

independIT Integrative Technologies GmbH
Bergstraße 6
D-86529 Schrobenhausen



schedulix!focus

Hierarchische Batch Strukturen

Dieter Stubler

Ronald Jeninga

November 25, 2016

Copyright © 2016 independIT GmbH

Rechtlicher Hinweis

Dieses Werk ist urheberrechtlich geschützt.

Alle Rechte, auch die der Übersetzung, des Nachdrucks und der Vervielfältigung des Artikels, oder Teilen daraus, vorbehalten. Kein Teil des Werkes darf ohne schriftliche Genehmigung der independIT GmbH in irgendeiner Form (Fotokopie, Mikrofilm oder ein anderes Verfahren), auch nicht für Zwecke der Unterrichtsgestaltung, reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.

Hierarchische Batch Strukturen

Einführung

Prozessabläufe in Data Warehouse Umgebungen bestehen oft aus einer Vielzahl von Arbeitsschritten, welche koordiniert und in der richtigen Reihenfolge zur Ausführung gebracht werden müssen.

Je feiner die einzelnen Arbeitsschritte in Teilprozesse zerlegt sind (Process Decomposition), umso besser lassen sich die Abläufe und deren Fortschritt überwachen und steuern. Beim Auftreten von Fehlern und der Wiederholung (Rerun) von Prozessen führt eine feine Granularität der Teilprozesse zu einer schnelleren Behebung von Fehlern und eine effizientere Nutzung der Hardwareressourcen durch Reduzierung von 'verlorener' Rechenzeit.

Die Zerlegung der Abläufe eines Data Warehouses in möglichst viele Einzelprozesse ist daher ein wichtiges Designziel für das Erreichen und Aufrechterhalten eines stabilen und zuverlässigen Data Warehouse Betriebes.

Verfolgt man dieses Ziel konsequent, entstehen schnell Abläufe mit mehreren hundert auszuführenden Teilprozessen.

Unterstützt das eingesetzte Scheduling System Abläufe mit derart vielen Teilprozessen nicht oder nur ungenügend, werden die Vorteile der Prozesszerlegung durch einen erhöhten Aufwand für Implementierung, Überwachung und Betrieb bezahlt. Dies setzt der Teilprozesszerlegung oft sehr frühe Grenzen.

Hierarchische Ordnung von Teilprozessen

Um Abläufe mit sehr vielen Teilprozessen einfach implementieren zu können und bei Überwachung und Betrieb solcher Abläufe nicht den Überblick zu verlieren, ermöglicht das schedulix Scheduling System die Gruppierung von Prozessen (Jobs) in hierarchischen Strukturen (Batches).

Abhängigkeiten können auf jeder dieser Hierarchieebenen definiert werden um eine möglichst einfache, übersichtliche und wartbare Realisierung auch komplexer Ablaufstrukturen zu ermöglichen. Dabei gilt, dass Abhängigkeiten von einem Subbatch erst als erfüllt betrachtet werden, wenn alle Jobs dieses Subbatches abgeschlossen sind. Ebenso 'erben' alle Jobs eines Subbatches alle Abhängigkeiten des übergeordneten Subbatches.

Beispiel: Ein einfacher Ablauf

Ein Ablauf soll 3 Reports (R1, R2 und R3) erstellen. Dazu müssen zuerst 3 Datenbanktabellen (T1, T2, T3) aktualisiert werden. Das Aktualisieren der Tabellen T1 und T2 kann parallel geschehen. T3 kann erst aktualisiert werden, wenn T1 aktualisiert wurde. Die Erstellung der einzelnen Reports kann parallel geschehen. Nach dem Erzeugen der Reports sollen diese in einem Webserververzeichnis publiziert werden.

Erzeugt man nun den Ablauf ohne hierarchische Ordnung so stellt sich dies wie in Abbildung 1 dar.

