Generische Konzeption eines Verhandlungsprozesses in ebXML

Philipp Thun, Michael Rebstock

Fachhochschule Darmstadt University of Applied Sciences
Fachbereich Wirtschaft, Haardtring 100, 64295 Darmstadt, Germany
philipp.thun@web.de, rebstock@fh-darmstadt.de

Abstract. Zielsetzung des Beitrags ist die Definition eines generischen Protokolls für bilaterale, multiattributive Verhandlungen auf der Grundlage von *Electronic Business XML* (ebXML). Der zwischenbetriebliche Ablauf bilateraler Verhandlungen wird unter Einsatz des *Business Process Specification Schema* (BPSS) modelliert. Eingesetzt werden hierzu UML-Diagramme. Im Rahmen der Analyse prüfen wir auch die Eignung von ebXML zur Abbildung komplexer Kollaborationen. Vorliegende ebXML-Prozessmodelle zu diesem Thema fließen in das Konzept mit ein und werden kritisch untersucht. Über die Spezifikation der Abläufe hinaus wird die Struktur der auszutauschenden Daten analysiert und die entsprechenden Klassendiagramme werden erstellt.

1 Einleitung

Elektronische Verhandlungen sind aktueller Gegenstand von Forschungsaktivitäten (für einen Überblick vgl. [1], [2]) als auch Thema kommerzieller Anwendungsentwicklung (etwa [3], [4], [5], [6], [7]). Eine elektronische Verhandlung ist dabei ein auf elektronischem Informationsaustausch beruhender Prozess der Entscheidungsfindung zwischen zwei oder mehr Parteien, der in der Regel eine Einigung zum Ziel hat (ähnlich etwa [8], [9]). Dieser Prozess besteht in der Regel aus einer systemgestützten Abgabe von Geboten und etwaigen Gegengeboten so lange, bis die Einigung erzielt oder der Vorgang abgebrochen wurde (vgl. [2]).

Zahlreiche Arbeiten befassen sich mit der (Voll-)Automatisierung des Verhandlungsprozesses und haben die Entwicklung agentenbasierter Anwendungen zum Ziel (u.a. [10], [11]; für einen Überblick vgl. [2]). Das Ziel dieser Anwendungen ist in der Regel die abschließende Aushandlung eines Attributes der Markttransaktion, nämlich des Preises. Betrachtet man dagegen Prozesse der bestehenden, nicht-elektronischen Geschäftsabwicklung, so fällt auf, dass dort im Rahmen einer interaktiven Kommunikation der beteiligten Geschäftspartner eine Vielzahl von Attributen wie Produktspezifikationen, Qualitäten, Mengen, Preise, Liefer-, Zahlungs- und Gewährleistungsbedingungen ausgehandelt werden. Multiattributive elektronische Verhandlungen werden bisher wenig betrachtet (eine - konzeptionelle - Ausnahme stellt etwa [12] dar). Einer vollständig automatisierten Lösung versperren sich multiattributive Verhandlungen (vgl. [2]). Im Fokus dieses Beitrags steht daher die elektronische Unterstüt-

zung des interaktiven, multiattributiven Verhandlungsvorgangs und nicht die vollständige Automatisierung des Prozesses.

Aufgrund des integrativen und interorganisationalen Charakters elektronischer Verhandlungen ist die Berücksichtigung von Standards ein entscheidender Erfolgsfaktor für Entwicklung und Einsatz entsprechender Systeme. *Electronic Business XML* (ebXML) stellt einen Standard für die globale elektronische Geschäftsabwicklung dar, der unter der Federführung von UN/CEFACT und OASIS entwickelt wird (vgl. [13]). Im Zentrum des Standards steht der elektronische Geschäftsdatenaustausch auf Basis der *Extensible Markup Language* (XML; vgl. [14]). Hierbei liegt ein besonderes Augenmerk auf der Standardisierung und Wiederverwendbarkeit von Prozessen (vgl. [15]). Aufgrund seines umfassenden Ansatzes und seiner Herstellerunabhängigkeit wurde ebXML gegenüber konkurrierenden Ansätzen (wie BizTalk oder xCBL) gewählt (vgl. auch [16]).

