Petrinetz-basierte Modellierung und Steuerung unternehmensübergreifender Geschäftsprozesse[1]

M. Merz, K. Müller-Jones, W. Lamersdorf

Universität Hamburg

Fachbereich Informatik

Vogt-Kölln-Str. 30 D-22527 Hamburg

[merz|kmueller|lamersd]@informatik.uni-hamburg.de

Zusammenfassung

Die Entwicklung einer Software-gestützten Ablaufsteuerung unternehmensübergreifender Geschäftsprozesse ist mit erheblich höheren Koordinationskosten verbunden als eine abteilungs- oder unternehmensinterne. Die gemeinsame Koordinationsinfrastruktur bietet daher Potential für eine weitere Effizienzsteigerung.

Der vorliegende Beitrag zeigt anhand eines exemplarischen Geschäftsprozesses, wie im Projekt COSM (Common Open Service Market) eine abstrakte Ablauf*modellierung* auf der Basis gefärbter Petrinetze in eine Ablauf*steuerung* automatisch umgesetzt wird. Bei reduziertem Konfigurationsaufwand sind so auch Teilnehmer externer Unternehmen in diesen Geschäftsprozeß integrierbar. Ebenso können Folgeprozesse über Unternehmensgrenzen hinweg ausgelöst werden.

1 Motivation

Die innerbetrieblichen Koordination von Geschäftsprozessen wird durch zentralisierte Weisungsbefugnis erreicht und aufgrund von Unternehmensstandards unterstützt. Eine derartige organisatorische Infrastruktur wird durch die Einrichtung einer Software-basierten, automatisierten Steuerung von Geschäftsprozessen unterstützt. Unternehmensübergreifende Geschäftsprozesse verursachen hingegen aufgrund der gegebenen Autonomie der beteiligten Organisationen erheblich höheren Koordinationsaufwand, wenn eine automatisierte Steuerung angestrebt wird. Hierbei gilt es nicht nur, Heterogenität im Bereich von Software-Anwendungen zu überbrücken, sondern auch organisationale Unterschiede zwischen unternehmensspezifischen Prozessen zu berücksichtigen. Während dabei die Vereinheitlichung von Dokumentenformaten, auf die sich die aktuelle Standardisierungsbemühung vor allem konzentriert (z.B. im Bereich der EDI-Standards) [Kimb91], einen Fortschritt erzielen mag, so scheint doch die Integration von Geschäftsprozessen über Unternehmensgrenzen hinweg nur im Rahmen langfristiger Kooperationen sinnvoll, wie z.B. zwischen Automobilherstellern und ihren Zulieferern. Ein Hinderungsfaktor ist hierbei das Fehlen einer gemeinsamen Kommunikationsinfrastruktur, welche über den reinen Nachrichtentransfer hinaus auch die Koordination von Einzelaktivitäten, aus denen sich Geschäftsprozesse zusammensetzen, steuert. Ähnlich der gemeinsamen Konvention eines EDI-Standards, würde eine derartige Infrastruktur die Rüstkosten der Konfiguration bei den beteiligten Unternehmen senken, so daß eine Integration der Geschäftsprozesse auf dieser Basis vorteilhaft werden kann. Diese Koordinationsinfrastruktur sollte andererseits so generisch bleiben, daß die Einbindung existierender Anwendungs-Systeme (*legacy code*) weitgehend unterstützt wird.

Ein unternehmensübergreifender Geschäftsprozeß kann die Einbeziehung externer Teilnehmer (z. B. Sachverständige, Zeugen etc.) in einen ansonsten internen Arbeitsablauf erfordern. Die

---

[1] Apeared in: GI/SI Jahrestagung ‘95, Zürich, 18.-20. Sept. 1995

Realisierung einer Prozeßsteuerung sollte hier eine flexible Bereitstellung von Werkzeugen für die Einbindung des extenen Teilnehmers umfassen. In anderer Form kann eine überbetriebliche Prozeßintegration auch durch das Initiieren von Folgeprozessen beim Partnerunternehmen erreicht werden. Hierbei sollte die Koordinationsinfrastruktur so gestaltet sein, daß einzelne Aktivitäten eines Prozesses wiederum Folgeprozesse auslösen können.

