Pragmatisches Business Activity Monitoring für Workflow Management-Systeme in einer SOA

Karin Büscher, Andreas Hausotter, Arne Koschel, Jan Seidel

Fakultät Wirtschaft und Informatik
Hochschule Hannover
Ricklinger Stadtweg 120
30459 Hannover
karin.buescher@web.de
andreas.hausotter@hs-hannover.de
akoschel@acm.org
janseidel@t-online.de

Abstract: Um im Wettbewerb zu bestehen, müssen Unternehmen ihre durch Anwendungssysteme (semi-) automatisierten Geschäftsprozesse flexibel ändern können. Hier sind u.a. Workflow-Management-Systeme (WfMSe) in Kombination mit Service-orientierten Architekturen (SOAn) einsetzbar. Das WfMS dient der Steuerung von Service-Aufrufen innerhalb von Geschäftsprozessen. Mittels Geschäftsprozess-Controlling werden Prozesse analysiert und durch Geschäftsprozess-Monitoring (GPM) während der Prozessausführung anfallenden Informationen und auftretenden Ereignisse überwacht, dies auch zeitnah mittels Business Activity Monitoring (BAM). Dieser Artikel liefert hier Beiträge durch eine auf einem Data-Warehouse-Ansatz basierende Architektur für BAM, die einige Nachteile „klassischer" BAM-Werkzeuge vermeidet und auch GPM-Anforderungen erfüllt. Ein größeres Fallbeispiel aus dem Versicherungswesen zeigt die Tragfähigkeit der erarbeiteten Konzepte.

1 Einleitung

Prozess- und Service-Orientierung sind aktuelle Konzepte zur Gestaltung der IT-Landschaft in Unternehmen. Dieser Artikel betrachtet ausgewählte Fragestellungen, die sich bei der Umsetzung dieser Konzepte mittels Workflow-Management-Systemen und deren Kombination mit Service-orientierten Architekturen ergeben.

1.1 Workflow-Management-Systeme und Service-orientierte Architekturen

Workflow-Management-Systeme unterstützen Unternehmen bei der Koordination ihrer Geschäftsprozesse nebst zugehörigen Datenbeständen (vgl. [GHS95] und [Ho95]). Die Steuerung von Workflow-Client-Anwendungen (Client Applications) wird größtenteils durch ein WfMS vorgenommen. Individuen erhalten ihre Aufgaben (Tasks) aus einem

zentralen Arbeitskorb (Worklist), der durch die Geschäftsprozesse gefüllt wird (vgl. [Ho95]).

Vielversprechend ist die Kombination von WfMS mit Service-orientierten Architekturen. Eine Service-orientierte Architektur (SOA) ist das konzeptuelle Ergebnis der Aufteilung des Gesamtgeschäfts eines Unternehmens in Geschäftsdomänen (vgl. [Er05], [OA06], [KBS07] und [En08]). Entsprechend bietet sie den Zugriff auf Geschäftsfunktionen und -daten des Unternehmens, technisch bereitgestellt in Form von Services. Ein Service stellt somit eine klare Funktionalität hinter einer wohl definierten Schnittstelle bereit, dies in einer meistens logisch und technisch verteilten Systemumgebung.

Häufig wird die Implementierung einer SOA ausschließlich mit dem Einsatz von Web Services assoziiert (vgl. [Er05] und [[LN05]). Folglich werden dann für Prozesse meist nur die Web Services-Standards WSBPEL, BPEL4PEOPLE und WS HumanTask betrachtet (vgl. [OA07], [Kl05] und [Ag07]).

Allerdings greift diese Sichtweise in der Praxis oft zu kurz. Unternehmen können oder wollen aus betrieblichen Gründen keine (reine) Web Services-Lösung einsetzen, z.B. weil sich eine solche Lösung nicht nahtlos in die gewachsene IT-Landschaft integrieren lässt. Gleichwohl soll die Anwendungslandschaft Service-orientiert gestaltet werden, dies aber auf Basis anderer Technologien.

Eine gut geeignete und immer noch Standard-basierende Technologiekombination stellen WfMS gemäß der Spezifikation der Workflow Management Coalition (vgl. [Ho95]) kombiniert mit Java Enterprise (Java EE) Application Server-Lösungen oder anderen Integrationstechnologien wie CORBA (vgl. [HV09] und [Du08]) oder De-facto-Standards wie .NET (vgl. [Sc08] und [Du02]) dar.

Im Fokus dieses Artikels stehen also Service-orientierte Architekturen, die auf Workflow-Management-Systemen und anderen Integrationstechnologien basieren.

1.2 Anwendungshintergrund: Deutsches Versicherungswesen

Die Tragfähigkeit der Kombination WfMS und SOA als solide und aktuelle Basis für die zentrale Software-Architektur eines Unternehmens wird mittlerweile als gegeben angesehen. Sie kommt beispielsweise auch in deutschen Versicherungsunternehmen zum Einsatz.

Standardisierte Geschäftsprozesse aus dem Versicherungswesen stellen auch das Anwendungsumfeld des vorliegenden Beitrags dar. Sie sind in Deutschland mit der Versicherungsanwendungsarchitektur (VAA, vgl.[GD13]) gegeben. Somit sind die in diesem Artikel behandelten Fragestellungen potentiell für alle Arten deutscher Versicherungsunternehmen bedeutsam. Der VAA entstammt der hier als Beispiel vorgestellte Standardprozess „Kulanz bearbeiten". Da die in diesem Beitrag erzielten Ergebnisse i.W. anwendungsneutral sind, lassen sie sich potentiell auch auf andere Anwendungsfelder übertragen.

Wird die Kombination von WfMS und SOA technisch betrachtet, so heißt das, WfMS können zur Implementierung von Geschäftsprozessen eingesetzt werden. Diese Prozesse

nutzen wiederum die Services einer SOA. WfMS stellen also Geschäftsprozess-Management-Funktionen bereit und die Services der SOA liefern die wesentlichen Ein- und Ausgaben sowie Berechnungsfunktionen hierzu.

1.3 Geschäftsprozess-Controlling, Geschäftsprozess-Monitoring und Business Activity Monitoring

WfMSe persistieren alle während der Prozess-Ausführung anfallenden Informationen in der so genannten „Audit-Trail-Datenbank". Sie schaffen damit die technische Voraussetzung für das sog. Geschäftsprozess-Controlling (Business Process Controlling, BPC). Hierbei werden die gesammelten Informationen abgeschlossener Prozess- und Aktivitätsinstanzen aufbereitet, aggregiert und analysiert und betriebswirtschaftlich motivierte Kennzahlen (Key Performance Indicator, KPI) generiert.

Primäres Ziel des Geschäftsprozess-Controlling ist die kontinuierliche Optimierung der Prozessmodelle im Rahmen des Geschäftsprozess-Lebenszyklus. Dieser umfasst die Phasen Geschäftsprozess-Analyse und -Design, Workflow-Implementierung, Workflow-Ausführung und Workflow-basiertes Controlling (vgl. [Mu02]).

