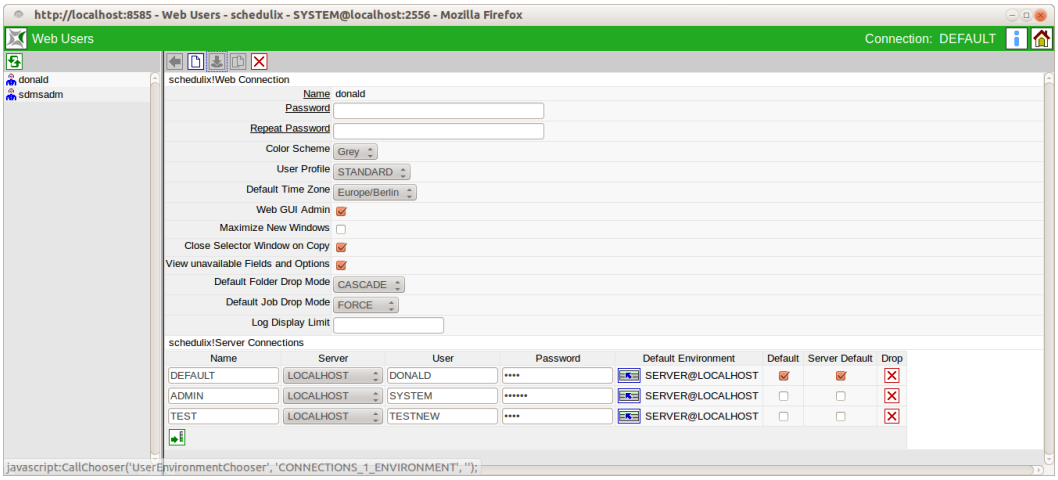


Abbildung 14.1: Benutzerverwaltung für das Web Frontend

14.2 Konzept

14.2.1 Kurzbeschreibung

Die Benutzerverwaltung dient der Definition von schedulix Oberflächenbenutzern. Jeder Benutzer, welcher mit der schedulix Oberfläche arbeiten will, sollte eine eigene Benutzerkennung haben. Der Benutzer muss sich während der Anmeldung mit dieser Benutzerkennung und deren Passwort anmelden.

14.2.2 Ausführliche Beschreibung

Zusätzlich zu Darstellungsoptionen und Sicherheitsmechanismen, ist in diesem Dialog die Verbindung zum Server pflegbar. Der schedulix Server verwendet eine eigene Benutzerverwaltung, welche unabhängig von der Benutzerverwaltung des Oberflächensystems ist. Aus diesem Grund ist es möglich, dass alle Oberflächenbenutzer nur einen Serverbenutzer benutzen oder dass eine 1:1 Abbildung der Ober-

flächenbenutzer auch im Server erfolgt. Bitte fragen Sie bei Änderungen an den Benutzeraccounts immer bei Ihrem Systemadministrator nach.

14.3 Navigator

Im Navigationsbildschirm werden alle Benutzerkennungen angezeigt.

14.4 Editor

Im Editor werden alle Daten eines gewählten Benutzers gepflegt.

14.4.1 schedulix!Web Connection

Die Felder im Web-GUI-Connect haben folgende Bedeutung:

Name Der Anmeldename des Benutzers.
Der Anmeldename ist im Dialog Login als Benutzername einzugeben.

Password Das *Password* dient zur Authentifizierung jedes Benutzers, welcher sich am schedulix System anmeldet und muss beim Login als Passwort eingegeben werden.
Das Passwort wird in der Anzeige verdeckt und muss identisch mit dem *Repeat Password* sein.
Da das Passwort im Dialog nicht angezeigt wird, ist es nicht ersichtlich, ob ein Passwort bereits eingegeben wurde oder dies noch notwendig ist. Es muss allerdings nur einmal eingegeben werden.

Repeat Password Das *Repeat Password* muss bei der Eingabe des Passwortes identisch zum Password sein.
Die doppelte Eingabe ist nötig, da ein Tippfehler beim blinden Eingeben des Passwortes zu Problemen führen kann. Da es unwahrscheinlich ist, dass zweimal der selbe Tippfehler gemacht wurde, geht man davon aus, dass bei einer korrekten zweifachen Eingabe des Passwortes der gewollte Begriff richtig eingetragen wurde.

Color Scheme Hier wird das für das Interface verwendete Farbschema ausgewählt.

User Profile Hier wird das Profil des Users eingegeben.

Default Time Zone Dieser Parameter definiert, welche Zeitzone bei der Neuanlage von Time Schedules voreingestellt wird. Weiterhin werden alle Zeitstempel in dieser Zeitzone angezeigt.

Time Stamp Format Hier kann ein vom Default (%Y.%m.%d %H:%M:%S Beispiel 2014.06.18 14:06) abweichendes Format für die Anzeige von Zeitstempeln eingestellt werden. Das Format entspricht dem der strftime C Standardfunktion.

Web-GUI-ADMIN Handelt es sich bei diesem Benutzer um einen Administrator der Weboberfläche, muss dieses Häkchen gesetzt werden.

Maximize New Windows Neue Dialoge werden immer maximiert angezeigt, wenn das Häkchen gesetzt wurde. Ist es nicht gesetzt, erscheinen die Fenster immer in der normalen Größe.

Close Selector Window on Copy Ist das Häkchen gesetzt, werden Fenster, welche zu Selektion von Objekten geöffnet werden, nach dem Betätigen des *Copy* Buttons automatisch geschlossen.

View unavailable Fields and Options Ist das Häkchen gesetzt, werden die nicht zur Verfügung stehenden Felder und Optionen angezeigt.

Default Folder Drop Mode Hierbei handelt es sich um den Modus, der verwendet wird, wenn ein Ordner im Dialog Batches and Jobs gelöscht werden soll. Es gibt folgende Optionen:

1. NORMAL

   In diesem Modus ist es nur möglich, einen Ordner zu löschen, wenn er leer ist, also keine Unterordner und keine Jobs, Batches oder Milestones besitzt.

   Dies ist der "Default"-Wert des Systems.

2. CASCADE

   In diesem Modus wird der Ordner kaskadierend gelöscht. Es werden alle Unterordner und alle darunter liegenden Jobs, Batches und Milestones gelöscht.

   Achtung: Diese Einstellung ist mit großer Vorsicht zu genießen, da es damit möglich ist, viele Objekte auf einmal zu löschen, obwohl man dies gar nicht beabsichtigt hat.

Default Job Drop Mode Hierbei handelt es sich um den Modus, welcher verwendet wird, wenn ein Job bzw. Batch oder Milestone gelöscht werden soll.
Es gibt folgende Optionen:

1. NORMAL

In diesem Modus ist es nur möglich, einen Job, Batch oder Mileston zu löschen, falls keine anderen Jobs mehr Abhängigkeiten zu diesem Job haben. Auch darf das Scheduling Entity keine Child-Beziehungen mehr besitzen.

Dies ist der "Default"-Wert des Systems.

2. FORCE

   In diesem Modus wird das Scheduling Entity gelöscht, obwohl noch andere Scheduling Entities Abhängigkeiten auf ihn besitzen und obwohl er noch Child-Beziehungen hat.

14.4.2 schedulix Server Connections

Hier kann eine oder mehrere Serververbindung konfiguriert werden.
Auf dem Main Desktop kann über ein Optionsfeld in der Kopfzeile gewählt werden, welche Serververbindung für das nächste zu öffnende Fenster verwendet werden soll.
Nach dem Editieren der Serververbindungen muss evtl. der Main Desktop im Browser neu geladen werden, damit die Serververbindungen im Optionsfeld der Kopfzeile richtig angezeigt werden.

Name Der Name der Serververbindung für die Auswahl in der Kopfzeile des Main Desktops.

Server Der Name des Servers wie er vom schedulix!Web Systemadministrator in /Custom/SDMSServers (in Zope) eingetragen wurde.

User Der Name des schedulix Server Benutzeraccounts. Dieser kann unterschiedlich zum Namen der Web Users sein.

Password Mit dieser Benutzerkennung und dem *Password* verbindet sich die schedulix Oberfläche mit dem schedulix Server.
Das Passwort wird in der Anzeige verdeckt.

Default Environment Das Default Environment wird bei Jobs standardmäßig eingetragen.

Default Ist das Häkchen gesetzt, so wird diese Serververbindung als Grundeinstellung verwendet. Es muss für genau eine Zeile gesetzt werden.

Server Default Ist das Häkchen gesetzt, so wird diese Serververbindung als Grundeinstellung verwendet, falls nur der Server bekannt ist. Es muss für jeden Server in genau einer Zeile gesetzt werden.

15 schedulix Server Users

15.1 Bild

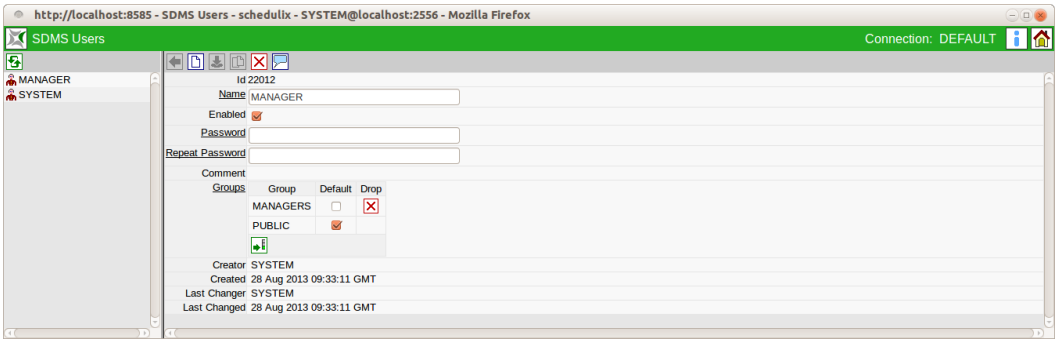


Abbildung 15.1: schedulix Server Benutzerverwaltung

15.2 Konzept

15.2.1 Kurzbeschreibung

Der schedulix Server User ist für die Legitimierung beim Scheduling Server notwendig.

15.2.2 Ausführliche Beschreibung

Da die Bedienung des Scheduling Servers nicht ausschließlich über die in dieser Dokumentation beschriebene Weboberfläche, sondern auch mit Hilfe anderer Schnittstellen wie etwa *sdmsh* erfolgen kann, ist eine Zugangskontrolle im Server notwendig. Die Pflege der Benutzer bzw. ihre Gruppenzugehörigkeit kann mittels des hier beschriebenen Dialogs durchgeführt werden.

15.3 Navigator

Im Navigationsbildschirm werden alle Benutzer angezeigt.
Der Navigatorbereich wird nur angezeigt, wenn der aktive Benutzer Mitglied der Gruppe ”ADMIN” ist. Allen anderen Benutzern wird nur der Editor angezeigt,

über welchen diese ihr eigenes Passwort und die als Grundeinstellung zu wählende Gruppe ändern können.

15.4 Editor

Im Editor werden alle Daten des gewählten oder neu angelegten Benutzers gepflegt. SDMS User dürfen nur von Benutzern, welche Mitglied der Gruppe "ADMIN" sind, editiert werden. Für alle anderen Benutzer sind alle Eingabefelder "read only".
Folgende Felder sind im Editor zu sehen:

Name Der Anmeldename des Benutzers.
Der Anmeldename ist im Dialog Login als Benutzername einzugeben.

Enabled Ist der aktive Benutzer Mitglied der Gruppe "ADMIN", so kann hier festgelegt werden, ob ein Login dieses Benutzers erlaubt werden soll.

Password Das *Password* dient zur Authentifizierung jedes Benutzers, welcher sich am schedulix System anmeldet und muss beim Login als Passwort eingegeben werden.
Das Passwort wird in der Anzeige verdeckt und muss identisch mit dem *Repeat Password* sein.
Da das Passwort im Dialog nicht angezeigt wird, ist es nicht ersichtlich, ob ein Passwort bereits eingegeben wurde oder dies noch notwendig ist. Er muss allerdings nur einmal eingegeben werden.
Die maximale Länge eines Passwortes beträgt 64 Zeichen. Alle Zeichen, also auch Leerzeichen, sind zulässig.
Das Passwort kann immer geändert werden, wenn der aktive Benutzer Mitglied der Gruppe "ADMIN" oder der zu ändernde Benutzer der aktive Benutzer ist (man kann natürlich sein eigenes Passwort ändern).

Repeat Password Das *Repeat Password* muss bei der Eingabe des Passwortes identisch zum Password sein.
Die doppelte Eingabe ist nötig, da ein Tippfehler beim blinden Eingeben des Passwortes zu Problemen führen kann. Da es unwahrscheinlich ist, dass zweimal der selbe Tippfehler gemacht wurde, geht man davon aus, dass bei einer korrekten zweifachen Eingabe des Passwortes, der gewollte Begriff richtig eingetragen wurde.

Groups In der Tabelle "Groups" sind die Gruppen aufgeführt, denen der Benutzer angehört.
Es gibt folgende Spalten:

Group Gruppen, in denen der angegebene User Mitglied ist.

Default Der Default zeigt an zu welcher Gruppe ein neu angelegtes Objekt gehört, wenn keine Gruppe spezifiziert ist.

Drop Dieser Button dient zum Löschen der Gruppenzugehörigkeit. Vor dem Löschen erfolgt eine Sicherheitsabfrage.

16 Groups

16.1 Bild

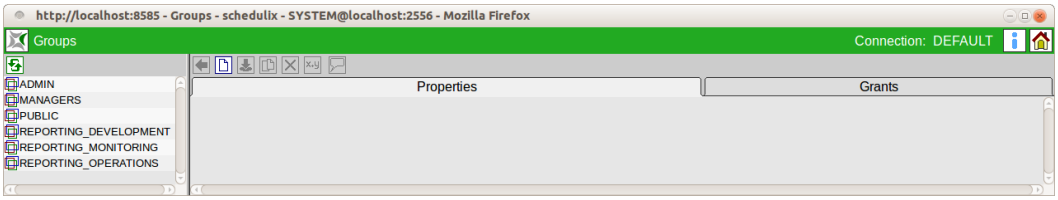


Abbildung 16.1: Gruppenverwaltung

16.2 Konzept

16.2.1 Beschreibung

Die Gruppen sind die Rechteträger. Die einzelnen Benutzer gehören Gruppen an und sie haben die Rechte die den Gruppen gehören. Ein User kann mehreren Gruppen angehören. Folgende Gruppen sind immer vorhanden:

- Public: Jeder Benutzer ist immer ein Mitglied der Gruppe *Public*.
- ADMIN: Jeder Benutzer der Gruppe *ADMIN* hat alle Rechte im System.

16.3 Navigator

Im Navigator stehen alle sichtbare Gruppen in einer Liste. Sichtbar in diesem Kontext bedeutet, dass man entweder Mitglied der Gruppe ist, oder aber das 'View' Privileg für die Gruppe hat.

16.4 Editor

16.4.1 Tab Properties

Dieser Tab dient der Pflege der Gruppeneigenschaften.

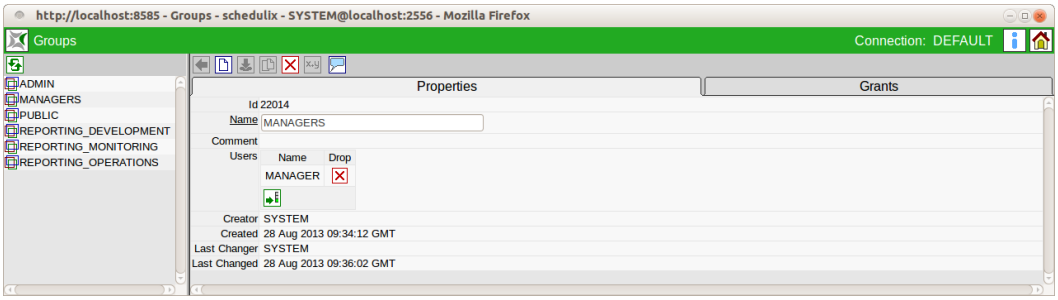


Abbildung 16.2: Gruppeneigenschaften

Folgende Felder sind zu sehen:

ID In diesem Feld wird die Group Id angezeigt.

Name Im Feld *Name* steht der Name der Gruppe.

Users In diesem Feld steht die Liste der User, die dieser Gruppe angehören.

Name Im Feld *Name* steht der Benutzer der Gruppe.

16.4.2 Grants

Der ”Grants” Tab zeigt alle Objekte an, für die der Gruppe Privilegien zugeordnet wurden. Der Objekttyp ist am jeweiligen Icon erkennbar. Die Liste wird immer nach Objektname sortiert.
Wird der Mauszeiger über das Privilegienfeld einer Zeile gebracht, so erscheint eine Pop-up-Info, welche den Privilegien-String erklärt.

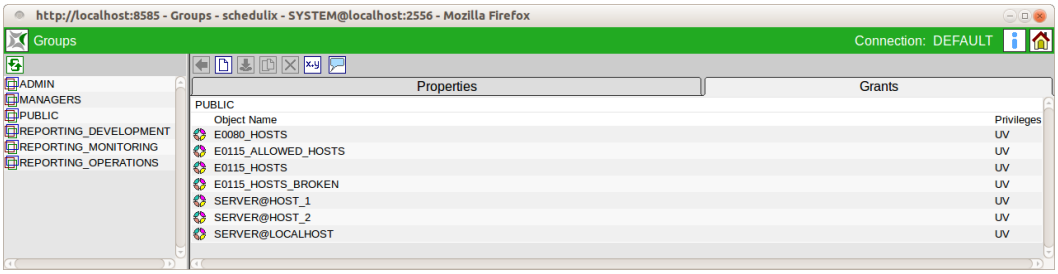


Abbildung 16.3: Objektprivilegien einer Gruppe

17 Time Scheduling

17.1 Bild

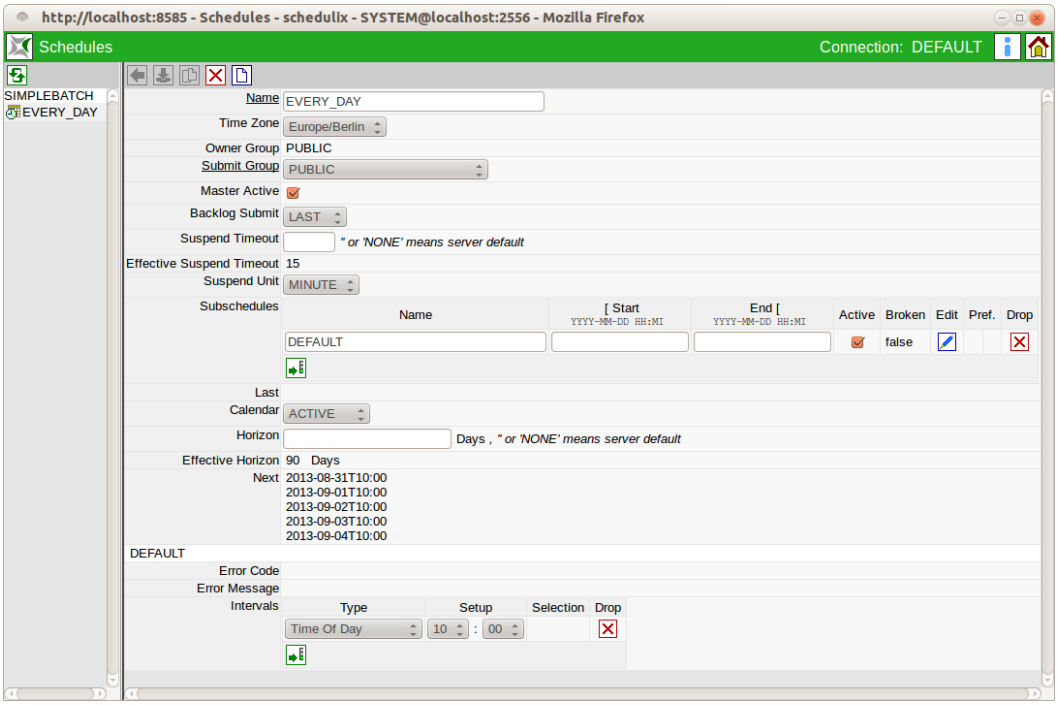


Abbildung 17.1: Time Scheduling

17.2 Konzept

17.2.1 Kurzbeschreibung

Das Time Scheduling erlaubt das Starten von Batches und Jobs zu bestimmten Zeitpunkten. Das Time Scheduling erlaubt eine einfache Definition komplexer Zeitpläne für Jobs und Batches.
Eine Anzahl von Zeitpunkten, die angeben, wann der Job gestartet werden soll, nennt man einen Zeitplan (Schedule).
Zeitpläne gelten immer nur für einen Master Job.

17.2.2 Ausführliche Beschreibung

Im Dialog "Time Scheduling" wird zwischen Main Schedules, welche in der Navigation als Elemente angezeigt werden und Sub Schedules, welche innerhalb eines Main Schedules verwaltet werden können, unterschieden.
Main Schedules sind untereinander unabhängig und es findet keine Interaktion statt.
Sub Schedules können vom Zeitintervall getrennt oder überschneidend definiert werden. Damit können verschiedene Ausfallzeiten simuliert oder unterschiedliche Zeitpläne eingestellt werden, die damit immer aktiv sind.
Falls auf Zuruf die Schedules geändert werden sollen, kann dies als Interimszeitplan eingegeben werden. Um den Interimsplan zu aktivieren, muss ihm die höchste Priorität zugeteilt werden. Der ursprüngliche Zeitplan bleibt dabei erhalten. Der ursprüngliche Zeitplan wird wieder aktiviert, indem man ihm wieder die höchste Priorität zuteilt.

17.3 Navigator

Die Navigation zeigt die für dieses Scheduling Entity vorhandenen Zeitpläne (Main Schedules) an. Ein Scheduling Entity kann mehrere Main Schedules besitzen. Diese Main Schedules werden unabhängig berücksichtigt. Das heißt, es werden die Startzeiten aller Main Schedules berücksichtigt.
Wenn mehrere Main Schedules identische Startzeiten vorsehen, wird das Scheduling Entity mehrfach gestartet.

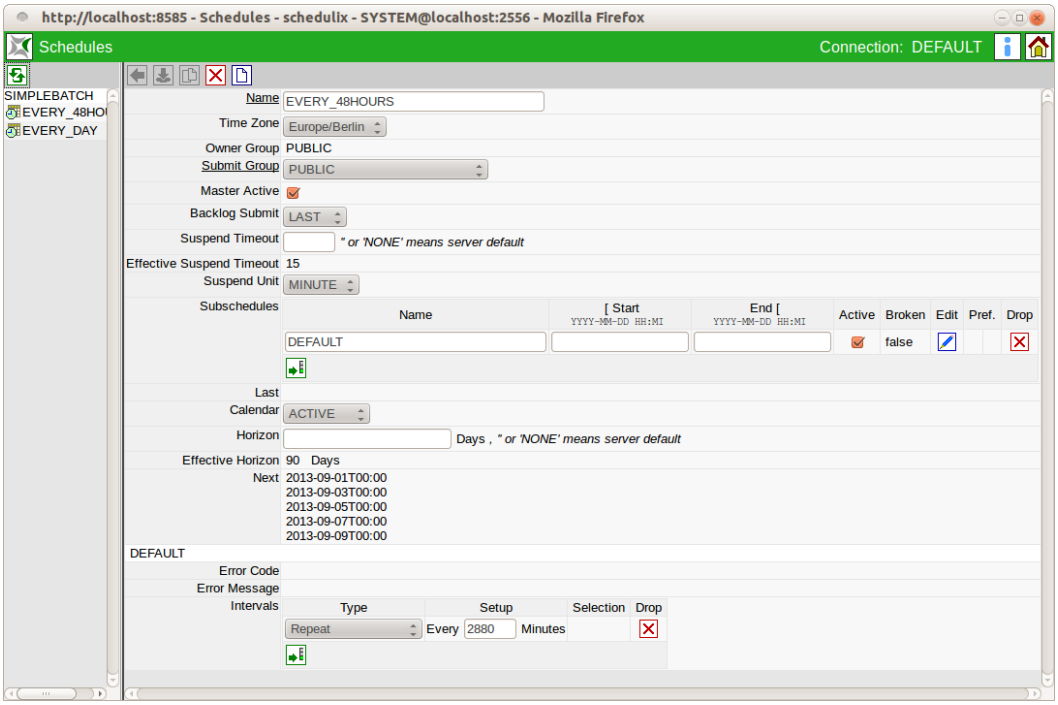


Abbildung 17.2: Time Scheduling Navigator

17.4 Editor

Der Editor zeigt für den in der Navigation gewählten Main Schedule, alle ”Detail”-Elemente an. Hier können Sub Schedules angelegt und eine Liste von Zeitpunkten gepflegt werden, an denen der gewählte Job oder Batch gestartet werden soll. Die

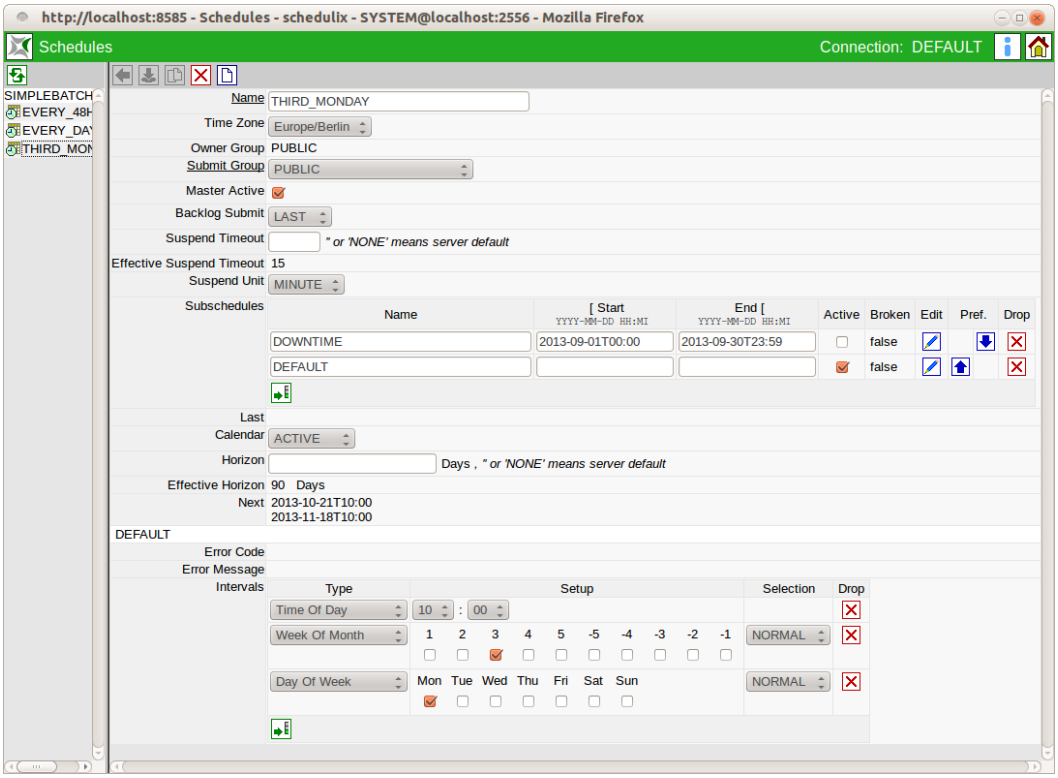


Abbildung 17.3: Time Scheduling Editor

”Editor” Maske sieht aus wie in Abbildung 17.3
Die obigen Felder haben folgende Bedeutung:

Name Das ist der Name des Main Schedules. Beim ersten Anlegen eines neuen Main Schedules wird dieser automatisch mit ”MASTER” vorbelegt. Er ist aber beliebig änderbar. Sollten mehrere Main Schedules angelegt werden, ist jeweils ein unterschiedlicher Name anzugeben.

