Workflows selbst weiterläuft sowie dass der lokale Workflow keine Daten an den auslösenden Workflow zurückgeben kann. N-faches Auslösen eines Startereignisses für einen lokalen Subworkflow löst diesen auch N mal aus! Im Workflow-Protokoll findet man in diesem Fall N Instanzen zu diesem lokalen Workflow.

10.8 Dynamische Blöcke

Ein dynamischer Block bearbeitet eine Liste mit gleichartigen Elementen (Objektreferenzen), deren Elementanzahl zur Definitionszeit nicht bekannt ist.

In früheren Releases gab es dafür die dynamische Parallelverarbeitung. Dieses Verfahren ist in [2] ausführlich beschrieben. Wollte man mehrere Schritte dynamisch parallel verarbeiten, so musste man ein eigenes Workflow-Muster anlegen, das dann dynamisch parallel aufgerufen wurde.

Die dynamischen Blöcke lösen das Problem viel eleganter. Zwischen den Blockbegrenzern (Pfeil nach oben, Pfeil nach unten) können beliebig viele Workflow-Schritte eingefügt werden. Diese bilden dann quasi einen eigenen, eingebetteten Workflow mit eigenem Container und Datenfluss zum umgebenden Workflow.

Es sind zwei Arten von Blöcken möglich:

- Dynamisch sequenzieller Block
  Das ist letztlich ein Iterator auf Workflow-Ebene.
- Dynamisch paralleler Block
  Das ist eine dynamische Parallelverarbeitung.

Blöcke werden durch X-Workitems repräsentiert. Bei sequenziellen Blöcken existiert genau ein Block-Workitem mit genau einem Blockcontainer. Dieser Container wird nacheinander von den Workitems des Blockes benutzt.

Bei dynamisch-parallelen Blöcken existieren so viele Block-Workitems mit je einem Blockcontainer wie es Einträge in der Objektliste gibt. Die Workitem-Hierarchie in der Kopftabelle SWWWIHEAD wird ergänzt:

- SWWWIHEAD-WI_CHCKWI
  Workitem des umgebenden Flows
- SWWWIHEAD-PARENT_WI
  Block-Workitem des umgebenden Blockes

Bei Block-Workitems sind PARENT_WI und WI_CHCKWI gleich.

10.8.1 Dynamisch sequenzieller Block = Workflow-Iterator

Der sequenzielle dynamische Block entspricht genau einem Iterator im Workflow-Kontext.

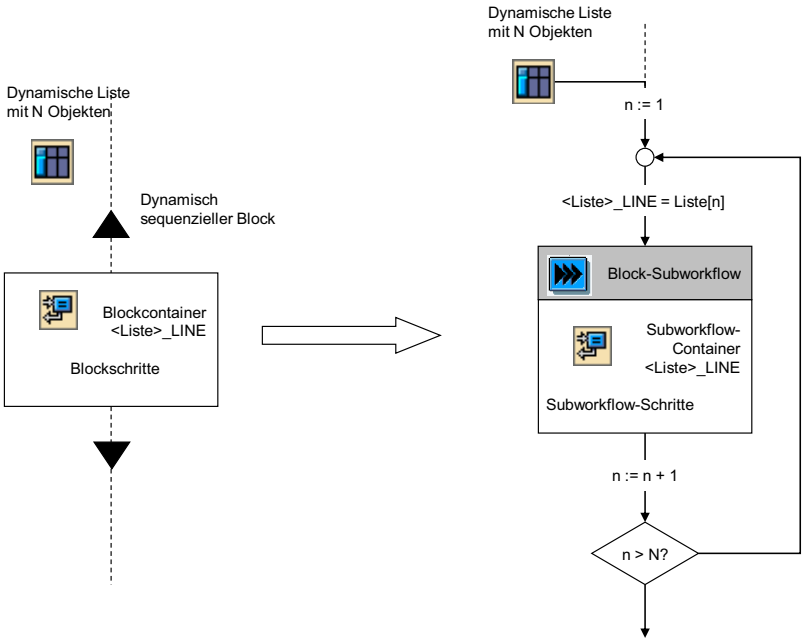


Abb. 10–26
Dynamisch sequenzieller Block als eingebetteter Subworkflow

10.8.2 Dynamisch paralleler Block

Der dynamisch parallele Block bearbeitet die N Objekte in einem parallelen Abschnitt, dessen Zweiganzahl erst zur Laufzeit bekannt ist.

