A Workflow Approach to Stream Processing

Author(s):
Biörnstad, Biörn Johan

Publication Date:
2008

Permanent Link:

https://doi.org/10.3929/ethz-a-005671753 →

Diss. ETH No. 17600

A Workflow Approach to Stream Processing

A dissertation submitted to
ETH ZURICH

for the degree of
Doctor of Technical Sciences

presented by
BIÖRN JOHAN BIÖRNSTAD
Dipl. Informatik-Ing. ETH
born February 3, 1978
citizen of Raperswilen TG
and Wigoltingen TG

accepted on the recommendation of
Prof. Dr. Gustavo Alonso, examiner
Prof. Dr. Heiko Schuldt, co-examiner
Prof. Dr. Cesare Pautasso, co-examiner

2008

Abstract

Today, workflow languages are widely used for service composition. Workflow and business process management (BPM) systems are typically based on a step-by-step execution model where a task is started, the result is received, and then the next task is scheduled for execution in a similar fashion. To track the execution of individual service invocations and of the overall workflow process, a state machine based approach is used. The model corresponds to the request-response nature of many service interfaces and maps directly to technologies such as Web services or business process modeling specifications such as WS-BPEL. However, there are services which do not follow this interaction pattern but rather proactively produce new information to be consumed by an application. Examples include RSS feeds listing the latest bids at an auction, result tuples from a data stream management system (DSMS), stock price tickers or a tag stream from an RFID reader.

This dissertation shows how to extend traditional state-based workflow management techniques with the necessary features to integrate streaming data services and combine them with conventional request-response services. First, we study the problem of accessing a stream from within a workflow process. We discuss different alternatives in terms of expressiveness and performance. One approach involves the extension of the traditional request-response task model. We show how to accomplish this on the level of the workflow language and describe the necessary changes in a workflow engine. Our solution provides a notion of stream which abstracts from the mechanism or protocol used to access the content of the stream. This makes the service composition independent of what type of stream is processed, e.g., RSS items, RFID tags or database tuples.

The invocation of a streaming service leads to a stream of result elements independently flowing through the invoking process, thereby creating a pipelining effect. This leads to safety problems in the execution of the process, as state-based workflow execution models are not designed for such parallel processing. E.g., considering that the tasks of a process do not always take the same time to execute, a task might not be ready to process a stream element when the element is offered to the task. This can lead to loss of data in the stream processing pipeline. We discuss different solutions to avoid the safety problems connected with pipelined processing and identify the minimal necessary extension to the semantics of a workflow language. This extension is based on a flow control mechanism which controls the flow of data between tasks in a process and allows the safe use of pipelining in the execution of a process.

Our extended semantics for pipelining will block a task when it is not safe to execute it. However, if the services that are composed into a stream processing pipeline show variations in their response time, this will decrease the throughput

of the pipeline, as our measurements show. Therefore, based on the flow control semantics, we show how to use a buffered data transfer between tasks in a pipelined process. The buffers will smooth the irregularities in the task duration and allow a pipeline to run at its maximum possible throughput.

Finally, to evaluate our approach, the stream processing extensions proposed in this thesis have been implemented in an existing workflow system. Apart from describing the implementation in detail, we present several performance measurements and an application built on top of the extended system. The application is a Web mashup which integrates the data from a live Web server log with a geolocation database in order to provide a real-time view of the visitors to a Web site together with their geographic locations on a map.

Kurzfassung

Die Verwendung von Workflow-Sprachen zur Komposition von Diensten ist heute weitverbreitet. Workflow- und Businessprozess-Systeme (BPM) basieren meist auf einer schrittweise Ausführung von Prozessen, bei der eine Aufgabe gestartet, das Resultat erhalten wird und anschliessend die als nächstes auszuführenden Aufgaben ermittelt werden. Um den Fortschritt der einzelnen Aufgaben wie auch des ganzen Prozesses zu verfolgen, verwenden herkömmliche Workflow-Systeme endliche Automaten. Dieses Model entspricht dem "request-response" Verhalten vieler Dienste und entspricht direkt Technologien wie Web services oder Spezifikationen für Prozessmodellierung wie WS-BPEL. Es existieren jedoch auch Dienste, welche nicht nach diesem Muster kommunizieren, sondern aktiv Informationen produzieren, die von Applikationen konsumiert werden können. Einige Beispiele hierfür sind RSS-Feeds mit den neusten Geboten an einer Auktion, Resultat-Tupel eines Data-Stream-Management-Systems (DSMS), Börsenticker oder der Datenstrom eines RFID-Lesegeräts.