Time Zone Die Zeitzone, für die der Schedule berechnet werden soll. Welche Zeitzonen hier zur Verfügung stehen, kann vom schedulix!Web Administrator konfiguriert werden.

Owner Group Im Feld *Owner Group* ist der Eigentümer des Schedules definiert.

Submit Group Die *Submit Group* bestimmt die Eigentümergruppe des Jobs, der submitted wird.

Master Active Ist das Flag im *Master Active* gesetzt, wird der Schedule vom Time Scheduling System berücksichtigt, wenn nicht, wird er ignoriert.

Backlog Submit Der Backlog bestimmt, wie sich das Time Scheduling System nach einem Server Downtime verhält:

- None: Die ”versäumten” Jobs werden nicht mehr ausgeführt.
- Last: Es wird nur der letzte der ”versäumten” Jobs ausgeführt.
- All: Es werden alle ”versäumten” Jobs ausgeführt.

Suspend Timeout, Suspend Unit Ein Job wird nach einem Server Downtime suspended submitted, wenn die geplante Submit-Zeit länger als die eingegebene Zeit in den Feldern *Suspend Timeout* und *Suspend Unit* her ist. Wenn keine Zeit eingetragen ist, gilt der serverweite Default.

Liste Sub Schedules In der Liste ”Sub Schedules” werden alle Sub Schedules des Main Schedules angezeigt. Die Schedules sind nur in dem eingegebenen Zeitraum gültig. Die Sub Schedules mit der höchsten Präferenz (zu oberst stehende) werden als Erstes berücksichtigt. Diese ’überdecken’ für ihren Zeitraum weiter unten stehende Sub Schedules. Dies bedeutet, dass zu einem Zeitpunkt immer nur genau ein Sub Schedule gültig sein kann.
Die Felder der Liste haben folgende Bedeutung:

Name Im Feld *Name* wird der Name des Sub Schedules eingetragen.

Start Im Feld *Start* wird das Datum und die Uhrzeit eingetragen, ab dem dieses Sub Schedule gültig ist. Wird das Feld leer gelassen, so gilt der Sub Schedule als ”schon immer gültig”.

End Im Feld *End* wird das Datum und die Uhrzeit eingetragen, ab dem dieses Sub Schedule ungültig wird. Wird das Feld leer gelassen, so gilt der Sub Schedule als ”ewig gültig”.

In der Liste der Sub Schedules stehen folgende Buttons zur Verfügung:

Active Nur, wenn das Flag gesetzt ist, werden die Jobs submitted. Durch Löschen des Flags kann das Submitten folglich verhindert werden. Downtimes werden typischer Weise als hoch präferenzielle Sub Schedules mit nicht gesetztem Active Flag implementiert.

Broken Mittels des *Broken* Feldes kann geprüft werden, ob beim Submit des Jobs ein Fehler aufgetreten ist. Ist dies der Fall, wird bei diesem Sub Schedule das Feld *Broken* mit TRUE angegeben.

Edit Nach Betätigung des *Edit* Buttons erscheint die zu dem Sub Schedule gehörende Intervallbeschreibung, in der man präzise festlegen kann, wann der Job gestartet wird.

Pref. Über die *Preference* Buttons lässt sich die Präferenz der einzelnen Sub Schedules festlegen. Dazu werden die Zeilen nach Betätigen der Buttons nach oben oder unten verschoben.

Drop Dieser Button dient zum Löschen des Objektes.

Last Im Feld *Last* wird der letzte Ausführungszeitpunkt des Jobs durch das Scheduling System angezeigt. Wurde der Job durch das Scheduling System noch nie gestartet, ist dieses Feld leer.
Das Datumsformat sieht folgendermaßen aus:
YYYY:MM:DDTHH:MM
Die Platzhalter entsprechen denen im Feld *Start*.

Calendar Dieses Feld bestimmt, ob die geplanten Ausführungszeiten des Jobs in den Kalender aufgenommen werden. Ist dies der Fall, haben auch die beiden nächsten Felder eine Bedeutung.

Horizon Das Feld *Horizon* bestimmt, wie weit im Voraus Kalendereinträge gemacht werden sollen. Die Zeitangabe erfolgt in Tagen.

Effective Horizon Falls kein Eintrag in das Feld *Horizon* gemacht wird, gilt der konfigurierte Default-Wert als Horizont. Das Feld *Effective Horizon* zeigt den aktuell gültigen Wert.

Next Im Feld *Next* wird der oder die nächsten geplanten Ausführungszeitpunkte des Tasks durch das Scheduling System angezeigt. Sind noch keine Zeitpläne und

-punkte angelegt, so ist dieses Feld leer. Sobald ein Zeitplan angelegt wird und Zeiten definiert wurden, wird der nächstmögliche Termin nach diesem Zeitplan angezeigt. Ist kein Zeitpunkt möglich, weil sich ausschließende Zeiten definiert wurden, so wird ebenfalls kein Wert angezeigt.
Wenn Kalendereinträge erstellt werden, wird hier nicht nur der nächste Ausführungszeitpunkt, sondern die nächsten fünf Ausführungszeitpunkte, soweit diese innerhalb des Horizonts liegen, gezeigt.
Das Datumsformat ist identisch mit dem des Feldes *Start*.

17.4.1 Teilbereich Sub Schedule Details

Dieser Teilbereich zeigt alle "Detail"-Informationen zum ersten Sub Schedule oder einen mittels des *Edit* Buttons gewählten anderen Sub Schedules an. Die Sub Schedule Details beginnen auf der Maske mit einer grauen Zeile, welche den Namen des aktuell ausgewählten Sub Schedules anzeigt, gefolgt von folgenden Feldern:

Error Code Im Feld *Error Code* wird, falls ein Fehler bei der Ausführung des Jobs im Time Scheduling aufgetreten ist, der übermittelte Fehlercode angezeigt. Ist kein Fehler aufgetreten, bleibt das Feld leer.

Error Message Im Feld *Error Message* wird, falls ein Fehler bei der Ausführung des Jobs im Time Scheduling aufgetreten ist, die übermittelte Fehlermeldung angezeigt. Ist kein Fehler aufgetreten, bleibt das Feld leer.

Liste Intervals Wird hier kein Intervall eingetragen (die Liste ist leer) so wird, falls der Sub Schedule als Active markiert ist, genau einmal zum Startzeitpunkt des Sub Schedules ein Job submittet.
Eine nicht leere Liste muss genau ein "driving" Intervall vom Typ "Time Of Day", "Repeat" oder "Calendar (Driver)" enthalten.
Das "driving" Intervall gibt den Takt an, zu dem potentiell ein Submit stattfindet.
Die anderen "filtering" Intervalle bestimmen durch ihre Filterwirkung, welche durch das "driving"-Intervall erzeugten Takte tatsächlich zu einem Submit führen.

Type Das Feld *Type* gibt die Art des Intervalls an. Abhängig vom Typ des Intervalls wird in der Spalte *Setup* eine andere Maske angezeigt.

Setup Hier wird abhängig vom Type des Intervalls eine Maske zur Konfiguration des Intervalls angezeigt.

Selection Bei Intervallen, welche eine Selektion ermöglichen (z.B.: ”Day Of Week”) kann in dieser Spalte ausgewählt werden, ob die Selektion ”NORMAL” oder ”INVERSE” erfolgen soll.

Es gibt folgende Intervall Typen:

- Repeat

  Hierbei handelt es sich um ein ”driving” Intervall, das den zeitgesteuerten Start des Tasks in regelmäßigen Abständen vorsieht. Die Anzahl der Minuten kann im Feld *Setup* eingeben werden.

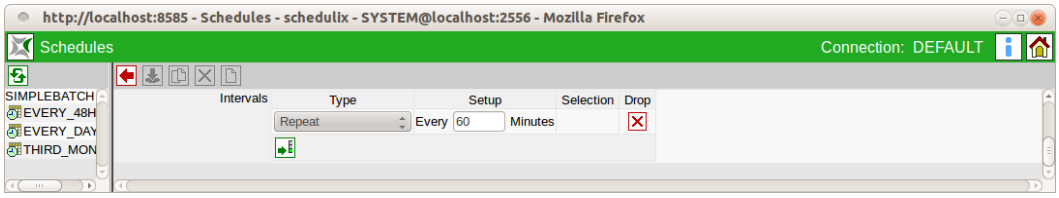


Abbildung 17.4: Repeat Driver

Im oberen Beispiel wird der Job alle 60 Minuten gestartet.

- Time of Day

  Mittels dieses ”driving”-Intervall-Typs kann der Job zu einer vorgegebenen Tageszeit starten. Die Tageszeit wird im Feld *Setup* mittels der Auswahl der Stunden (24H) und der Minuten eingetragen werden.

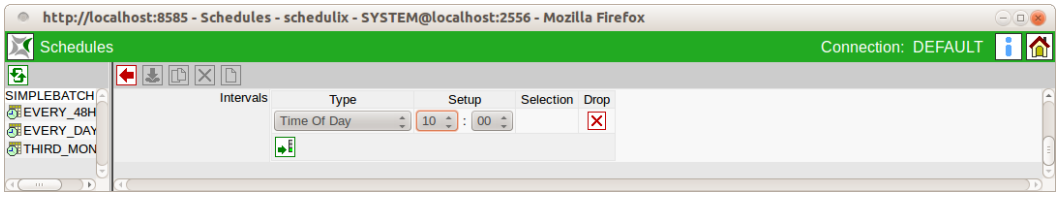


Abbildung 17.5: Time Of Day Driver

Im oberen Beispiel wird der Job jeden Tag um 10:00 Uhr morgens gestartet.

Es können mehrere ”Time Of Day”-Intervalle angelegt werden. Damit ist es dann leicht möglich, mehrere Ausführungszeitpunkte an einem Tag zu definieren.

- Range of Day

  Dieser ”filtering”-Intervall-Typ, ist nur in Kombination mit den ”driving” Intervall Typen ”Repeat” und ”Calendar (Driver)” erlaubt. Er erlaubt die Beschränkung auf einen oder mehrere Zeitbereiche innerhalb eines Tages.

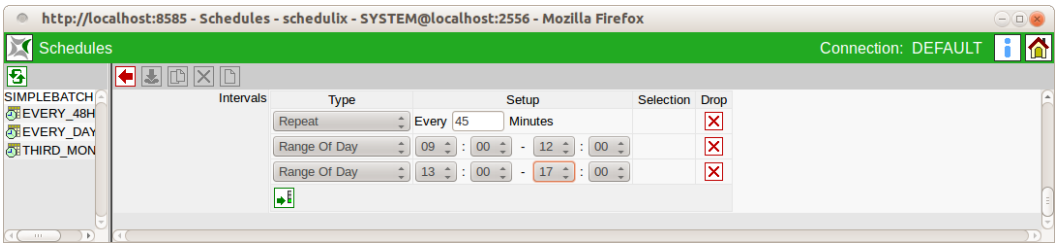


Abbildung 17.6: Range of Day Filter

Es können mehrere ”Range Of Day”-Intervalle angelegt werden. Damit ist es dann leicht möglich, mehrere Ausführungszeiträume an einem Tag zu definieren.

- Day of Week

  Mittels des ”filtering”-Intervall-Typs ”Day of Week” kann ausgewählt werden, an welchen Wochentagen der Job ausgeführt werden soll. Dies wird im Feld *Setup* durch Auswahl der einzelnen Wochentage festgelegt.

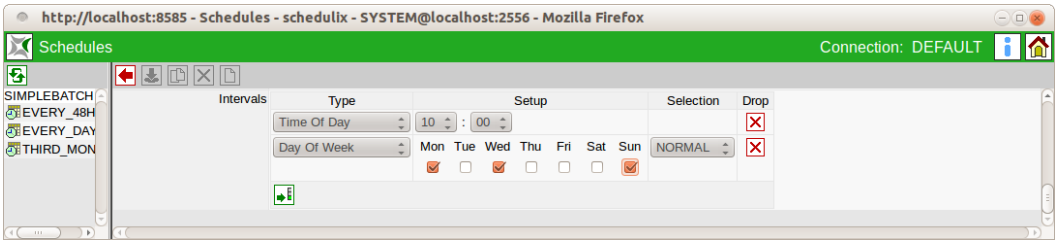


Abbildung 17.7: Day of Week Filter

Im oberen Beispiel wird der Job jeden Montag, Mittwoch und Sonntag ausgeführt.

- Day of Month

  Mittels des ”filtering”-Intervall-Typs ”Day of Month” kann ausgewählt werden, an welchen Tagen des Monats der Job ausgeführt werden soll. Diese werden im Feld *Setup* durch Auswahl der einzelnen Tage im Monat festgelegt.

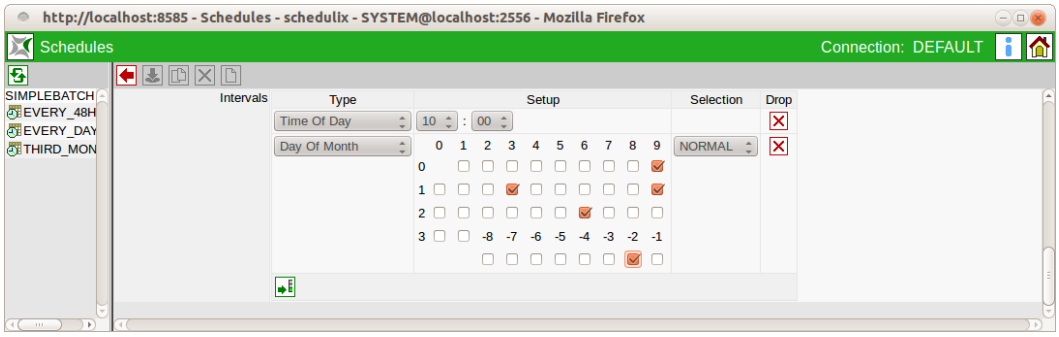


Abbildung 17.8: Day of Month Filter

Im oberen Beispiel wird der Job am 9., 13., 19., 26. und am vorletzten Tag jeden Monats ausgeführt.

- Week of Month

  Mittels des ”filtering”-Intervall-Typs ”Week of Month” kann ausgewählt werden, in welcher Woche des Monats der Job ausgeführt werden soll. Diese werden im Feld *Setup* durch Auswahl der einzelnen Wochen im Monat festgelegt.

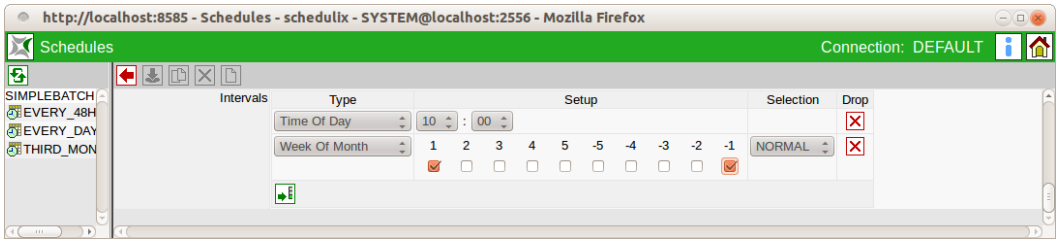


Abbildung 17.9: Week of Month Filter

Im oberen Beispiel wird der Job in der ersten und letzten Woche eines jeden Monats ausgeführt. Dabei gilt, dass die erste Woche als die ersten 7 Tage des Monats, und die letzte Woche als die letzten 7 Tage des Monats definiert ist.

- ISO Week of Month

  Mittels des ”filtering”-Intervall-Typs ”ISO Week of Month” kann ausgewählt werden, in welcher ISO Kalenderwoche des Monats der Job ausgeführt werden soll. Diese werden im Feld *Setup* durch Auswahl der einzelnen Wochen im Monat festgelegt.

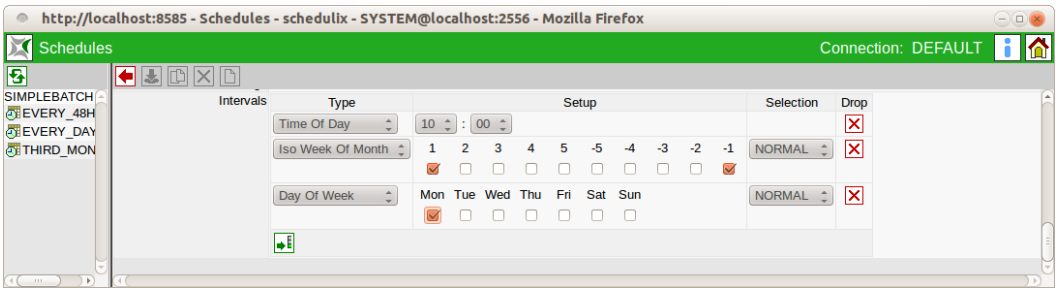


Abbildung 17.10: ISO Week of Month Filter

Im oberen Beispiel wird der Job am Montag der ersten ISO Woche eines jeden Monats ausgeführt. ISO Wochen beginnen immer Montags. Wochen werden einem Monat zugeordnet, wenn mindestens 4 Tage der Woche in diesem Monat liegen. Ist zum Beispiel der erste Wochentag eines Monats ein Mittwoch, so beginnt die erste ISO Woche des Monats schon im Vormonat.

- ISO Week of Year

  Mittels des ”filtering”-Intervall-Typs ”ISO Week of Year” kann ausgewählt werden, in welcher ISO Kalenderwoche des Jahres der Job ausgeführt werden soll. Diese werden im Feld *Setup* durch Auswahl der einzelnen Wochen im

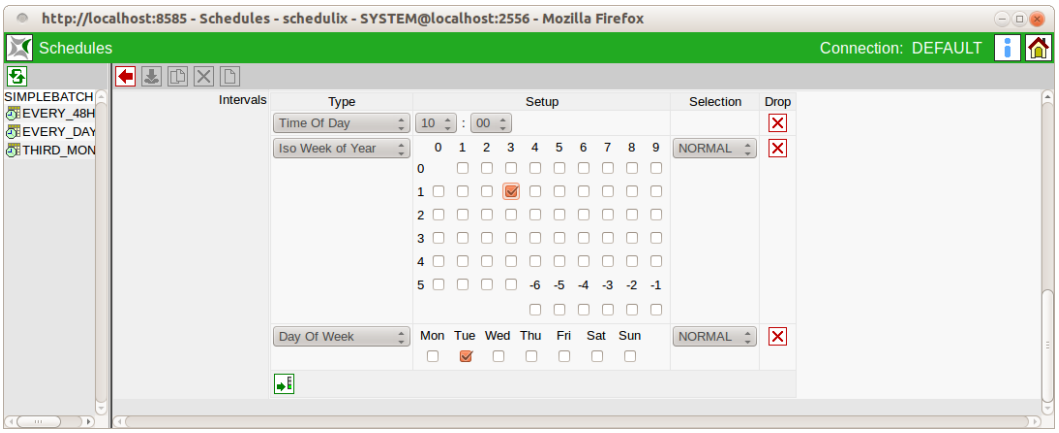


Abbildung 17.11: Week of Year Filter

Jahr festgelegt. Die Option sieht im Bild 17.11 aus.

Im oberen Beispiel wird der Job am Dienstag der 31. ISO Kalenderwoche ausgeführt. ISO Wochen beginnen immer Montags. Wochen werden einem Jahr zugeordnet, wenn mindestens 4 Tage der Woche in diesem Jahr liegen. Ist zum Beispiel der erste Wochentag eines Jahres ein Mittwoch, so beginnt die erste ISO Woche des Jahres schon im Vorjahr.

- Month of Year

  Mittels des "filtering"-Intervall-Typs "Month of Year" kann ausgewählt werden, in welchem Monat des Jahres der Job ausgeführt werden soll. Diese werden im Feld *Setup* durch Auswahl der einzelnen Monate im Jahr festgelegt.

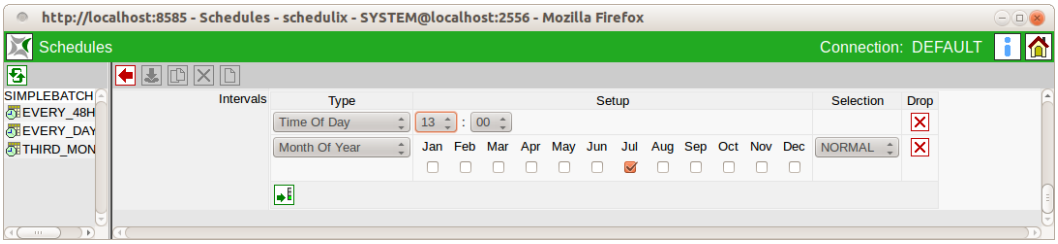


Abbildung 17.12: Month of Year Filter

Im oberen Beispiel wird der Job jeden Tag im Juli um 13:00 Uhr ausgeführt.

- Calendar (Driver)

  Mittels des "driving"-Intervall-Typs "Calendar (Driver)" können die Ausführungszeitpunkte aus einem vordefinierten Kalender entnommen werden. Diese Option steht nur zur Verfügung, wenn der Administrator solche Kalender angelegt und in schedulix!Web bekannt gemacht hat.

  Im Feld *Setup* kann dann der Kalender ausgewählt und durch die Option "with select on" bestimmt werden, auf welche Einheit (DAY, WEEK, MONTH, YEAR) sich die folgende Auswahl bezieht.

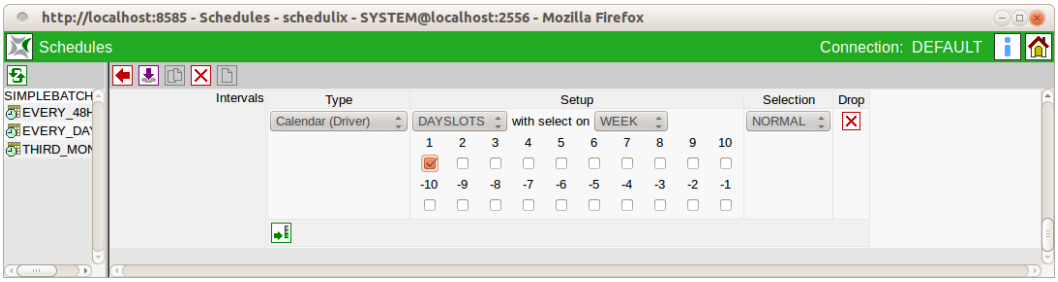


Abbildung 17.13: Calendar Driver

Im oberen Beispiel wird aus einem Kalender mit Namen ”DAYSLOTS” der jeweils erste Eintrag jeder Kalenderwoche ausgewählt.

- Calendar (Filter)

  Mittels des ”filtering”-Intervall-Typs ”Calendar (Filter)” können die Ausführungszeitpunkte mittels eines vordefinierten Kalender eingeschränkt werden. Diese Option steht nur zur Verfügung, wenn der Administrator solche Kalender angelegt und in schedulix!Web bekannt gemacht hat.

  Im Feld *Setup* kann dann der Kalender ausgewählt und durch die Option ”with select on” bestimmt werden, auf welche Einheit (DAY, WEEK, MONTH, YEAR) sich die folgende Auswahl bezieht.

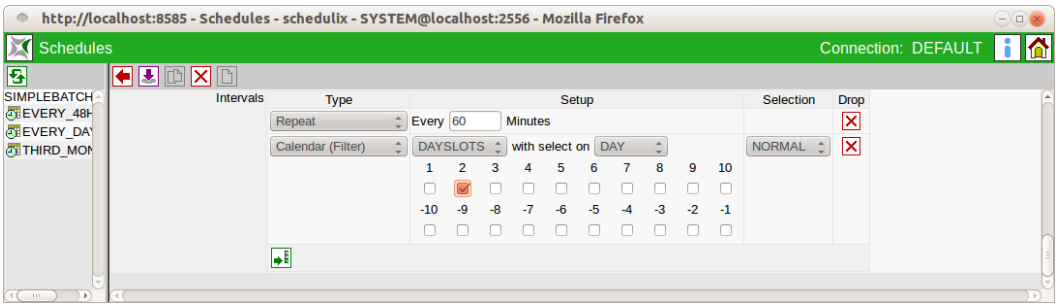


Abbildung 17.14: Calendar Filter

Im oberen Beispiel wird alle 60 Minuten ein Submit ausgeführt, falls sich dieser im Kalender ”DAYSLOTS” im zweiten Block eines jeden Tages befindet.

Setup Das Feld *Setup* gibt die genauere Definition des Intervalls an. Es ist je nach Intervalltyp unterschiedlich.

18 Submit Batches und Jobs

18.1 Bild

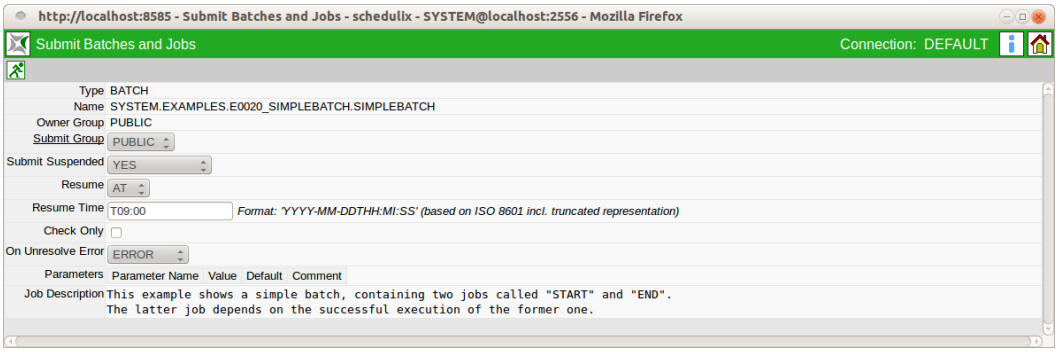


Abbildung 18.1: Submit Batches und Jobs

18.2 Konzept

18.2.1 Kurzbeschreibung

Der Dialog "Submit Batches and Jobs" dient zum manuellen Starten von, im Dialog Batches and Jobs, definierten Master Jobs. Durch den Submit wird dem schedulix Server mitgeteilt, das dieser Master Job aktiviert werden soll.