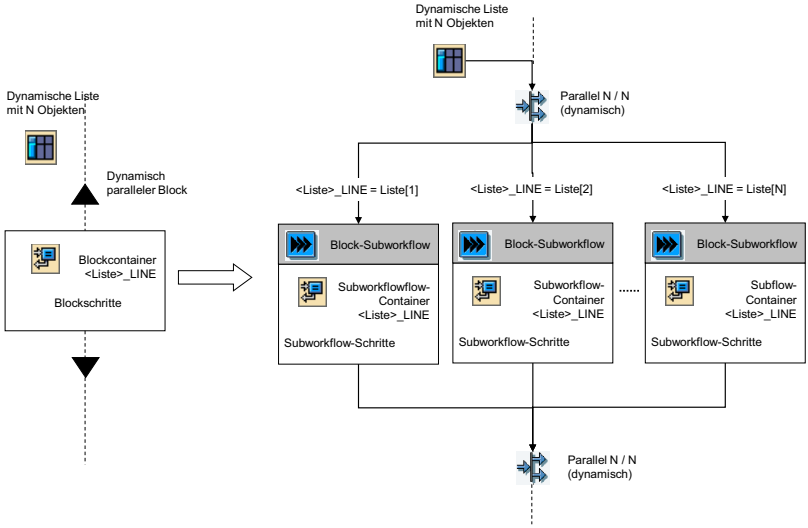


Abb. 10–27
Dynamisch paralleler Block als eingebetteter Subworkflow

10.8.3 Beispiel: Bewertungs-Workflow für Vertrag

Im folgenden Beispiel wird ein Vertrag dynamisch parallel durch N Mitarbeiter bewertet. Jeder Mitarbeiter kann über eine Sachbearbeiterentscheidung (SBE) von minimal 0 bis maximal 3 Punkte vergeben. Die Parallelverarbeitung soll abgebrochen werden, wenn insgesamt bereits N*2 Punkte erreicht wurden.

Variablen im Workflow-Container:

Vertrag	Ref To ZCL_VERTRAG
UserListe	Ref To ZCL_USER, mehrzeilig
UserCount	Type I, Anzahl der User in der Liste (=N)
PunkteGesamt	Type I, Anzahl der vergebenen Gesamtpunkte
PunkteLimit	Type I, notwendige Punkteanzahl für Abbruch 2*N

Der erste Block demonstriert die dynamisch-sequenzielle Verarbeitung. Als Blockschritt wurde irgendein Schritt gewählt.

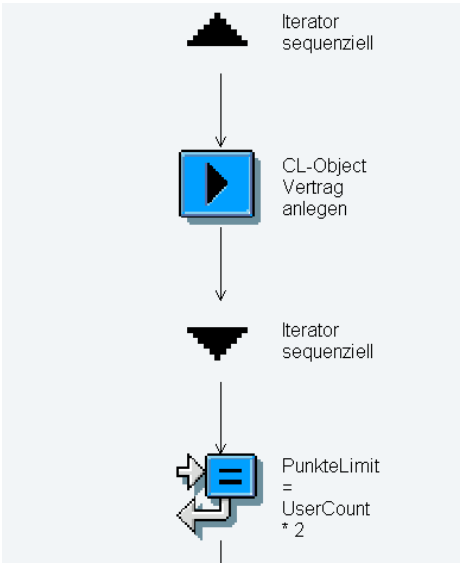


Abb. 10–28
Dynamisch-sequenzielle Bearbeitung der UserListe

Der Datenfluss Workflow ←→ Blockcontainer sieht folgendermaßen aus: In &_WF_PARFOREACH_INDEX& steht der aktuelle Index der UserListe. Das Element mit diesem Index wird übergeben und könnte bearbeitet werden. Hier wird als Rückgabewert einfach eine 1 auf den UserCount im Workflow-Container addiert. Zum Schluss hat man also die Anzahl der Einträge in der dynamischen Liste ermittelt.