Zielsetzung des vorliegenden Beitrags ist die Definition eines generischen Protokolls für bilaterale, multiattributive Verhandlungen auf der Grundlage von ebXML. Ein solches Protokoll stellt eine Basis weitergehender Anwendungsentwicklung dar. Dazu wird der zwischenbetriebliche Ablauf bilateraler Verhandlungen mit Hilfe des *Business Process Specification Schema* (BPSS; vgl. [17]) modelliert (Kapitel 2). Hierbei prüfen wir die Eignung von ebXML zur Abbildung komplexer Kollaborationen. Bereits zu diesem Thema bestehende Konzepte der Prozessmodellierung fließen in die Arbeit mit ein und werden kritisch betrachtet. Über die Spezifikation der Abläufe hinaus wird die Struktur der auszutauschenden Daten analysiert und ein Modell vorgestellt, das multiattributive Verhandlungen in XML abbilden kann (Kapitel 3).

2 Prozessmodellierung

Die Spezifikation des Verhandlungsprozesses (*Business Process Specification*) definiert den Austausch von Geschäftsdaten in Form von Dokumenten, welche im Rahmen von Transaktionen von einem Geschäftspartner zum anderen transportiert werden. Mehrere atomare Transaktionen lassen sich dabei zu komplexeren Kollaborationen kombinieren. Das *Business Process Specification Schema* steht sowohl als XML DTD-, als auch als XML *Schema*-Dokument (vgl. [18]) zur Verfügung. Der modellierte Prozess bildet insgesamt ein konkretes XML-Dokument, welches im Folgenden in Auszügen dargestellt wird.

2.1 Vereinbarungsphase

Eine elektronische Verhandlung stellt einen Prozess innerhalb der Vereinbarungsphase einer elektronischen Markttransaktion dar (Abbildung 1).

Gegenüber vorliegenden Darstellungen (etwa [19], [20]) sind die hier relevanten Phasenübergänge und Dokumente hervorgehoben. Weitere mögliche Abläufe im Rahmen der Vereinbarungsphase, wie etwa das Bewerten unterschiedlicher Gebote, werden im Rahmen dieser Arbeit nicht näher untersucht.

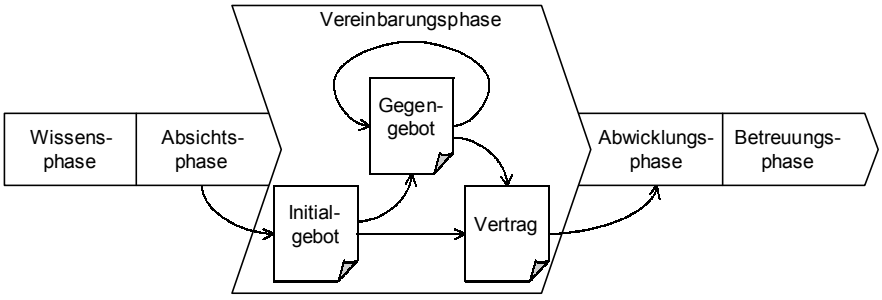


Fig. 1. Phasen einer Markttransaktion

Zu den Dokumenten *Gebot* (Initialgebot, Gegengebot) und *Vertrag* lässt sich jeweils ein Teilprozess der Vereinbarungsphase identifizieren: das Führen der Verhandlung („RequesterNegotiation“) sowie das Schließen des Vertrags („RequesterContract“). Abbildung 2 zeigt das entsprechende UML *Use Case*-Diagramm (zur UML-Notation vgl. [21]).

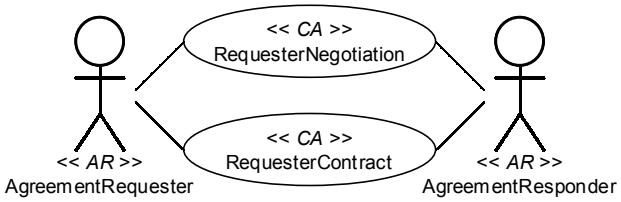


Fig. 2. Use Case-Diagramm der Kollaboration „ReachAgreement“

Der entsprechende Ausschnitt der *Business Process Specification* findet sich in Abbildung 3.

```
<BinaryCollaboration name="ReachAgreement">
   <InitiatingRole name="AgreementRequester"
     ... />
   <RespondingRole name="AgreementResponder"
     ... />
   <CollaborationActivity name="RequesterNegotiation"
     binaryCollaboration="ConductNegotiation"
     fromAuthorizedRole="AgreementRequester"
     toAuthorizedRole="AgreementResponder"
     ... />
   <CollaborationActivity name="RequesterContract"
     binaryCollaboration="CloseContract"
     fromAuthorizedRole="AgreementRequester"
     toAuthorizedRole="AgreementResponder"
     ... />
```