18.2.2 Ausführliche Beschreibung

Die manuelle Freigabe von Master Jobs ist eine von zwei Methoden, um einen Job zu starten. Die zweite Möglichkeit besteht durch den Eintrag des Jobs im Time Scheduling.
Beim manuellen Start können darüber hinaus noch bestimmte Parameter gesetzt, und bestimmt werden, ob es sich um einen echten Start oder nur einen Check auf Startfähigkeit handelt.
Weiter kann bestimmt werden, ob der Master Job gleich suspended werden soll.

18.3 Navigator

Die Navigation zeigt alle verfügbaren Master Jobs in einer Ordnerhierarchie an. Sie entspricht dabei dem Navigationsbildschirm aus dem Batches und Jobs Dialog. Als Unterschied werden hier allerdings nur Objekte angezeigt, welche das Flag "Submit as Master allowed" gesetzt haben.

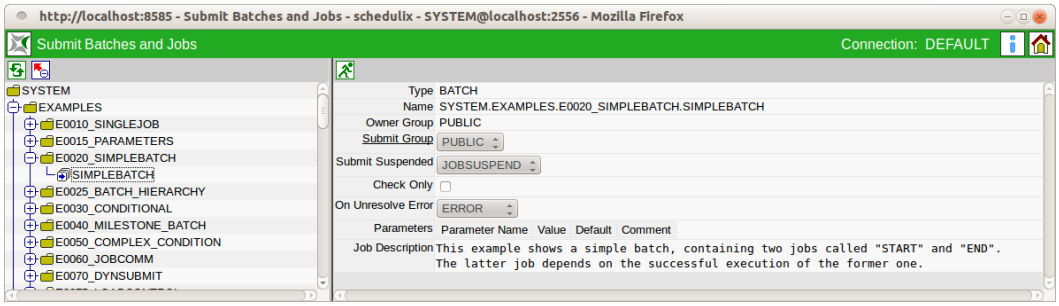


Abbildung 18.2: Submit Navigation

18.4 Editor

Mit dem Editor "Frame" können Startoptionen und Parameter übergeben und ein Master Job submitted werden.

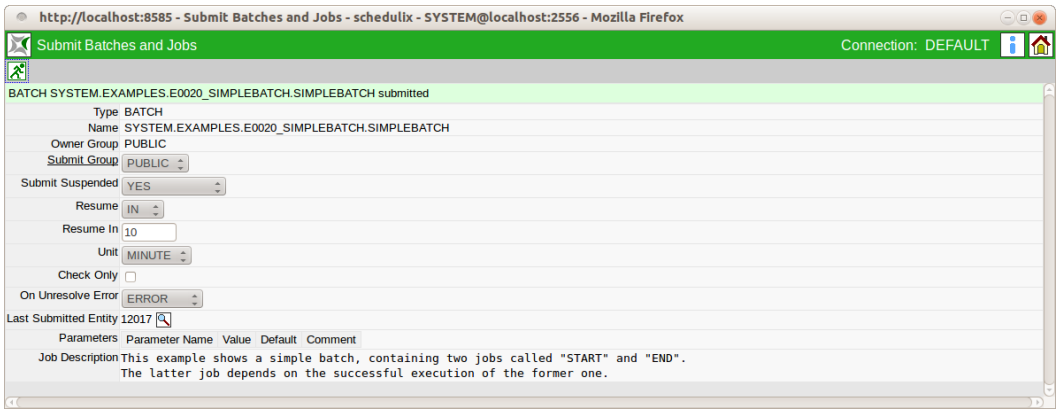


Abbildung 18.3: Submit Editor

Folgender Button ist auf dieser Maske aktiv:

 Submit

Mit dem *Submit* Button wird der Job gestartet (Flag "Check Only" nicht gesetzt), oder es wird geprüft, ob der Master Job startfähig ist (Flag "Check Only" gesetzt). Die obigen Felder haben folgende Bedeutung:

Type Der Typ des Master Jobs. Dieser kann entweder ein Job oder Batch sein.

Name Das ist der vollständige Name (mit Pfad) des Master Jobs.

Owner Group Im Feld *Owner Group* ist der Eigentümer des Jobs definiert.

Submit Group Die *Submit Group* bestimmt die Eigentümergruppe des Jobs, der submitted wird.

Submit Suspended Dieses Feld gibt an, ob der Job gleich nach dem Start in den Suspend Status gehen soll oder nicht.
Die folgenden Eingabefelder *Resume*, *Resume Time*, *Resume In*, *Unit* werden nur angezeigt, falls dieses Feld auf 'YES' gesetzt wird.
Es gibt folgende Optionen:

1. YES

   Das Child wird beim Submit als Suspended angelegt und muss durch eine explizite Freigabe gestartet werden.

2. NO

   Das Child wird beim Submit nicht Suspended und kann sofort starten.

3. JOBSUSPEND

   Ob eine Verzögerung stattfindet oder nicht, hängt vom Feld *Suspend* des zu submittenden Jobs ab. Das heißt, je nachdem, wie die Einstellung im Job definiert wurde, wird diese übernommen.

Resume Hier kann ausgewählt werden, ob ein automatischer Resume stattfinden soll.
Es gibt folgende Möglichkeiten:

- NO: Wählt diese Funktionalität ab und es werden keine anderen Eingabefelder angezeigt.
- AT: Wählt einen automatischen Resume zu einem festen Zeitpunkt. Das Eingabefeld *Resume Time* wird angezeigt.

- IN: Wählt einen automatischen Resume nach Ablauf einer Zeit. Die Eingabefelder *Resume In* und *Unit* werden angezeigt.

Resume Time Hier wird der gewünschte Resume Zeitpunkt im Format "YYYY-MM-DDTHH:MI:SS" eingegeben.
Das Format orientiert sich an der ISO Norm 8601 und erlaubt auch unvollständige Angaben. Die Eingabe von 'T09:00' wird den Job um 09:00 Uhr resumen (ausgehend von der aktuellen Zeit).

Resume In Hier wird angegeben, wie viele Zeiteinheiten (siehe *Unit*) bis zum Resume gewartet werden soll.

Unit Hier wird eingegeben, ob es sich bei der Eingabe im *Resume In* um Minuten (MINUTE), Stunden (HOUR), oder Tage (DAY) handeln soll.

On Unresolve Error Im Feld *On Unresolve Error* wird beim Submitten bestimmt, wie man im Falle von nicht aufgelösten Abhängigkeiten verfährt.
Es gibt folgende Möglichkeiten:

- Error: Die Abhängigkeiten die nicht aufgelöst werden können, werden als Fehler betrachtet. Die fehlenden Abhängigkeiten führen zum Submit-Abbruch.
- Suspend: Beim Suspend erfolgt der Submit, als wären diese Abhängigkeiten nicht definiert. Allerdings wird der Job suspended submitted.
- Ignore: Beim Ignore wird submittet, als wären diese Abhängigkeiten nicht definiert.

Check Only Dieses Flag gibt an, ob der Job submitted wird, oder ob nur eine Prüfung der Startfähigkeit stattfindet.

Liste Parameters Benötigt der Job Parameter, müssen die Parameterwerte vor dem Submit in dieser Liste eingetragen werden. Mehr zum Thema Parameter finden Sie im Kapitel 13.5.10.

Job Description Die Beschreibung des Jobs aus der Job Definition.

19 Bookmarks

19.1 Bild

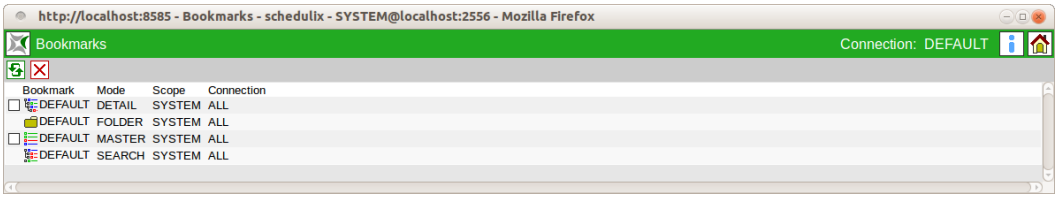


Abbildung 19.1: Bookmarks

19.2 Konzept

19.2.1 Kurzbeschreibung

Im Dialog "Bookmark" ist es möglich, gespeicherte Abfragen aus den Dialogen "Running Master Jobs" und "Search Running Jobs" abzurufen. Hiermit sind immer wiederkehrende Suchabfragen und Übersichtsfenster leicht zu speichern und wieder neu aufzurufen.

19.2.2 Ausführliche Beschreibung

Bookmarks können für den aktuellen Benutzer oder systemweit zur Verfügung gestellt und geändert werden. Darüber hinaus ist es möglich, bestimmte Bookmarks immer sofort nach dem Anmelden ins System zur Verfügung zu haben. Diese werden dementsprechend automatisch gestartet.
Die Liste der Bookmarks enthält folgende Felder:

Bookmark Hier wird der Name des Bookmarks angezeigt. Durch Anwahl des Namens erscheint ein neues Fenster mit der als Bookmark gespeicherten Anfrage.

Mode Im Feld *Mode* steht die Art eines Bookmarks. Es gibt folgende Arten:

1. FOLDER

   Hierbei handelt es sich um einen Folder Bookmark.

2. MASTER

   Hierbei handelt es sich um einen Bookmark, welcher von der Suchmaske die im Kapitel 20 beschrieben wird, gespeichert wurde. Mittels dieses Bookmarks sind nur Master Jobs sichtbar.

3. SEARCH

   Der Mode Search beschreibt einen Bookmark, welcher von der Suchmaske die im Kapitel 21 beschrieben wird, gespeichert wurde. Hiermit sind alle Jobs auffindbar.

4. DETAIL

   Hiermit wird ein Bookmark beschrieben, der die "Detail"-Ansicht (mehr hierzu im Kapitel 20) beschreibt. Dieses CHILD hängt immer mit einem Bookmark des Modes MASTER zusammen (ist das Child der "Master" Ansicht).

   Welcher Bookmark gewählt wird, wird wie folgt ermittelt:

   Definiert der Master Job oder Batch einen Parameter DETAIL_BOOKMARK so wird dieser verwendet. Ist dies nicht der Fall, wird das Feld 'Detail Bookmark' des Master Bookmarks ausgewertet. Ist dies ebenso nicht gesetzt, so wird der Detail Bookmark 'DEFAULT' verwendet.

Scope Mit dem Scope Parameter wird angegeben, ob der Bookmark systemweit oder nur für den aktuellen Benutzer sichtbar ist. Es gibt folgende Optionen:

1. SYSTEM

   Ein Bookmark mit dem Scope SYSTEM ist für alle Benutzer im gesamten System sichtbar. Er wird in der Bookmark-Liste von allen Benutzern angezeigt und darf von allen verwendet werden. Werden Änderungen an einem Bookmark vom Scope SYSTEM durchgeführt, gelten diese systemweit für alle Benutzer.

   Bookmarks vom Typ SYSTEM dürfen allerdings nur von Benutzern mit Web-GUI-Administrationsrechten (siehe Web-Users) angelegt werden. Ein normaler Benutzer darf zwar SYSTEM Bookmarks ändern, der Bookmark wird aber anschließend als USER Bookmark gespeichert. Das heißt, die lokale Änderung überschreibt den Bookmark nur für den aktuellen Benutzer, nicht für andere. Löscht man diesen Bookmark, erscheint wieder der Original SYSTEM Bookmark.

2. USER

   Hat der Scope den Wert USER, dann ist dieser Bookmark von dem aktuellen User angelegt worden und nur für diesen verfügbar.

Connection Das Feld *Connection* informiert über die Gültigkeit des Bookmarks bei Verwendung mehrerer Serververbindungen. Folgende Optionen existieren:

1. CURRENT

   Der Bookmark gilt nur für die aktuelle Connection. Wird das Bookmark-Fenster für eine andere Connection geöffnet, ist dieser Bookmark nicht sichtbar.

2. ALL

   Der Bookmark ist für alle Server Connections gültig.

19.3 Navigation

Der Bookmark-Dialog besteht nur aus einem Navigationsfenster. Wird aus dem Navigationsfenster ein Eintrag gewählt, wird dieser in einem neuen Fenster angezeigt.

20 Running Master Jobs

20.1 Bild

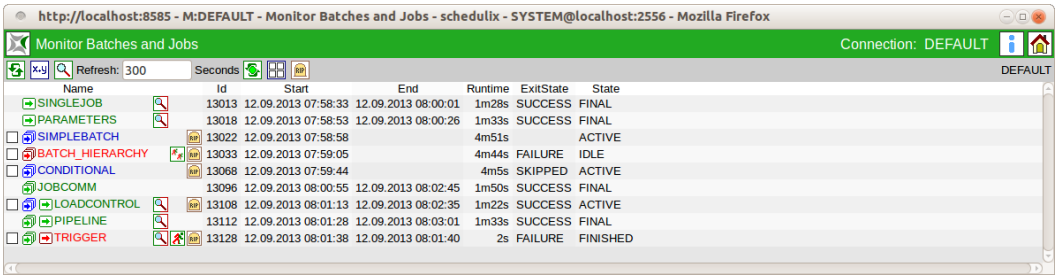


Abbildung 20.1: Running Master Jobs

20.2 Konzept

20.2.1 Kurzbeschreibung

Der Dialog *Running Master Jobs* dient zur Anzeige der aktuell laufenden oder historisch gelaufenen (je nach Einstellung) Master Jobs des Systems.

20.2.2 Ausführliche Beschreibung

Der Dialog *Running Master Jobs* dient zur Anzeige der aktuell laufenden Jobs (Monitoring), sowie die Betreuung der aktiven Jobs (Operating). Die Anzeige und Filterkriterien des Running Master Job-Dialoges können über den Bookmark (Default, Master, System) angepasst werden. Dieser Bookmark beschreibt die Anzeige des kompletten Dialoges.

20.3 Master Navigator

Die Navigation nimmt den ganzen Raum des Dialoges ein. Hier erscheint eine Liste aller aktuell laufender 'Master submittable'- Jobs und Batches.

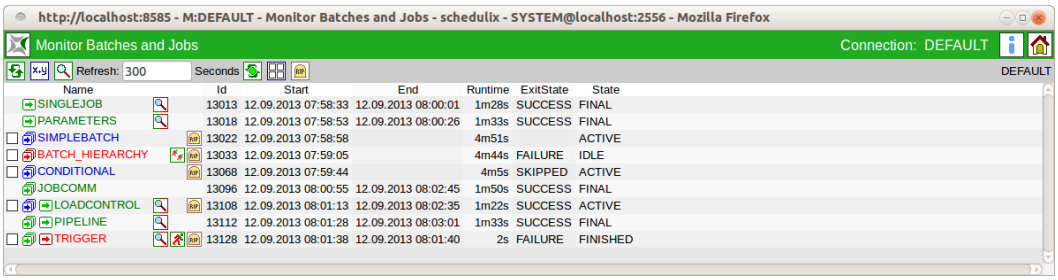


Abbildung 20.2: Running Master Jobs Navigator

Find

Betätigt man den *Find* Button, so kommt man zur Seite für die Eingabe der Suchkriterien sowie Einstellung des Bildschirms.

Auto-Refresh On/Off

Der *Auto-Refresh On* Button ist ebenfalls ein Schalter und kann in zwei Stellungen betrieben werden. In der Stellung "Auto-Refresh On" wird in der eingestellten Zeit eine Aktualisierung automatisch durchgeführt.
Im Eingabefeld links vom Button kann die Zeit, die zwischen zwei Refreshes liegt, in Sekunden eingeben werden. Der Default-Wert ist 300 Sekunden. Anhand der Farbe des Buttons kann der Zustand des Auto-Refresh abgelesen werden. Grün = On, Rot = Off.
Die im Dialog erscheinende Liste gibt alle Master Jobs an, die im aktuellen System aktiv sind (default) oder, falls über die Suchmaske Filteroptionen eingegeben wurden, die dementsprechend gefilterten Ergebnisse. Folgende Spalten sind in der Liste zu sehen:

Name Hier erscheint, je nach Stellung des Buttons *Show Jobs*, der vollständige Pfad und Name des Master Jobs oder nur der Name des Master Jobs.
Der Name ist anklickbar und führt zur Detailmaske, falls es sich um einen Master Submittable Batch mit Children handelt, ansonsten führt das Klicken zur Detailmaske für Jobs.
Vor dem Namen können Icons angezeigt werden, welche bestimmte States und Situation beschreiben. Es gibt folgende Icons:

Beim angezeigten Objekt handelt es sich um einen Job der aktuell läuft oder laufen kann. Die Icon-Farbe ist blau.

Beim angezeigten Objekt handelt es sich um einen Job, der wartet. Die Icon-Farbe ist lila.

- Job Icon ohne Punkt: Der Job befindet sich im Dependency Wait.
- Job Icon mit einem Punkt: Der Job befindet sich im Synchronize Wait.
- Job Icon mit zwei Punkten: Der Job befindet sich im Resource Wait.

Beim angezeigten Objekt handelt es sich um einen Job, der vom Time Scheduling in der Zukunft submitted werden wird.

Beim angezeigten Objekt handelt es sich um einen Job, der vom Time Scheduling in der Zukunft Suspended submitted werden wird.

Beim angezeigten Objekt handelt es sich um einen Job, der gecancelt wurde. Der Zustand dieses Jobs kann sich nicht mehr ändern. Die Icon-Farbe ist braun.

Beim angezeigten Objekt handelt es sich um einen Job, der Final ist. Der Zustand dieses Jobs kann sich nicht mehr ändern. Die Icon-Farbe ist grün.

Beim angezeigten Objekt handelt es sich um einen Job, der Finished und Restartable, Unreachable oder Error ist. Die Icon-Farbe ist rot.

Beim angezeigten Objekt handelt es sich um einen Batch, der im Moment läuft. Die Icon-Farbe ist blau.

Beim angezeigten Objekt handelt es sich um einen Batch, der im Moment wartet. Die Icon-Farbe ist lila.

Beim angezeigten Objekt handelt es sich um einen Batch, der vom Time Scheduling in der Zukunft submitted werden wird.

Beim angezeigten Objekt handelt es sich um einen Batch, der vom Time Scheduling in der Zukunft Suspended submitted werden wird.

Beim angezeigten Objekt handelt es sich um einen Batch, der Final ist. Der Zustand des Batches kann sich nicht mehr ändern. Die Icon-Farbe ist grün.

Beim angezeigten Objekt handelt es sich um einen Batch, der gecancelt wurde. Der Zustand dieses Batches kann sich nicht mehr ändern. Die Icon-Farbe ist braun.

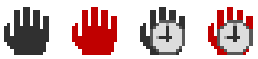
Beim angezeigten Objekt handelt es sich um einen Batch, der Unreachable ist, oder von dem mindestens einer der Children Finished und Restartable, Unreachable oder Error ist. Die Icon-Farbe ist rot.

Beim angezeigten Objekt handelt es sich um einen Milestone, der wartet. Die Icon-Farbe ist lila.

Beim angezeigten Objekt handelt es sich um einen Milestone, der gecancelt wurde. Die Icon-Farbe ist braun.

Beim angezeigten Objekt handelt es sich um einen Milestone, der Final ist. Die Icon-Farbe ist grün.

Beim angezeigten Objekt handelt es sich um einen Milestone, der Unreachable ist, oder von dem mindestens einer der Children Finished und Restartable, Unreachable oder Error ist. Die Icon-Farbe ist rot.



Diese Icons teilen dem Benutzer mit, dass dieses Objekt Suspended wurde. Ist das Icon rot, so wurde der Batch bzw. Job Restricted Suspended. In diesem Fall kann ein Resume nur von einem Benutzer, welcher Mitglied in der Gruppe ADMIN ist, wieder Resumed werden. Das Icon mit der Uhr zeigt an, dass für das Objekt ein automatischer Resume eingestellt wurde. Die Farben des Namens sind in ihrer Bedeutung analog zu den farblich identischen Buttons.

Der *Cancel Run* Button erscheint, falls der Job auf Resources oder Abhängigkeiten wartet oder noch auf einem Jobserver zugeteilt wurde. Durch Drücken des *Cancel Run* Buttons kann der geplante Job-Lauf abgebrochen werden. Der Job bekommt den State "Cancelled". Es ist nun kein Rerun und kein Resume möglich. Der Job bekommt auch keinen Exit State. Jobs, welche auf einen gecancelten Job warten, müssen diesen Job ignorieren oder ebenfalls gecancelt werden.

Mittels dieses Buttons kann ein neues Fenster aufgemacht werden, welches den Inhalt des Logfiles darstellt. Sollte dies nicht möglich sein, wenden Sie sich bitte an Ihren Systemadministrator.

 Rerun

Der *Rerun* Button erscheint, falls der aktuelle Job beendet wurde und sich in einem Restartable-Zustand befindet. Das heißt, er besitzt einen Exit State, welcher nicht Final ist. Durch Drücken des *Rerun* Buttons wird das Rerun-Programm aufgerufen (falls definiert, ansonsten das Run-Kommando). Hiermit ist ein erneutes Starten eines fehlerhaften Jobs nach der Problembehebung möglich.

 Rerun Children

Der *Rerun Children* Button wird angezeigt, wenn sich Children des aktuellen Jobs in einem Restartable-Zustand befinden. Das heißt, sie besitzen einen Restartable Exit State. Durch Drücken des *Rerun Children* Buttons wird das Rerun für alle Children gestartet, welche Restartable sind. Hiermit ist ein erneutes Starten fehlerhafter Children nach einer Problembehebung möglich.

Id Die *Id* ist der eindeutige Identifier der aktuellen "Master Job"-Laufzeitinstanz im System.

Start Beim Feld *Start* handelt es sich um den Zeitpunkt des Submits oder des Starts.

End Beim Feld *End* handelt es sich um den Zeitpunkt der Beendigung des Jobs. Wurde der Job noch nicht beendet, bleibt der Eintrag leer.

Runtime Die *Runtime* gibt die aktuelle Laufzeit des Jobs in Tagen (d), in Stunden (h), in Minuten (m) und Sekunden (s) an. Läuft der Job noch, wird die Laufzeit bis zum letzten Refresh des Dialoges angezeigt und bei jedem neuen Refresh aktualisiert.

Exit State Wurde der Job beendet, wird der *Exit State* den der Job bei der Beendigung bekommen hatte, angezeigt. Ist der Job noch nicht beendet, bleibt das Feld leer.

State Der *Job State* ist der aktuelle Laufzeitstatus des Jobs. Das Laufzeitsystem des schedulix Serversystems vergibt und ändert diese States, wenn der Job (das heißt, die Laufzeitinstanz des Jobs) seinen Lebensweg durch das schedulix System durchläuft. Der Job kann folgende States haben:

1. Submitted

   Der Job wurde manuell, durch einen Parent-Prozess oder über das Time Scheduling submitted und soll ausgeführt werden. Dies ist der initiale Status eines Jobs und ist normalerweise nicht sichtbar.

2. Dependency Wait

   Der Job wartet auf notwendige Abhängigkeiten, die erfüllt werden müssen.

3. Synchronize Wait

   Der Job wartet auf benötigte "Synchronizing Resources".

4. Resource Wait

   Der Job wartet auf die Bereitstellung ausreichender System Resources.

5. Unreachable

   Der Job kann nicht ausgeführt werden, da eine oder mehrere Abhängigkeiten (Dependencies) nicht erfüllt sind. Diese Situation kann durch das Ignorieren von Abhängigkeiten behoben werden.

6. Cancelled

   Der Job wurde manuell gecancelt und wird nicht mehr ausgeführt.

7. Error

   Durch einen Definitionsfehler kann der Job nicht ausgeführt werden. Ein Beispiel hierfür ist etwa eine benötigte Resource, die von keinem Jobserver zur Verfügung gestellt wird. In so einem Fall kann der Job restarted werden, nachdem die Fehlersituation behoben wurde.

   Es kommt auch häufig vor, dass bei der Eingabe des Run Programs ein Fehler gemacht wird. Wenn daraufhin der Jobserver nicht in der Lage ist den Prozess zu starten, wird der Job in den Zustand Error versetzt. Auch in dem Fall kann der betreffende Job restarted werden.

   Falls der Submit eines Masters über das Time Scheduling fehlschlägt, wird der Master zur Kennzeichnung dieses Fehlers im Status Error in das System

eingestellt. Damit ist der Fehler für den Verantwortlichen sichtbar. Der Job oder Batch ist jedoch nicht restartable und muss nach der Reparatur manuell submitted werden.

8. Runnable

   Der Job kann von einem Jobserver gestartet werden. Verbleibt ein Job längere Zeit in diesem Zustand, sollte der Jobserver überprüft werden.

9. Starting

   Der Job wurde einem Jobserver zum Starten übergeben.

10. Started

    Der Job wurde einem Jobserver zum Starten übergeben. Der Jobserver hat den Auftrag quittiert.

11. Running

    Das Run Command wurde vom Jobserver gestartet.

12. To Kill

    Der Benutzer hat den schedulix Server beauftragt das zu dem Job gehörige Kill Program aufzuführen. Das Kill Program kann mittels des *Cancel-Run* Buttons ausgeführt werden.

13. Killed

    Das zum Job gehörige Kill Program wurde ausgeführt.

14. Broken Active

    Der Jobserver hat seine Verbindung zum Run Program verloren. Der Prozess läuft zwar noch, aber das Ergebnis des Jobs (Exit-Code) kann nicht mehr an den Jobserver und dementsprechend an den schedulix Server zurückgegeben werden.

15. Broken Finished

    Hat ein Job einen State "Broken Active" und beendet sich, geht er in den State "Broken Finished" über. Das bedeutet, der Prozess läuft nicht mehr. Der Exit Code des Prozesses konnte nicht mehr vom Jobserver ermittelt werden. Eine manuelle Überprüfung der Job-Ergebnisse und ein manuelles Setzen des Exit State ist nötig. Das Setzen des Exit State erfolgt mittels des Buttons *Set State*.

16. Finished

    Der Job wurde beendet. Der Exit State ist im Feld *Exit State* zu sehen.

17. Final

    Der Job sowie alle seine Children sind beendet und haben ein Final Exit State.