1.1 Ein exemplarischer Geschäftsprozeß

Um den Anwendungshintergrund für die nachfolgenden Ausführungen zu verdeutlichen und um den präsentierten Ansatz anhand einer realistischen Fallstudie zu prüfen, sei an dieser Stelle ein Geschäftsprozeß aus dem Luftfahrtbereich dargestellt: Eine Fluggesellschaft ist verpflichtet, nach jedem absolvierten Flug einen *Flugbericht* (FB, engl. flight report, FR) anzufertigen. Dieses Dokument dient u.a. der Beschreibung besonderer Vorkommnisse im technischen Umfeld oder im Zusammenhang mit Fluggästen. Der FR wird vom Flugkapitän ausgefüllt und datentechnisch in einen *FR-Pool* eingestellt. Periodisch werden Flugberichte daraufhin von einem Sachbearbeiter bewertet und je nach Vorkommnis einem oder gleichzeitig mehreren Experten aus unterschiedlichen Fachbereichen zur Abgabe einer Stellungnahme weitergeleitet. Diese Experten gehören z.B. dem Wartungs- odere Sicherheitspersonal, dem Betriebsrat oder einem Zuliefererunternehmen an. Die involvierten Experten liefern daraufhin ihre Stellungnahme wieder beim Sachbearbeiter ab, so daß dieser anschließend eine Vorklassifizierung des Vorfalls durchführt und den Flugbericht entweder direkt auf optische Medien archiviert oder einem dedizierten Entscheidungsgremium zukommen läßt. Als Ergebnis dieses Treffens kann entweder ein weiterer Geschäftsprozeß zur Behandlung des Problemfalles in Gang gesetzt oder der Vorgang an den Sachbearbeiter zurückgeleitet werden mit dem Auftrag, weitere oder verfeinerte Stellungnahmen von den Experten einzuholen. Der u.U. initiierte Folgeprozeß kann beispielsweise dem betrieblichen Vorschlagswesen eines Partnerunternehmens zugeordnet sein.

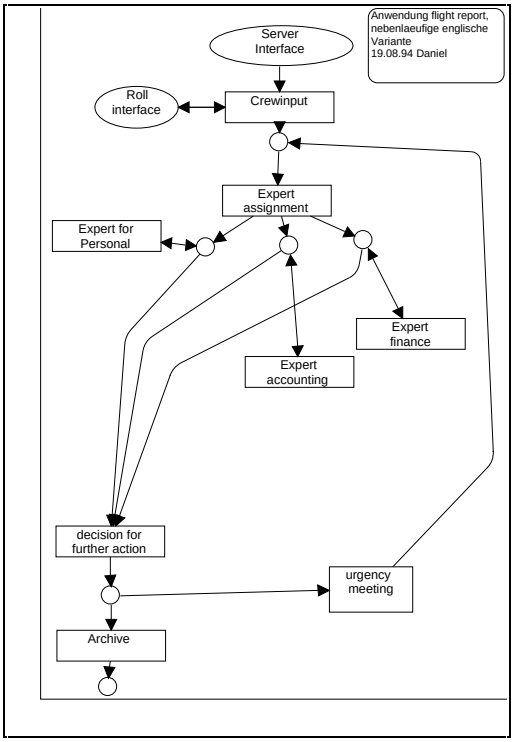

Abb. 1: Petrinetz-Modell für den FR-Vorgang

Abb. 1 stellt den Ablauf des FR-Prozesses anhand eines Petrinetzes dar. Aus Gründen der Übersichtlichkeit wurde bei der Darstellung auf die in diesem Beitrag für Ablaufpläne verwendeten *gefärbten Petrinetze* (coloured petri nets, CPN) verzichtet [Jens92].

1.2 Gliederung des Beitrags

Im zweiten Kapitel werden die für diesen Beitrag relevanten Grundlagen im modelltheoretischen Bereich sowie bezüglich einer geeigneten Kommunikationsinfrastruktur dargestellt. Darauf basierend stellt das dritte Kapitel ein Modell zur generischen Unterstützung der Ablaufsteuerung dar. Exemplarisch wird dabei die automatische Koordination von Aktivitäten des FR-Prozesses dargestellt. Den aktuellen Stand der Implementierung beschreibt das vierte Kapitel.