Über die im Rahmen des Geschäftsprozess-Controlling vorgenommene Analyse und Aufbereitung der Audit-Trail-Daten hinaus sind auch die aktuell in Ausführung befindlichen Prozess- und Aktivitätsinstanzen von besonderem Interesse. Fragestellungen wie z.B. „Wie lassen sich kritische Zustände, etwa die Schwellwertüberschreitung der Prozessdurchlaufzeit oder der Liegezeit einer Aktivitätsinstanz, rechtzeitig erkennen?" und „Welche Aktionen sind infolge kritischer Zustände auszuführen?" stehen per definitionem nicht im Fokus des Geschäftsprozess-Controlling. Sie lassen sich vielmehr dem sogenannten Geschäftsprozess-Monitoring (Business Process Monitoring, BPM) zuordnen, das die zeitnahe Überwachung aktiver Prozessinstanzen zum Gegenstand hat, um diese zielgerichtet zu beeinflussen und schnell auf auftretende Probleme und Engpässe reagieren zu können.

Damit ist das Geschäftsprozesse-Monitoring konzeptionell Teil des – allgemeineren – Business Activity Monitoring (BAM). Bei diesem von Gartner im Zusammenhang mit Enterprise Application Integration (EAI) geprägten Begriff handelt es sich um die Echtzeiterfassung kritischer Performance-Indikatoren mit dem Ziel, die Geschwindigkeit und Effektivität von Geschäftsfunktionen zu verbessern (vgl. [HF02]). Im Gegensatz zum Prozessmonitoring werden also bei BAM nicht nur Prozessinstanzen, sondern auch Middleware-Systeme – etwa Workflow-Management-Systeme und Datenbanksysteme – und Backend-Anwendungen einer Anwendungslandschaft berücksichtigt.

Geschäftsprozess-Controlling und -Monitoring unterstützen ein kontinuierliches Geschäftsprozessmanagement, das einerseits steuernde Eingriffe in die Prozessabläufe und andererseits deren agile und flexible Anpassung an geänderte Marktbedingungen ermöglicht (vgl. [Be07]). Zusammen mit dem Business Activity Monitoring liefert es damit einen wesentlichen Beitrag zur Sicherstellung der Wettbewerbsfähigkeit der Unternehmen.

BAM-Werkzeuge namhafter Hersteller sind auf dem Markt verfügbar, so z.B. von IBM, Oracle, Microsoft, TIBCO, SAP und der Software AG (vgl. [WSG07]). Da jedoch BAM

im Kontext von EAI entwickelt wurde, zeichnet es sich durch eine hohe Komplexität seiner Komponenten und eine nicht immer zufriedenstellend zu lösende Integration in vorhandene System- und Anwendungslandschaften aus. Als ggfs. problematisch ist der hohe Ressourcenbedarf anzusehen. Er resultiert vor allem aus dem Anspruch, kritische Anwendungs- und Systemzustände in Echtzeit zu erfassen und zu propagieren.

1.4 Ergebnisse

Da einerseits das Monitoring von Anwendungssystemen und Systemplattformen häufig durch spezialisierte Lösungen – etwa HP Open View – übernommen wird, andererseits eine Echtzeit-Überwachung aktiver Prozess- und Aktivitätsinstanzen im Rahmen des Geschäftsprozess-Monitoring oftmals nicht erforderlich ist, ist der BAM-Ansatz nach Gartner (vgl. [HF02]) für Unternehmen jedoch vielfach nicht die Methode der Wahl.

Hier setzen die Forschungsbeiträge der vorliegenden Arbeit an, die in Kooperation mit IT-Abteilungen aus mehreren mittelgroßen deutschen Versicherungsunternehmen entstanden sind. Die auf einem Data-Warehouse-Ansatz basierende Lösung erfüllt einerseits die Anforderungen an ein zeitnahes Geschäftsprozess-Monitoring; sie vermeidet andererseits aber den hohen Ressourcenbedarf „klassischer" BAM-Werkzeuge. Im Einzelnen sind die Beiträge dieses Artikels:

- Architekturkonzepte für ein Geschäftsprozess-Monitoring in Service-orientierten Architekturen mit Workflow. Alle Alternativen nutzen dabei die mit Informationen aus der Audit-Trail-Datenbank des WfMS befüllte „Foundation Database" (vgl. [Ha11]).
- eine Bewertung der Konzepte mit den Methoden der Nutzwertanalyse – unter Berücksichtigung der Wiederverwendbarkeit, Angemessenheit und des Realisierungsaufwands – und die anschließende Auswahl einer optimalen Alternative
- die prototypische Umsetzung des ausgewählten Architekturkonzepts
- die Evaluierung der realisierten Alternative anhand eines umfangreichen Fallbeispiels aus der VAA, dem Geschäftsprozess „Kulanz bearbeiten".

Für den von uns erstellten BAM-Prototypen wurden mehrere Leistungsmessungen durchgeführt, die u.a. ergaben, dass die Umsetzung in der jetzigen Form nicht für echtzeitnahe Anwendungsfälle geeignet ist. Komplette Durchläufe von der Ereigniserzeugung bis zur abschließenden Auswertung durch die Regelverarbeitung können auch im mehrstelligen Sekundenbereich liegen. Dies war jedoch aus Sicht der Kooperationspartner für deren Anwendungsbereiche in der Regel unkritisch und wurde für sie von der Flexibilität und Offenheit des Gesamtsystems mehr als aufgewogen.

Dennoch untersuchen wir in aktuell laufenden Arbeiten von uns auch den Transfer unserer Ergebnisse in Hybrid Cloud-basierte Umgebungen, wovon wir uns neben betrieblichem Optimierungspotential auch Performanz-Vorteile durch Parallelisierung versprechen.

Der Rest dieses Artikels ist wie folgt aufgebaut: Kapitel 2 behandelt verwandte Arbeiten. Die Kapitel 3 und 4 beschreiben das Anwendungsszenario und die Architekturkonzepte.

Kapitel 4 hat die prototypische Umsetzung der optimalen Alternative und die Evaluierung der Lösung zum Gegenstand. Zusammenfassung und Ausblick schließen diesen Artikel ab.

2 Verwandte Arbeiten

Neben der bereits einleitend durchgeführten Abgrenzung unserer Arbeiten zu Web Services-basierten Ansätzen wie WSBPEL etc. gibt es eine Reihe weiterer zum vorliegenden Artikel verwandte Veröffentlichungen, auf die im Folgenden näher eingegangen werden soll.

Allgemeine WfMS-Beiträge betrachten generelle WfMS-Konzepte und entsprechende Implementierungsaspekte (vgl. [GHS95], [De06] und [Ho95]). Sie liefern damit eine konzeptuelle Basis für weitergehende Forschungen. Demzufolge werden die hier vorgestellten Arbeiten in den konzeptuellen Kontext des Workflow-Referenzmodells der Workflow Management Coalition (WfMC, vgl. [Ho95]) eingeordnet. Diese Beiträge liefern jedoch nur allgemeine Konzepte; naturgemäß erreichen sie nicht den Spezialisierungsgrad der vorliegenden Arbeit.

In einem eng verwandten Kontext stehen Arbeiten, die Workflow-Management im SOA-Umfeld betrachten. So wird ein dezentralisiertes WfMS-Modell nebst Infrastruktur in [WML08] behandelt. Unsere BPM- und BAM-Architektur könnte eine nützliche Erweiterung derartiger Ansätze darstellen. Wie bereits erwähnt, sind die Konzepte unserer Arbeit übertragbar und damit vielleicht sogar geeignet, Web Services-Standards zu erweitern. Auf der anderen Seite könnten unsere Forschungen auch von Konzepten innerhalb der WSBPEL (vgl. [OA07]) und verwandten Spezifikationen profitieren.