(Natürlich würde man dazu nicht extra einen Block anlegen, sondern eher eine Utility-Methode; hier geht es nur um die Demonstration

einer speziellen Variante des Datenflusses). Das PunkteLimit wird mit PunkteLimit = 2 * UserCount festgelegt.

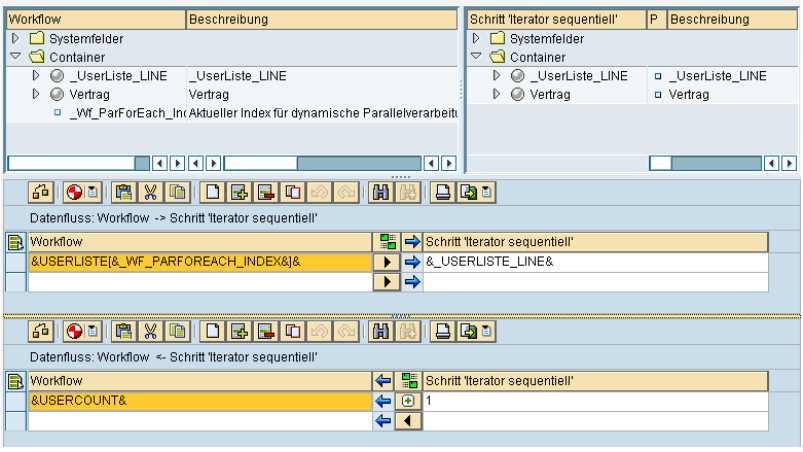


Abb. 10–29
Datenfluss zum dynamisch-sequenziellen Block

Der dynamisch-parallele Block erzeugt je User in der UserListe eine SBE mit den vier Bewertungsergebnissen. Als Bearbeiter wird ein Ausdruck der Form &_USERLISTE_LINE.GET_PDUSER()& benutzt. Die Returningmethode setzt einfach ein »US« vor den Usernamen der Instanz.