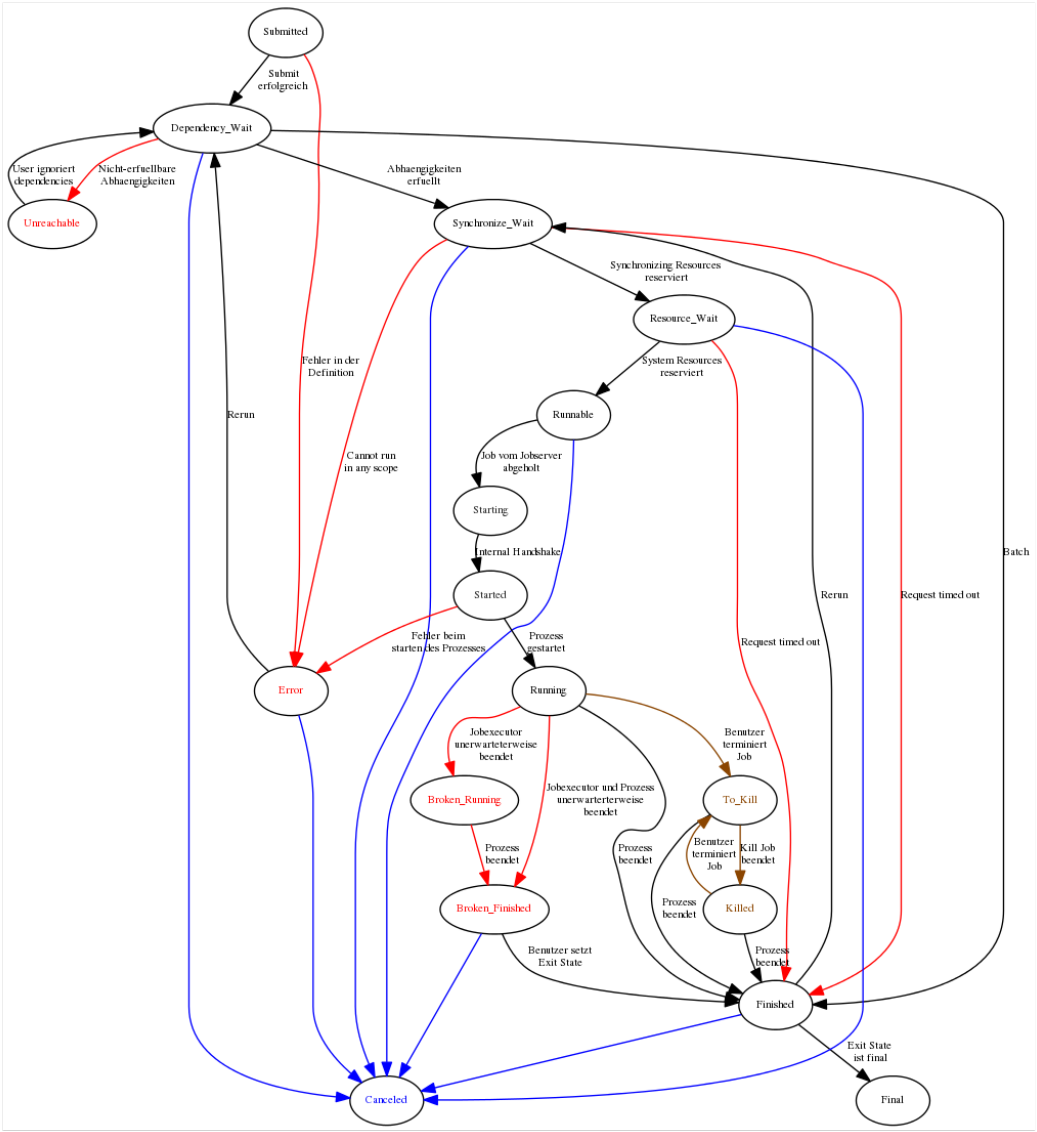


Abbildung 20.3: Statusdiagramm von Batches und Jobs

Die Abbildung 20.3 gibt die Statusübergänge des schedulix Laufzeitsystems wieder.

Variables Nach dem State können, je nach Konfiguration, die dort definierten Variablen in der definierten Reihenfolge und Farbe stehen.

20.3.1 Master Navigator Query-Maske

Durch Drücken des *Settings* Button, erscheint die Query-Maske. Die Query-Maske dient zum Einstellen der Query-Anforderungen für den aktuellen Dialog. Hier können Filterkriterien und Anzeigekriterien editiert werden. Der Dialog sieht folgendermaßen aus:

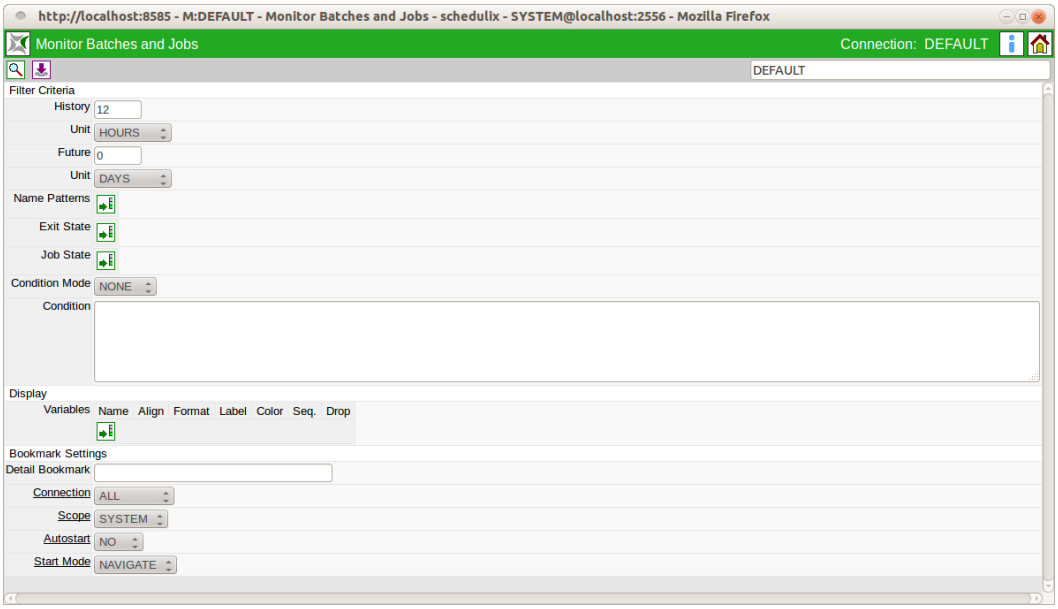


Abbildung 20.4: Running Master Jobs Query-Maske

Durch Veränderung in der Maske und anschließendes Abspeichern unter dem aktuellen oder einem neuen Bookmark-Namen (Eingabefeld in der Kopfzeile rechts oben), kann die Änderung für eine spätere Wiederverwendung gespeichert werden. Soll die Veränderung nur temporärer Natur sein, so kann sie einfach eingetragen und zum Navigationsbildschirm zurückgesprungen werden. Damit sind die Änderungen nur für die aktuelle Suche gültig. Wird der Dialog anschließend geschlossen, sind die Änderungen verloren.
Die obigen Felder haben folgende Bedeutung:

History/Unit Die maximale Zeit, die seit Ende eines Ablaufes vergangen sein darf, damit er in der Anzeige erscheint. Sollen zum Beispiel alle Jobs, die aktuell laufen, und alle Jobs, welche in der letzten Stunde beendet wurden, angezeigt werden, muss hier der Wert 1 und im Feld *Unit* die Einheit Hours eingestellt werden. Sollen zum Beispiel alle laufenden und alle in der letzten Stunde beendeten Jobs angezeigt werden, muss im Feld *History* der Wert 1 und in der Unit die Einheit Hours eingestellt werden. Es gibt folgende Möglichkeiten:

1. Minutes

2. Hours

3. Days

Future/Unit Über das Feld *Future* wird festgelegt, ob und wie weit zukünftig zu submittende Batches bzw. Jobs in der Masterliste angezeigt werden. Unabhängig von dem im Feld *Future* ausgewählten Intervall, kann für Schedules ohne aktivierten Kalender nur der nächste Start angezeigt werden. Für Schedules mit Kalender können nur geplante Starts im Kalenderhorizont des Schedules angezeigt werden. Das Feld *Unit* wird wie bei der History behandelt.

Liste Name Patterns In der Liste der Name Patterns kann eine Menge von Namens Pattern stehen, nach denen die Ergebnismenge gefiltert werden soll. Ein Name Pattern kann der vollständige Name, ein Namensteil oder ein regulärer Ausdruck sein.
Werden in der Liste mehrere Einträge gesucht, wird nach jedem Vorkommen all dieser Einträge gesucht. Diese werden mit "Oder" verknüpft.
Beispiel:
Es soll nach dem Auftreten des Namens "Pipeline" gesucht werden.
Folgende Name Patterns sind möglich:

- Vollständig: **Pipeline** (Groß-/Kleinschreibung spielt keine Rolle)
- Teil-String: **eline**
- Regulärer Ausdruck: **P.*line**

Durch die *Hinzufügen* und *Entfernen* Buttons ist die Änderung der Liste der Name Patterns möglich.

Liste Exit State In der Liste können Exit States eingetragen werden, welche die gesuchten Jobs haben müssen, damit sie in der Ergebnisliste angezeigt werden können.
Beispiel:
Es sollen nur Jobs angezeigt werden, deren Exit State "Failure" ist.
Durch Hinzufügen des Exit State "Failure" in die Liste der Exit States, kann nun nach diesen Jobs gesucht werden.
Enthält die Liste keine Einträge, werden die Exit States bei der Suche nicht berücksichtigt.
Durch die *Hinzufügen* und *Entfernen* Buttons ist die Änderung der Liste der Exit States möglich.

Liste Job State In der Liste kann nach Job States gesucht werden, die sich in einem der angegebenen Zustände befinden.
Enthält die Liste keine Einträge, werden die Job States bei der Suche nicht berücksichtigt.
Durch die *Hinzufügen* und *Entfernen* Buttons, ist die Änderung der Liste der Job States möglich.

Condition Mode Der Condition Mode sagt aus, ob und wie die nachfolgende Condition ausgewertet wird.
Es gibt folgende Möglichkeiten:

- None: Die Condition wird nicht berücksichtigt.
- And: Die Suchanfrage und die Condition werden mit ”Und” verknüpft.
- Or: Die Suchanfrage und die Condition werden mit ”Oder” verknüpft.
- Minus: Die Suchanfrage und die Negation der Condition werden mit ”Und” verknüpft.
- Only: Es wird nur die Condition berücksichtigt.

Condition Mit der Condition lässt sich die Suchanfrage verfeinern. Für die genaue Syntax und die verschiedenen Möglichkeiten wird auf die Syntaxdokumentation (List Job) verwiesen.

Liste Variables In der Liste Variables können alle, in der Navigationsliste anzuzeigenden Job-Variablen und Parameter, definiert werden. Hiermit ist eine leichte Übersicht über wichtige Parameter des Jobs möglich. Die Anzeige erfolgt am Ende jeder Zeile in der Navigationsliste.
Bei den Variables müssen folgende Felder eingetragen werden:

Name Der Name der Variablen oder des Parameters muss hier eingetragen werden.

Align Mit dem Feld *Align* kann angeben werden, wie die Anzeige der Variablen in der Liste erfolgen soll. Folgende Möglichkeiten können ausgewählt werden:

- LEFT

  Die Anzeige erfolgt linksbündig.

- CENTER

  Die Anzeige erfolgt zentriert.

- RIGHT

  Die Anzeige erfolgt rechtsbündig.

Format Hiermit kann das Format für die Ausgabe angegeben werden. Es gibt folgende Möglichkeiten:

- KEINE

  Es erfolgt keine Formatierung.

- NUMBER

  Es erfolgt eine numerische Formatierung mit Tausendertrenner und Nachkommastellen (falls nötig).

Label Hiermit kann die Überschrift, welche im Listen-Kopf für die Variable erscheinen soll, festgelegt werden. Wird kein Label festgelegt, dann wird der Variablenname als Überschrift verwendet.

Color Hiermit kann die Hintergrundfarbe der Spalte für die Variable hinterlegt werden. Im ”Drop Down” Menü des Feldes ist eine Vorschau für die jeweilige Farbe zu sehen.

Seq. Über die Buttons *Up* und *Down*, kann die Reihenfolge der Variablen in der Liste eingestellt werden. Die Buttons sind nur zu sehen, wenn mehr als ein Eintrag in der Liste der Variablen erscheint.

Bookmark Settings Die Eingabefelder im Bereich *Bookmark Settings* sind nur beim Speichern eines Bookmarks relevant. Die folgenden Eingabefelder beschreiben Eigenschaften des zu speichernden Bookmarks.

Detail Bookmark Wird im Feld *Detail Bookmark* der Name eines Detail Bookmarks eingegeben, so wird beim Öffnen eines Detail Navigation Fensters aus einem über diesen Bookmark geöffneten Master Navigation Fensters der angegebene Detail Bookmark verwendet. Existiert dieser noch nicht, so wird dieser Bookmark-Name in der Detail Navigation in eckigen Klammern angezeigt. Diese verschwinden beim Anlegen des Detail Bookmarks in der Detail Navigation durch Speichern des Bookmarks. Das Feld *Detail Bookmark* wird nur in den *Bookmark Settings* der Master Navigation angezeigt.

Connection Hier wird festgelegt, für welche Connections der aktuelle Bookmark gelten soll. Es gibt zwei Ausprägungen:

1. ALL

   Der Bookmark kann für alle Server Connections verwendet werden.

2. CURRENT

   Der Bookmark kann nur für die aktuelle Server Connection verwendet werden.

Scope Hier wird der Gültigkeitsbereich definiert, für den der aktuelle Bookmark gelten soll. Es gibt zwei Ausprägungen:

1. USER

   Der Bookmark ist nur für den aktuellen Benutzer sichtbar. Alle anderen Benutzer sehen diesen Bookmark nicht, oder, falls es sich um eine Änderung an einen systemweit verfügbaren Bookmark handelt, ist die Änderung nur lokal für diesen Benutzer sichtbar.

2. SYSTEM

   Der Bookmark oder die aktuelle Änderung ist für alle Benutzer sichtbar. Das heißt, alle Benutzer bekommen diesen neuen Bookmark in ihrer Bookmark-Liste zu sehen. Handelt es sich um eine Änderung an einem bestehenden systemweiten Bookmark, trifft die Änderung auf alle Benutzer zu.

Autostart Durch Autostart kann angegeben werden, ob ein Bookmark sofort nach dem Anmelden aktiv werden soll. Das Feld *Autostart* wird nur in den *Bookmark Settings* der Master Navigation angezeigt. Es gibt zwei Ausprägungen:

1. YES

   Der Bookmark wird sofort nach dem Anmelden gestartet. Das heißt, für jeden Bookmark mit Autostart "Yes" erscheint nach dem Anmelden ein Fenster auf dem Bildschirm.

2. NO

   Der Bookmark muss explizit durch Anklicken im Bookmark-Dialog gestartet werden und ist nicht automatisch aktiv. Das ist der Default-Wert.

Start Mode Durch den Start Mode wird festgelegt, ob die Query sofort nach dem Start des Bookmarks ausgeführt und das Navigationsfenster angezeigt werden soll, oder ob der Query-Dialog angezeigt werden soll. Es gibt zwei Ausprägungen:

1. NAVIGATE

   Die Query wird sofort ausgeführt und der Navigationsbildschirm mit allen Ergebnissen angezeigt.

2. QUERY

   Der Query-Dialog wird zuerst angezeigt.

20.4 Detail Navigation

Hat man auf einen der Master Jobs (welcher Children hat) geklickt, erscheint das "Detail"-Fenster des Master Jobs.

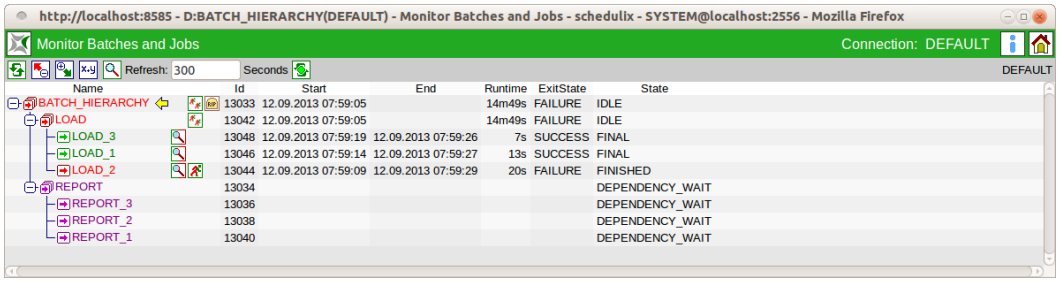


Abbildung 20.5: Running Master Jobs Detailfenster

Das "Detail"-Fenster ähnelt dem Master-Fenster. Folgende Unterschiede sind festzustellen:

20.4.1 Tree Darstellung

Statt der flachen Liste im Master-Fenster werden alle Children des aktuellen Batches und deren Children in einer Tree Hierarchy-Darstellung angezeigt. Die erste Ebene ist aufgeklappt, jede weitere Ebene nicht. Eine Ausnahme hiervon stellt die Suche mittels 'Name Patterns' dar, da hier jeder gefundene Job aufgelistet wird und dazu die jeweilige Ebene und die darüber liegenden Ebenen ebenfalls automatisch aufgeklappt werden.

20.4.2 Suchergebnis

Das Ergebnis einer Suche wird mittels des Pfeilsymbols

dargestellt. Hierdurch können gefundene Einträge die aufgeklappt wurden, da ein Child-Eintrag den Suchkriterien entspricht, von Treffern unterschieden werden. Nur Treffer werden mittels des Pfeilsymbols angezeigt.

20.4.3 Detail Navigation Query-Maske

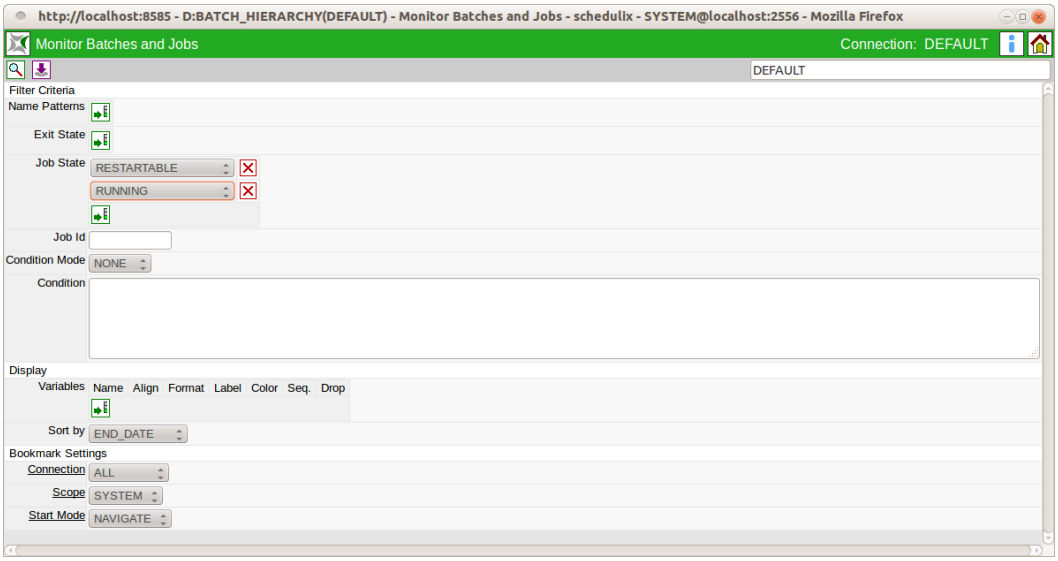


Abbildung 20.6: Detail Navigation Query-Maske

Die Query-Maske für die Detail Navigation entspricht der Query-Maske für die Master Navigation, mit folgenden Unterschieden:
Die Felder *History*, *Unit*, *Detail Bookmark* und *Autostart* entfallen und es gibt einige zusätzliche Felder.
Folgende Felder sind auf der Maske zu sehen:

Liste Job State In der Liste Job State kann nach Jobs gefiltert werden, welche den eingetragenen Job State haben. Es kann jeder im Feld *State* beschriebene State ausgewählt und nach ihm gefiltert werden.
Werden keine Job States eingetragen, werden diese bei der Suche nicht berücksichtigt.

Job Id In diesem Feld kann eine konkrete Job Id eingetragen werden, nach der gesucht werden soll. Als Ergebnis kann immer nur maximal ein Job gefunden werden.

Sort By Hiermit kann die Liste nach verschiedenen Kriterien sortiert werden. Es sind folgende Kriterien möglich.

1. Name

   Die Ergebnisliste soll alphabetisch nach dem Namen des Jobs sortiert werden.

2. Start Date

   Die Ergebnisliste soll chronologisch nach dem Zeitpunkt des Start des Jobs sortiert werden.

3. End Date

   Die Ergebnisliste soll chronologisch nach dem Zeitpunkt des Beendens des Jobs sortiert werden.

20.5 Detailmaske für Jobs

Klickt man im "Detail"- oder im "Master"-Fenster einen Job ohne Children an, wird automatisch die Detailmaske für Jobs angezeigt. Sie sieht folgendermaßen aus:

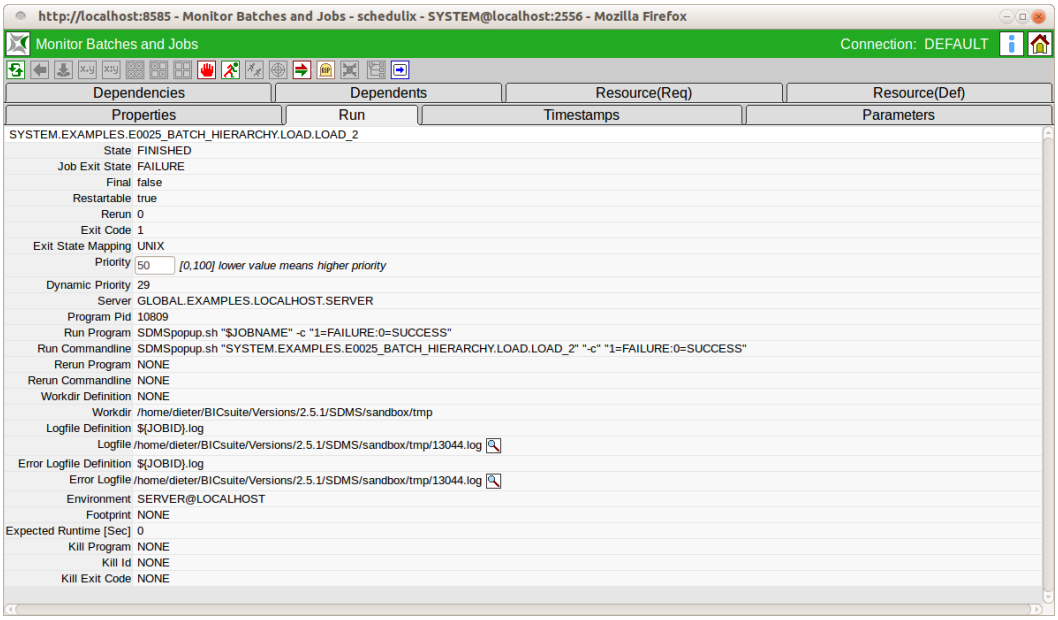


Abbildung 20.7: Detailmaske für Jobs

In dieser Maske werden alle Laufzeitinformationen angezeigt und manche können verändert werden. Des Weiteren stehen Aktions-Buttons zur Verfügung, die eine Manipulation an den aktuellen Objekten zulassen.

20.5.1 Buttons

Die folgenden Buttons sind alle Aktions-Buttons. Das heißt, durch diese Buttons wird eine bestimmte Aktion auf dem gewählten Job durchgeführt. Um dies zu erreichen, wird nach der Betätigung des Buttons der Confirm Bildschirm angezeigt.

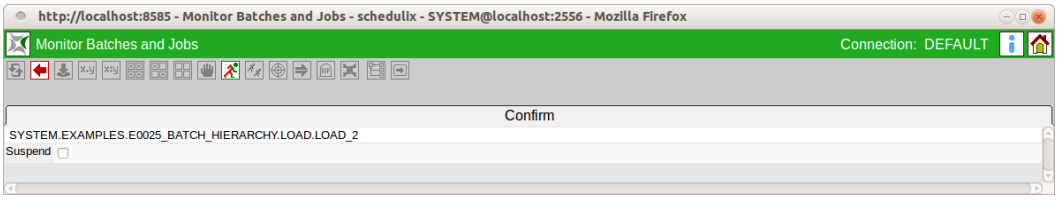


Abbildung 20.8: Confirm Maske

(Siehe Abbildung 20.8)

Im Textfeld kann nun ein Kommentar eingegeben werden, welcher den Grund der Aktion oder zusätzliche Bemerkungen beschreibt. Dieser Text wird später im Tab "Audit", im Feld *Reason/Comment* angezeigt. Durch erneutes Betätigen des jeweiligen Aktions-Buttons, wird die Aktion durchgeführt. Durch Betätigen des *Cancel* Buttons, wird die Aktion abgebrochen und zum vorherigen Bildschirm zurückgesprungen. Abhängig von der Aktion enthält der Confirm Bildschirm weitere Eingabefelder zur Steuerung der Aktion.

 Suspend

Mittels des *Suspend* Buttons, können Jobs Suspended werden. Dies ist nur möglich, falls die Jobs sich nicht schon im State "Suspended" befinden und noch nicht beendet sind. Nach dem erfolgreichen Suspend haben die Jobs den State "Suspended".

Suspend Local Wird das Flag *Suspend Local* gesetzt, so wirkt sich der Suspend nicht auf die Children der Batches oder Jobs aus. Die Children werden weiter verarbeitet, der Suspended Batch oder Job wird jedoch nicht FINAL werden. Abhängige Batches oder Jobs müssen warten, bis der Batch oder Job Resumed wurde.

Suspend Restricted Diese Checkbox wird nur angezeigt, falls der Benutzer der Gruppe ADMIN angehört. Ist beim Suspend dieses Flag gesetzt, so kann der Batch, bzw. Job nur von einem Benutzer der ADMIN-Gruppe wieder Resumed werden.

Zusätzliche Eingabefelder: *Resume*, *Resume Time*, *Resume In*, *Unit Suspend Restricted*

Resume Hier kann ausgewählt werden, ob ein automatischer Resume stattfinden soll.
Es gibt folgende Möglichkeiten:

- NO: Wählt diese Funktionalität ab und es werden keine anderen Eingabefelder angezeigt.
- AT: Wählt einen automatischen Resume zu einem festen Zeitpunkt. Das Eingabefeld *Resume Time* wird angezeigt.
- IN: Wählt einen automatischen Resume nach Ablauf einer Zeit. Die Eingabefelder *Resume In* und *Unit* werden angezeigt.

Resume Time Hier wird der gewünschte Resume Zeitpunkt im Format "YYYYMMDD HH:MM" eingegeben.

Resume In Hier wird angegeben, wie viele Zeiteinheiten (siehe *Unit*) bis zum Resume gewartet werden soll.

Unit Hier wird eingegeben, ob es sich bei der Eingabe im *Resume In* um Minuten (MINUTE), Stunden (HOUR), oder Tage (DAY) handeln soll.

 Resume
Mittels des *Resume* Buttons können Jobs, welche sich im State "Suspended" befinden, wieder aktiviert werden. Das heißt, es erfolgt eine erneute Prüfung der Ressourcen und Abhängigkeiten und, falls alle Prüfungen erfolgreich sind, wird der Job auf einem Jobserver gestartet.
Zusätzliche Eingabefelder: *Resume*, *Resume Time*, *Resume In*, *Unit*

Resume Hier kann die Art des Resumes spezifiziert werden.