2. Basistechnologie und Hintergrund des Ansatzes

Grundlage des unten beschriebenen Ansatzes zur Ablaufsteuerung ist das Klient/Server-Modell, bei dem Klienten-Anwendungen auf der Basis des entfernten Prozeduraufrufes (Remote Procedure Call, RPC) Server-Operationen aufrufen. Ein Geschäftsprozeß entspricht dabei systemtechnisch einer Folge von RPCs bei einem dedizierten *Anwendungsserver*. Im Gegensatz zu einer Vielzahl von Implementierungen (z. B. [Rein93]) ist im vorgestellten Modell keine zentrale Vergabe von Aufträgen durch einen *Dispatcher* realisiert, sondern das Prinzip der *Auftragsanziehung*: Hier befindet sich der Teilnehmer eines Geschäftsprozesses in der Rolle des Klienten, welcher von der Prozeßsteuerung die Berechtigung zur Ausführung (den unten definierten *Ablaufplan*) einer Aktivität anfordert.

Vor der Einführung des zugrundeliegenden Modells der Ablaufsteuerung soll an dieser Stelle der Begriff des *Geschäftsprozesses*, wie er im Rahmen dieses Beitrages verstanden wird, definiert werden:

Hierbei setzt sich ein Geschäftsprozeß, bzw. ein *Ablauf*, aus einer Menge von *Aktivitäten* A, einer Menge von *Rollen* R sowie der Abbildung $A \rightarrow R$, welche jeder Aktivität eine *berechtigende Rolle* zuweist, zusammen. Benutzer können in beliebigen Rollen agieren, so daß eine N:M-Relation zwischen Benutzern und Rollen besteht. Ferner definiert ein Geschäftsprozeß einen *Kontrollfluß* zwischen Aktivitäten, welcher auch Zyklen von Aktivitäten zuläßt. Eine Aktivität gilt als *bereit*, wenn gemäß einer Präzedenzrelation $A_{t1} \Rightarrow A_{t2}$ alle direkt vorgelagerten Aktivitäten beendet sind, und ihre Vorbedingung zum Wert `True` evaluiert. Jede Aktivität setzt sich wiederum aus folgenden Komponenten zusammen (in Anlehnung z.B. an [Rein93]):

- Einer *Aktion* als der eigentliche Kern der Aktivität, welcher einen externen Effekt bewirkt,
- einer *Vorbedingung* als Prädikat auf dem aktuellen Evaluationszustand des Ablaufplans,
- Eingangsdaten, welche als Parameter für die Aktion verwendet werden sowie
- Ausgangsdaten, als Resultat der Aktion.

Aktivitäten können nebenläufig stattfinden, solange sie nicht im Konflikt miteinander stehen, d.h. wenn sie nur lesend auf gemeinsame Ressourcen zugreifen. Neben dem Kontrollfluß findet zwischen einzelnen Aktivitäten ein Datenfluß statt, wobei Ausgangsdaten vorgelagerter Aktivitäten als Aktions-Parameter nachgelagerter verwendet werden. Auf diese Weise findet eine indirekte Kommunikation durch Operationen auf gemeinsam genutzen Datenstrukturen statt.

Der *Ablaufplan* eines individuellen Geschäftsprozesses ist seine zur Laufzeit des Steuerungssystems verfügbare Ablauf*beschreibung* sowie auch die *Repräsentation* des aktuellen Abarbeitungs*zustandes*. Sie legt anhand einer eingebetteten Petrinetz-Repräsentation den Kontrollfluß zwischen Aktivitäten fest und enhält Informationen über die Schnittstelle des Anwendungsservers, dem der Prozeß zugeordnet ist, sowie zusätzliche Informationen, die weiter unten im Zusammenhang mit dem *generischen Klienten* beschrieben werden. Die

Berechtigung zur Ausführung einer Aktivität geht einher mit dem Erhalt des Ablaufplanes. Aus der gleichen CPN-Spezifikation resultierende Ablaufpläne gehören einem *Ablauftyp* an.