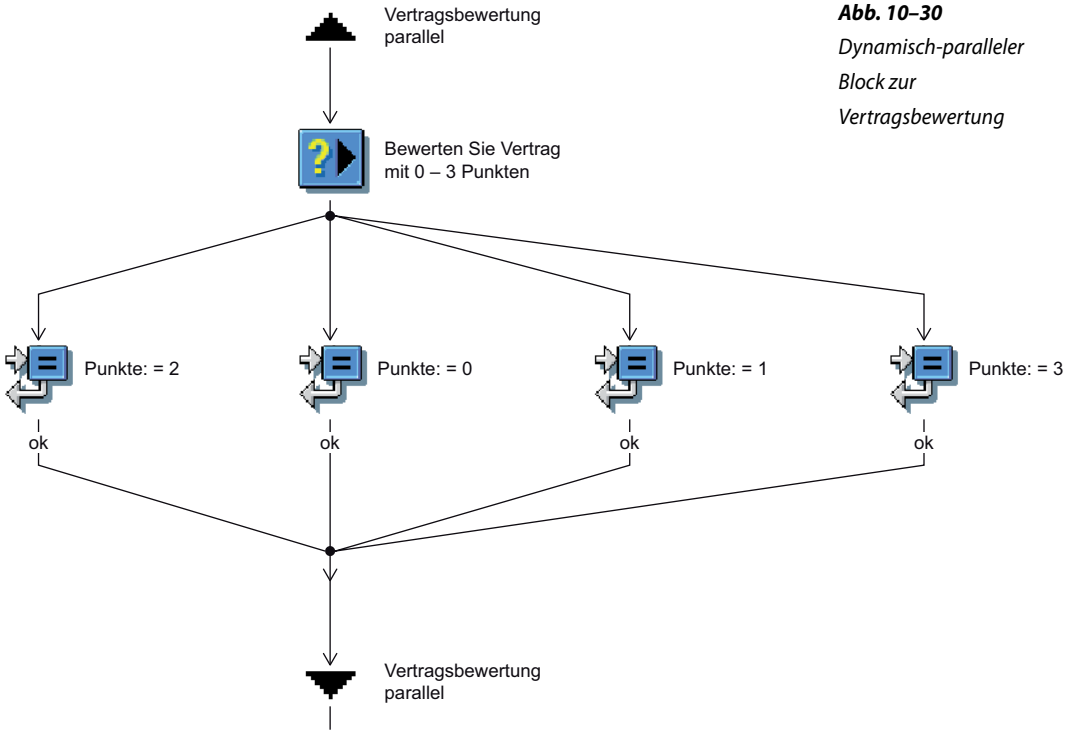


Abb. 10–30
Dynamisch-paralleler Block zur Vertragsbewertung

Dem Blockcontainer wird wieder das n-te Element der UserListe übergeben, zusätzlich der zu bewertende Vertrag. Das Ergebnis einer Bewertung ist die vom jeweiligen User vergebene Punkteanzzahl, sie wird im Datenfluss vom Blockcontainer zum Workflow auf die PunkteGesamt aufaddiert.

Abb. 10–31
Datenfluss zwischen Workflow-Container und Blockcontainer

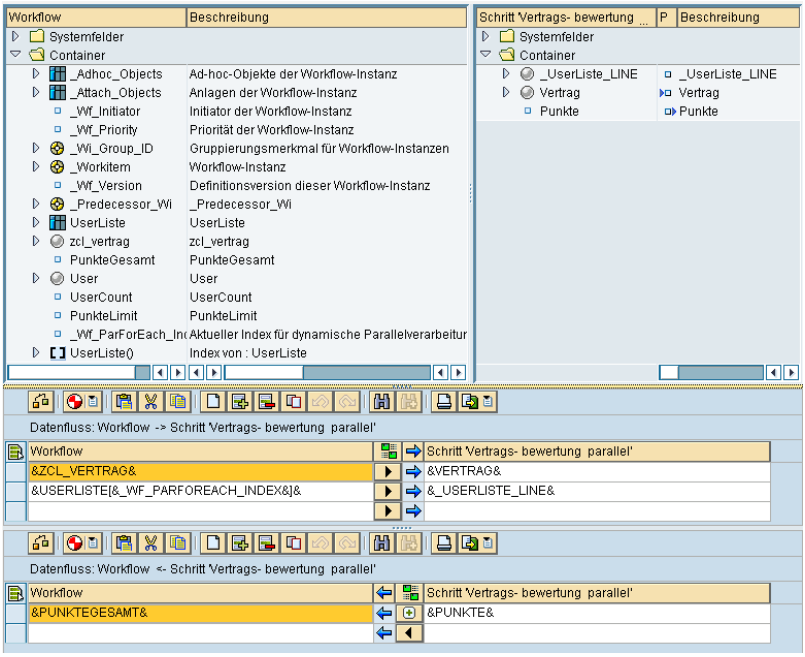
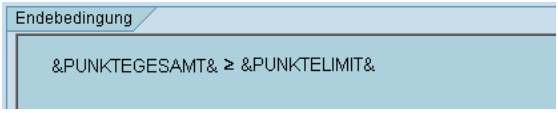
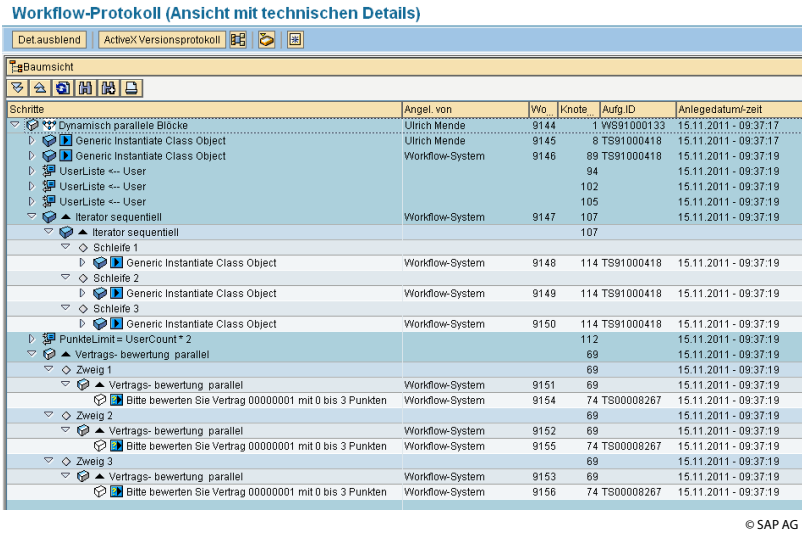