- OFF: Steht nur zur Auswahl, falls der Job suspended und ein Autoresume (IN/AT) aktiv ist. Diese Option deaktiviert diesen Autoresume.
- NOW: Es findet ein sofortiger Resume statt. Es werden keine anderen Eingabefelder angezeigt.
- AT: Wählt einen automatischen Resume zu einem festen Zeitpunkt. Das Eingabefeld *Resume Time* wird angezeigt.
- IN: Wählt einen automatischen Resume nach Ablauf einer Zeit. Die Eingabefelder *Resume In* und *Unit* werden angezeigt.

Resume Time, Resume In, Unit Siehe oben (Aktion Suspend)

 Rerun
Der *Rerun* Button erscheint, falls der aktuelle Job beendet wurde und sich in einem "Restartable"-Zustand befindet. Das heißt, er besitzt einen Exit State, welcher nicht Final ist. Durch Drücken des *Rerun* Buttons, wird das Rerun-Programm aufgerufen (falls definiert, ansonsten das Run-Kommando). Hiermit ist ein erneutes Starten eines fehlerhaften Jobs nach der Problembehebung möglich.
Zusätzliche Eingabefelder: *Suspended*, *Resume*, *Resume Time*, *Resume In*, *Unit*

Suspended Wird dieses Flag gesetzt, so wird der Job Suspended und das Auswahlfeld *Resume* wird angezeigt. Ist dieses Feld nicht gesetzt, so werden keine weiteren Felder angezeigt.

Resume, Resume Time, Resume In, Unit Siehe oben (Aktion Suspend)

 Rerun Children

Der *Rerun Children* Button wird angezeigt, wenn sich Children des aktuellen Jobs in einem "Restartable"-Zustand befinden. Das heißt, sie besitzen einen Restartable Exit State. Durch Drücken des *Rerun Children* Buttons wird das Rerun für alle Children gestartet, welche Restartable sind. Hiermit ist ein erneutes Starten fehlerhafter Children nach einer Problembehebung möglich.
Zusätzliche Eingabefelder: Siehe *Rerun*

 Kill

Der *Kill* Button erscheint nur, falls der aktuelle Job im Moment läuft (State Running) und in der Job Definition ein Kill-Programm angegeben wurde. Durch Drücken des Buttons *Kill* wird das jeweilige Kill Program aufgerufen und der State des Jobs auf "TO KILL" umgesetzt. Wird das Kill Program erfolgreich durchgeführt, ändert sich der State des Jobs auf "KILLED". Hiermit ist ein Terminieren des Jobs im Programmlauf möglich.

 Set State

Ermöglicht das manuelle Setzen eines Exit States für einen Job. Der *Set State* Button erscheint, falls ein Job mindestens eine der folgenden Bedingungen erfüllt:

- Der Job ist in einem Fehler Status (Restartable)
- Der Job ist in einem PENDING Status
- Der Job ist Suspended und nicht aktiv und nicht in einem FINAL Status

Zusätzliche Eingabefelder: *Set Exit State*, *Force*, *Resume*

Set Exit State Hier kann im Feld *Set Exit State* ein gültiger Exit State ausgewählt werden.

Force Hier wird ausgewählt, ob man einen Exit State zulassen will, der nicht vom Job selbst über seinen Exit Code erreicht werden kann.

Resume Hier wird ausgewählt, ob ein Suspended Job nach dem Set State Resumed werden soll.

 Cancel Run

Der *Cancel Run* Button erscheint, falls der Job auf Resources oder Abhängigkeiten wartet oder noch auf einem Jobserver zugeteilt wurde. Durch Drücken des *Cancel Run* Buttons kann der geplante Job-Lauf abgebrochen werden. Der Job bekommt den State "Cancelled". Es ist nun kein Rerun und kein Resume möglich. Der Job bekommt auch keinen Exit State. Jobs, welche auf einen gecancelten Job warten, müssen diesen Job ignorieren oder ebenfalls gecancelt werden.

 Comment

Mittels des *Comment* Buttons kann ein beliebiger Kommentar bzgl. des Jobs eingetragen werden. Es wird außer dem Kommentareintrag keine weitere Aktion am Job durchgeführt.

 Edit Job

Beim *Edit Job* Button handelt es sich nicht um einen Aktions-Button, da hier keine Änderung am aktuellen Laufzeitobjekt durchgeführt werden kann. Wird der *Edit Job* Button betätigt, erscheint ein neues Fenster mit dem Batches and Jobs Definitionsscreen der Job Definition. Die angezeigten Daten beziehen sich auf die Definition des Ablaufobjektes zum Zeitpunkt des Submits.

20.5.2 Tab Properties

Der Tab ”Properties” enthält alle Daten bzgl. der Job Definition und des aktuellen States. Er sieht folgendermaßen aus:

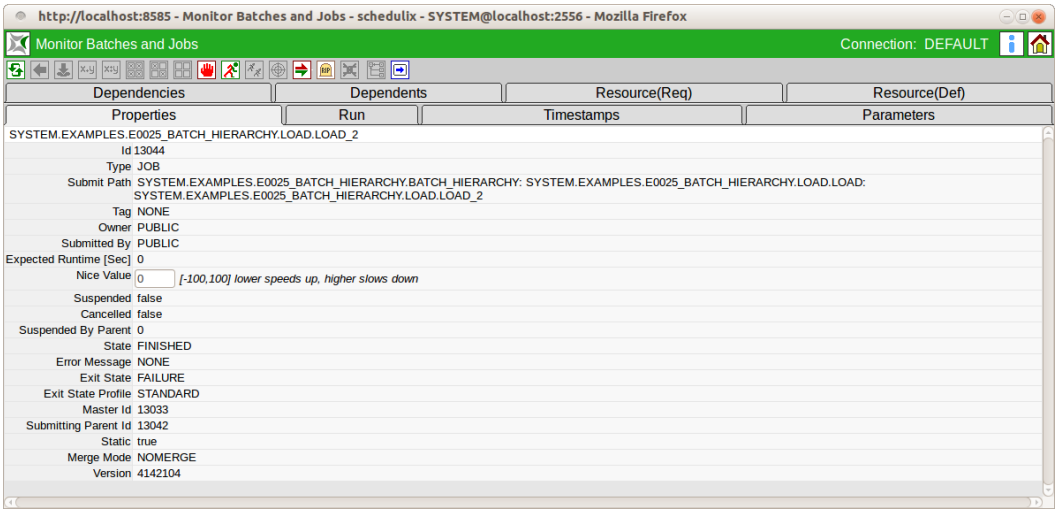


Abbildung 20.9: Job und Batch Properties

Die Felder des Tabs ”Properties” haben folgende Bedeutung:

Id Das ist die eindeutige Identifikationsnummer des Job-Laufzeitobjektes.

Type Das ist der Typ des Laufzeitobjektes. Mehr zu Typen finden Sie im Kapitel 13.5.1.

Submit Path Das ist der vollständige Submit-Pfad des Laufzeitobjektes. Das heißt, hier wird kommagetrennt angegeben, über welchen Pfad der Submit stattgefunden hat.

Tag Das ist der Name des Child Tags. Dies ist nur bei einem dynamischen Child notwendig und gibt eine eindeutige Identifikation der aktuellen Child-Instanz. Mehr zu dynamischen Children finden Sie im Kapitel 13.5.4.1.

Owner Das ist der Name der Gruppe, welche die Job Definition für dieses Laufzeitobjekt erstellt hat.

Submitted By Das ist der Name der Gruppe, welche die Laufzeitinstanz submitted hat. Wurde der Job dynamisch durch ein Parent-Programm submitted, ist dies der Benutzer, welcher den Parent submitted hat.

Unresolved Handling Das Feld *Unresolved Handling* ist nur in der Maske des Master Jobs sichtbar. Es zeigt die Einstellung des Feldes *On Unresolved Error* aus der Submit-Maske.

- Ignore: Die Unresolved Dependencies werden grundsätzlich ignoriert.
- Suspend: Im Falle von Fehlern, aufgrund Unresolved Dependencies wird der gesamte Ablauf "suspended" submitted.
- None: Default-Verhalten im Falle von Unresolved Dependencies, wird behandelt, wie in der Dependency Definition vorgegeben.

Nice Value Bei diesem Feld handelt es sich um ein Eingabefeld. Der Nice Value kann hier eingegeben werden. Mehr zum Nice Value finden Sie im Kapitel 13.5.4.1. Da es sich hier um das einzige Eingabefeld des Dialoges handelt, muss die Tabulatortaste nach der Eingabe betätigt werden, damit der *Save* Button aktiv wird, um die Änderung abzuspeichern.

Suspended Das Feld gibt an, ob das Laufzeitobjekt aktuell Suspended ist (TRUE) oder nicht (FALSE). Durch die Buttons *Suspend* und *Resume* kann der Suspend State geändert werden.

Suspend by Parent Gibt an, ob das Laufzeitobjekt durch den Parent suspended wurde (1) oder nicht (0).

State Zeigt den aktuellen State des Laufzeitobjektes an. Mehr zum State finden Sie im Kapitel 20.3.

Error Message Wurde im Feld *State* der State "ERROR" gemeldet, wird eine nähere Beschreibung des Fehlers genannt.

Exit State Ist der Job beendet, wird hier der resultierende Exit State des Jobs mitgeteilt.

Exit State Profile Hier wird das Exit State Profile, welches in der Job Definition eingetragen wurde, angezeigt. Mehr zum Exit State Profile finden Sie im Kapitel 13.5.1.

Master Id Hierbei handelt es sich um die Id des Master Jobs, welcher submitted wurde, um dieses Laufzeitobjekt zu erzeugen. Ist das Objekt selbst als Master Job submitted worden, ist diese identisch mit der Id.

Submitting Parent Id Hierbei handelt es sich um die Id des Parent-Laufzeitobjektes, welcher den aktuellen Job submitted hat. Hat der Job kein Parent, wird "NONE" angezeigt.

Static *Static* ist das Kennzeichen das angibt, ob es sich beim aktuellen Laufzeitobjekt um einen statisch (TRUE) oder um ein dynamisch submittetes Child (FALSE) handelt. Mehr zu dynamischen Children finden Sie im Kapitel 13.5.4.1.

Merge Mode Gibt den aktuell verwendeten Merge Mode an. Mehr zum Merge Mode finden Sie im Kapitel 13.5.4.1.

Version Gibt die Versionsnummer der beim Submit aktuellen Job Definition an.

20.5.3 Tab Run

Der Tab ”Run” enthält alle Informationen, welche sich auf die aktuellen Informationen zum Run Program und zur Laufzeitumgebung und des Rechners beziehen. Der Tab sieht folgendermaßen aus:

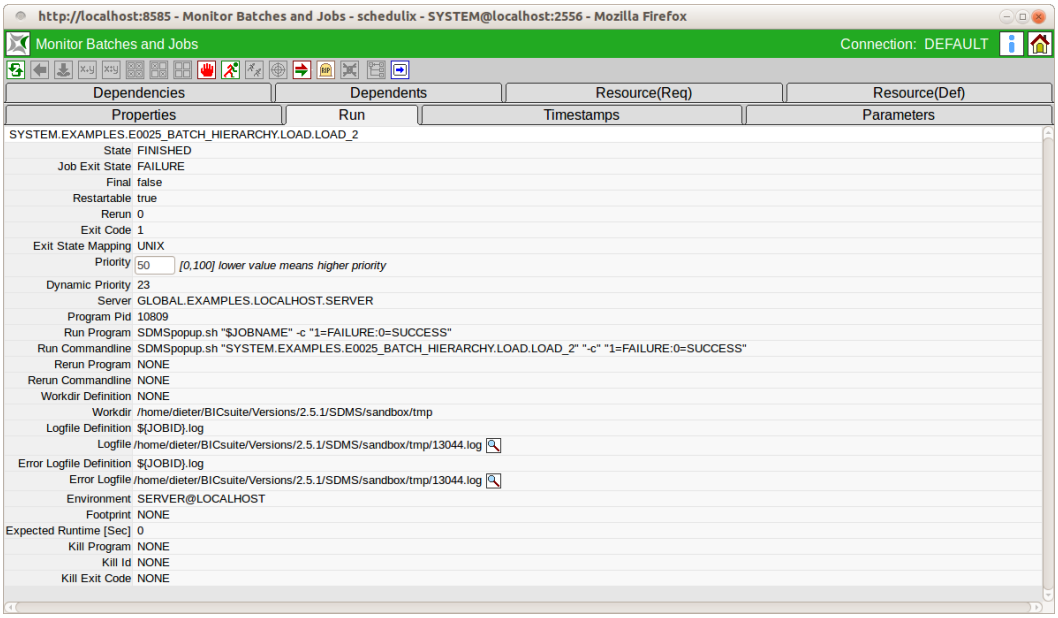


Abbildung 20.10: Batch und Job Run Information

Die Felder des Tabs ”Run” haben folgende Bedeutung:

State Zeigt den aktuellen State des Laufzeitobjektes an. Mehr zum State finden Sie im Kapitel 20.3.

Job Exit State Der Job Exit State enthält den aktuellen Exit State des Jobs, falls der Job beendet ist, ansonsten ”NONE”.

Final Handelt es sich bei dem Job Exit State um einen Final State, dann steht der Wert des Feldes auf ”TRUE”. Ist der Job Exit State ”NONE” oder kein Final State, dann steht der Wert des Feldes auf ”FALSE”.

Restartable Ist der Job Restartable, dann steht der Wert auf ”TRUE”, ansonsten auf ”FALSE”. Falls das Feld den Wert TRUE hat, ist der Button *Rerun* aktiv.

Rerun Im Feld *Rerun* steht die Anzahl der Neustarts des Jobs.

Exit Code Beim Exit Code handelt es sich um den Exit-Wert, den das Run Program bei der Beendigung des Prozesses hatte. Mehr zu Exit Code finden Sie im Kapitel 13.5.2.

Exit State Mapping Das im Batches and Jobs-Dialog definierte Exit State Mapping des Jobs, welches beim Umwandeln des Exit Codes in den Job Exit State verwendet wurde. Mehr zu Exit State Mapping finden Sie im Kapitel 13.5.2.

Priority Hierbei handelt es sich um ein Eingabefeld. Es wird die aktuelle Priorität angezeigt und diese kann verändert werden. Durch anschließendes Betätigen der Tabulatortaste und Drücken des *Save* Buttons wird die neue Priorität gespeichert. Mehr zum Thema Priorität finden Sie im Kapitel 13.5.2.

Dynamic Priority Hierbei handelt es sich um die dynamische (effektive) Priorität des Jobs, welche bei ansteigender Zeit vom Submit bis zum Ausführen steigt. Mehr zur dynamischen Priorität finden Sie im Kapitel 12.4.2.3.

Server Das Feld *Server* gibt den aktuellen Jobserver an, auf welchem der Prozess ausgeführt wird oder wurde.

Program PId Die *Program PId* ist die Prozess Id des Run Programs.

Run Program Hierbei handelt es sich um die im Batches and Jobs-Dialog definierte Kommandozeile des Run Programs. Variablen und Parameter werden hier noch mit ihren Namen angezeigt und werden nicht ersetzt.

Run Command Line Die im Batches and Jobs-Dialog definierte Kommandozeile des Run Programs mit ersetzten Werten. Das ist die Darstellung, wie sie der zu verarbeitenden Shell übergeben wurde. Alle Variablen und Parameter wurden mit ihren aktuellen Werten ersetzt.

Rerun Program Die im Batches and Jobs-Dialog definierte Kommandozeile des Rerun Programs. Variablen und Parameter werden hier noch mit ihren Namen angezeigt und werden nicht ersetzt. Wurde kein Rerun Program definiert, steht hier NONE.

Rerun Command Line Die im Batches and Jobs-Dialog definierte Kommandozeile des Rerun Programs, mit ersetzten Werten. Das ist die Darstellung, wie sie der zu verarbeitenden Shell übergeben wurde. Alle Variablen und Parameter wurden mit ihren aktuellen Werten ersetzt. Ist kein Rerun durchgeführt worden, steht hier NONE.

Workdir Definition Im Feld *Workdir Definition* ist das Verzeichnis, von dem der Prozess ausgeführt wird, angezeigt. Steht hier "None", wird das Default-Arbeitsverzeichnis des ausführenden Jobservers genutzt.

Workdir Hierbei handelt es sich um das im aktuellen Job verwendete bzw. zu verwendende Workdir. Ist der Job beendet oder läuft er aktuell, werden die ersetzten Parameter und Variablen angezeigt, ansonsten die Parameter.

Logfile Definition Die *Logfile Definition* entspricht der Definition des Logfiles im Batches and Jobs- Dialog.

Logfile Hier steht der Name und der Pfad des aktuell verwendeten Logfiles (falls der Job beendet ist oder noch läuft), ansonsten NONE. Mögliche Variablen und Parameter werden durch die tatsächlichen Werte ersetzt.
Mittels des Buttons

kann ein neues Fenster aufgemacht werden, welches den Inhalt des Logfiles darstellt. Sollte dies nicht möglich sein, wenden Sie sich bitte an Ihren Systemadministrator.

Error Logfile Definition Die Error Logfile Definition entspricht der Definition des Error Logfiles im Batches and Jobs-Dialog.

Error Logfile Hier steht der Name und der Pfad des aktuell verwendeten Error Logfiles (falls der Job beendet ist oder noch läuft) ansonsten NONE. Mögliche Variablen und Parameter werden durch die tatsächlichen Werte ersetzt.
Der Button

hat die gleiche Funktion, wie soeben bei Logfile beschrieben.

Environment Das Environment entspricht der Definition des Environments im Batches and Jobs-Dialog.

Footprint Der Footprint entspricht der Definition des Footprints im Batches and Jobs-Dialog.

Expected Runtime Die Expected Runtime entspricht der Definition im Batches and Jobs-Dialog.

Kill Program Das Kill Program entspricht der Definition des Kill Programs im Batches and Jobs-Dialog.

Kill Id Die Kill Id enthält die Prozess Id des Kill Programs, falls dieses gestartet wurde. Wurde noch kein Kill durchgeführt, ist die Kill Id NONE.

Kill Exit Code Der Kill Exit Code ist das Ergebnis, welches das Kill Program nach Beendigung des Kill-Vorganges an den Jobserver zurückliefert. Hiermit kann der Erfolg oder Misserfolg eines Kill-Versuches ermittelt werden, falls dies vom Kill Program vorgesehen ist.

20.5.4 Tab Timestamps & Statistics

Der Tab "Timestamps & Statistics" gibt verschiedene Zeitpunkte für den jeweiligen Job an. Mit diesem Zeitpunkt kann ein Überblick über das Zeitverhalten und die Dauer der Job-Ausführung gegeben werden. Der Tab sieht folgendermaßen aus:

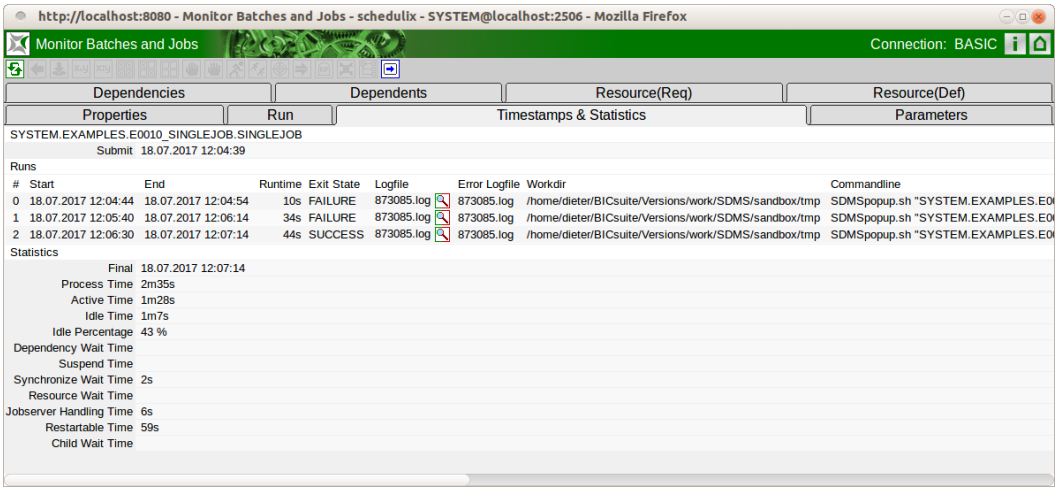


Abbildung 20.11: Batch und Job Timestamps

Der Tab ist nur informativ, es können keine Eingaben gemacht werden.
Die Felder des Tabs "Timestamps & Statistics" haben folgende Bedeutung:

Submit Im Timestamp *Submit* steht die Zeit des Submit-Zeitpunktes für den aktuellen Job. Handelt es sich um einen statischen Job, ist dies der Submit-Zeitpunkt für den Master Job, bei dynamischen Jobs die Zeit, an dem der Parent Job den dynamischen Submit durchgeführt hat. Der Submit-Zeitpunkt stimmt mit dem Zeitpunkt, zu dem der Job an den State Dependency Wait gewechselt hat, überein.

In der Tabelle *Runs* werden alle Ausführungen des Jobs aufgelistet. Im Normalfall enthält diese Tabelle eine Zeile. Wird der Job allerdings aufgrund von Fehlern neu gestartet, werde mehrere Zeilen angezeigt.

Start In der Spalte *Start* steht die Zeit des Starts des Run Programs durch den Jobserver.

End In der Spalte *End* steht die Zeit der Beendigung des Jobs.

Runtime Die Spalte *Runtime* zeigt die Laufzeit des jeweiligen Laufes an.

Exit State Die Spalte *Exit State* enthält den Exit Status des Laufes.

Logfile Die Spalte *Logfile* enthält den Namen des Logfiles für standard out. Durch einen Klick auf das Icon (Lupe) neben dem Namen kann das Logfile angezeigt werden.

Error Logfile Die Spalte *Error Logfile* enthält den Namen des Logfiles für standard error. Durch einen Klick auf das Icon (Lupe) neben dem Namen kann das Error Logfile angezeigt werden. Das Icon wird nur angezeigt, falls sich die Namen von Logfile und Error Logfile unterscheiden.

Workdir In der Spalte *Workdir* wird das Arbeitsverzeichnis des Jobs angezeigt.

Commandline Die Spalte *Commdline* zeigt die zur Ausführung gebrachte Kommandozeile an.

Jobserver Die Spalte *Jobserver* zeigt den Namen des Jobserver Agents an, welcher den den Job ausgeführt hat.

Exit Code Die Spalte *Exit Code* enthält den Exit Code der jeweiligen Ausführung.

PID Die Spalte *PID* enthält die System Process Id des ausgeführten Programms.

Synchronize Die Spalte *Synchronize* zeigt an, zu welchem Zeitpunkt der Job in den State "Synchronize Wait" übergegangen ist. Das heißt, wann alle möglichen Abhängigkeiten des Jobs erfüllt wurden.

Resource Die Spalte *Resource* wird die Zeit des Statusüberganges in Resource Wait angezeigt. Das heißt, zu diesem Zeitpunkt wurden alle geforderten Synchronizing Resources erfüllt und dem Job zugeteilt.

Runnable Die Spalte *Runnable* gibt an, wann der Job in den State "Runnable" eingetreten ist. Das heißt, alle Dependencies und Resource-Anforderungen konnten erfüllt werden. Ab jetzt kann der Job von einem Jobserver abgearbeitet werden.

Im folgenden werden Informationen und Statistiken ausgegeben, welche sich auf die gesamte Lebenszeit des Jobs beziehen.

Final Im Timestamp *Final* steht die Zeit der Beendigung des Jobs, die einen Final State erzeugt hat. Wurde der Job beendet, besitzt aber einen Restartable State, wird hier NONE eingetragen.

Process Time Die *Process Time* ist die Zeit, welche der Job oder Batch verbracht hat, nachdem alle Abhängigkeiten erfüllt waren.

Active Time Die *Active Time* ist die Zeit, in der der Job oder eines seiner Kinder aktiv waren.

Idle Time Die *Idle Time* ist die Zeit, in der der Job inaktiv und keines seiner Kinder aktiv waren.

Idle Percentage Die *Idle Percentage* ist der Anteil der *Idle Time* an der *Process Time*.

Dependency Wait Time Die *Dependency Wait Time* ist die Zeit, die der Job aufgrund von Abhängigkeiten gewartet hat.

Suspend Time Die *Suspend Time* ist die Zeit, die der Job suspended war.

Synchronize Wait Time Die *Synchronize Wait Time* ist die Zeit, die der Job auf synchronisierende Ressourcen gewartet hat.

Resource Wait Time Die *Resource Wait Time* ist die Zeit, die der Job auf System Ressourcen gewartet hat.

Jobserver Handling Time Die *Jobserver Handling Time* ist der Zeit Overhead des Jobserver Agents.

Restartable Time Die *Restartable Time* ist die Zeit, die der Job nach einem Fehler auf einen Restart gewartet hat.

Child Wait Time Die *Child Wait Time* ist die Zeit, die der Job darauf gewartet, hat bis alle seinen Kinder FINAL waren.

20.5.5 Tab Dependencies

Im Tab ”Dependencies” werden alle Abhängigkeiten des aktuellen Jobs zu anderen Jobs angezeigt. Der Tab sieht folgendermaßen aus:

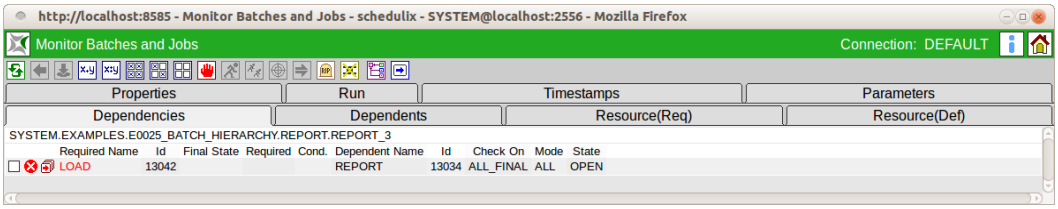


Abbildung 20.12: Batch und Job-Abhängigkeiten

Der Dialog gibt eine Übersicht über alle notwendigen Abhängigkeiten, ihren aktuellen State und die Möglichkeit Abhängigkeiten bewusst zu ignorieren.