Diese Dissertation zeigt auf, wie man herkömmliche zustandsbasierte Workflow-Techniken erweitern kann, um Datenströme integrieren und mit konventionellen "request-response" Diensten kombinieren zu können. Als erstes untersuchen wir das Problem des Zugriffs auf einen Datenstrom aus einem Prozess heraus. Wir diskutieren verschiedene Alternativen, welche sich in der Aussagekraft und der Performanz unterscheiden. Einer der Ansätze beinhaltet die Erweiterung des herkömmlichen request-response Models einer Aufgabe. Wir zeigen, wie die Workflow-Sprache entsprechend erweitert werden muss, und wir beschreiben die nötigen Änderungen im Workflowsystem. Unsere Lösung bietet eine abstrakte Sicht auf einen Datenstrom, welche unabhängig ist vom konkreten Protokoll für den Zugriff auf den zugehörigen Dienst. Dadurch wird eine Dienste-Komposition unabhängig vom Typ des verarbeiteten Datenstromes, z. B. RSS-Elemente, RFID-Kennungen oder Datenbanktupel.

Der Empfang eines Datenstroms von einem Dienst führt dazu, dass mehrere Datenelemente unabhängig durch den Prozess fliessen, welcher den Dienst aufgerufen hat, wodurch ein Pipeliningeffekt ensteht. Dieser Effekt führt zu Sicherheitsproblemen bei der Ausführung des Prozesses, da diese Art der überlappenden Ausführung in Workflow-Sprachen nicht vorgesehen ist. Weil die einzelnen Aufgaben in einem Prozess nicht immer gleich viel Zeit benötigen, kann es zum Beispiel vorkommen, dass eine Aufgabe nicht bereit ist, ein Datenstromelement zu bearbeiten, weil sie noch mit dem vorhergehenden Element beschäftigt ist. Dies kann zum Verlust des betreffenden Datenelements führen. Wir diskutieren verschiedene Alternativen zur Vermeidung der Sicherheitsprobleme und erörtern die minimal benötigte Erweiterung der Semantik einer Workflowsprache. Diese Erweiterung basiert auf der Kontrolle des Datenflusses zwischen den verschiedenen Aufgaben in einem Prozess. Diese

Flusskontrolle erlaubt den gefarhlosen Einsatz von Pipelining bei der Ausführung eines Prozesses.

Unsere erweiterte Semantik für Pipelining blockiert eine Aufgabe, falls sie nicht sicher ausgeführt werden kann. Falls jedoch die Dienste, welche in einem Prozess kombiniert sind, unregelmässige Ausführungszeiten aufweisen, führt dies zu einem verringerten Durchsatz in der Verarbeitung eines Datenstromes, wie unsere Messungen zeigen. Auf der Flusskontrolle aufbauend, zeigen wir, wie ein gepufferter Datenaustausch zwischen Aufgaben verwendet werden kann, um diese Unregelmässigkeiten auszugleichen. Dies erlaubt einem Prozess mit Pipelining, seinen maximalen Durchsatz zu erreichen.

Um unseren Ansatz zu beurteilen, haben wir schliesslich die vorgeschlagenen Erweiterungen für die Bearbeitung von Datenströmen in einem bestehenden Workflowsystem implementiert. Abgesehen von einer detaillierten Beschreibung der Implementation, diskutieren wir verschiedene Messungen und eine Applikation, welche aufbauend auf dem erweiterten System realisiert wurde. Die Applikation ist ein "Web mashup", welches Daten aus einem Webserver-Log mit einer Geolocation-Datenbank integriert, um in Echtzeit die Besucher einer Website zusammen mit ihren geografischen Standorten auf einer Karte zu visualisieren.