Weitere verwandte Arbeiten betrachten die Kombination von WfMS und Data Warehouses. Bei [Mu01] beispielsweise wird ein WfMS und ein Data Warehouse für prozessorientierte Management-Informations-Systeme kombiniert und eine Taxonomie zur Auswertung der Audit-Trail-Datenbank sowie Komponenten zur Prozessüberwachung vorgestellt. Allerdings liegt bei diesem Artikel der Fokus – im Sinne eines BPC – eher auf der Analyse abgeschlossener Prozess- und Aktivitätsinstanzen und weniger auf der zeitnahen Reaktion auf kritische Prozesszustände – also BPM oder allgemeiner BAM – wie in unseren ForschungsarbeitenText.

Ebenfalls im Kontext stehende Arbeiten widmen sich betrieblichen und konzeptionellen Aspekten des Business Activity Monitoring. Bei [We09] wird eine Methodik zur Korrelation von KPIs und Quality of Service (QoS)- bzw. Prozessmetriken entwickelt. Die Arbeit von [Pe08] wiederum stellt einen semantischen Ansatz zur Ableitung betrieblich relevanter Informationen aus technischen Low-Level-Informationen der Audit-Trail-Datenbank vor.

Mehrere Veröffentlichungen legen den Fokus auf die Ereignisverarbeitung. So wird in [JFK04] ein regelbasiertes Framework für das BAM und in [JP06] die Ereignisverarbeitung in einer SOA für das BAM auf der Grundlage der SOA Lösung

SOPware der Deutschen Post vorgestellt. Ferner ist die Arbeit von [Gr06] zu erwähnen, da sie der fachlichen Domäne der Versicherungsunternehmen zuzurechnen ist. Die Autoren gehen auf die Neugestaltung der Prozesslandschaft nach SOA-Prinzipien ein und fokussieren dann auf die Verarbeitung von BPEL-Events und solchen, die aus einer "Event Cloud" durch ein "Complex Event Processing (CEP)"-System verarbeitet werden. Eine weitere Fallstudie diskutiert die Stärken und Schwächen verschiedener Performance-Management-Tools und stellt Architekturen für das Measurement und das Business Activity Monitoring vor (vgl. Ku05]).

Zusammengefasst könnten obige Beiträge in Teilgebieten der künftigen Ergänzung der vorliegenden Arbeit dienen, z.B. im Bereich der KPIs, QoS- bzw. Prozessmetriken. Umgekehrt könnten Arbeiten aus dem WfMS, BPC-, BPM- und BAM-Kontext ggf. auf unseren Arbeiten aufbauen.

Keine dieser Arbeiten stellt jedoch eine auf einem Data-Warehouse-Ansatz basierende integrierte Lösung für das Geschäftsprozess-Controlling, Geschäftsprozess-Monitoring und Business Activity Monitoring in Service-orientierten Architekturen mit WfMS bereit

3 Anwendungsszenario

3.1 Szenario

Die prototypische Implementierung des in diesem Artikel beschriebenen Business Activity Monitoring wurde einer umfassenden Evaluation unterzogen. Hierzu wurde ein Geschäftsprozess aus der Versicherungswirtschaft herangezogen und mit Hilfe einer Workflow-Client-Anwendung zur Bearbeitung von Aufträgen, die eine Benutzerinteraktion erfordern, umgesetzt.

3.2 Geschäftsprozess „Kulanz bearbeiten"

Ein Standardgeschäftsprozess der Versicherungswirtschaft ist „Kulanz bearbeiten", der innerhalb der VAA des Gesamtverbandes der Deutschen Versicherungswirtschaft (GDV) spezifiziert ist (vgl. [GD13]).

Im Verlauf dieses Prozesses werden zunächst die Vorschäden sowie die gezahlten Prämien des Versicherten ermittelt. Anschließend wird daraus der sogenannte Rentabilitätswert ermittelt. Bereits nach diesem Prozessschritt kann eine Ablehnung der Kulanzgewährung erfolgen. Sollte das nicht der Fall sein, wird ermittelt, ob das Vertragskonto des Versicherten ein negatives Saldo aufweist. Tritt dieser Fall ein, erfolgt ebenfalls eine Ablehnung. Andernfalls wird der zur Kulanz gemeldete Schaden mit einem festgelegten Wert verglichen. Sollte die Schadenshöhe unter diesem Wert liegen, wird die Kulanzhöhe errechnet und die Kulanz gewährt. Wenn die Schadenshöhe über diesem Wert liegt, erfolgt eine individuelle Beurteilung des Versicherten. Nach dieser Beurteilung wird

entschieden, in welcher Höhe eine Kulanz gewährt wird. Abschließend erfolgt in allen Fällen auch eine Korrespondenz mit dem Versicherten.

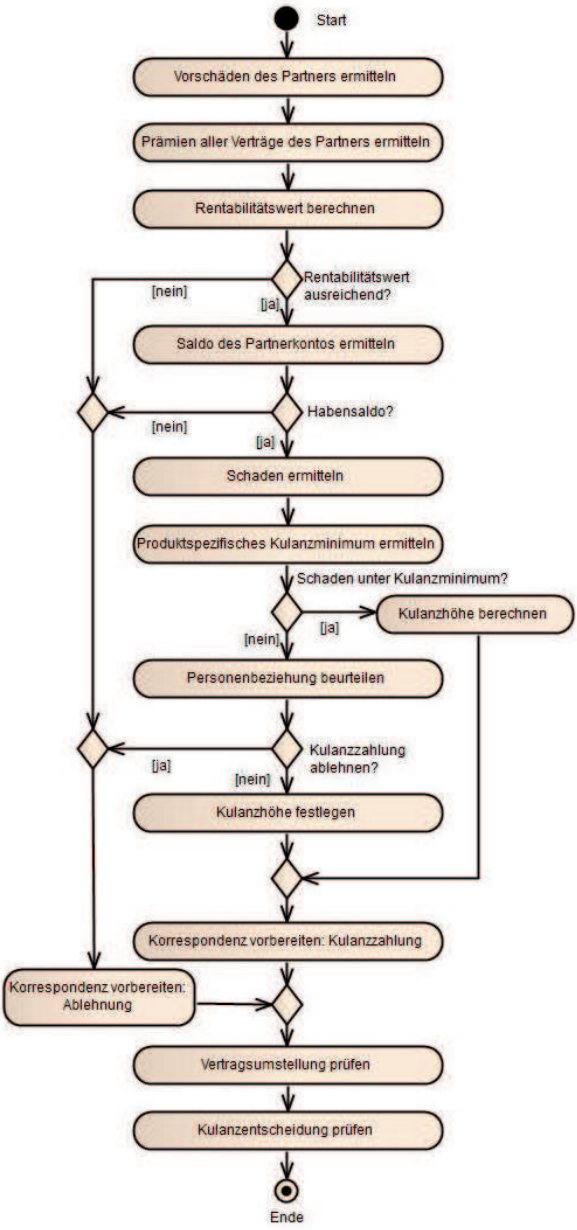

Abbildung 1: Geschäftsprozess „Kulanz bearbeiten“

Des Weiteren wird geprüft, ob eine Vertragsumstellung evtl. sinnvoll ist. Außerdem wird die Kulanzentscheidung abermals überprüft. Eine vereinfachte Darstellung des Prozesses wird im Beispiel „Kulanzprozess" (vgl. Abbildung 1) gezeigt.

3.3 Service-orientierte Realisierung

Das Anwendungsszenario, das diesem Beitrag zugrunde liegt, wurde aus Gründen der Wiederverwendbarkeit, Flexibilität und Agilität Service-orientiert gestaltet. Abbildung 2 zeigt den relevanten Ausschnitt aus der Service-orientierten Anwendungslandschaft.