Folgende Buttons sind auf der Maske sichtbar:

 Ignore Dependencies

Der Button *Ignore Dependencies* wird verwendet, um eine oder mehrere Abhängigkeiten bewusst zu ignorieren. Alle zu ignorierenden Abhängigkeiten müssen vorher in der Maske ausgewählt werden. Anschließend wird der Button gedrückt. Nun erscheint ein neues Fenster zur Auswahl des Ignore Modus und zur Eingabe eines Vermerks. Das Fenster sieht folgendermaßen aus:

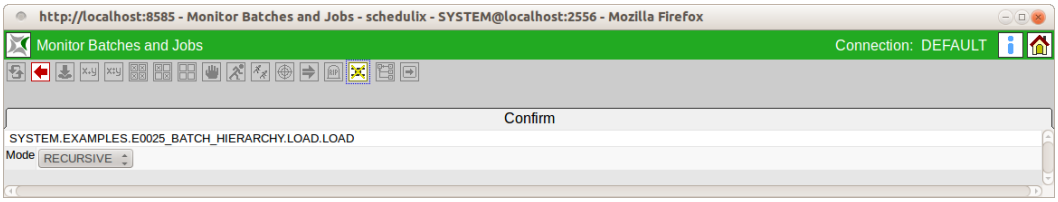


Abbildung 20.13: Batch und Job Ignore Dependencies Confirm Maske

Auf dieser Maske muss der Modus des Ignore eingegeben werden. Per Default erben alle Children eines Submitted Entities dessen Abhängigkeiten: Wird B von A benötigt, so ist auch jedes Child von A (CA) von B abhängig. Mittels des Modus des Ignore ist nun das Verhalten der vererbten Abhängigkeit bei den Children bestimmbar.
Es gibt folgende Optionen:

1. Recursive

   Dies ist das Default-Verhalten.

   Das Ignore bezieht sich nicht nur auf den aktuellen Job, sondern auch auf die vererbten Abhängigkeiten, die Children des aktuellen Jobs zu dem benötigten Job haben. Das heißt, der Parent Job ignoriert diese Abhängigkeit und alle seine Children ignorieren diese vererbte Abhängigkeit ebenfalls. Sollte ein Child eine explizit angegebene Abhängigkeit, welche nicht vererbt wurde, zum benötigten Job haben, so wird diese nicht ignoriert, sondern beachtet.

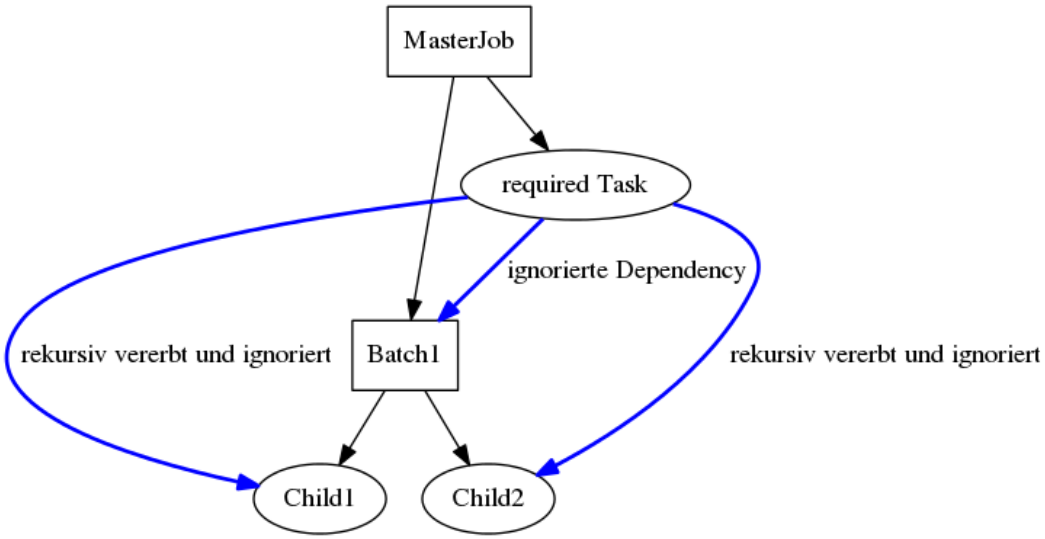


Abbildung 20.14: Beispiel für Recursive Ignore

(Abbildung 20.14)

2. Job Only

   Nur der Job selbst ignoriert die Abhängigkeit. Alle Children beachten die vererbte Abhängigkeit und warten auf den benötigten Job. Damit ist es möglich, den Parent Job laufen, seine Children allerdings warten zu lassen.

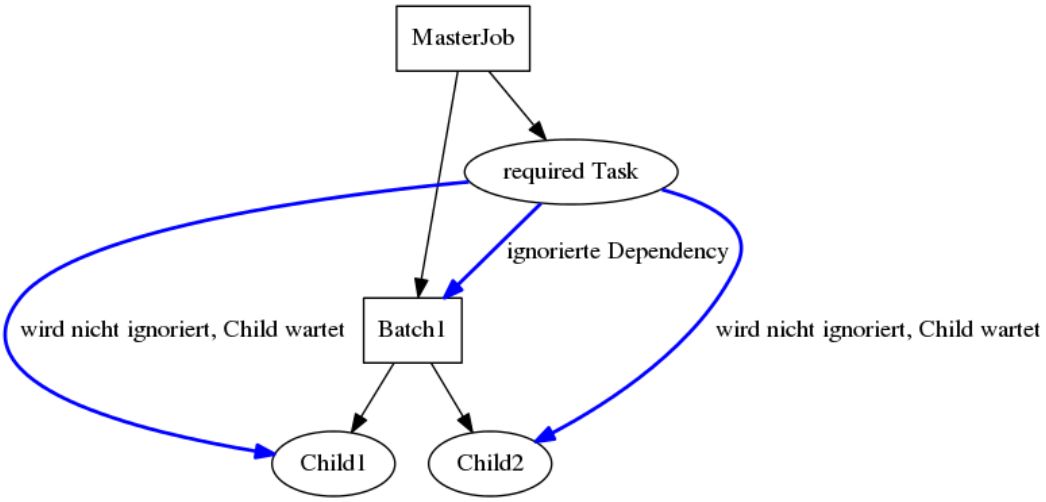


Abbildung 20.15: Beispiel für Job Only Ignore

(Abbildung 20.15)

Nach der Eingabe des Ignore Modus und der Eingabe eines Kommentars im Kommentarfeld, kann durch ein erneutes Betätigen des *Ingore Dependency* Buttons die Aktion durchgeführt werden.
Die Liste des Tabs zeigt alle Dependencies, welche zum Starten des Jobs erfüllt werden müssen, und ihren aktuellen State an. Innerhalb der Liste kann eine Selektion stattfinden, die mittels des Buttons *Ignore Dependencies* zum Ignorieren der Abhängigkeiten benutzt werden kann.
Folgende Felder sind in der Liste sichtbar:

Icon-Feld Über das *Icon*-Feld ist der aktuelle State der Abhängigkeit ersichtlich. Es gibt folgende Möglichkeiten:

Die Abhängigkeit wurde noch nicht erfüllt oder kann nicht mehr erfüllt werden. Sollte bei einer Abhängigkeit mit dem Dependency Mode ALL dieses Icon zu sehen sein, kann durch diese Abhängigkeit der aktuelle Job nicht ausgeführt werden.
Im Dependency Mode ANY muss mindestens eine Abhängigkeit nicht rot sein, damit der Job ablaufen kann.

Die Abhängigkeit wurde erfüllt. Im Dependency Mode ANY muss mindestens eine Abhängigkeit den State grün haben, damit der Job laufen kann. Im Dependency Mode ALL müssen alle Einträge grün sein, damit der Job laufen kann.

Required Name Hierbei handelt es sich um den Namen des Jobs, welcher als Voraussetzung für den Start des aktuellen Jobs gelaufen sein muss. Durch Anklicken des Namens kann zu seiner Laufzeitdefinition gewechselt und dessen State überprüft werden.

Id In diesem Feld wird die Laufzeit-Id des Required Jobs angezeigt.

Final State Wurde der Required Job abgearbeitet, steht hier der finale State des Jobs. Ist hier ein Final State eingetragen und die Abhängigkeit ist nicht erfüllt, muss der Grund ein falscher Final State für diese Abhängigkeit sein. Das heißt, es gibt einen Unterschied zwischen Final State und Required State. Soll der Job nun doch ausgeführt werden, kann dies nur durch das Ignorieren über den Ignore *Dependency* Button geschehen. Befinden sich die Abhängigkeiten der Liste im Dependency Mode ANY, muss mindestens eine andere Abhängigkeit erfüllt sein.

Required In diesem Feld wird der benötigte Final State des Jobs angezeigt. Damit die Abhängigkeit erfüllt ist, muss der Final State identisch mit dem Required State sein oder der Required State NONE sein.

Dependent Name In diesem Feld wird der Job genannt, welcher diese Abhängigkeit fordert. Dies ist im Normalfall der aktuell gewählte Job, es kann aber auch einer der Parents des Jobs sein.

Check On Das Feld *Check On* beschreibt das Verhältnis der Beendigung des Required Jobs. Es gibt folgende Ausprägungen:

1. All Final

   Der Required Job und alle sein Children müssen Final sein.

2. Job Only

   Nur der Required Job muss final sein, die Children werden nicht beachtet.

Mehr zum Thema Check On finden Sie im Kapitel 13.5.6.1.

Id Hierbei handelt es sich um die Id des abhängigen Jobs.

Mode Der *Mode* legt fest, wie die Liste der Jobs zu beachten ist. Es gibt folgende Ausprägungen:

1. ALL

   Es müssen alle Abhängigkeiten der Liste erfüllt werden, damit der Job starten kann.

2. ANY

   Es muss mindestens eine Abhängigkeit der Liste erfüllt werden, damit der Job starten kann.

Mehr zum Mode finden Sie im Kapitel 13.5.6.

State Hierbei handelt es sich um den aktuellen State der Abhängigkeitsbeziehung. Es gibt folgende Ausprägungen:

1. OPEN

   Der Required Job hat noch keinen Final State erreicht (ist noch nicht gelaufen, läuft noch, ist auf Fehler gelaufen). Die Abhängigkeit ist noch nicht erfüllt, kann aber durch Erreichen des richtigen Final States erfüllt werden.

2. FULFILLED

   Der Required Job ist beendet und hat den notwendigen Final State erreicht. Die Abhängigkeit ist erfüllt worden.

3. FAILED

   Der Required Job ist beendet und hat einen falschen Final State erreicht. Die Abhängigkeit kann nicht mehr erfüllt werden. Im Dependency Mode ALL kann der aktuelle Job nur durch Ignorieren der Dependency zum Laufen gebracht werden. Im Dependency Mode ANY muss mindestens eine der anderen Abhängigkeiten erfüllt werden.

20.5.6 Tab Dependents

Im Tab "Dependents" werden alle Abhängigkeiten von anderen Jobs zum aktuellen Jobs angezeigt. Der Tab sieht folgendermaßen aus:

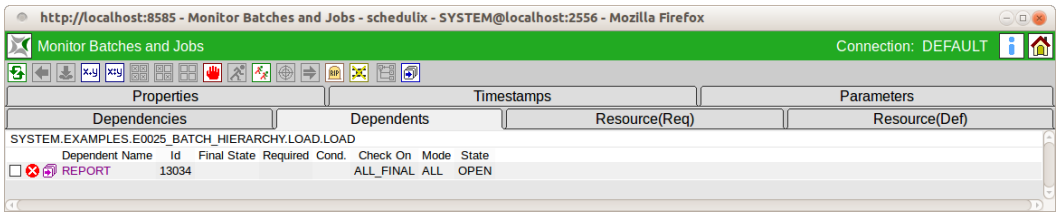


Abbildung 20.16: Batch und Job Dependents Tab

Der Dialog gibt eine Übersicht über alle bestehenden Abhängigkeiten, ihren aktuellen State und die Möglichkeit Abhängigkeiten bewusst zu ignorieren.

Folgende Buttons sind auf der Maske sichtbar:

 Ignore Dependencies

Der Button *Ignore Dependencies* wird verwendet, um eine oder mehrere Abhängigkeiten bewusst zu ignorieren. Alle zu ignorierenden Abhängigkeiten müssen vorher in der Maske ausgewählt werden. Anschließend wird der Button gedrückt. Nun erscheint ein neues Fenster zur Auswahl des Ignore Modus und zur Eingabe eines Vermerks. Das Fenster sieht folgendermaßen aus:

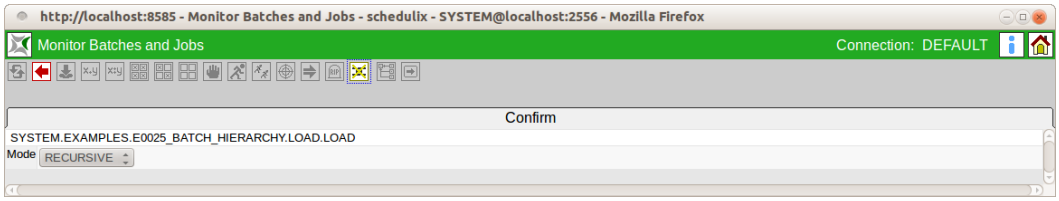


Abbildung 20.17: Batch und Job Ignore Dependencies

Auf dieser Maske muss der Modus des Ignore eingegeben werden. Per Default erben alle Children eines Submitted Entities dessen Abhängigkeiten: Wird B von A benötigt, so ist auch jedes Child von A (CA) von B abhängig. Mittels des Modus des Ignore ist nun das Verhalten der vererbten Abhängigkeit bei den Children bestimmbar.

Es gibt folgende Optionen:

1. Recursive

   Dies ist das Default-Verhalten.

   Das Ignore bezieht sich nicht nur auf den abhängigen Job, sondern auch auf die vererbten Abhängigkeiten, die Children des abhängigen Jobs zum aktuellen Job haben. Das heißt, der Parent Job ignoriert diese Abhängigkeit und alle seine Children ignorieren diese vererbte Abhängigkeit ebenfalls. Sollte ein Child eine explizit angegebene Abhängigkeit, welche nicht vererbt wurde, zum benötigten Job haben, so wird diese nicht ignoriert, sondern beachtet.

2. Job Only

   Nur der abhängige Job selbst ignoriert die Abhängigkeit. Alle Children beachten die vererbte Abhängigkeit und warten auf den aktuellen Job. Damit ist es möglich, den Parent Job laufen, seine Children allerdings warten zu lassen.

Nach der Eingabe des Ignore Modus und der Eingabe eines Kommentars im Kommentarfeld, kann durch ein erneutes Betätigen des *Ingore Dependency* Buttons die Aktion durchgeführt werden.
Die Liste des Tabs zeigt alle Dependencies, welche zum Starten anderer Jobs erfüllt werden müssen, und ihren aktuellen State an. Innerhalb der Liste kann eine Selektion stattfinden, die mittels des Buttons *Ignore Dependencies* zum Ignorieren der Abhängigkeiten benutzt werden kann.
Folgende Felder sind in der Liste sichtbar:

Icon-Feld Über das *Icon*-Feld ist der aktuelle State der Abhängigkeit ersichtlich. Es gibt folgende Möglichkeiten:

Die Abhängigkeit wurde noch nicht erfüllt oder kann nicht mehr erfüllt werden. Sollte bei einer Abhängigkeit mit dem Dependency Mode ALL dieses Icon zu sehen sein, kann durch diese Abhängigkeit der abhängige Job nicht ausgeführt werden.
Im Dependency Mode ANY muss mindestens irgendeine andere, hier nicht sichtbare Abhängigkeit des abhängigen Jobs nicht rot sein, damit der abhängige Job laufen kann.

Die Abhängigkeit wurde erfüllt. Im Dependency Mode ANY muss mindestens eine Abhängigkeit den State grün haben, damit der abhängige Job laufen kann. Im Dependency Mode ALL müssen evtl. auch andere, hier nicht sichtbare Abhängigkeit des abhängigen Jobs erfüllt sein, damit der abhängige Job laufen kann.

Dependent Name Hierbei handelt es sich um den Namen des Jobs, welcher davon abhängig ist, dass der aktuelle Job gelaufen ist. Durch Anklicken des Namens, kann zu seiner Laufzeitdefinition gewechselt und dessen State überprüft werden.

Id In diesem Feld wir die Laufzeit-Id des abhängigen Jobs angezeigt.

Final State Wurde der abhängige Job bereits abgearbeitet, steht hier der finale State des Jobs.

Required In diesem Feld wird der benötigte Final State des aktuellen Jobs angezeigt.

Cond In diesem Feld wird angezeigt, ob eine zusätzliche Bedingung für die Abhängigkeit definiert ist.

Check On Das Feld *Check On* beschreibt das Verhältnis der Beendigung des aktuellen Jobs. Es gibt folgende Ausprägungen:

1. All Final

   Der aktuelle Job und alle sein Children müssen Final sein.

2. Job Only

   Nur der aktuelle Job selbst muss Final sein, die Children werden nicht beachtet.

Mehr zum Thema Check On finden Sie im Kapitel 13.5.6.1.

Mode Der *Mode* legt fest, wie die Liste der Jobs zu beachten ist. Es gibt folgende Ausprägungen:

1. ALL

   Es müssen alle Abhängigkeiten des abhängigen Jobs erfüllt werden, damit dieser starten kann.

2. ANY

   Es muss mindestens eine Abhängigkeit des abhängigen Jobs erfüllt werden, damit dieser starten kann.

Mehr zum Mode finden Sie im Kapitel 13.5.6.

State Hierbei handelt es sich um den aktuellen State der Abhängigkeitsbeziehung. Es gibt folgende Ausprägungen:

1. OPEN

   Der aktuelle Job hat noch keinen Final State erreicht (ist noch nicht gelaufen, läuft noch, ist auf Fehler gelaufen). Die Abhängigkeit ist noch nicht erfüllt, kann aber durch Erreichen des richtigen Final States erfüllt werden.

2. FULFILLED

   Der aktuelle Job ist beendet und hat den notwendigen Final State erreicht. Die Abhängigkeit ist erfüllt worden.

3. FAILED

   Der aktuelle Job ist beendet und hat einen falschen Final State erreicht. Die Abhängigkeit kann nicht mehr erfüllt werden. Im Dependency Mode ALL kann der abhängige Job nur durch Ignorieren der Dependency zum Laufen gebracht werden. Im Dependency Mode ANY muss mindestens eine andere, hier nicht sichtbare Abhängigkeit des abhängigen Jobs erfüllt werden.

20.5.7 Tab Resource(Req)

Im Tab "Resource(Req)" werden alle Informationen angezeigt, welche über den aktuellen Zustand der angeforderten Resources Auskunft geben.
Der Tab sieht folgendermaßen aus:

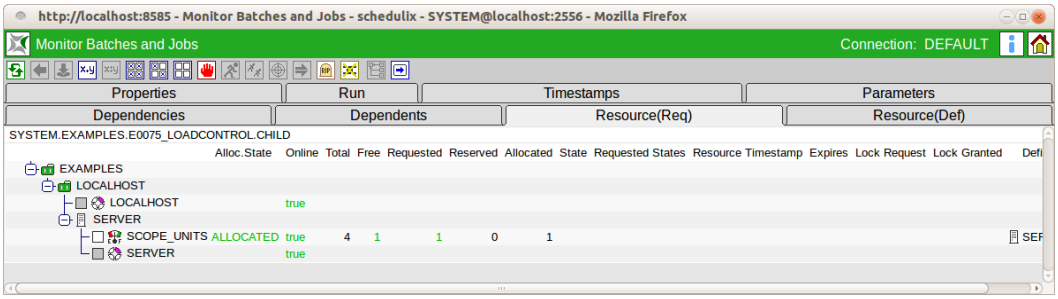


Abbildung 20.18: Job Resource Requirements

Im Tab ist es möglich, alle Resource-Instanzen und deren Zuordnung zu Scopes zu betrachten. Des Weiteren besteht die Möglichkeit Anforderungen an Resources zu ignorieren.
Der Button *Ignore Resources* hat folgende Bedeutung:

 Ignore Resources

Der Button *Ignore Resources* wird verwendet, um eine oder mehrere Resource-Anforderungen bewusst zu ignorieren. Alle zu ignorierenden Resource-Anforderungen müssen vorher in der Maske ausgewählt werden. Anschließend wird der Button gedrückt. Nun erscheint eine neue Maske zur Eingabe eines Vermerks. Die Maske sieht folgendermaßen aus:

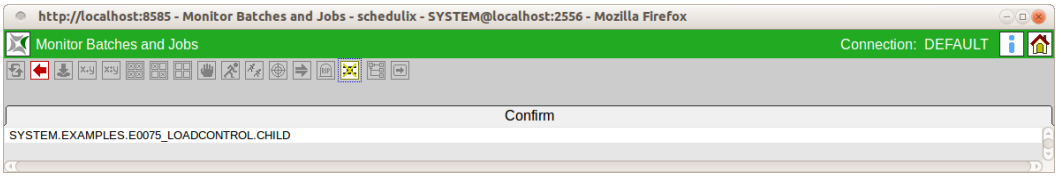


Abbildung 20.19: Job Ignore Resource Requirement

Nach der Eingabe eines Kommentars im Kommentarfeld wird durch ein erneutes Betätigen des *Ignore Resource* Buttons die Aktion durchgeführt.
In der Liste Resources werden alle angeforderten Resources des Jobs angezeigt. Die Anzeige erfolgt hierarchisch mittels der Scopes und Jobserver, in dem sich die angeforderte Resource befindet. Folgende Spalten sind in der Liste sichtbar:
In der ersten Spalte der Liste werden hierarchisch (sollten sich die Resources innerhalb von Scopes und Jobservern befinden) alle angeforderten Resources mit ihren

Namen angezeigt.
Durch einen vorangestellten Auswahlknopf ist es möglich, die Resources für ein Ignorieren mittels des Buttons *Ignore Dependencies* zu wählen. Ist das vorangestellte Feld grau, ist dies nicht möglich (bei statischen Resources).
Durch das Anklicken des Resource-Names kann zum Dialog 12.4.2.3 gesprungen werden. Hier ist eine Untersuchung der aktuellen Belegung der Resource möglich.

Allocation State Beim Allocation State handelt es sich um den aktuellen State der Resource-Anforderung. Es gibt folgende Ausprägungen:

1. Keine

   Hierbei handelt es sich um statische oder System Resources, die zum Submit-Zeitpunkt geprüft werden. Sollten diese nicht mehr vorhanden oder offline sein, erfolgt eine Fehlermeldung zur Submit-Zeit.

2. Blocked

   Die Resource wird von einem anderen Job benutzt und kann nicht belegt werden. Der Job wartet nun auf diese Resource. Es kann verschiedene Gründe für eine Blockierung geben. Die Kriterien die eine Belegung verhindern werden in der Liste rot angezeigt.

3. Allocated

   Die Resource wurde vom aktuellen Job belegt und die Anforderung ist damit erfüllt.

4. Available

   Die Resource ist verfügbar, aber noch nicht belegt.

20.5.8 Tab Resource(Def)

Im Tab "Resource(Def)" werden alle Informationen angezeigt, welche über den aktuellen Zustand der von diesem Submitted Entity definierten Resources Auskunft geben.
Der Tab sieht folgendermaßen aus:

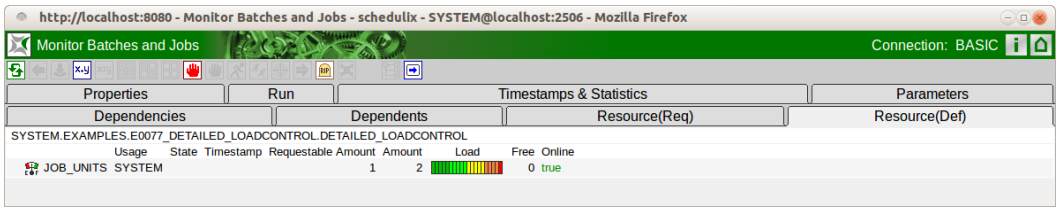


Abbildung 20.20: Batch und Job Defined Resources

In der Liste Resources werden alle definierten Resources des Jobs angezeigt. Folgende Spalten sind in der Liste sichtbar:
In der ersten Spalte der Liste werden alle definierten Resources mit ihren Namen angezeigt.
Durch das Anklicken des Resource-Namens kann zum Dialog 12.4.2.3 gesprungen werden. Hier ist eine Untersuchung der aktuellen Belegung der Resource möglich.

Usage Die *Usage* gibt an, um welchen Typ "Resource" es sich handelt. Mehr zu Typen von Resources finden Sie im Dialog Named Resource.

State Das Feld *State* zeigt den aktuellen Status der Resource an, wenn die Usage der Resource "Synchronizing" ist und ein Resource State Profile zugeordnet wurde.

Timestamp Der *Timestamp* gibt die Zeit des letzten Statuswechsels einer Resource an, wenn die Usage der Resource "Synchronizing" ist und ein Resource State Profile zugeordnet wurde.

Requestable Amount Wenn die Usage der Resource "Synchronizing" oder "System" ist, gibt das *Requestable Amount* die Menge an, die von einem Job maximal angefordert werden kann.

Amount Wenn die Usage der Resource "Synchronizing" oder "System" ist, gibt das *Amount* die Menge an, die für alle Jobs insgesamt zur Verfügung steht.

Load Wenn die Usage der Resource "Synchronizing" oder "System" ist, zeigt das *Load* die Belegung der Resource graphisch an.

Free Wenn die Usage der Resource "Synchronizing" oder "System" ist, gibt das *Free* die nicht belegte Menge an, die für alle Jobs noch zur Verfügung steht.

Online Zeigt an, ob die Resource "online" ist, also aktuell angefordert werden kann.

20.5.9 Tab Parameters

Im Tab "Parameters" werden alle aktuell in diesem Job definierten Ein- und Ausgangsparameter und ihre aktuellen Werte angezeigt. Der Tab sieht folgendermaßen aus:

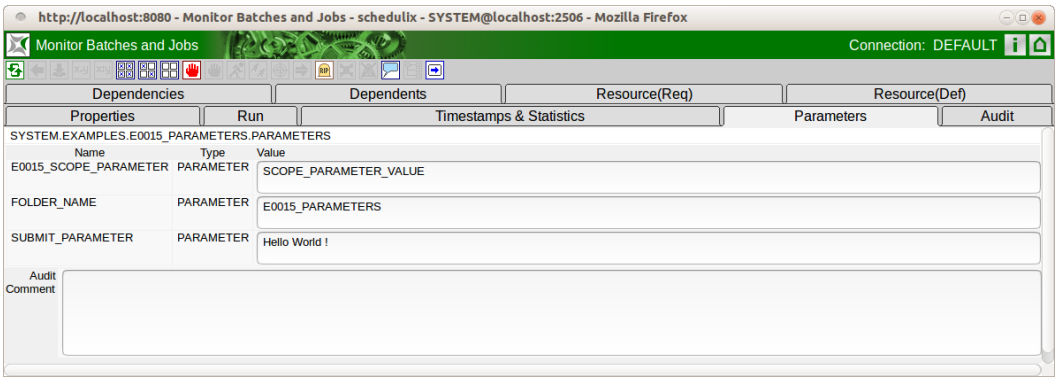


Abbildung 20.21: Batch und Job Parameters

In der Liste "Parameter" werden alle Parameter angezeigt, welche aktuell im Job definiert sind.
Ist der Benutzer in der Gruppe ADMIN oder hat OPERATE-Privilegien auf den Job oder Batch, können Parameter vom Typ PARAMETER und RESULT editiert werden. In diesem Fall wird auch das Feld *Audit Comment* angezeigt, in dem eine Begründung der Änderung angegeben werden kann, welche im Audit Trail festgehalten werden.
Informationen zu Parameter finden Sie im Kapitel 13.5.10.