Ein RPC-Dienst, welcher den Transfer von Ablaufplänen und für Klienten den Zugriff auf Anwendungssysteme realisiert, sollte in hohem Maße der dynamischen Typisierung von Aufrufparametern gerecht werden: Da bei der Einbeziehung externer Teilnehmer zur Laufzeit Bindungen zu Anwendungsdiensten eingegangen werden, deren operationale Schnittstelle a-priori unbekannt ist, sind statisch typisierte Mechanismen wie die Generierung von *Stubs* beim Sun ONC RPC oder beim *Distributed Computing Environment* der OSF (DCE) [Schi93] ungeeignet. Eine Klientenkomponente sollte für den Zugriff auf Anwendungsdienste nicht neu kompiliert und gebunden werden müssen. Dieser Installationsaufwand würde gerade die Kosten der unternehmensübergreifenden Koordination erhöhen.

Das *Dynamic Invocation Interface* (DII), welches ein Bestandteil der *Common Object Request Broker Architecture* (CORBA) [OMG91] ist, stellt eine geeignete Grundlage für den Entwurf heterogener, verteilter Klient/Server-Anwendungen dar. Es bietet eine standardisierte Schnittstelle zur Definition von Repräsentationen für dynamisch typisierte Operationsaufrufe. Parameter liegen während des Prozeduraufrufes in Form einer *Named Value List* vor, welche Software-Komponenten, die direkt oder indirekt am Aufruf beteiligt sind, die Möglichkeit bietet, Parameterwerte aufzulisten, zu inspizieren, zu löschen oder um zusätzliche Parameter zu ergänzen. In Verbindung mit dem Ablaufplan und der dort enthaltenen Schnittstellenbeschreibung eines Anwendungsservers kann ein Klient von einem beliebigen Knoten im Netzwerk aus Operationsaufrufe auf der Basis des DII wiederum typsicher durchführen.

3 Modell der Ablaufsteuerung

Grundvoraussetzung für das in diesem Beitrag dargestellte Klient/Server-Modell ist eine Kommunikationsinfrastruktur, die es Klientenanwendungen ermöglicht, basierend auf dem CORBA DII Server-Operationen aufzurufen. Hierbei findet im Modell eine Trennung in drei Systemkomponenten statt: Anwendungsserver, Koordinationsserver und Klienten [MMML94] (Abb. 2).

3.1 Der Anwendungsserver

Definitionsgemäß besteht ein Geschäftsprozeß aus einer Menge von Aktivitäten, deren Aktion der entfernte Aufruf einer Server-Operation ist. Diese Aufrufe erfolgen bei einem oder mehreren *Anwendungsservern*, welche jeweils das Anwendungssystem eines Geschäftsprozesses implementieren. Für den Fall des FR-Managements stellt der FR-Server beispielsweise die Operationen `CrewInput`, `ExpAssign` etc. bereit. Ein Anwendungsserver kann sich aus einem bestehenden Software-Modul und dediziertem Schnittstellenkode zusammensetzen, der den Zugang für Klientenaufrufe realisiert. Zielsetzung ist hierbei, bestehende Software-Systeme (*legacy code*) unverändert für Klienten nutzbar zu machen. Eine dem Kontrollfluß entsprechende Steuerung der Klientenzugriffe erfolgt dabei durch einen vom Anwendungsserver separierten *Koordinationsserver*:

3.2 Der Koordinationsserver

Dieser Server verwaltet für jeden einzelnen Prozeß einen dedizierten Ablaufplan einschließlich des jeweiligen Evaluationszustandes. Dieser Ablaufplan enthält unter anderen die Petrinetz-Repräsentation, welche über zulässige Folgeoperationen informiert. Die beim Koordinationsserver vorliegenden Ablaufpläne befinden sich je nach Prozeß in unterschiedlichen Evaluationszuständen und können verschiedenen Ablauftypen angehören. Anwender greifen unter Angabe ihrer Rollenidentifikation auf den Koordinationsserver zu, so daß dieser in der Lage ist, einen für den Anwender zulässigen Ablaufplan zu finden und an

diesen auszuliefern. Aufgabe des Koordinationsservers ist somit, sicherzustellen, daß ein Ablaufplan nur dann ausgeliefert wird, wenn der Empfänger aufgrund seiner Rolle zur weiteren Abarbeitung berechtigt ist. Nach der Ausführung der Aktion (des RPC beim Anwendungsserver) sendet der Klient den Ablaufplan an den Koordinationsserver zurück. Aufgrund der durchgeführten Operationsaufrufe ist beim Ablaufplan inzwischen eine Zustandsänderung eingetreten, so daß der Kooperationsserver ermitteln kann, für welche Rollen der Ablaufplan anschließend zur Verfügung steht.