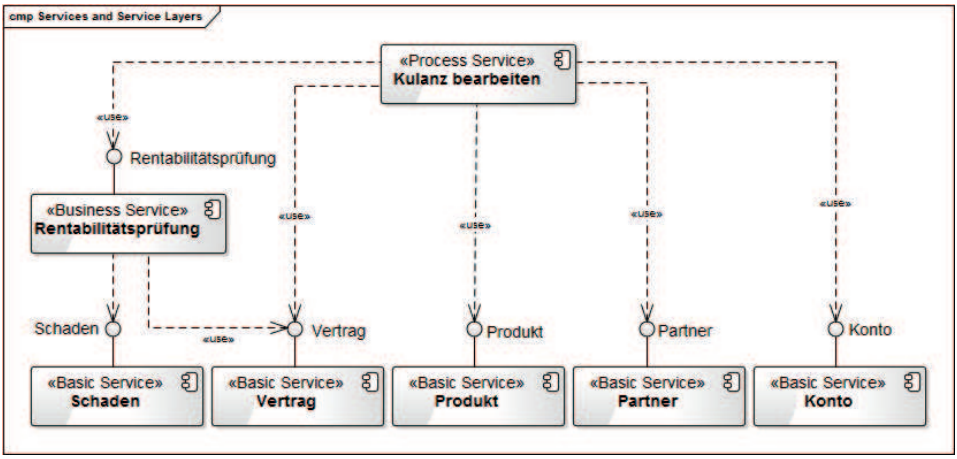

Abbildung 2: Services und Service Layers

„Schaden“, „Vertrag“, „Produkt“, „Partner“ und „Konto“ stellen datenzentrierte Basic Services dar, die die von den Backend-Systemen bereit gestellte Funktionalität über ihre Schnittstellen anbieten. „Rentabilitätsprüfung“ ist ein Business Service, der komplexe Rentabilitätsberechnungen durchführt und auf die Dienste der Services „Vertrag“ und „Schaden“ zugreift. Der Service „Kulanz bearbeiten“ übernimmt als Process Service die Orchestrierung.

Da aus betrieblichen Gründen eine reine Web Services-Lösung für die Kooperationspartner keine Option darstellte, kamen zur Realisierung der SOA-Alternative, Standardbasierende Technologiekombinationen wie Java Enterprise (Java EE) Application Server-Lösungen und ein Workflow-Management-System (WfMS) zur Anwendung.

So wurden „Schaden“, „Vertrag“, „Produkt“, „Partner“, „Konto“ und „Rentabilitätsprüfung“ als Java EE Stateless Session Beans realisiert. Da die Fachlogik, nach denen die Prüfung der Rentabilität erfolgt, häufigen Änderungen unterzogen ist, soll aus Flexibilitätsgründen in nachfolgenden Forschungsarbeiten der Service „Rentabilitätsprüfung“ mittels eines regelbasierten Ansatzes umgesetzt werden. Der Service „Kulanz bearbeiten“ wurde als Prozess modelliert. Die Instanziierung der im XPDL-Format vorliegen-

den Prozessbeschreibungen und deren Ausführung erfolgt durch die Workflow Engine des WfMS.

4 Business Activity Monitoring (BAM)

4.1 Grundlagen des BAM

Der Begriff Business Activity Monitoring wurde im Jahr 2003 durch die Gartner, Inc. geprägt (vgl. [HF02]). Business Activity Monitoring stellt dabei das Erfassen, Analysieren, Melden und Alarmieren wichtiger Ereignisse in Echtzeit dar. Im Gegensatz zum Business Process Monitoring werden beim BAM nicht ausschließlich Geschäftsprozesse überwacht, sondern zum Beispiel auch Anwendungssysteme und Middleware (vgl. Abbildung 3).

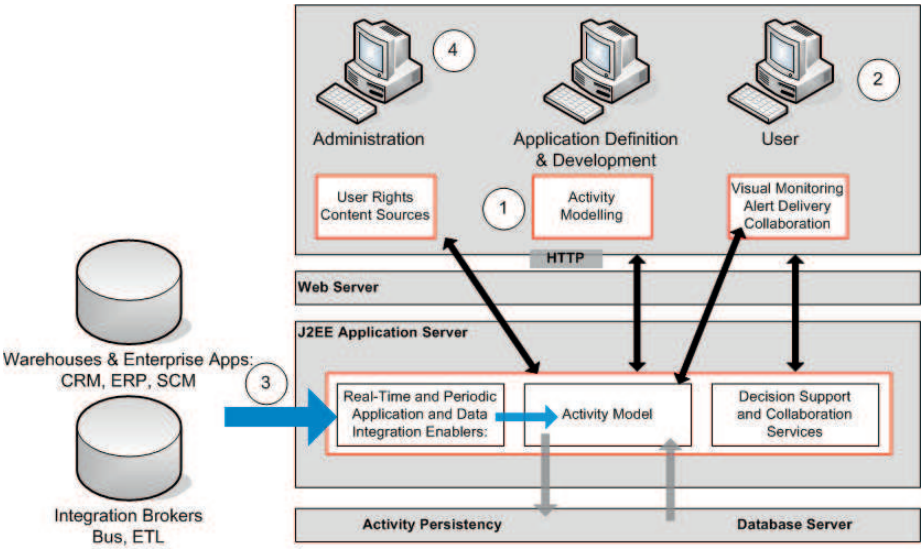

Abbildung 3: BAM nach Gartner

Im BAM muss zwar auf die anfallenden Daten in Echtzeit reagiert werden, dies führt aber nicht direkt zu einer Änderung am Ablauf des Geschäftsprozesses. Aus diesem Grund kann weiterhin ein Service getriebenes Architekturkonzept wie SOA eingesetzt werden. Allerdings müsste dieses Architekturkonzept dafür um eine Komponente zum Entdecken von Ereignissen in Echtzeit erweitert werden, wie es zum Beispiel eine Event Driven Architecture (EDA) bietet. Grundsätzlich ist dies auch eine denkbare (hier nicht gewählte) Option, z.B. basierend auf Complex Event Processing (CEP)-Ansätzen.

4.2 Architekturkonzepte

Für die Umsetzung der geforderten Funktionen hätte theoretisch auch eine am Markt befindliche „Standardlösung" eingesetzt werden können. Dies wurde jedoch gleich zu Beginn von den Kooperationspartnern aus unternehmensstrategischen und den in der Einleitung genannten fachlich-technischen Gründen – hohe Komplexität der Komponenten, hoher Aufwand bei einer Integration in die vorhandene System- und Anwendungslandschaft, hoher Ressourcenbedarf – verworfen.

Als Anforderung wurde genannt, dass verschiedene einfache Ereignisse, z.B. der Start eines Prozesses bzw. einer Aktivität und der Abschluss eines Prozesses bzw. einer Aktivität überwacht werden sollen. Aus diesen wiederum lassen sich verschiedene komplexe Ereignisse ableiten wie z. B. die Bearbeitungsdauer eines Prozesses bzw. einer Aktivität oder auch die Anzahl verschiedener Prozessinstanzen während eines bestimmten Zeitraums.

Im ersten untersuchten Grobkonzept werden zu Beginn des Programmablaufs durch einen Administrator die Regeln festgelegt, die zur Bildung von komplexen Ereignissen und somit zum Überwachen von Kennzahlen benötigt werden (vgl. Abbildung 4).