Fig. 3. Definition von „ReachAgreement“ (Teil 1) gemäß BPSS

Zunächst wird eine Kollaboration (<*BinaryCollaboration*>) mit dem Namen „ReachAgreement“ definiert, welcher eine initiierende sowie eine antwortende Rolle zugewiesen wird (<*InitiatingRole*> und <*RespondingRole*>). Im Sinne von ebXML sind beides autorisierte Rollen (*Authorized Roles*). Im obigen UML-Diagramm sind diese daher mit der Abkürzung „AR“ versehen. Da ein Initialgebot sowohl ein Verkaufsangebot als auch eine Kaufnachfrage beinhalten kann, werden die hier verwendeten Rollendefinitionen unabhängig von der Käufer- oder Verkäuferrolle festgelegt. Ein Käufer kann also beide generischen Rollen einnehmen, da er sowohl eine Nachfrage abschicken (als „AgreementRequester“) als auch ein Angebot erhalten kann (als „AgreementResponder“). Entsprechendes gilt für einen Verkäufer.

Im Anschluss werden die beiden oben identifizierten Teilprozesse der Vereinbarungsphase beschrieben. Das Element <*CollaborationActivity*> („CA“ im Diagramm) gibt an, dass es sich hierbei wiederum um komplexe, eigenständige Kollaborationen handelt, welche an anderer Stelle im Dokument spezifiziert und hier lediglich instanziiert werden. Aus der Namensgebung der Instanzen (hier „RequesterNegotiation“ und „RequesterContract“) ist die Rolle ersichtlich, die sie jeweils initiiert (hier „AgreementRequester“). An dieser Stelle hätten die Namen der Kollaborationen auch für die Instanzen beibehalten werden können; der Vorteil des obigen Benennungsschemas kommt jedoch dann zum Tragen, wenn es mehrere Instanzen derselben Kollaboration gibt (siehe Abschnitt 2.2). Die beiden instanziierten Kollaborationen besitzen ebenfalls je zwei beteiligte Rollen. Über die Attribute *fromAuthorizedRole* und *toAuthorizedRole* werden diese mit den gerade definierten Rollen auf der Ebene der Kollaboration „ReachAgreement“ verknüpft.

Zur Ablaufbeschreibung wird ein UML *State*-Diagramm verwendet (Abbildung 4). Dort werden, wie im entsprechenden ebXML-Dokument (Abbildung 5), Zustände und Zustandsübergänge beschrieben.

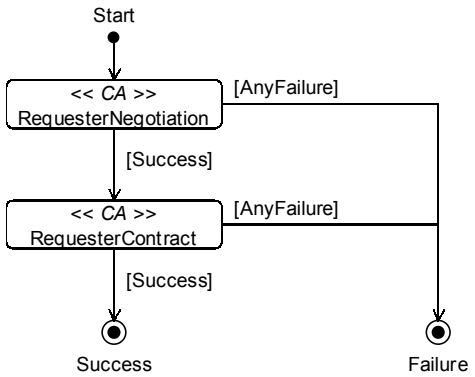


Fig. 4. State-Diagramm der Kollaboration „ReachAgreement“

Jede zuvor festgelegte Kollaborationsaktivität („RequesterNegotiation“ als Instanz von „ConductNegotiation“ und „RequesterContract“ als Instanz von „CloseContract“) stellt einen möglichen Zustand dieser Phase dar; hinzu kommen Start- und Endzustände.

```
    <Start
      toBusinessState="RequesterNegotiation"
      ... />
    <Transition
      fromBusinessState="RequesterNegotiation"
      toBusinessState="RequesterContract"
      conditionGuard="Success"
      ... />
    <Failure
      fromBusinessState="RequesterNegotiation"
      conditionGuard="AnyFailure"
      ... />
    <Success
      fromBusinessState="RequesterContract"
      conditionGuard="Success"
      ... />
    <Failure
      fromBusinessState="RequesterContract"
      conditionGuard="AnyFailure"
      ... />
</BinaryCollaboration>
```