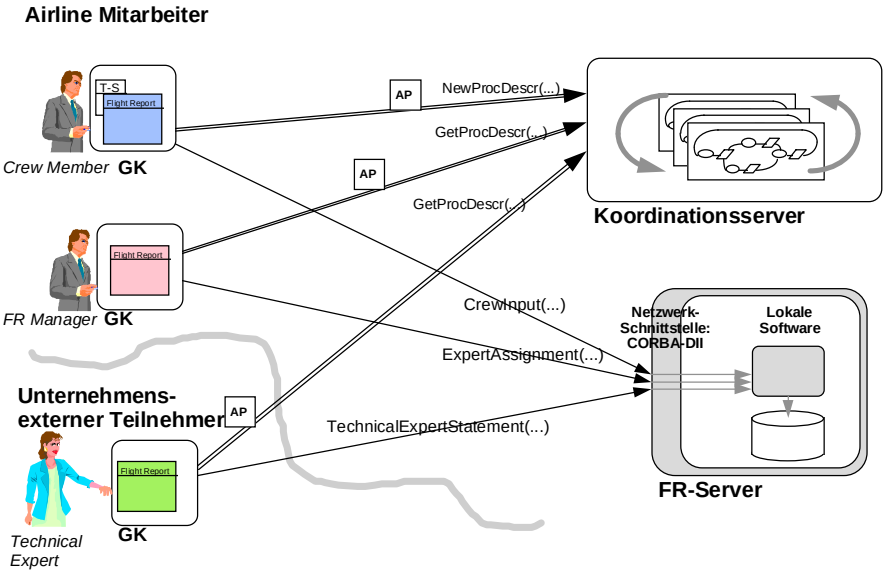

Abb. 2: Die Anforderung von Ablaufplänen vom Koordinationsserver durch den generischen Klienten und die Ausführung entfernter Operationen beim Anwendungsserver

Durch den Koordinationsserver kann ein Anwender einen neuen Ablaufplan mit einem Initialzustand generieren lassen, wenn die Berechtigung für die Initialisierungstransition gegeben ist. Analog wird ein Ablaufplan aus dem Betrieb entfernt und archiviert, wenn er in den Endzustand eingetreten ist - in diesem Zustand ist der betreffende Prozeß beendet.

Der Koordinationsserver wird durch die Klienten über zwei Operationen zum Zugriff auf Ablaufpläne gesteuert:

- NewProcessDescription(RoleID, ProcessType) liefert als Ergebnis einen initialisierten Ablaufplan für den spezifizierten Prozeßtyp
- GetProcessDescription(RoleID, ProcessType) liefert einen vom Koordinationsserver ermittelten Ablaufplan. Es ist der internen Verwaltung überlassen, welcher Ablaufplan geliefert wird.

Zusätzlich besitzt der Koordinationsserver eine Management-Schnittstelle, über die Ablaufpläne verwaltet werden können.

3.3 Der generische Klient

Um die geforderte Flexibilität auf der Seite des Klienten zu erlangen, ist eine Softwarekomponente erforderlich, welche für den Anwender unabhängig vom jeweiligen Prozeß den korrekten Aufruf der zulässigen Server-Operation steuert. Die Unabhängigkeit dieses *generischen Klienten* (GK) besteht sowohl bzgl. der Syntax der Anwendungsserver-schnittstelle als auch bzgl. der Semantik einzelner Aktivitäten. Dem Anwender wird über die graphische Benutzerschnittstelle des generischen Klienten lediglich die Information bereitgestellt, die durch den Ablaufplans repräsentiert ist [MML94]:

- ein dynamisch generiertes Dialogfenster zur Dateneingabe,

- aufrufbare Server-Operationen, welche durch das Betätigen von Schaltflächen ausgeführt werden können (*Call-Buttons*), sowie
- Bindungen zu weiteren Diensten können aufgrund von Server-Referenzen eingegangen werden, die im Dialogfenster angezeigt werden (*Binding-Buttons*).