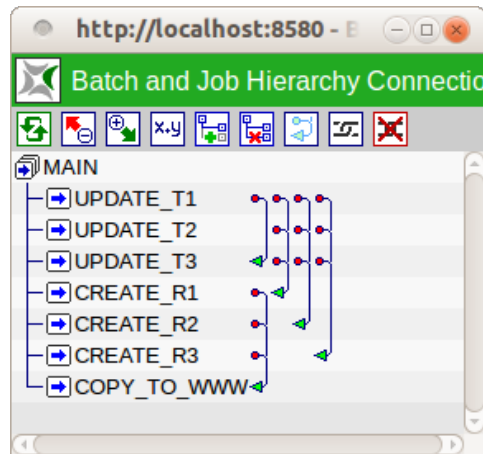


Figure 1: Batchstruktur ohne hierarchische Ordnung

Schon dieser vergleichsweise kleine Ablauf aus 7 Prozessen ist nicht wirklich übersichtlich und erfordert die Definition von 13(!) Abhängigkeitsbeziehungen. Versucht man ohne hierarchische Batch Struktur die Anzahl notwendiger Abhängigkeiten zu reduzieren, so kann dies durch Verkettung der einzelnen Jobs erreicht werden. Abbildung 2 zeigt diese Lösung. Das sieht zwar jetzt etwas übersichtlicher aus, birgt jedoch Probleme.

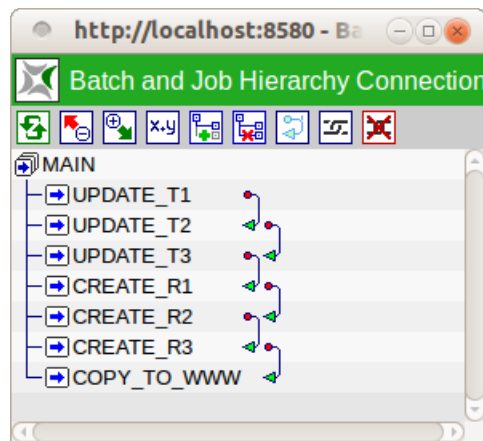


Figure 2: Batchstruktur als Kette

Die einzelnen Prozesse laufen jetzt nicht mehr parallel, sondern seriell hintereinander ab, was zu einer unnötig hohen Gesamtlauzeit des Ablaufes führt. Die Information über tatsächlich bestehende Abhängigkeiten geht verloren und die Wartbarkeit wird verringert bzw. Optimierungspotential geht verloren, da ein

späterer Entwickler nicht mehr entscheiden kann ob die serielle Abarbeitung fachlich notwendig oder aus technischen Gründen einfacher zu realisieren war. Durch den Einsatz hierarchischer Batch Strukturen kann dies deutlich vereinfacht werden. Abbildung 3 verdeutlicht dies.

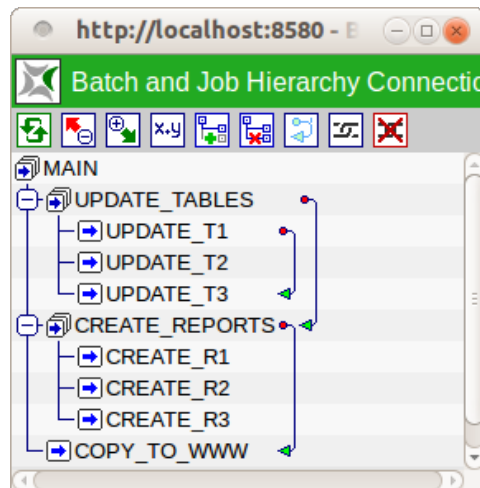


Figure 3: Batchstruktur mit hierarchischer Ordnung

Durch die Gruppierung der UPDATE... Jobs und der CREATE... Jobs in die (Sub)Batches UPDATE_TABLES und CREATE_REPORTS ist das ganze Ablaufkonstrukt deutlich überschaubarer geworden. Die Abhängigkeit des Batches CREATE_REPORTS vom Batch UPDATE_TABLES gilt erst als erfüllt, wenn alle Jobs in UPDATE_TABLES abgeschlossen sind. Das selbe gilt für die Abhängigkeit des Jobs COPY_TO_WWW vom Batch CREATE_REPORTS. Die Jobs CREATE_Rn 'erben' implizit die Abhängigkeit vom Batch UPDATE_TABLES von ihrem übergeordneten Batch CREATE_REPORTS. Daher sind nun nur noch 3 Abhängigkeitsbeziehungen nötig um die Jobs in der richtigen Reihenfolge abzuarbeiten.