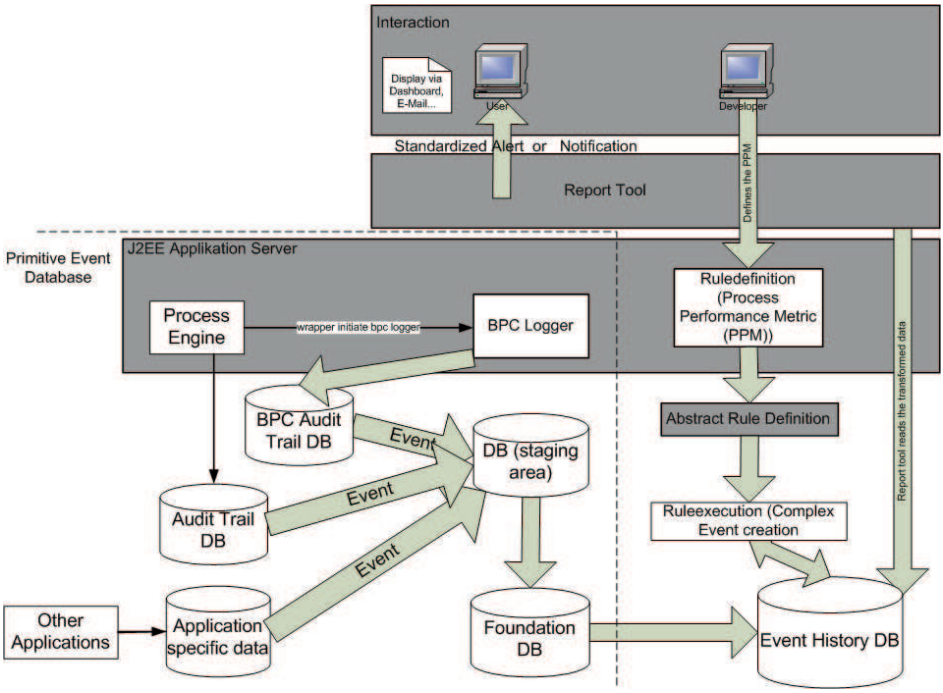

Abbildung 4: Grobkonzept mit BPC Logger und Foundation Database

Anschließend müssen die Regeln in eine abstrakte Regelsprache überführt werden. Diese abstrakte Regelsprache ermöglicht es, die Regelverarbeitung generisch zu halten. Die

mittels der abstrakten Regelsprache formulierten Regeln werden dann im Anschluss mit Hilfe eines Adapters in die produktspezifische Regelsprache überführt. Dies könnte z.B. durch Datenbanktrigger realisiert werden. Dazu würde ein Adapter an der Komponente der abstrakten Regelerzeugung für jede definierte Regel eine Implementation eines Triggers erzeugen.

Durch die Prozess Engine findet dann die Erzeugung von Ereignissen statt: So stößt die Engine über Wrapper den BPC Logger an. Dieser reichert die Daten um weitere Informationen an und speichert diese in der BPC Audit Trail DB. Aus dieser DB, der Audit Trail DB und eventuell aus applikationsspezifischen Datenbanken werden die Daten mittels ETL-Prozess in die Staging Area DB überführt und von dort durch einen ETL-Prozess in die Foundation DB geladen. Um dann einheitlich strukturierte Ereignisse in der Event History DB speichern zu können, werden die Daten erneut mittels ETL-Logik aus der Foundation DB extrahiert, transformiert und in die Event History DB geladen.

Durch die Komponente Ruleexecution werden die in der Event History DB gespeicherten Ereignisse gemäß der definierten Regeln auf Ereignismuster durchsucht und eventuell komplexe Ereignisse erzeugt, die ebenfalls in der Event History DB gespeichert werden. Das Report Tool sucht nun alle zu meldenden Ereignisse aus der Event History DB und meldet sie an die dafür zuständigen Personen. Für die Meldungen kann zum Beispiel ein Dashboard verwendet werden (vgl. Abbildung 4).

Das zweite untersuchte Grobkonzept ist ähnlich dem oben beschriebenen Grobkonzept modelliert. Der wesentliche Unterschied besteht darin, dass statt eines ETL-Prozesses zwischen der Foundation DB und der Event History DB ein Adapter zum Einsatz kommt.

Im dritten untersuchten Grobkonzept sollen die anfallenden Daten direkt in der Audit Trail DB und weiteren applikationsspezifischen Datenbanken gespeichert werden. Diese Datenbanken werden wiederum von Triggern überwacht. Von den Triggern werden die Ereignisse standardisiert und an die Regelverarbeitung übermittelt. Die Regelverarbeitung wendet die zuvor definierten Regeln an, um die gemeldeten Ereignisse zu verarbeiten und eventuell komplexe Ereignisse zu erzeugen. Von der Regelverarbeitung werden dann alle Ereignisse in der Event History DB gespeichert.

4.3 Bewertung der drei Grobkonzepte

Um eine Entscheidung zur Verwendung eines Grobkonzeptes treffen zu können, wurden diese mit Hilfe eines gewichteten Kriterienkatalogs bewertet. Die Grobkonzepte wurden dabei in sechs verschiedenen Kategorien bewertet. Diese Kategorien sind die Funktionalität, die Zuverlässigkeit, die Benutzbarkeit, die Effizienz, die Wartbarkeit und die Übertragbarkeit. Diese Kategorien wurden jeweils wiederum in mehrere Kriterien unterteilt.

Das erste Grobkonzept mit BPC Logger und Foundation Database hat eine im Vergleich zu den anderen Optionen bessere Bewertung erhalten. Dieser Entscheidung zugrunde liegt der höhere Reifegrad wegen der Wiederverwendung bestehender Komponenten und der geringere Realisierungsaufwand.

5 Prototypische Umsetzung und Evaluierung

Die Bewertung der drei Grobkonzepte auf der Grundlage des mit den Kooperationspartnern abgestimmten Kriterienkatalogs ergab eine deutliche Präferenz für das Konzept mit BPC Logger und Foundation Database (vgl. Abbildung 4), sodass in nachfolgenden Studien dieses konkretisiert und zu einem Feinkonzept weiterentwickelt wurde.