21 Search Running Jobs

21.1 Bild

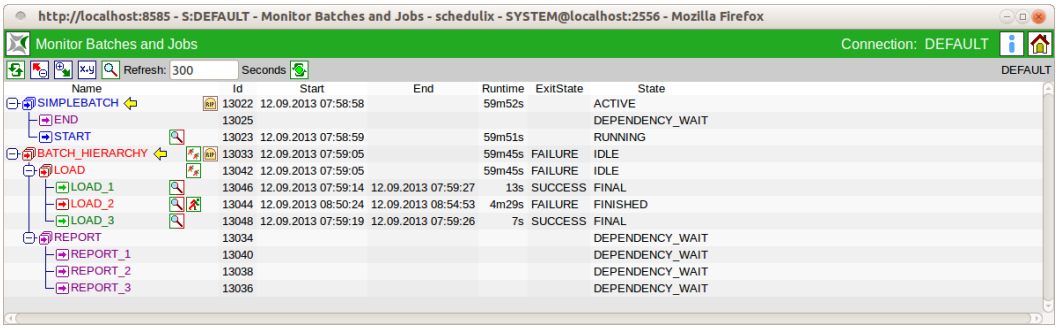


Abbildung 21.1: Search Running Jobs

21.2 Konzept

21.2.1 Kurzbeschreibung

Im Dialog *Search Running Jobs* können alle aktuell laufenden oder beendeten Jobs, Batches und Milestones gesucht und angezeigt werden.

21.2.2 Ausführliche Beschreibung

In dieser Konsole ist es möglich, alle Jobs zu überprüfen, fehlerhafte neu zu starten, Resources zu ignorieren etc.

Auch die Anzeige Search Running Jobs ist nur eine ”Default”-Ansicht (die des Bookmarks ”Default Search System”) und kann dementsprechend ebenfalls vom Benutzer angepasst und verändert werden. Wir beschreiben hier die ”Default”-Implementierung.

21.3 Navigator

Der Navigationsbildschirm ist analog zur "Detail" Maske des Running Master Jobs. Weitere Informationen finden Sie dort.

21.4 Detail Navigator Query-Maske

Nach dem Aufruf des Dialoges erscheint im Gegensatz zum Running Master Jobs Dialog zuerst die Query-Maske.

Die Query-Maske sieht folgendermaßen aus:

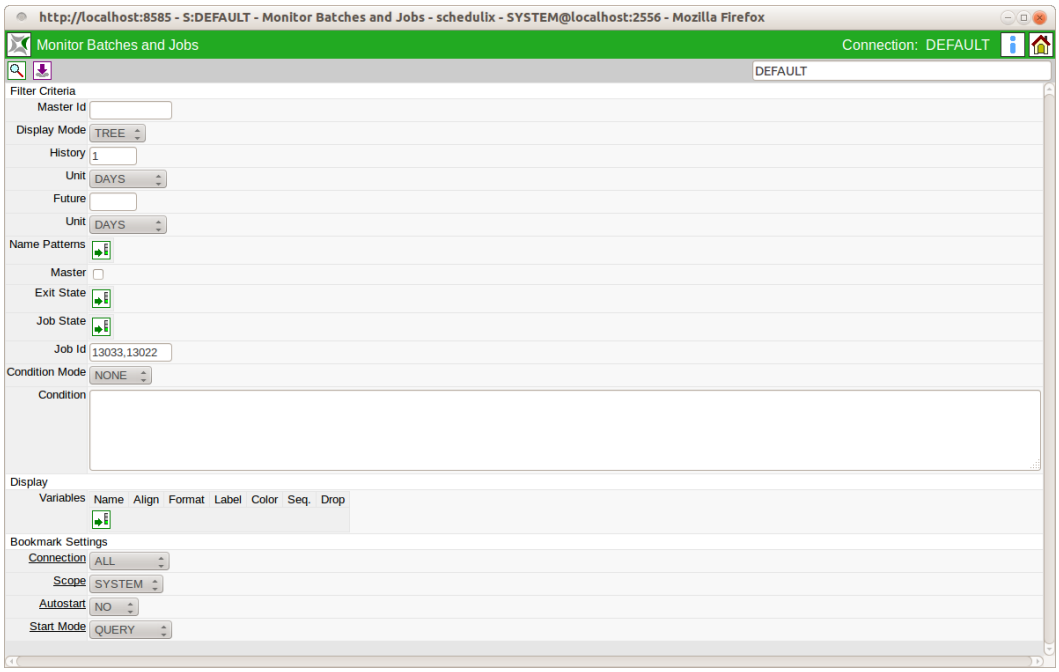


Abbildung 21.2: Search Running Jobs Query-Maske

Die Query-Maske für die Detail Navigation entspricht der Query-Maske für Running Master Jobs.

Zusätzlich sind folgende Felder definiert:

Master Id Im Feld *Master Id* kann eine Laufzeit-Id eines Master Jobs eingetragen werden. Alle gefundenen Jobs müssen von dieser Master Id submitted worden sein.

Display Mode Im Feld *Display Mode* kann die Listendarstellung eingestellt werden. Es gibt folgende Möglichkeiten:

1. LIST

   Die Ausgabe wird als Liste angezeigt.

2. TREE

   Die Ausgabe erfolgt hierarchisch nach der Parent-Child-Hierarchie. Es ist möglich, in einenn Tree zu expandieren und ihn wieder zu schließen.

Master Beim Feld *Master* handelt es sich um einen Filter, wenn der Schalter gesetzt wurde, mit dem nur Master Submittable Jobs angezeigt werden sollen.

Liste Job State In der Liste *Job State* kann nach Jobs gefiltert werden, welche den eingetragenen Job State besitzen. Werden keine Job States eingetragen, so werden diese bei der Suche nicht berücksichtigt.

Job ID In diesem Feld kann eine konkrete *Job Id* eingetragen werden, nach der gesucht werden soll. Als Ergebnis kann immer nur maximal ein Job gefunden werden.

22 Kalender

22.1 Bild

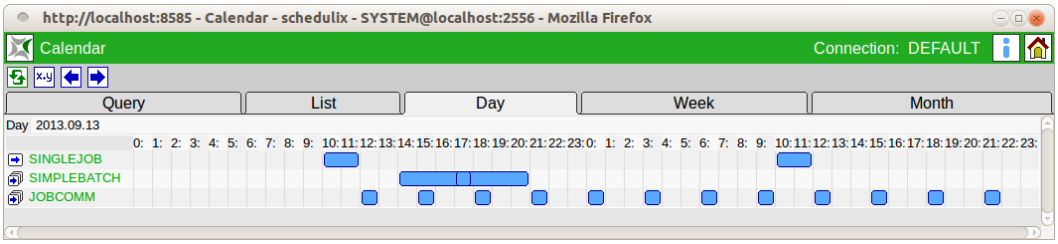


Abbildung 22.1: Kalender

22.2 Konzept

22.2.1 Kurzbeschreibung

Die Kalenderfunktion bietet eine Übersicht über die mittels des Time Scheduling Moduls geplante Ausführungen von Jobs und Batches.
Damit ein Job oder Batch in dieser Übersicht aufgenommen wird, muss beim Anlegen eines Schedules die Kalenderfunktion aktiviert werden (Calendar = active).

22.3 Ausführliche Beschreibung

Der Kalender wird auf unterschiedliche Weise dargestellt. Es gibt zum einen die einfache Liste, chronologisch sortiert, auf der die anstehenden Submits aufgeführt werden. Um die Liste nicht unnötig lang, und damit unübersichtlich zu machen, kann die Auswertungsperiode geändert werden. Standardmäßig werden die nächsten 24 Stunden angezeigt.
Die zweite Darstellung ist die Tagesdarstellung. Dabei wird der Tag als Periode von 24 Stunden dargestellt. Balken geben an, wann welcher Job gestartet werden soll, und, so weit diese Information vom Benutzer gepflegt wurde, wie lange er dauert.
Die dritte und vierte Darstellung entspricht der Tagesdarstellung, zeigt die Periode aber mit niedriger Granularität. Die Wochendarstellung zeigt sieben Tagen,

aufgeteilt in jeweils 6 Stunden-Intervalle. Die Monatsdarstellung zeigt (maximal) 31 Tage.
In der Wochen- und Monatsdarstellung werden die auszuführenden Jobs bzw. Batches ebenfalls mit Balken dargestellt. Allerdings ist die Laufzeit der Objekte weniger deutlich sichtbar als in der Tagesdarstellung, da die Balken mindestens so lang sind wie die kleinste Einheit in der jeweiligen Ansicht.
So würde ein Batch, der eine erwartete Laufzeit von acht Stunden hat, in der Tagesansicht auch tatsächlich mit 8 Stunden angezeigt. In der Wochenansicht werden zwei Felder, also 12 Stunden angezeigt. In der Monatsansicht wird ein ganzer Tag belegt.

22.3.1 Tab Query

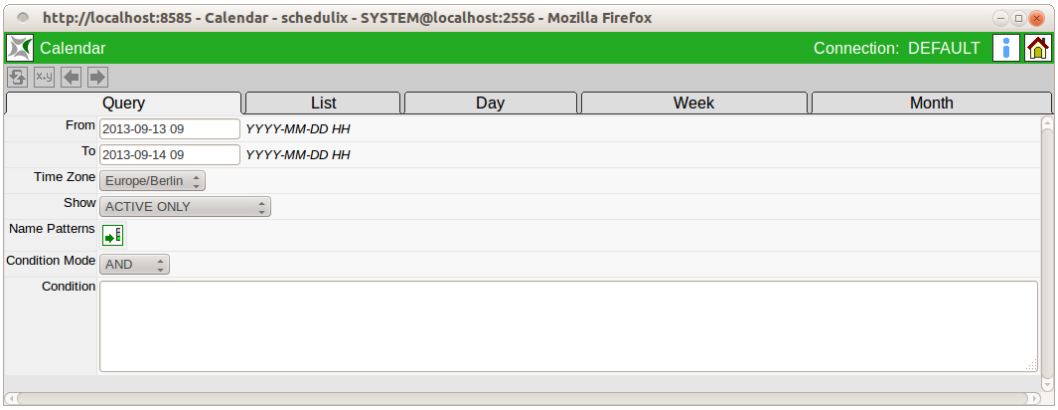


Abbildung 22.2: Kalender Query Tab

Auf dem Tab "Query" können Suchkriterien für die anzuzeigenden Job Definitions spezifiziert werden. Alle weitere Tabs sind von diesen Einschränkungen betroffen.
Die Felder *From* und *To* (Zeile) definieren die Periode für die Startzeiten der anzuzeigenden Jobs und Batches.
Das Feld *Time Zone* erlaubt die Auswahl der Zeitzone, in der der Kalender dargestellt werden soll.
In das Feld *Condition* können einfache Bedingungen verknüpft mit Booleschen Operatoren eingegeben werden. In den einfachen Bedingungen können Parameter auf bestimmte Inhalte getestet werden. Die Parameter werden mit **JOB.***parametername* adressiert. Folgendes Beispiel zeigt eine (syntaktisch) gültige Bedingung:

```
job.developer == 'ronald' or
job.developer == 'dieter'
```

22.3.2 Tab List

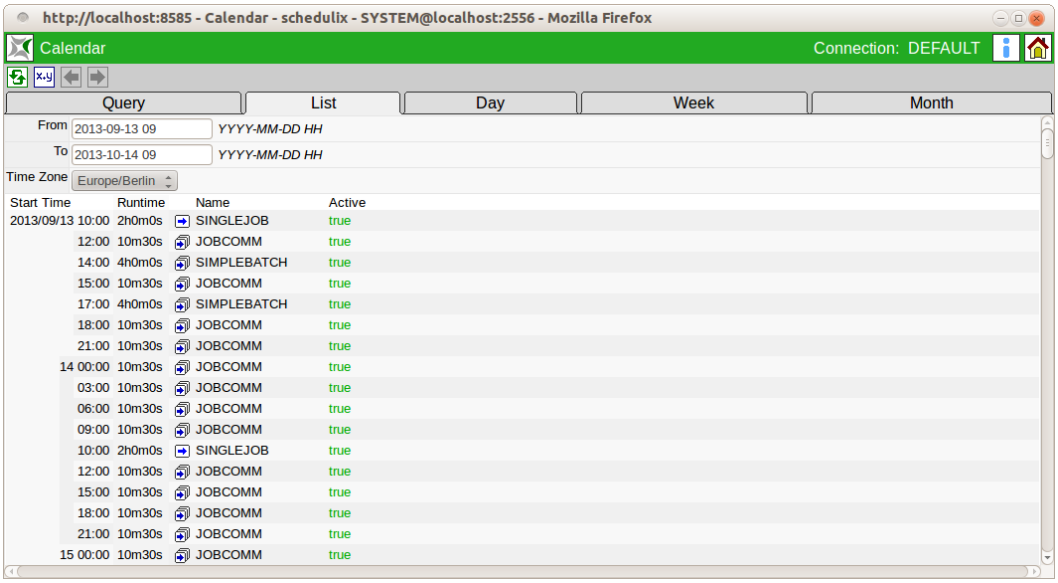


Abbildung 22.3: Kalender Tab List

Auf dem Tab "List" werden, für die in den Feldern *From* und *To* eingestellte Periode, alle Startzeiten der eingeplanten Jobs und Batches in einer einfachen Liste angezeigt.

Das Feld *Time Zone* erlaubt die Auswahl der Zeitzone, in der der Kalender dargestellt werden soll.

In der Spalte *Start Time* steht die geplante Startzeit. Um die Übersichtlichkeit zu erhöhen, werden in dieser Spalte und Zeile nur die Daten angezeigt, die sich im Vergleich mit der vorherigen Zeile geändert haben. Wenn sich zwei aufeinanderfolgenden Zeilen auf den gleichen Tag und Stunde beziehen, wird in der zweiten Zeile nur der Minutenteil des Zeitpunktes angezeigt.

22.3.3 Tab Day

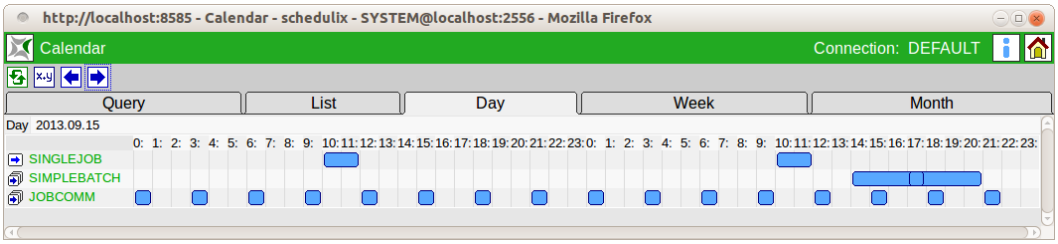


Abbildung 22.4: Kalender Tab Day

Auf dem Tab "Day" werden alle geplante Jobs und Batches für den selektierten Tag sowie den Tag darauf graphisch angezeigt.

Falls ein Job oder Batch am Anfang voraussichtlich noch aktiv ist, wird dies wie im Bild in der Zeile für "KEEP_RESOURCE" dargestellt. Auch die Darstellung überlappender Läufe ist in der Zeile sichtbar.

22.3.4 Tab Week

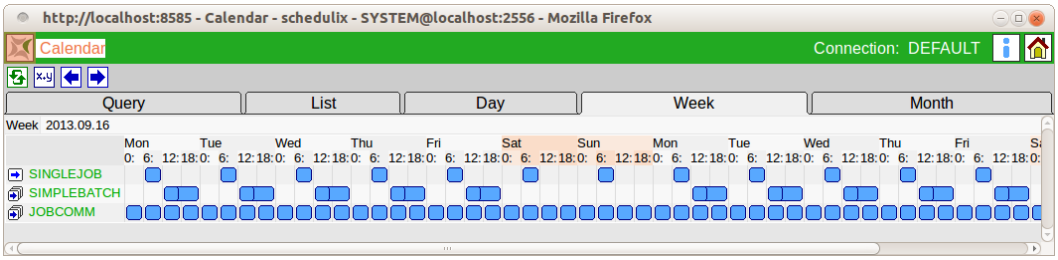


Abbildung 22.5: Kalender Tab Week

Auf dem Tab "Week" werden alle geplante Jobs und Batches für die selektierte Woche sowie die Woche darauf graphisch angezeigt.

22.3.5 Tab Month

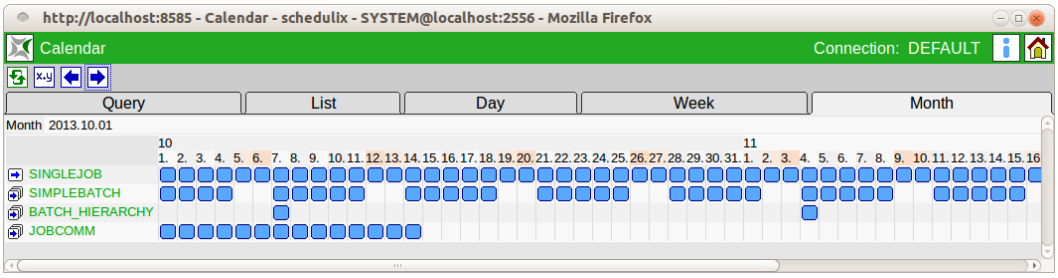


Abbildung 22.6: Kalender Tab Month

Auf dem Tab ”Month” werden alle geplante Jobs und Batches für den selektierten Monat sowie den Monat darauf graphisch angezeigt.

23 System Information

23.1 Bild

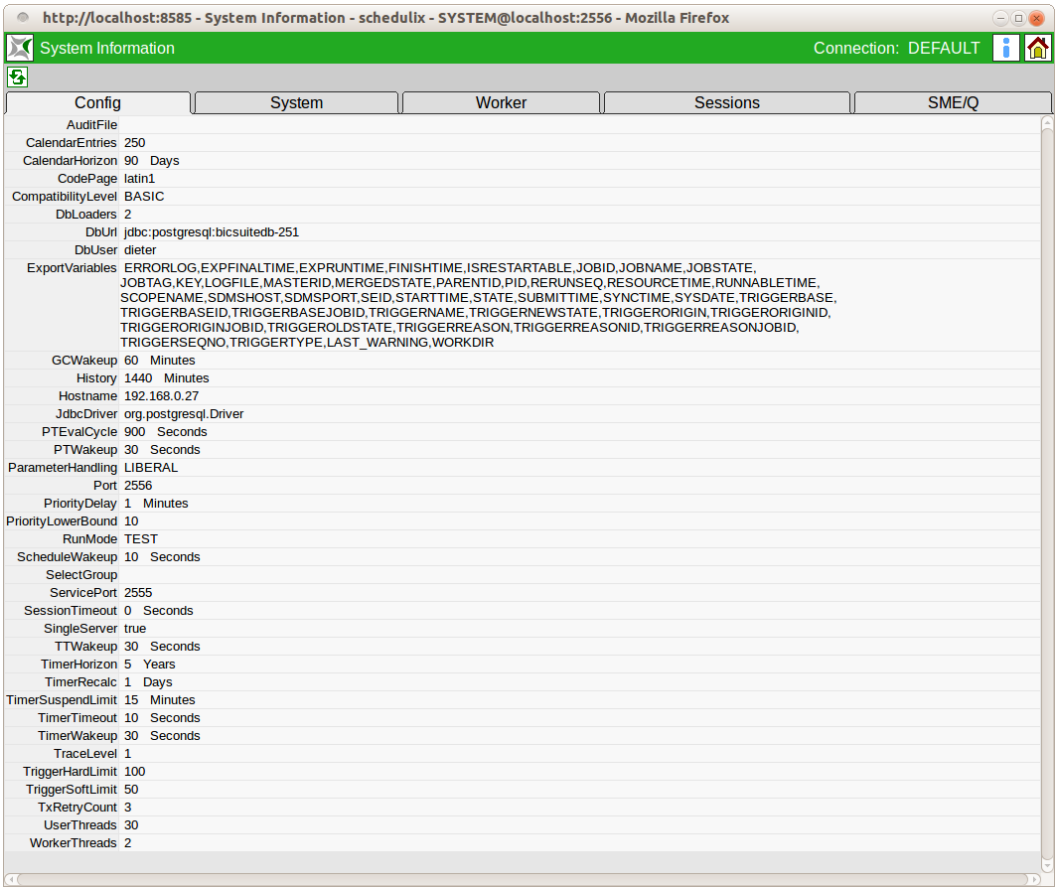


Abbildung 23.1: System Information

23.2 Konzept

23.2.1 Kurzbeschreibung

Der Menüpunkt ”System Information” im Hauptmenü liefert eine Übersicht der Server- und Systemparameter, eine Liste der Workeraktivitäten, sowie eine Liste der aktuellen Sessions.

23.2.2 Tab Config

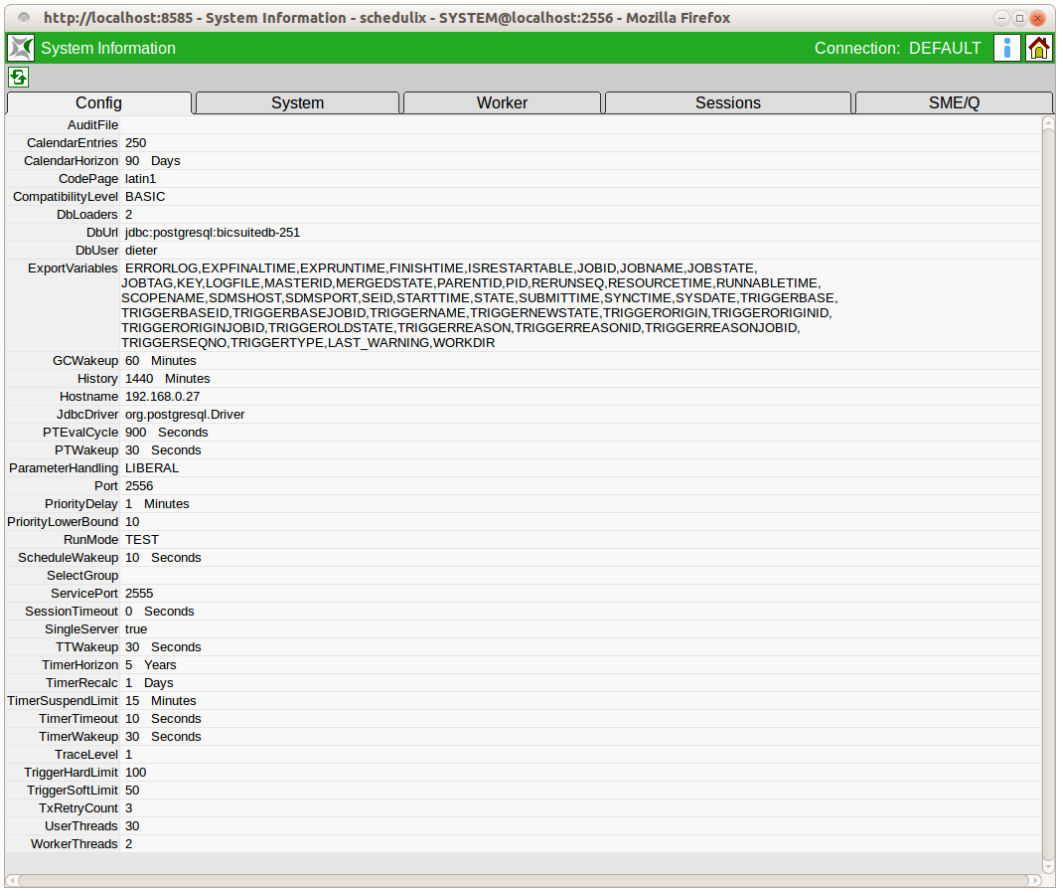


Abbildung 23.2: System Konfiguration

Auf dem Tab ”Config” wird die aktuelle Serverkonfiguration gezeigt. Dabei sind die Parameter in alphabetische Reihenfolge sortiert. Um Parameter zu ändern, muss die Datei `$BICSUITECONFIG/server.conf` angepasst und der Scheduling Server neu gestartet werden. Falls die Umgebungsvariable `BICSUITECONFIG` nicht gesetzt ist, wird als Konfigurationsverzeichnis `$BICSUITEHOME/etc` ver-

wendet.
Aus Sicherheitsgründen werden nicht alle Serverparameter dargestellt. Insbesondere Zugangsinformation für das Datenbanksystem wird nicht dargestellt.
Die derzeitige Serverparameter sind:

CalendarEntries

Der Parameter **CalendarEntries** definiert die maximale Anzahl Kalendereinträge pro Scheduled Event. Normalerweise wird der Parameter **CalendarHorizon** zuerst Wirkung zeigen, aber für den Fall, dass innerhalb des Horizonts sehr viele Einträge anfallen, verhindert dieser Parameter, dass unsinnig viele Kalendereinträge vorgenommen werden. Da der Horizont pro Schedule definiert werden kann, könnte damit eine Art Denial of Service-Angriff vorgenommen werden.
Falls der Parameter nicht definiert wird, gilt als Default-Wert 300 Einträge.

CalendarHorizon

Der Parameter **CalendarHorizon** definiert den Default-Horizont für Kalender.
Falls der Parameter nicht definiert wird, gilt als Default-Wert 62 Tage (immer mindestens zwei Monate).

CodePage

Der Parameter **CodePage** definiert den aktiven Zeichensatz. Der Parameter ist inzwischen obsolet.

CompatibilityLevel

Der Parameter **CompatibilityLevel** definiert, welche Optionen und Befehle vom Server als gültig erkannt werden. Mögliche Werte sind: **BASIC**, **PROFESSIONAL** und **ENTERPRISE**. Das maximale Level ist abhängig von der lizenzierten Version.
Per Default wird das Compatibility Level auf die lizenzierte Version gesetzt.

DbLoaders

Der Parameter **DbLoaders** bestimmt wie viele Threads beim Startup des Servers für das Laden des Objektspeichers gestartet werden. Abhängig von der Anzahl vorhandenen Prozessoren kann eine höhere Anzahl Threads zu einer Beschleunigung des Serverstarts führen.
Als Default wird die Anzahl vorhandener Prozessoren, jedoch maximal 5 genommen.

DbUrl

Der Parameter **DbUrl** definiert, welche Datenbank der Server nutzen soll. Die gültige Syntax ist abhängig vom eingesetzten Datenbanksystem.
Verständlicherweise gibt es für diesen Parameter keine Default-Belegung.

DbUser

Der Parameter **DbUser** gibt an, unter welchem Benutzernamen mit der Datenbank gearbeitet werden soll.
Auch für diesen Parameter gibt es kein Default.