Einfluß auf die Wartbarkeit

Die hierarchische Strukturierung erhöht die Wartbarkeit von Abläufen durch Reduzierung von Änderungsaufwand und Fehlerrisiko.

Beispiel: Einbinden eines zusätzlichen Prozesses

Zur Verdeutlichung soll in unserem Beispiel ein vierter Report R4 erzeugt werden und der dazu nötige Prozess in den Ablauf eingebunden werden.

Geschieht dies im Modell ohne hierarchische Ordnung, so müssen neben der Definition des Jobs CREATE_R4 und dessen Einhängen in den Batch MAIN noch 4(!) zusätzliche Abhängigkeiten definiert werden, damit auch CREATE_R4 auch auf die Jobs UPDATE_T1, UPDATE_T2 und UPDATE_T3 wartet und COPY_TO_WWW

jetzt auch auf den neuen Job CREATE_R4 wartet. Abbildung 4 zeigt die neue Struktur des Ablaufes.

Sehr leicht kann hier eine Abhängigkeit vergessen werden und es kommt zu fehlerhaften Ergebnissen des Data Warehouse Betriebes. Vergessen wir den COPY_TO_WWW Job vom neuen Job CREATE_R4 abhängig zu machen, führt dies dazu, dass falls CREATE_R4 einmal länger läuft als die anderen CREATE_... Jobs, der COPY_TO_WWW schon anläuft bevor der Report R4 erzeugt wurde. Es wird dann ein fehlerhafter oder veralteter Report ausgeliefert. Erschwerend kommt hinzu, dass dieser Fehler nicht immer auftritt, sondern abhängig ist vom Zeitverhalten der beteiligten Prozesse, was eine Fehlersuche erschwert.

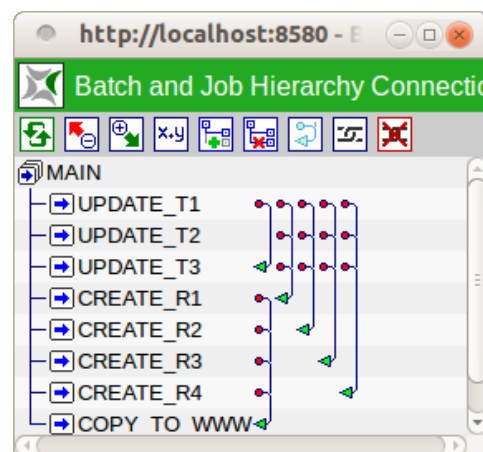


Figure 4: Batchstruktur ohne hierarchischer Ordnung nach dem Einbinden des Reports R4

In unserer hierarchisch organisierten Ablaufdefinition genügt es den Job CREATE_R4 zu definieren und diesen in den Subbatch CREATE_REPORTS einzuhängen. Es müssen keine zusätzlichen Abhängigkeiten mehr erzeugt werden. Aufwand und Fehlrisiko sind in dieser Variante deutlich geringer als ohne hierarchische Ordnung des Ablaufes. Abbildung 5 zeigt die neue Struktur des Ablaufes.

Beispiel: Zerlegung eines Prozesses in Teilprozesse (Process Decomposition)

Auch der Prozess der Teilprozesszerlegung wird durch das hierarchische Modell des schedulixScheduling Systems optimal unterstützt, was wir in folgendem Beispiel verdeutlichen wollen.

Der Prozess UPDATE_T1 läuft inzwischen zu lange und soll in die beiden parallel ausführbaren Prozesse UPDATE_T1_A und UPDATE_T1_B zerlegt werden. Dazu sind im schedulix Scheduling System nur folgende Handgriffe nötig:

1. Definition der Jobs UPDATE_T1_A und UPDATE_T1_B

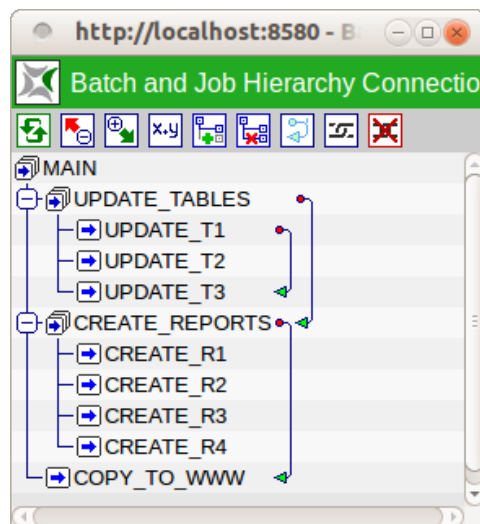


Figure 5: Batchstruktur mit hierarchischer Ordnung nach dem Einbinden des Reports R4

2. Ändern des Types von UPDATE_T1 nach 'BATCH'
3. Einhängen der Jobs aus 1. in den Batch UPDATE_T1

Sämtliche Abhängigkeitsbeziehungen bleiben erhalten und der Aufwand die Änderung im Scheduling System zu realisieren, sowie das Fehlerrisiko bleiben minimal. Abbildung 6 zeigt unseren Ablauf nach der Durchführung der Teilprozesszerlegung. Zur Übersichtlichkeit wurde in der Darstellung der Subbatch CREATE_REPORTS geschlossen.

Wollte man diese Teilprozesszerlegung in einer Ablaufstruktur ohne hierarchische Ordnung implementieren, wären zusätzlich zu den beschriebenen Arbeitsschritten noch eine Menge neuer Abhängigkeitsbeziehungen zu erzeugen um eine korrekte Abarbeitungsreihenfolge zu gewährleisten.

Weitere Eigenschaften von hierarchisch strukturierten Abläufen

Neben den gezeigten Eigenschaften, unterstützen hierarchisch organisierte Ablaufstrukturen weitere Funktionen deren ausführliche Vorstellung hier den Rahmen sprengen würde.

Dazu gehören unter anderem:

- Wiederverwendbarkeit durch Parameterisierung
- Automatische Fehlerbehandlung und Notifications (Trigger)
- Hierarchisches Operating (Suspend, Resume, Rerun, Cancel, ... auf (Sub)Batchebene)

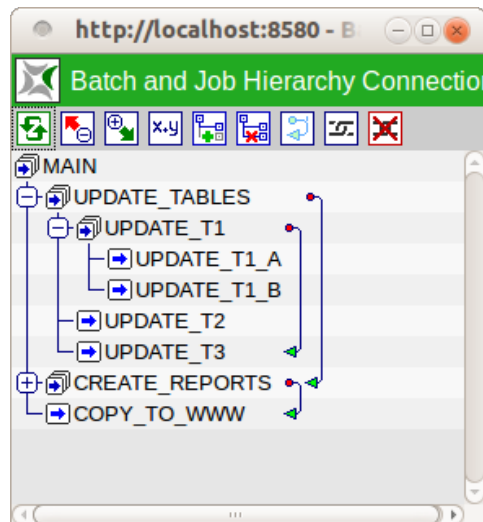


Figure 6: Batchstruktur nach Zerlegung des Prozesses UPDATE_T1 in parallele Teilprozesse

- Bedingte Ausführung von Ablaufteilen
- Wiederholte Ausführung von Ablaufteilen
- ...

Wir werden dem einen oder anderen Aspekt einen zukünftige schedulix!focus widmen.

Schlußbemerkung

Schon die sehr einfachen Beispiele in diesem Dokument machen deutlich, welche Vorteile hierarchische Batch Strukturen in der Modellierung von Abläufen bieten. In grösseren Abläufen mit einer hohen Zahl von Teilprozessen wird diese Funktionalität des eingesetzten Scheduling Systems zu einer Notwendigkeit ohne die eine optimale und sinnvolle Teilprozesszerlegung nicht mehr möglich ist.

Das schedulix Scheduling System unterstützt mit seinem ausgereiften und mächtigen Konzept zur hierarchischen Abbildung von Prozessen die Teilprozesszerlegung optimal und erhält dabei Übersichtlichkeit und Wartbarkeit auch sehr grosser und komplexer Auflaufstrukturen.