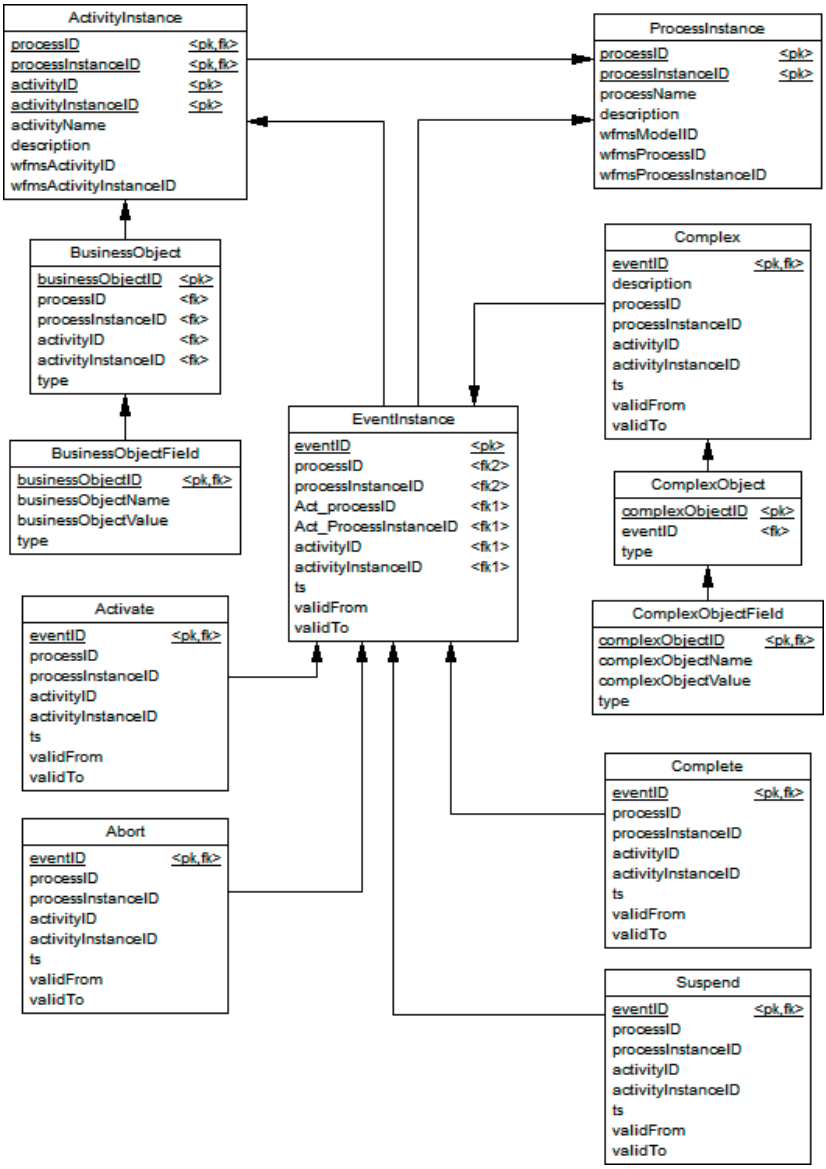

Abbildung 5: Datenbankschema der Foundation Database

In diesem Kapitel sollen zunächst die wesentlichen Konzepte zur Regelmodellierung und -verarbeitung vorgestellt und ein Einblick in die Implementierung gegeben werden. Abschließend wird auf Probleme, die sich während der Implementierung ergeben haben, eingegangen und eine Einschätzung der Laufzeiteigenschaften unserer Lösung vorgenommen.

5.1 Verarbeitung elementarer und komplexer Ereignisse

In der Event History DB werden elementare Ereignisse abgespeichert. Mittels eines Primärschlüssels kann eine Ereignisinstanz eindeutig identifiziert werden. Eine Spezialisierung der Ereignisinstanz wird über die Tabellen Activate, Complete, Abort, Suspend sowie Complex vorgenommen (vgl Abbildung 5).

Die Tabelle mit den komplexen Ereignissen wird erst im späteren Verlauf durch die auf der Event History DB arbeitenden Trigger befüllt. Um die Information, aus welchen elementaren Ereignissen ein komplexes Ereignis entstanden ist, nicht zu verlieren, wird diese in Einträgen der entsprechenden Tabellen ComplexObject in Verbindung mit der Tabelle ComplexObjectField hinterlegt. Dort kann jedoch nicht die Modellierungsvorschrift sondern nur die Zusammensetzung eines speziellen komplexen Ereignisses nachvollzogen werden. Die Modellierungsvorschrift muss in jenen Datenbanktriggern implementiert werden, die die komplexen Ereignisse erzeugen. Vorteilhaft ist bei dieser Umsetzung, dass bei neuen komplexen Ereignissen keine Anpassungen am Datenbankmodell erforderlich werden, sondern lediglich ein neuer Trigger implementiert werden muss.

Außerdem wird durch die Modellierung der Tabellen Complex, ComplexObject und ComplexObjectField vermieden, dass Daten von einem Trigger gelesen werden, um sie in einer Transaktion zu verarbeiten, während diese Daten noch von einer anderen Transaktion geändert werden und noch nicht bestätigt sind.

Durch diese Modellierung wird somit sichergestellt, dass jede Transaktion nach dem ACID-Prinzip korrekt abgearbeitet wird, da sich keine gleichzeitig stattfindenden Transaktionen gegenseitig beeinflussen.

Während der Entwicklung traten kurzzeitig Schwierigkeiten rund um die Datenkonsistenz auf. Durch lesende Datenzugriffe eines Triggers, während diese von einem anderen Trigger in einer Transaktion versucht wurden zu verändern, kam es zu Seiteneffekten. Durch die Anpassung im zugrundeliegenden Datenmodell konnte eine Vermeidung von Lese-Schreib-Konflikten sichergestellt werden.

5.2 Aufbau der eingesetzten Trigger zur Regelüberwachung

Da nach der Befüllung der Event History DB alle elementaren Ereignisse vorhanden sind, können im Anschluss auch die komplexen Ereignisse erzeugt werden. Mit jedem auftretenden elementaren Ereignis kann auch wieder ein komplexes Ereignis entstehen. Um diese Ereignismuster zu überwachen – also bei jedem Auftreten eines elementaren

Ereignisses eventuell auch ein komplexes Ereignis zu erzeugen – werden die Ereignismuster mittels Datenbanktriggern realisiert.

Die Durchführbarkeit dieses Vorhabens soll anhand eines beispielhaft mittels SQL implementierten Triggers aufgezeigt werden. Dafür wurde das komplexe Ereignis loggingNumberPassesOfActivityPerDay herangezogen. Dies beschreibt die Protokollierung der Anzahl der Durchläufe einer Aktivität pro Tag. Abbildung 6 zeigt den zeitlichen Ablauf des Triggers.

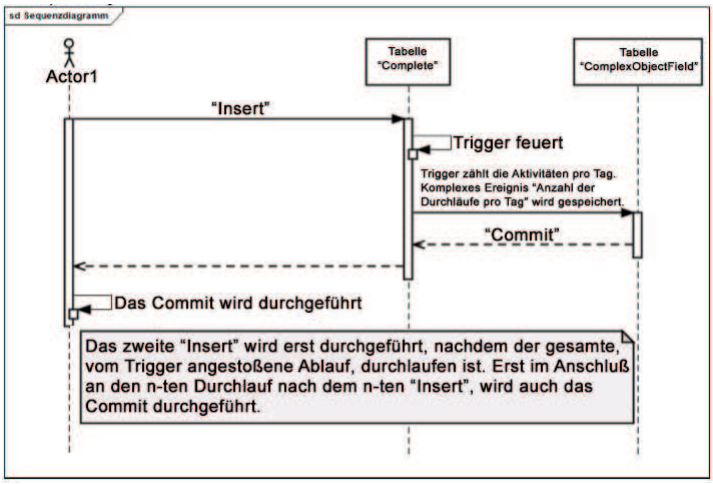

Abbildung 6Ablauf eines Triggers

Neben den komplexen Ereignissen, die aufgrund ihrer fachlichen Notwendigkeit definiert wurden, sollen auch auftretende Fehler durch ein komplexes Ereignis erkannt und verarbeitet werden. Dazu wurde ein Trigger definiert, der auf jede mögliche fehlerhafte logische Kombination von elementaren Ereignissen, wie z. B. ein elementares Ereignis, das vor einem anderen auftreten muss aber einen späteren Zeitstempel trägt, reagiert. Durch diesen Trigger werden somit fehlerhafte Eingaben durch Nutzer sowie fehlerhaft ausgeführte automatisch ablaufende Prozesse erkannt.

Um den erkannten Fehler zu protokollieren, erzeugt der Trigger wiederum ein komplexes Ereignis. In diesem Ereignis werden die fehlerhaften Parameter protokolliert.