Fig. 5. Definition von „ReachAgreement“ (Teil 2) gemäß BPSS

2.2 Verhandlung

Abbildung 6 zeigt die Detaillierung des Teilprozesses „ConductNegotiation“. (Da die Struktur der Definition dieser Kollaboration mit der oben dargestellten in großen Teilen übereinstimmt, wird an dieser Stelle auf die XML-Version der Prozessdarstellung verzichtet.)

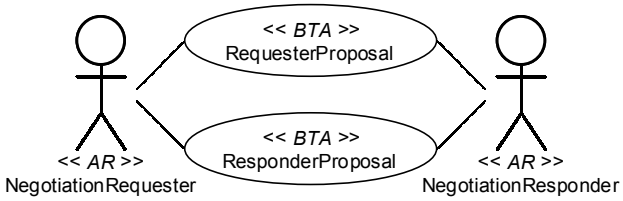


Fig. 6. Use Case-Diagramm der Kollaboration „ConductNegotiation“

Aufgrund der direkten Verknüpfung von Rollen auf unterschiedlichen Detailebenen des Prozesses sind die Rollen „AgreementRequester“ und „NegotiationRequester“ bzw. „AgreementResponder“ und „NegotiationResponder“ miteinander identisch. Innerhalb dieser Kollaboration gibt es zwei sogenannte Transaktionsaktivitäten („BTA“; *Business Transaction Activity*). Beide sind Instanzen derselben Transaktion („SubmitProposal“), jedoch mit vertauschter Rollenzuordnung.

Abbildung 7 zeigt die Definition der Transaktion „SubmitProposal“ als Auszug aus der *Business Process Specification*.

```
<BusinessTransaction name="SubmitProposal"
  ... >
    <RequestingBusinessActivity
      ... >
        <DocumentEnvelope
          businessDocument="NegotiableProposal"
          ... />
    </RequestingBusinessActivity>
    <RespondingBusinessActivity
      ... >
        <DocumentEnvelope
          businessDocument="NPAcceptance"
          isPositiveResponse="true"
          ... />
        <DocumentEnvelope
          businessDocument="NPCounterproposalAdvice"
          isPositiveResponse="true"
          ... />
        <DocumentEnvelope
          businessDocument="NPRejection"
          isPositiveResponse="false"
          ... />
    </RespondingBusinessActivity>
</BusinessTransaction>
```

Fig. 7. Definition von „SubmitProposal“ gemäß BPSS

Die Transaktion (<*BusinessTransaction*>) wird durch eine anfragende sowie eine antwortende Aktivität beschrieben. Die Aktivitäten referenzieren jeweils Dokumente, die gesendet oder empfangen werden. Die Spezifikation der Dokumente ist in eigenen Beschreibungen abgelegt. Abbildung 8 zeigt ein Beispiel für das Anfragedokument. Innerhalb der Definition wird wiederum auf den Ort der Strukturspezifikation verwiesen.

```
<BusinessDocument name="NegotiableProposal"
  specificationLocation=
    "http://www.ebxml.org/bd/NegotiableProposal.xsd"
  ... />
```

Fig. 8. Definition von „NegotiableProposal“ gemäß BPSS

Die Antwortdokumente innerhalb dieser Transaktion (beschrieben in <*RespondingBusinessActivity*>) enthalten keinerlei Nutzdaten, sondern dienen lediglich als Ausdruck der Absicht des antwortenden Verhandlungspartners. Die übertragene Information findet sich bereits im Namen des gesendeten Dokuments („NPAcceptance“, „NPRejection“ oder „NPCounterproposalAdvice“). Abbildung 9 zeigt den Dokumentaustausch im Überblick.