Unabhängig vom Einsatz im Bereich der Ablaufsteuerung bietet der generische Klient die Möglichkeit der dynamischen Bindung an beliebige Dienste, welche in der Art der Anwendungsdienste eine Beschreibung ihrer operationalen Schnittstelle bereitstellen. Da auch der Koordinationsserver selbst für den generischen Klienten auf diese Weise als Anwendungsserver zugreifbar ist, kann zunächst eine Operation des Koordinationsservers aufgerufen werden. Dies führt der Benutzer über den generischen Klienten durch Drücken der mit der Operation `GetProcDescr` verbundenen Schaltfläche durch. Als Resultat wird ein Ablaufplan geliefert, anhand dessen wiederum eine Benutzerschnittstelle für den jeweiligen Anwendungsdienst generiert wird.

3.4 Ein Beispielablauf für den Flugbericht-Prozeß

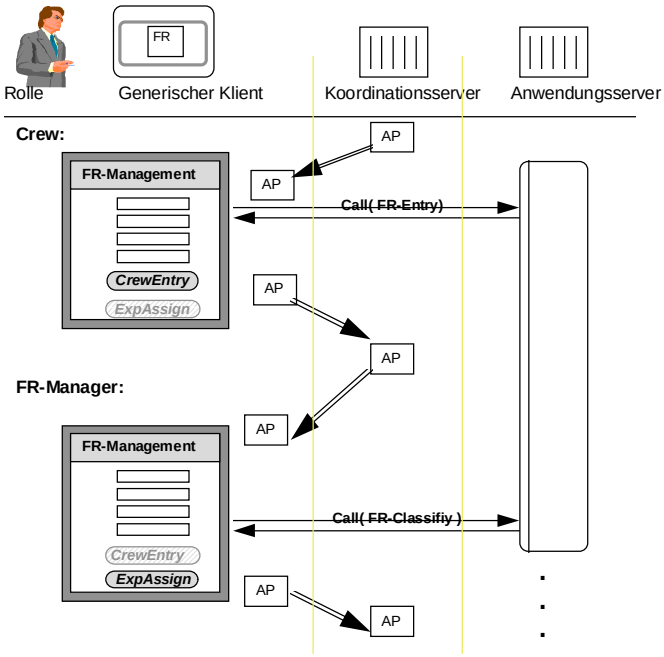

Abb. 3: Kontrollfluß zwischen generischem Klient, Koordinations- und Anwendungsserver

Der Ablauf der Prozeßsteuerung sei im folgenden am Beispiel des FR-Vorgangs erläutert: Der FR-Vorgang beginnt mit dem Zugriff des CREWMEMBERS auf den Koordinationsserver, um einen neuen Ablaufplan generieren und an den Klienten senden zu lassen (`NewProcDescr`). Dort wird aus der Schnittstellenbeschreibung des Ablaufplans ein Dialogfenster generiert. Durch das Betätigen der Schaltfläche `CrewEntry` ruft das CREWMEMBER beim Anwendungsserver (Hier dem FR-Server) die entsprechende Operation auf und trägt die Daten des Flugberichts ein. Hierbei ergeben sich durch diesen RPC Modifikationen am Ablaufplan, so daß sich dieser nach dem Schalten der korrespondierenden Transition in einem Folgezustand befindet. Die Operation `ExpertAssignment` ist nun aktiviert, jedoch nur für die Rolle FR-MANAGER. Somit sind für das CREWMEMBER im aktuellen Zustand weder die Operation `FREntry` aufrufbar (da im Netz die korrespondierende Transition nicht mehr aktiviert ist) noch die Operation `FRClassify`. Sie ist zwar dem Netzzustand nach aktiviert, das CREWMEMBER besitzt jedoch nicht die für diese Transition berechtigende Rolle. Folglich kann der Ablaufplan nur an den Koordinationsserver zurückgeliefert werden. Von dort steht er im nächsten Zyklus einem FR-MANAGER zur Verfügung, welcher berechtigt ist, die Operation `ExpertAssignment` auszuführen. Abb. 3 verdeutlicht diesen Transfer des Ablaufplanes zu

wechselnden Anwendern. Wenn nach diesem Schritt der Klassifikation die nebenläufige Einholung von Stellungnahmen erfolgt, muß der Koordinationsserver den Ablaufplan duplizieren, und zwar entsprechend der Anzahl der nebenläufig auftretenden Rollen.