5.3 Erzeugung der Trigger durch die Abstract Rule Definition

Damit spätere Anwender des Systems keine tiefer gehenden Datenbankkenntnisse besitzen müssen, um für jedes neu zu berücksichtigende Ereignismuster einen eigenen Trigger realisieren zu können, soll die Erzeugung der Trigger durch die Komponente - Abstract Rule Definition - erfolgen. Außerdem wird dadurch eine Entkopplung der zugrundeliegenden Technologie erreicht. Zur Nutzung dieser Komponente ist ausschließlich die Kenntnis einer abstrakten Regelsprache notwendig.

Die Erzeugung der Trigger findet über einen vordefinierten Ablauf statt. Dieser ist in der Komponente Abstract Rule Definition hinterlegt. Eine Verfeinerung dieser Komponente und ihr Zusammenspiel mit der Komponente Ruleexecution ist in Abbildung 7 zu sehen.

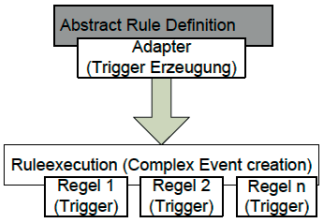

Abbildung 7: Prozess der Trigger-Erzeugung

Zu Beginn der Verarbeitung existiert ein XML-Dokument, deren Inhalt die abstrakte Definition der zu verarbeitenden Regel darstellt. Die Erzeugung dieses XML-Dokumentes erfolgt im Vorfeld manuell. In einer späteren Projektphase soll die Definition abstrakter Regeln durch ein graphisches Werkzeug erfolgen können.

Die Abstract Rule Definition stellt dabei eine Metasprache zur Beschreibung von Regeln dar. Diese Regeln sollen durch Regelcompiler in produktspezifische Regeln überführt werden. In dem entwickelten Feinkonzept wird die abstrakte Regeldefinition über den oben gezeigten Ablauf in Trigger überführt. Die Menge der Trigger bilden somit die Regeln in der Datenbank ab. Jede abstrakte Regel wird in einen Trigger in der Datenbank überführt, außerdem wird für jede Regel ein eigenes XML-Dokument angelegt. Die XML-Dokumente werden von einem Parser wie zum Beispiel JAXP geparst. Die daraus gewonnenen Informationen müssen von der Rule Builder-Komponente weiter verarbeitet und in Trigger-Funktionen überführt werden. Das Anlegen der Trigger in der Datenbank kann zum Beispiel über eine JDBC-Schnittstelle stattfinden.

5.4 Verwendung einer Abstrakten Regelsprache

In der Komponente Abstract Rule Definition muss eine abstrakte Regelsprache eingesetzt werden, um aus einem Ereignismuster einen Trigger zu erzeugen. Dieser Zwischenschritt dient dazu, die Ereignismuster unabhängig von der zugrundeliegenden Technologie entwickeln zu können. Ereignismuster werden definiert und in ein XML-Dokument, oder eine grafische Benutzeroberfläche eingegeben, ohne dass ein Wechsel des Datenbanksystems darauf Auswirkungen hat. Die abstrakte Regelsprache übernimmt also die Aufgabe vorgegebene Ereignismuster in einen datenbankspezifischen Trigger zu überführen (vgl. Abbildung 7). Eine mögliche Umsetzung einer abstrakten Regelsprache stellt RuleML (vgl. [BTW01]) dar.

Ebenso wie die Trigger muss auch die verwendete abstrakte Regelsprache einige Anforderungen erfüllen, um alle denkbaren Ereignismuster interpretieren und umwandeln zu können. Es ist nötig, dass mathematische Operatoren (+, -, *, /), logische Operatoren und gängige Vergleichsoperatoren direkt unterstützt oder durch die Verkettung mehrerer Bedingungen mittels eines UND-Operators gebildet werden können. Zudem müssen ein SUM()-Operator und ein CAUSES-Operator unterstützt werden.

5.5 Implementierung und Evaluierung

Die erarbeiteten Konzepte wurden durch eine prototypische Umsetzung evaluiert. Hierzu wurde eine Implementierung der Event History DB sowie der beschriebenen Trigger auf der Basis Der Oracle DB vorgenommen. ETL-Prozesse wurden mittels des Pentaho Data Integration Tools umgesetzt. Die Bestandteile wurden separaten Funktionstests unterzogen. Um die prototypische Umsetzung in Gänze zu evaluieren wurden umfangreiche Lasttests durchgeführt. Verwandt wurden hierfür virtuelle Maschinen, die wiederum auf mehreren herkömmlichen Arbeitsplatz PCs betrieben wurden. Die Leistungsmessung erfolgte auf Basis generierter SQL-Skripts für verschiedene fiktive Eingaben. Somit ließ sich sowohl ermitteln, welche Bearbeitungszeiten für einzelne Eingaben benötigt werden, als auch, wie viele Eingaben parallel erfolgen können, ohne das die Bearbeitungszeiten merkbar negativ beeinflusst wurden.

Die Dauer der Erzeugung einzelner Ereignisse wie auch die Bearbeitung vollständiger Workflows befand sich dabei im Millisekundenbereich. Insgesamt war die Bearbeitung von ca. 1.000 Workflows pro Minute möglich ohne die Bearbeitungszeiten negativ zu beeinflussen. Dabei belief sich die Datenmenge der abgearbeiteten Workflows inklusive der zugehörigen Ereignisse auf ca. 1 MB pro 1.000 Workflows.

Ein abschließender Performancevergleich mit am Markt befindlichen „Standardlösungen“ hat nicht stattgefunden, da der Einsatz einer solchen Lösung von den Kooperationspartnern nicht in Betracht gezogen wurde und dementsprechend keine Alternative darstellte. Auch wurden die zunächst entwickelten Grobkonzepte, nachdem sie verworfen worden sind, nicht umgesetzt, sodass auch hier keine vergleichenden Performancetests stattgefunden haben.

Zusammenfassend standen Performanceuntersuchungen nicht im Fokus unserer Arbeiten. Vielmehr besaßen für die Kooperationspartnern die Kriterien Wiederverwendung bestehender Komponenten, Flexibilität und Offenheit des Gesamtsystems eine deutlich höhere Priorität als etwa die Performance (vgl. Kapitel 4).

6 Zusammenfassung und Ausblick

Unternehmen stehen vor erheblichen Herausforderungen: Um im globalisierten Markt bestehen zu können, müssen in immer kürzer werdenden Zyklen vorhandene Geschäftsprozesse flexibel an sich schnell ändernde Anforderungen angepasst werden. Serviceorientierte Architekturen scheinen geeignet, diese Anforderungen an Flexibilität und Agilität erfüllen zu können. Technisch kann eine SOA zum Beispiel auf Basis von Web Services-Standards implementiert werden. Da SOA jedoch ein technologieneutrales Konzept ist, können gut auch andere technische Lösungen zum Einsatz kommen, die ggf. besser zur IT-Landschaft eines Unternehmens passen.