Abb. 10–32
Abbruchbedingung für dynamisch-parallelen Block



Die dynamische Parallelverarbeitung wird abgebrochen, wenn das PunkteLimit erreicht wurde, bevor alle Bewertungen vorliegen.

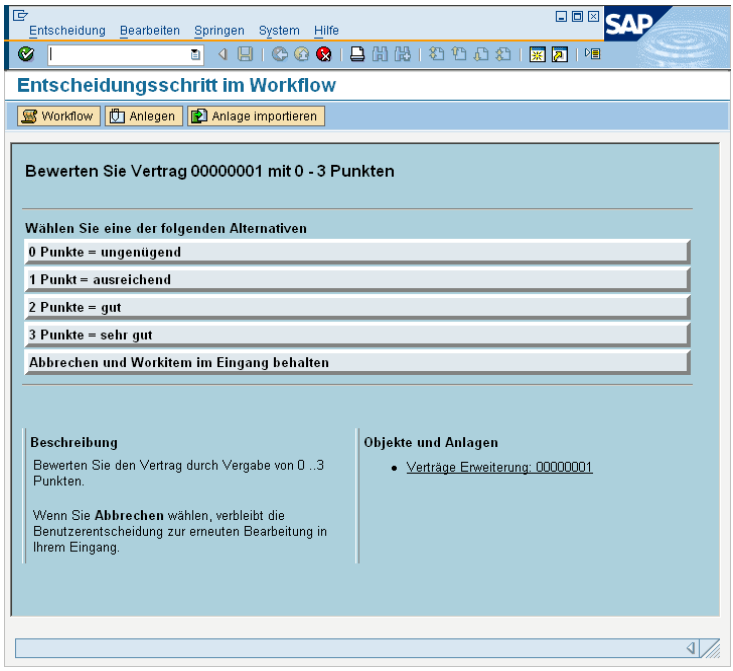
Das Workflow-Protokoll zeigt den Stand nach Abarbeitung des sequenziellen Blocks (ohne Dialogschritte) und nach Erzeugung aller drei Dialogschritte im parallelen Block. Es ist zu sehen, dass der sequenzielle Block für die drei Durchläufe nur ein Block-Workitem hat (9147). Demgegenüber hat der parallele Block drei Block-Workitems (9151,9152, 9153).



© SAP AG

Abb. 10–34
Protokoll eines Workflows mit Blöcken

In der SBE werden von jedem User die Punkte vergeben. In einem Link soll das zu bewertende Objekt, hier also der Vertrag, anzeigbar sein.



© SAP AG

Abb. 10–33
SBE zur Vertrags- bewertung mit Link zum Vertrag

Leider funktionieren die im Container vordefinierten _ADHOCOBJECTS nicht für Referenzen von Klassen. Man muss also für jeden Klassenlink eine eigene Variable im Container anlegen. Aber das ist kein großer Aufwand, zumal man den Datenfluss ohnehin angeben muss.