3.5 Prozeßintegration

Das FR-Beispiel zeigt, daß als Entscheidungsergebnis ein weiterer Prozeß, z.B. des betrieblichen Vorschlagwesens des Zulieferers, ausgelöst werden kann. Dieser Folgeprozeß kann direkt aus dem FR-Server heraus initiiert werden, indem dieser als Klient den dortigen Koordinationsserver mit der Erzeugung eines neuen Ablaufplanes (PartSuppl.NewProcessDescription(Role: FR-Server, ProcessType: VorschlProc)) bewirkt. Da der FR-Server als Klient agiert, erhält er den initialisierten Ablaufplan und muß die Operation in diesem Vorschlags-Prozeß automatisch ausführen, zu der er gemäß Zustand und Rolle berechtigt ist. Für außerbetriebliche Rolleninhaber ist für die Teilnahme an einem Geschäftsprozeß folgende Konfiguration erforderlich: erstens muß am externen Standort ein generischer Klient installiert werden und zweitens dem extenen Teilnehmer eine Rollenbezeichnung zugewiesen werden.

4. Netzentwurf und Generierung des Ablaufplans

Im folgenden ist das schrittweise Vorgehen bei der Generierung von Ablaufplänen skizziert:

Zunächst wird der Prozeßablauf anhand eines Petrinetz-basierten Werkzeuges modelliert. Hierbei bieten Werkzeuge, wie z.B. Design/CPN [MSC93], die Möglichkeit, neben der eigentlichen Modellierung auch Simulationsläufe oder z.B. Tests auf Verklemmungsfreiheit durchzuführen. Stellen und Transitionen des Netzes repräsentieren Datenstrukturen, welche später zur Laufzeit Bestandteil des Ablaufplanes sind. Kanten zu Ausgangsstellen einer Transition führen zu Output-Parametern im Ablaufplan. Das im Netzmodell mit einer Transition assoziierte Prädikat wird als Bestandteil einer Operationsbeschreibung generiert. Auf diese Weise kann anhand aller Transitionen des Petrinetzes die operationale Schnittstelle des Anwendungssdienstes beschrieben und diese Beschreibung im Ablaufplan abgelegt werden. Die initiale Markenverteilung sowie die Gesamtheit von Zustandsvariablen definieren den Ausgangszustand eines Ablaufplans.

Im zweiten Schritt erfolgt mit Hilfe eines *Rolleneditors* die Zuweisung einer berechtigenden Rolle zu jeder Operationsbeschreibung. Da seitens des generischen Klienten ein so beschiebener RPC beim Server nur durch das Betätigen einer Schaltfläche ausgeführt wird, ist zudem eine Spezifikation der Benutzerschnittstelle erforderlich, die für den jeweiligen Server generiert wird. Als weitere Komponente des Ablaufplans beschreibt sie das Layout der Benutzerschnittstelle sowie die Dialogsteuerung, d.h., wie Schaltflächen und Operationsaufrufe in Beziehung stehen.

Der in dieser Weise generierte Ablaufplan wird bei einem Koordinationsserver über dessen Management-Schnittstelle registriert. Bei jeder Erzeugung eines neuen Prozesses mit `NewProcessDescription` wird eine Kopie des Ablaufplans erzeugt.

Implementation

Die Implementierung der dargestellten Systemkomponenten basiert auf den bisherigen Ergebnissen des Projektes COSM (Common Open Service Market). Im Rahmen dieser Aktivität ist bereits der generische Klient und ein Hauptbestandteil des Ablaufplans - die *Dienstrepräsentation* - sowie verschiedene Softwarekomponenten zur Verwaltung und Vermittlung von Ablaufplänen prototypisch implementiert. Aktuell befindet sich die Einbettung der Petrinetz-Repräsentation sowie der Koordinationsserver in der Realiserungsphase.