In unseren Arbeiten wurde deshalb eine SOA auf Basis des Java EE-Standards und einem Workflow-Management-System umgesetzt. Im SOA-Kontext realisieren dann die Prozessbeschreibungen die sogenannten Prozess-Services, die durch die Workflow En-

gine instanziiert und ausgeführt werden. Ein Geschäftsprozess-Monitoring unterstützt die Effizienzkontrolle der SOA. Im Themenkontext SOA, WfMS und Geschäftsprozess-Monitoring liegen folglich die Beiträge dieser Arbeit. Diese sind:

- verschiedene Architekturkonzepte für ein Geschäftsprozess-Monitoring in Service-orientierten Architekturen mit WfMS (vgl. Unterkapitel 4.2)
- eine Bewertung der Konzepte mit den Methoden der Nutzwertanalyse hinsichtlich Wiederverwendbarkeit, Angemessenheit und Realisierungsaufwand (vgl. Unterkapitel 4.3)
- ein Prototyp zur Umsetzung des gewählten Architekturkonzepts (vgl. Kapitel 5) sowie die Evaluierung der Ergebnisse anhand eines umfangreichen Fallbeispiels aus der VAA, dem Geschäftsprozess „Kulanz bearbeiten".

Auf Basis der erfolgreich erzielten Ergebnisse erweitern wir in aktuellen Arbeiten unsere Gesamtarchitektur um den Aspekt der Geschäftsregeln (Business Rules Management) und untersuchen die Verknüpfung der bisherigen Ergebnisse mit Cloud-basierten IT-Umgebungen.

Literaturverzeichnis

[Ag07] Agrawal, A. et al. (2007), Web Services Human Task (WS-HumanTask), Version 1.0, Technical report, Active Endpoints, Adobe, BEA, IBM, Oracle, SAP.

[Be07] Becker, M. (2007), Geschäftsprozess-Controlling auf der Basis von Business-Intelligence-Konzepten und Data-Warehouse-Systemen: Architekturen , Datenmodell, Vorgehensmodell und Anwendungsszenarien, Shaker: Aachen.

[BTW01] Boley, H., Tabet, S., Wagner, G. (2001), Design Rationale for RuleML: A Markup Language for Semantic Web Rules, DFKI GmbH, Kaiserslautern.

[De06] De Jong, P. (2006), Going with the Flow, in: ACM Queue, pp 26--32.

[Du08] Dunkel, J., Eberhart, A., Fischer, S., Kleiner, C., Koschel, A. (2008), Systemarchitekturen für Verteilte Anwendungen, Hanser, München

[En08] Engels, G., Hess, A., Humm, B., Juwig, O., Lohmann, M. und Richter, J.-P. (2008), Quasar Enterprise: Anwendungslandschaften serviceorientiert gestalten, dpunkt.verlag, Heidelberg.

[Er05] Erl, T. (2005), Service-Oriented Architecture: Concepts, Technology, and Design, 5th edt., Prentice Hall, NJ.

[GD13] GDV (2001), VAA Final Edition. Das Objektorientierte Fachliche Referenzmodell Version 2.0, Berlin, http://www.gdv-online.de/vaa/vaafe_html/dokument/otrm.pdf, Abruf am 05.12.2013.

[GHS95] Georgakopoulos, D., Hornick, M., Sheth, A. (1995), An Overview of Workflow Management: From Process Modelling to Workflow Automation Infrastructure, in: Distributed and Parallel DBS v. 3, p. 119--153, USA.

[Gr06] Greiner, T., Duster, W., Pouatcha, F., von Ammon, R. (2006), Business Activity Monitoring of Norisbank Taking the Example of the Application easyCredit and the Future Adoption of Complex Event Processing (CEP), in: Proc. IEEE Services Comp. Workshops SCW '06, pp 83.

[Ha11] Hausotter, A., Kleiner, C., Koschel, A., Zhang, D., Gehrken, H. (2011), Always Stay Flexible! WfMS-independent Business Process Controlling in SOA, in: Enterprise Distributed Object Computing Conference Workshops (EDOCW), 2011, 15th IEEE International', pp 184–193.

[HF02] Hellinger, M., Fingerhut, S. (2002), Business Activity Monitoring: EAI Meets Data Warehousing, in: EAI Journal, pp 18--21.

[Ho95] Hollingsworth, D. (1995), The Workflow Reference Model, Wf Management Coalition, UK.

[HV09] Henning, M., Vinoski, S. (2009), Advanced CORBA Programming with C++, Add.Wesley, USA.

[JFK04] Jeng, J. J., Flaxer, D., Kapoor, S. (2004), RuleBAM: a rule-based framework for business activity management, in: Proc. IEEE Intl. Conference on Services Computing (SCC 2004), pp 262--270.

[JP06] Jobst, D., Preissler, G. (2006), Mapping clouds of SOA- and business-related events for an enterprise cockpit in a Java-based environment, in: PPPJ '06: Proceedings of the 4th international symposium on Principles and practice of programming in Java, ACM, NY, USA, pp 230--236.

[KBS07] Krafzig, D., Banke, K., and Slama, D. (2007), Enterprise SOA: Best Practices für Serviceorientierte Architekturen -- Einführung, Umsetzung, Praxis, mitp-Verlag, Heidelberg.

[Kl05] Kloppmann, M.et al. (2005), WS-BPEL Extension for People -- BPEL4People, Technical report, SAP AG, IBM Corp.

[Ku05] Kung, P., Hagen, C., Rodel, M., Seifert, S. (2005), Business process monitoring and measurement in a large bank: challenges and selected approaches, in: Proc. Sixteenth International Workshop on Database and Expert Systems Applications, pp 955--961.

[LN05] Lomow, G., Newcomer, E. (2005), Understanding Service-Oriented Architecture (SOA) with Web Services, Addison Wesley, USA.

[Mu01] zur Muehlen, M. (2001), Process-driven management information systems.

[Mu02] zur Muehlen, M. (2002), Workflow-based Process Controlling, Logos Verlag: Berlin.

[MWL08] Martin, D., Wutke, D., Leymann, F. (2008), Synchronizing control flow in a tuplespace-based, distributed workflow management system, in: Proceedings of the 10th international conference on Electronic commerce (ICEC '08). ACM, NY, USA.

[OA06] OASIS (2006), Reference Model for SOAs 1.0, 2006, http://docs.oasis-open.org/soa-rm/v1.0.

[OA07] OASIS (2007), WSBPEL -- Web Services Business Process Execution Language Version 2.0, OASIS Standard, 2007. -- combining data warehouses and workflow technology. in: Proc. 4th Intl. Conf. on EC Research (ICECR-4), B. Gavish, Ed., Dallas, TX, Nov 2001, pp 550--566.

[Pe08] Pedrinaci, C., Lambert, D., Wetzstein, B., van Lessen, T., Cekov, L., Dimitrov, M. (2008), SENTINEL: a semantic business process monitoring tool, in: OBI '08: Proc. 1st international workshop on Ontology-supported business intelligence, ACM, New York, NY, USA, pp 1--12.

[Sc08] Schwichtenberg, H.(2008), Microsoft .NET 3.5 - Crashkurs: Presentation, Communication, Workflow, Visual Basic 2008, Visual C\# 2008, ADO.NET 3.5, ORM, LINQ, WPF, WCF, Microsoft Press, München.

[We09] Wetzstein, B., Leitner, P., Rosenberg, F., Brandic, I., Dustdar, S., Leymann, F. (2009), Monitoring and Analyzing Influential Factors of Business Process Performance, in: Proc. IEEE International Enterprise Distributed Object Computing Conference EDOC '09, pp 141--150.

[WML08] Wutke, D., Martin, D., Leymann, F. (2008), Model and infrastructure for decentralized workflow enactment, in: Proc. ACM SAC '08. ACM, NY, USA, pp 90--94.

[WSG07] Wagner, pp, Sonntag, M., Gostmann, H. (2007), Vergleich von Business Activity Monitoring Werkzeugen, Fachstudie, IAAS, Universität Stuttgart, Stuttgart.