ExportVariables

Der dargestellte Parameter **ExportVariables** ist eine Zusammensetzung zweier Konfigurationsparameter: **ExportVariables** und **UserExportVariables**. Die genannte Variablen werden beim Abholen eines Jobs als eine Liste von Key Value Pairs dem Jobserver übergeben. Dieser wiederum hat die Möglichkeit, die Variablen in die Umgebung des zu startenden Prozesses zu setzen. Ob der Jobserver dies tut, bzw. unter welchem Namen er die Variable in die Umgebung setzt, ist abhängig von der Konfiguration des Jobservers.
Diese Funktionalität wurde auf zwei Parameter aufgeteilt, da die Default-Belegung des Parameters **ExportVariables** die Liste der Standardvariablen ist. Um zu verhindern, dass der Administrator die gesamte Liste abschreiben muss, damit nur ein weiterer Parameter exportiert werden kann, gibt es die Möglichkeit, die zusätzlichen Variablen im Parameter **UserExportVariables** zu spezifizieren.

GCWakeup

In regelmäßigen Abständen werden "veraltete" Objekte aus dem Objekt-Cache entfernt. Der Parameter **GCWakeup** definiert, wie viel Zeit zwischen den Aufräumarbeiten liegen soll. Es ist nicht sinnvoll, den Parameter mit einem niedrigen Wert zu belegen, da der Aufwand nur abhängig ist von der Anzahl vorhandener Objekte (im Wesentlichen die Submitted Entities) und nicht von der Anzahl aufzuräumender Objekte.
Der Default für diesen Parameter ist 240 (Minuten).

History

Der Parameter **History** definiert die Zeit, für die Information über Master-Instanzen von Job- bzw. Batch-Definitionen behalten wird. Ältere Jobs bzw. Batches werden nur dann im Objekt-Cache behalten, wenn sie noch in irgendeiner Weise aktiv, also weder final noch cancelled, sind.
Per Default wird die Information der letzten 10 Tage, also 14400 Minuten behalten.

History Limit

Der Parameter **History Limit** definiert die Zeit, für die Information über nicht mehr aktive Master-Instanzen von Job- bzw. Batch-Definitionen maximal behalten werden. Dieses Limit gilt, wenn der Parameter **MinHistoryCount** ungleich 0 gesetzt ist. Wenn History auf 10 Tage, History Limit auf 30 Tage und MinHistoryCount auf 10 gesetzt ist, werden für einen wöchentlichen Job bzw. Batch nur die letzten 4 bis 5 Ausführungen im Objekt-Cache gehalten.
Per Default wird die Information der letzten 10 Tage, also 14400 Minuten behalten.

Hostname

Der Parameter **Hostname** ist der Name des Scheduling Servers. Dies ist auch der Inhalt des Standard Job Parameters **SDMSHOST**.
Per Default wird hier "localhost" gesetzt.

JdbcDriver

Der Parameter **JdbcDriver** definiert, welche Klasse als JDBC Driver verwendet wird. Der Name dieser Klasse ist abhängig vom eingesetzten Datenbanksystem. Diese Klasse muss im CLASSPATH auffindbar sein. Die CLASSPATH-Umgebungsvariable wird in die Datei `$BICSUITECONFIG/java.conf` gesetzt.
Es spricht für sich, dass für diesen Parameter kein Default-Wert vorgesehen ist.

MinHistoryCount

Der Parameter **MinHistoryCount** definiert die minimale Anzahl von Master-Instanzen einer Job- bzw. Batch-Definition, die im Objekt-Cache gehalten werden. Ein Job bzw. Batch, der nur selten zur Ausführung kommt, kann so deutlich länger, als durch den Parameter **History** definiert, im Objekt-Cache gehalten werden. Die maximale Zeit, die ein solcher Job bzw. Batch im Objekt-Cache gehalten wird, wird jedoch trotzdem durch den Parameter **History Limit** begrenzt.
Per Default ist der Parameter mit dem Wert 0 belegt und somit nicht aktiv.

MaxHistoryCount

Der Parameter **MaxHistoryCount** definiert die maximale Anzahl von Master-Instanzen einer Job- bzw. Batch-Definition, die im Objekt-Cache gehalten werden. Ein Job bzw. Batch, der sehr häufig zur Ausführung kommt, kann so deutlich kürzer als durch den Parameter **History** definiert im Objekt-Cache gehalten werden. Ein exzessiver Speicherverbrauch durch solche Jobs bzw. Batches kann dadurch vermieden werden.
Per Default ist der Parameter mit dem Wert 0 belegt und somit nicht aktiv.

ParameterHandling

Der Parameter **ParameterHandling** definiert, wie der Server reagiert, wenn Parameter angesprochen werden, die nicht deklariert wurden. Die mögliche Werte sind:

Wert	**B**edeutung
STRICT	Falls ein nicht deklarierter Parameter benutzt wird, wird eine Fehlermeldung ausgegeben.
WARN	Es wird eine Warnung ins Logfile des Servers geschrieben. Der Wert der Variablen wird (so weit gefunden) zurückgegeben.
LIBERAL	Der Wert des Parameters wird, so weit gefunden, zurückgegeben.

Tabelle 23.1: ParameterHandling Optionen

Aus Sicht der Software Engineering sollte **ParameterHandling** als STRICT definiert werden. Aus Sicht der Bequemlichkeit ist WARN oder LIBERAL vorzuziehen.
Die Default-Einstellung ist LIBERAL.

Port

Der Parameter **Port** definiert, welcher TCP-Port der Server zur Kommunikation mit den Clients benutzt. Die Wahl des Ports ist frei, soweit kein Port unter 1024 genutzt wird, da nur hochprivilegierte Benutzer dies können. Da der Scheduling Server so konzipiert ist, dass keine besonderen Privilegien für den Betrieb benötigt werden, sollte der Server daher auch keine besonderen Privilegien erhalten.
Der Default Port ist 2506.

PriorityDelay

Der Parameter **PriorityDelay** definiert, nach wie viel Zeit die (effektive) Priorität eines Jobs hochgestuft wird. Die automatische Erhöhung der Priorität verhindert, dass Jobs unendlich auf Resources warten.
Der Default ist 30 (Minuten).

PriorityLowerBound

Der Parameter **PriorityLowerBound** bestimmt die höchste Priorität, die durch normales Altern von Jobs erreicht werden kann. Ist der Parameter größer als 0, dann bleiben eine Anzahl Prioritätsstufen frei, die vom Administrator genutzt werden können, um bestimmte Jobs garantiert vorne in eine Warteschlange einzureihen.
Per Default ist der PriorityLowerBound auf 10 gesetzt.

RunMode

Der Parameter **RunMode** sollte in produktiven Umgebungen auf PRODUCTION gesetzt sein. In der Entwicklungsumgebung von independIT steht der Parameter gelegentlich auf TEST. Falls der RunMode auf TEST steht, ist ein Befehl "RUN TEST ..." freigeschaltet. Es ist nicht definiert, welche Auswirkungen die Benutzung dieses Befehls hat, da er für Testzwecke genutzt wird. Es ist sehr wohl möglich, dass die Benutzung des Befehls zu Serverabstürzen führt.
Der RunMode steht auf PRODUCTION soweit er nicht explizit auf TEST gesetzt wurde.

ScheduleWakeup

Der Parameter **ScheduleWakeup** definiert, nach wie viel Zeit der Resource Scheduling Thread aufwacht. Da der Resource Scheduler nahezu keine Zeit beansprucht, wenn es nichts zu tun gibt, kann die Zeit zwischen zwei Aktivierungen relativ kurz gehalten werden.
Der Default-Wert beträgt 30 (Sekunden).

SelectGroup

Der Parameter **SelectGroup** definiert, welche Gruppe das Select Statement benutzen darf. Ist dieser Parameter nicht gesetzt, oder enthält er einen Namen einer nicht existierenden Gruppe, dürfen nur Mitglieder der Gruppe ADMIN das Select Statement benutzen. Dieser Parameter existiert aus GrÃijnden der Backward Compatibility.

ServicePort

Der Parameter **ServicePort** ist der Port, über den (nur) ein Administrator sich mit dem Server verbinden kann. Dieser Port kann genutzt werden um in Störungsfällen doch noch einen, in der Hauptsache lesenden, Zugriff auf dem Server zu haben.
Der Default-Wert ist die 2505.

SessionTimeout

Der Parameter **SessionTimeout** definiert, wie lange eine Session per Default idle sein darf, bevor der Server die Session terminiert. Im Normalfall spielt dieser Parameter nur für interaktive Sessions mit `sdmsh` eine Rolle.
Der Default ist 30 (Sekunden).

SingleServer

Der Parameter **SingleServer** ist ein Boolean-Wert. Steht er auf "true", geht der Server davon aus, dass kein weiterer Server Zugriff auf das Repository haben kann.

Beim Hochfahren nach einem ”ungraceful shutdown” ignoriert er gesetzte Tickets und setzt sein eigenes Ticket. Sollte ein zweiter Server unerwarteterweise doch aktiv sein, wird dieser feststellen, dass das Repository jetzt einem anderen Server gehört, und die Arbeit einstellen. Wenn beide Server den Parameter auf 'true” haben, wird eine Art Ping-Pong-Effekt die Folge sein.
Die Default-Einstellung ist ”false”.

TTWakeup

Der Parameter **TTWakeup** definiert, wie häufig die Tickets im Repository mindestens erneuert werden. Werden mehrere Server eingesetzt, sollten die Parameter aufeinander abgestimmt sein (d.h. möglichst gleich).
Der Default ist 30 (Sekunden).

TimerHorizon

Der Parameter **TimerHorizon** definiert, wie weit der Timer Thread in die Zukunft schauen soll, um den nächsten Ausführungszeitpunkt eines Jobs zu finden.
Der Default ist 5 (Jahre).

TimerRecalc

Der Parameter **TimerRecalc** bestimmt, nach wie viel Zeit erneut versucht wird einen nächsten Ausführungszeitpunkt eines Jobs zu finden, nachdem die letzte Suche aufgrund des Erreichens des TimerHorizon abgebrochen wurde.
Der Default ist 5 (Tage).

TimerSuspendLimit

Der Parameter **TimerSuspendLimit** definiert die Zeit, nach der ein Job suspended submitted wird, falls der Server zur geplanten Submit-Zeit nicht aktiv war. Diese Grenze kann für jedes Scheduled Event individuell gesetzt werden. Wird dies jedoch unterlassen, gilt der Wert des TimerSuspendLimits.
Die Default-Zeit beträgt 15 (Minuten).

TimerWakeup

Der Parameter **TimerWakeup** bestimmt, in welchen Abständen der Timer Thread aktiviert wird. Ein hoher Wert führt zu Ungenauigkeiten bezüglich des tatsächlichen Submit-Zeitpunktes sowie potentiell lange Laufzeiten des Time Schedule-Vorganges. Ein niedriger Wert ist dagegen deutlich weniger schädlich.
Gute Erfahrungen wurden bis jetzt mit dem Default-Wert von 30 (Sekunden) gemacht.

TraceLevel

Der Parameter **TraceLevel** bestimmt, wie viel und welche Art Logging-Information der Server ins Logfile schreibt. Die untenstehende Tabelle zeigt, welche Art von Meldungen bei welchem Trace Level protokolliert werden:

Art	**T**race Level	**B**emerkung
FATAL	Alle	FATAL-Meldungen werden bei einem schweren Fehlerzustand im Server protokolliert und sollten unverzüglich beim Support-Team von independIT gemeldet werden.
ERROR	Alle	ERROR-Meldungen werden bei auftretenden Fehlern protokolliert. Diese Art von Fehler sind deutlich weniger kritisch als Fehler der Klasse FATAL. Es ist empfehlenswert diese Art von Fehler zu untersuchen und, wenn möglich, ihre Ursachen zu beseitigen.
INFO	Alle	INFO-Meldungen geben eine wichtige oder interessante Information aus. Beispiele für solche Meldungen sind die Meldungen, die während der Startup-Phase des Servers geschrieben werden.
WARNING	> 0	WARNING-Meldungen sind interessante Meldungen, die häufig mit Bedienungsungenauigkeiten zu tun haben. Ein Beispiel dafür sind die Warnungen, die auftreten, wenn der Parameter 'ParameterHandling' auf WARN oder STRICT gesetzt ist.
MESSAGE	> 1	MESSAGE-Meldungen sind relativ unwichtige Meldungen, die nichtsdestotrotz sehr interessant sein können. So werden zum Beispiel alle ausgeführten Befehle und ihre Ausführungszeit als MESSAGE protokolliert.
DEBUG	3	DEBUG-Meldungen sind Meldungen, die beim Troubleshooting hilfreich sein können. Allerdings führt Trace Level 3 zu einer gewaltigen Flut an Meldungen. Für den Normalbetrieb sind diese Meldungen wenig aussagekräftig und eher behindernd.

Tabelle 23.2: Übersicht der Trace Level

TriggerHardLimit

Der Parameter **TriggerHardLimit** legt die maximale Anzahl, die ein und derselbe Trigger feuern kann, fest.

Die Default-Einstellung ist 100.

TriggerSoftLimit

Wenn Trigger angelegt werden, ohne dass ein FireLimit spezifiziert wird, legt der Parameter **TriggerSoftLimit** die maximale Anzahl, die eine Instanz dieses Triggers feuern kann, fest.
Die Default-Einstellung ist 50.

TxRetryCount

In manchen Fällen kann es sein, dass die Durchführung einer Transaktion fehlschlägt, obwohl die Transaktion semantisch und syntaktisch korrekt ist. In diesem Fall wird der Server erneut versuchen die Transaktion durchzuführen. Wie oft dies passiert, wird durch den Parameter **TxRetryCount** definiert.
Default-mäßig werden maximal drei Versuche unternommen.

UserThreads

Der Parameter **UserThreads** bestimmt die maximale Anzahl Sessions, die gleichzeitig connected sein können. Benutzer die das Web-Frontend benutzen, müssen nicht in der vollen Anzahl mitgezählt werden, da der Zope-Server für jeden Kommunikationsschritt eine neue Verbindung mit dem Scheduling-Server aufbaut und wieder abbaut.
Die Jobserver dagegen bleiben im Normalfall mit dem Scheduling-Server verbunden und belegen jeweils einen Thread.
Dieser Parameter hat den Default-Wert 10.

WorkerThreads

Der Parameter **WorkerThreads** definiert wie viele Threads für die Befriedigung lesender Abfragen gestartet werden sollen. Im Gegensatz zu den schreibenden Transaktionen sind viele lesende Transaktionen sehr aufwendig. Dabei werden insbesondere die lesenden Transaktionen aus dem Monitoring-Bereich genannt.
Per Default werden zwei solche Threads gestartet. In größeren Umgebungen ist es sinnvoll, die Zahl zu erhöhen.

23.2.3 Tab System

Auf diesem Tab werden einige Laufzeitinformationen sowie Informationen über die Java Virtual Machine dargestellt. Abhängig von der eingesetzten Hardware und dem Betriebssystem bzw. der Java Virtual Machine können letztgenannte Informationen von den dargestellten Informationen abweichen. Es wird deshalb für die korrekte Interpretation dieser Informationen auf die jeweilige Dokumentation verwiesen.

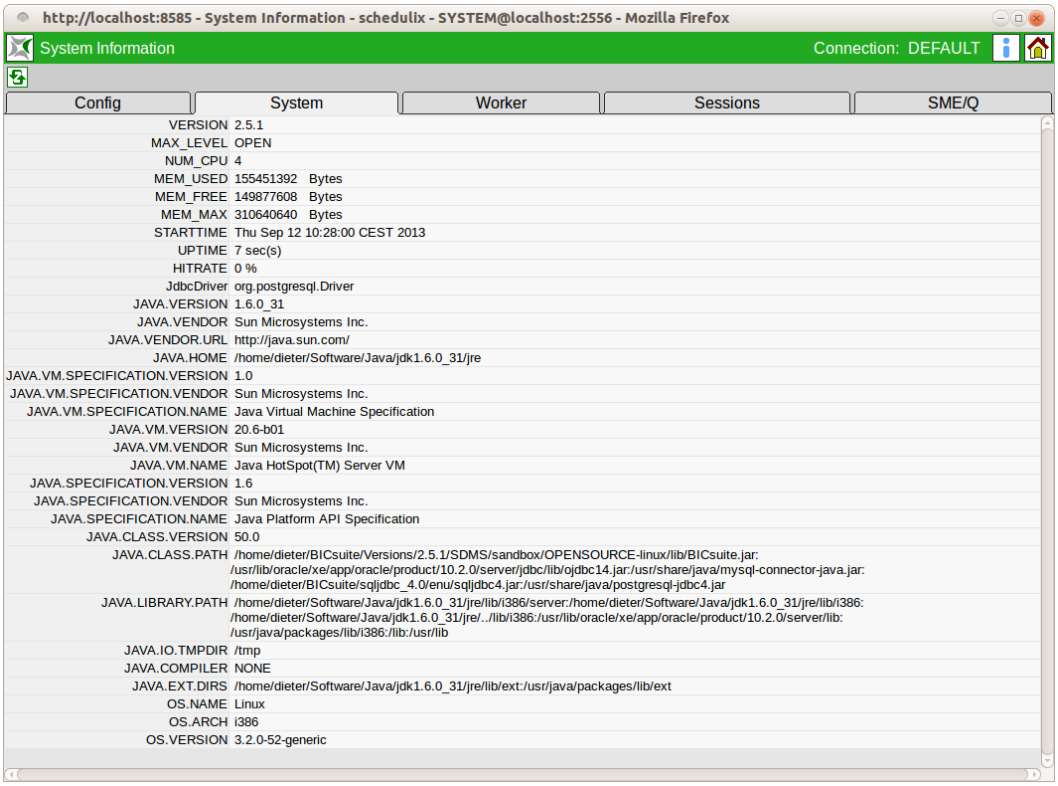


Abbildung 23.3: System Runtime Umgebung

VERSION

Das Feld *VERSION* zeigt die eingesetzte Software Version des Scheduling Servers.

MAX_LEVEL

Das Feld *MAX_LEVEL* zeigt die lizenzierte Version des Scheduling Servers.

NUM_CPU

Das Feld *NUM_CPU* zeigt die Anzahl von der Java Virtual Machine gemeldeten Prozessoren.

MEM_USED

Das Feld *MEM_USED* zeigt, wie viel Speicher momentan von der Virtual Machine belegt wird.

MEM_FREE

Das Feld *MEM_FREE* zeigt, wie viel des belegten Speichers noch für den Server zur Verfügung steht.

MEM_MAX

Das Feld *MEM_MAX* zeigt, wie viel Speicher die Virtual Machine maximal belegen darf. Falls *MEM_MAX* und *MEM_FREE* den gleichen Wert haben, bedeutet dies, dass außer dem unter MEM_FREE gemeldeten freien Platz, kein weiterer Speicher mehr zur Verfügung steht. An sich ist das kein Problem. Wenn jedoch die Menge MEM_FREE zuneige geht, wird die Virtual Machine sich immer häufiger um Garbage Collection kümmern müssen, was schwer auf die Performance des Systems lasten kann. In dem Fall ist ein Restart des Servers (mit einer eventuellen Anpassung der Speicherkonfiguration in `$BICSUITECONFIG/java.conf`) das kleinere Übel.

STARTTIME

Das Feld *STARTTIME* zeigt den Zeitpunkt des Starts des Servers.

UPTIME

Das Feld *UPTIME* dient nur der Bequemlichkeit und zeigt, wie lange der Server bereits läuft.

HITRATE

Das Feld *HITRATE* zeigt die Effektivität eines Caching-Algorithmus im Resource Scheduling Thread. Da dieser Cache nur im Fall eines kompletten Reschedule seine Wirkung zeigt, sind bereits relativ niedrige Werte eine Indikation für Effektivität.

JdbcDriver

Der Eintrag hinter dem Titel "JdbcDriver" wurde auf diesem Tab nur zur Vollständigkeit aufgenommen, da auch die sonstigen, Java betreffenden Konfigurationsparameter, hier vorhanden sind. Es handelt sich hier um den selben Wert wie auf dem Tab "Config".

Weitere Einträge

Alle weiteren Einträge sind abhängig von der eingesetzten Hardware, vom Betriebssystem und der Java Virtual Machine. Obwohl die Werte im Prinzip relativ leicht interpretierbar sind, wird für eine genaue Beschreibung der Bedeutung auf die Java-Dokumentation verwiesen.

23.2.4 Tab Worker

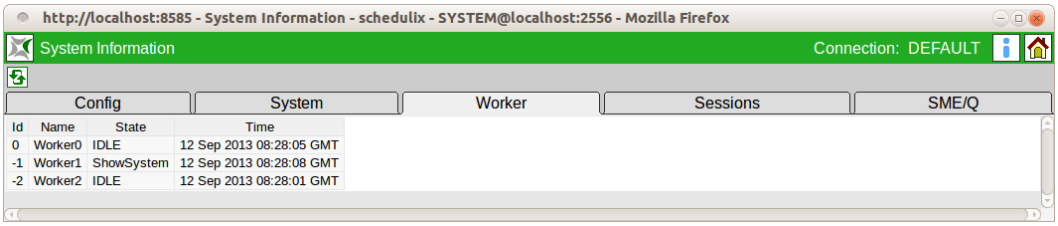


Abbildung 23.4: Worker Thread Information

Auf dem Tab "Worker" wird in einer Tabelle die Aktivität der Worker Threads gezeigt. Der Worker mit der Id 0 ist immer der Worker, der für die Verarbeitung der schreibenden Transaktionen zuständig ist. Alle weiteren Worker sind zuständig für die Verarbeitung der lesenden Transaktionen.
Sind regelmäßig alle Worker Threads tätig, ist das eine Indikation für eine zu geringe Menge Worker Threads und die Anzahl sollte erhöht werden.

Id

In der Spalte *Id* steht die interne Nummer des Workers.

Name

In der Spalte *Name* steht der Name des Workers.

State

In der Spalte *State* wird gezeigt, was der jeweilige Worker gerade macht. Hier steht entweder IDLE, falls der Worker nichts zu tun hat, oder der Name der Klasse des Objektes, welches er gerade verarbeitet. Im Normalfall lässt sich aus dem Namen der Klasse leicht ableiten, um welche Art Statement es sich handelt.

Time

In der Spalte *Time* steht die Startzeit des letzten ausgeführten Statements, bzw. des Statements, das gerade ausgeführt wird.

23.2.5 Tab Sessions

Auf dem Tab "Sessions" wird eine Tabelle aller momentan connected Sessions, sowie ihre Aktivität gezeigt.

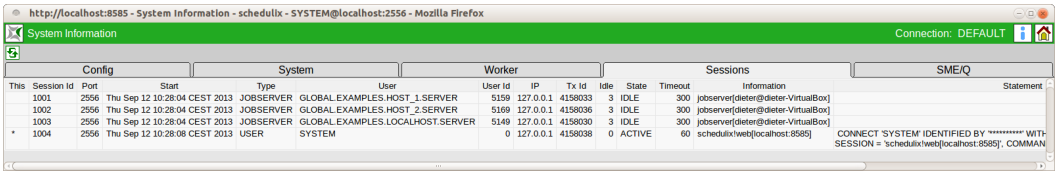


Abbildung 23.5: Session Information

This

Die Spalte *This* zeigt mit einem Asterisk (*), welche der gezeigten Sessions die eigene Session ist.

Session Id

In der Spalte *Session Id* steht die Nummer der Session. Die Nummer der Session wird benötigt, wenn man mit dem "KILL SESSION"-Befehl die Session terminieren möchte.

Port

In der Spalte *Port* steht, über welchen Port die Session mit dem Server connected ist.

Start

In der Spalte *Start* steht die Zeit, zu der die Session aufgebaut wurde.

Type

In der Spalte *Type* steht, um welche Art von Client es sich bei der Connection handelt. Die möglichen Werte sind USER, JOBSERVER oder JOB.

User

In der Spalte *User* steht der Name des Benutzers. Im Falle eines Users ist es sein Benutzername. Wenn es sich um einen Jobserver handelt, ist es der vollqualifizierte Name des Jobservers. Bei einem Job wird die Job Id gezeigt.

UserId

In der Spalte *UserId* steht die Objekt-Id des Benutzers, Jobservers oder Jobs.

IP

In der Spalte *IP* steht die IP-Adresse des Rechners, von dem die Connection aufgebaut wurde.

TxId

In der Spalte *TxId* steht die Nummer der zuletzt durchgeführte Transaktion. Wenn gerade ein Statement ausgeführt wird, ist es die Nummer der aktiven Transaktion.

Idle

In der Spalte *Idle* steht die Anzahl Sekunden, die nach der letzten Aktivität verstrichen sind. Idealerweise ist dieser Wert im Falle von Jobservern nicht höher als ihr NOP_DELAY.

State

In der Spalte *State* wird der Zustand der Session gezeigt.

Wert	**B**edeutung
IDLE	Die Session wartet auf Eingaben des Benutzers.
QUEUED	Die Session hat ein Statement in der Warteschlange zum Ausführen bereitstehen.
ACTIVE	Ein Statement der Session wird gerade ausgeführt.
COMMITTING	Ein Statement der Session ist in der Abschlussphase der Ausführung.
CONNECTED	Eine Session wurde initiiert, aber der Benutzer hat sich noch nicht authentifiziert.

Tabelle 23.3: Session Status

Timeout

In der Spalte *Timeout* steht, nach wie vielen Sekunden Idletime die Connection vom Server abgebrochen wird. Ein Wert von 0 bedeutet dabei, dass der Server die Connection nicht abbricht.

Statement

In der Spalte *Statement* steht das Statement, welches gerade von der Connection ausgeführt wird, soweit die Connection nicht IDLE ist. Es ist selbstverständlich, dass Passwörter in Connect Statements unerkennbar gemacht werden.

Allerdings ist das Statement nur dann sichtbar, wenn man über Administratorrechte verfügt.

23.2.6 Tab SME/Q

Auf dem Tab "SME/Q" wird eine Tabelle der je Quartal erzeugten Submitted Entities angezeigt. Der Tab "SME/Q" ist nur sichtbar, falls der Benutzer der Gruppe "ADMIN" angehört.

Year

Zeigt das Jahr an.

Quarter

Zeigt das Quartal an.

Total

Die Spalte *Total* enthält die Gesamtanzahl an Submitted Entities, welche in diesem Quartal erzeugt wurden.

Expected Total

Die Spalte *Expected Total* enthält im aktuellen, noch nicht abgeschlossen Quartal (1. Zeile der Tabelle), die hochgerechnete zu erwartende Anzahl zum Quartalsende. Diese Zahl ist unverbindlich und dient ausschließlich der Orientierung. Für alle vergangenen Quartale ist das Expected Total gleich dem Total.

Avg. SME / Day

Die Spalte *Avg. SME / Day* enthält die durchschnittlich pro Tag erzeugte Anzahl an Submitted Entities.

Index