Das generelle Anliegen des COSM-Projektes liegt in der Schaffung einer prototypischen Infrastruktur für *Dienstemärkte*, so daß ein dynamisch typisierter Zugriff auf entfernte Dienste für den menschlichen Benutzer möglich wird [MML94]. In der Basisvariante der Dienstrepräsentation findet noch keine Berücksichtigung von Rollen und Kontrollflüssen statt.

Verschiedene prototypische Anwendungen wurden mit Hilfe des CORBA DII und der Dienstrepräsentation implementiert. Hierbei stellt das DII eine Eigenentwicklung dar, welche wahlweise in Verbindung mit dem Sun-RPC oder dem DCE RPC eingesetzt werden kann. Der generische Klient wurde basierend auf der portablen GUI-Bibliothek *StarView* entwickelt. Entwicklungsplattform ist ein IBM RS/6000-Cluster, jedoch sind bereits Portierungen von Teilen der COSM-Infrastuktur auf Sun/Solaris bzw. IBM OS/2 durchgeführt worden.

5. Zusammenfassung

Während auf der *Kommunikations*ebene ein einheitlicher Vernetzungsstandard (das Internet) bereits heute gegeben ist, steht die Entwicklung einer systemtechnischen Infrastruktur zur *Koordination* von Abläufen in entsprechender geographischer Verteilung noch aus. Der vorliegende Beitrag zeigt anhand eines exemplarischen Geschäftsprozesses eine Möglichkeit der Integration von Ablaufmodellierung und -steuerung auf der Basis gefärbter Petrinetze und deren Repräsentation, dem *Ablaufplan*. Entsprechend der zugrundeliegenden Infrastruktur aus dem Projekt COSM (Common Open Service Market) agieren Benutzer in der Rolle des Klienten und sind prinzipiell in der Lage, sich bei moderatem Installationsaufwand an einem Geschäftsprozeß zu beteiligen. Erforderlich ist lediglich die Authentisierung innerhalb einer für den Geschäftsprozeß erforderlichen Rolle. Da für eine Installation des benötigten generischen Klienten nur ein Netzzugang erforderlich ist, kann auch ohne signifikanten Mehraufwand ein unternehmensfremder Teilnehmer einbezogen werden.

Eine Untersuchung der mit dieser organisationsübergreifenden, automatisierten Vernetzung verbundenen Auswirkungen auf die Autonomie beteiligter Unternehmen und auf die Koordinationseffizienz marktwirtschaftlicher Prozesse erlangt somit zunehmende Bedeutung als Forschungsgegenstand von Ökonomie und Informatik.

Literatur

[Gruh93] Gruhn, V.: "FunSoft Netze", in: Scheschonk, G. and Reisig, W. (Ed.): "Petri-Netze im Einsatz für Entwurf und Entwicklung von Informationssystemen", Informatik Aktuell, Springer Verlag, Berlin Heidelberg New York, 1993, S. 31ff

[Jens92] Jensen, K.: "Coloured Petri Nets", Volume 1, Springer Verlag, Berlin Heidelberg New York, 1992

[Kimb91] Kimberley, P.: ”Electronic Data Interchange”, McGraw-Hill, New York, 1991

[MML94] Müller, K., Merz, M., Lamersdorf, W.: ”Service trading and mediation in distributed computing systems”, in Proc. ICDCS ’94, IEEE Computer Society Press, 1994, S. 450ff

[MMML95] Merz, M., Moldt, D., Müller-Jones, K., Lamersdorf, W.: ”Workflow Modeling and Execution with Coloured Petri Nets in COSM”, submitted for publication, 1995

[MSC93] Meta Software Corporation: ”Design/CPN Tutorial”, 1993

[OMG91] OMG: ”The Object Request Broker: Architecture and Specification”: OMG Document No. 91.12.1, Object Management Group, Framingham, MA, USA, 1991

[Rein93] Reinwald, B.: ”Workflow-Management in verteilten Systemen”, Stuttgart, Leibzig, Teubner, 1993

[Schi93] Schill, A.: ”DCE: Das OSF Distributed Computing Environment: Grundlagen und Anwendungen”, Springer, New York, Berlin, Heidelberg, 1993