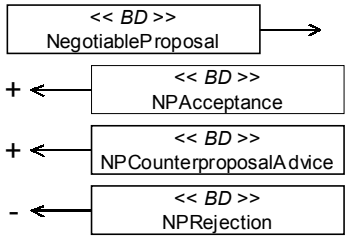


Fig. 9. Dokumente im Rahmen der Transaktion „SubmitProposal“

Plus- und Minus-Zeichen stehen dabei für das Attribut *isPositiveResponse*, welches der empfangenden Seite eine schnelle und zweifelsfreie Interpretation der ankommenden Dokumente ermöglicht. Abbildung 10 zeigt das *State*-Diagramm dieses Teilprozesses.

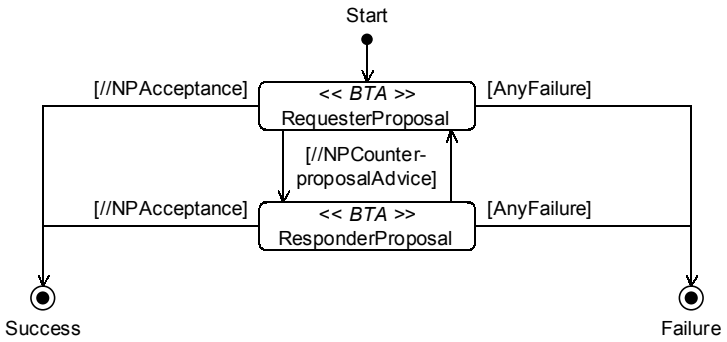


Fig. 10. State-Diagramm der Kollaboration „ConductNegotiation“

Wird „NPAcceptance“ übermittelt, so führt dies zu einem erfolgreichen Endzustand der Verhandlung und der darauf folgende Teilprozess („RequesterContract“) kann beginnen. Auf der anderen Seite führt eine vollständige Ablehnung des Gebots („NPRejection“) zu einem negativen Endzustand und damit auch zum Abbruch der gesamten Vereinbarungsphase (siehe Abbildung 4). Möchte der antwortende Geschäftspartner weder annehmen noch ablehnen, sondern ein Gegengebot abgeben, so erfolgt dies im Rahmen einer neuen Transaktion. Das Antwortdokument („NPCounterproposalAdvice“) wird in diesem Fall genutzt, um den Verhandlungspartner auf das Gegengebot zu avisieren. Der Grund für diese auf den ersten Blick etwas umständliche Sequenz liegt in der Strukturdefinition von *Business Transactions* (vgl. [17]). Diese erfordern genau ein anfragendes und kein oder ein antwortendes Dokument. Um aufeinander folgende Gegengebote abzubilden, müssen diese jeweils innerhalb einer neuen Transaktion versendet werden.

Das Dokument *E-Commerce Patterns* beschreibt u.a. ein mögliches Muster für Verhandlungen in ebXML (*Simple Negotiation Pattern* [vgl. 22]). Die an dieser Stelle vorgestellte Spezifikation weicht hiervon in einigen Punkten ab. Die wesentlichste Änderung bezieht sich auf die Abgabe eines Gegengebots, also den zentralen Aspekt einer Verhandlung. Das *Simple Negotiation Pattern* sieht den Hinweis auf ein folgen-

des Gegengebot nicht als zwingend erforderlich, sondern lediglich als optional an, und ordnet diese Möglichkeit der Ablehnung eines Gebots unter. Dies führt dazu, dass Ablehnungen und Gegengebote zunächst identisch behandelt werden und damit die aktuelle Instanz der Kollaboration, *Conversation* genannt, beenden. Die Zusendung eines Gegengebots erfolgt dann anschließend im Rahmen einer neuen Konversation. Die Zuordnung der Gegengebote muss demnach auf Anwendungsebene erfolgen.

Das in diesem Beitrag vorgestellte Konzept handhabt Gegengebote als Teil einer nicht unterbrochenen Konversation. Nur so kann das für den Empfang von Geschäftsdaten zuständige *Business Service Interface* sicherstellen, dass alle Nachrichten, die im Rahmen einer Verhandlung abgegeben wurden, auch in ihrer richtigen Reihenfolge beim jeweiligen Geschäftspartner eintreffen. Werden Gegengebote stets in einer neuen Konversation abgegeben, besteht diese Zuordnungsmöglichkeit nicht. Die Nutzung dieser in das *ebXML-Framework* integrierten Option, kann demnach nur dann erfolgen, wenn der Austausch von aufeinander bezogenen Gegengeboten als grundlegender Aspekt einer Verhandlung in die Spezifikation integriert wird.

Das *State*-Diagramm in Abbildung 10 weist zwei unterschiedliche Arten von Bedingungen auf. Neben den sogenannten *Condition Guards*, welche die Werte „Success“, „BusinessFailure“, „TechnicalFailure“ und „AnyFailure“ (als Generalisierung von „BusinessFailure“ und „TechnicalFailure“) enthalten können, ermöglicht ebXML die Nutzung von frei definierbaren *Condition Expressions*. Bei diesen Ausdrücken ist nicht nur die Bedingung, sondern auch ihre Sprache individuell wählbar, so können z.B. XPath (vgl. [23]) oder Java (vgl. [24]) zur Anwendung kommen. Abbildung 11 zeigt ein Beispiel. Die XPath-Bedingung dieses Beispiels sucht in der Antwort nach einem Element „NPCounterproposalAdvice“ und führt zu einem Zustandsübergang, falls die Bedingung positiv aufgelöst wird.

```
<Transition
  fromBusinessState="RequesterProposal"
  toBusinessState="ResponderProposal"
  ... >
    <ConditionExpression
      expressionLanguage="XPath"
      expression="//NPCounterproposalAdvice"/>
</Transition>
```

Fig. 11. Beispiel einer Transition gemäß BPSS

2.3 Vertragsabschluss

Abbildung 12 zeigt den *Use Case* des zweiten Teilprozesses der Vereinbarungsphase, „CloseContract“. Diese Kollaboration beinhaltet nur eine Transaktionsaktivität: „RequesterOffer“ ist eine Instanz der Transaktion „SubmitOffer“. Abbildung 13 zeigt die entsprechende XML-Beschreibung.

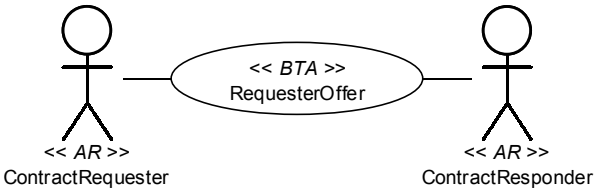


Fig. 12. Use Case-Diagramm der Kollaboration „CloseContract“

```
<BusinessTransaction name="SubmitOffer"
  ... >
    <RequestingBusinessActivity
      ... >
        <DocumentEnvelope
          businessDocument="BindingOffer"
          ... />
    </RequestingBusinessActivity>
    <RespondingBusinessActivity
      ... >
        <DocumentEnvelope
          businessDocument="BOAcceptance"
          isPositiveResponse="true"
          ... />
        <DocumentEnvelope
          businessDocument="BORejection"
          isPositiveResponse="false"
          ... />
    </RespondingBusinessActivity>
</BusinessTransaction>
```

Fig. 13. Definition von „SubmitOffer“ gemäß BPSS

Der Aufbau der Transaktionen weist große Ähnlichkeiten auf. So enthält das Anfragedokument den ausgehandelten Vertrag („BindingOffer“), welcher sich aus den im vorangegangenen Teilprozess definierten Punkten zusammensetzt und daher inhaltlich identisch ist mit dem zuletzt akzeptierten Vorschlagsdokument. Allerdings existieren nur zwei zulässige Antwortdokumente, Annahme („BOAcceptance“) und Ablehnung („BORejection“), da dieser Teilprozess nur noch dem verbindlichen Abschluss der Vereinbarung dient (Abbildung 14).

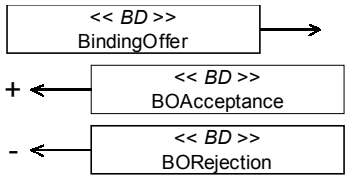


Fig. 14. Dokumente im Rahmen der Transaktion „SubmitOffer“

„BORejection“ kommuniziert die Absicht des Geschäftspartners wiederum bereits durch seinen Umschlag (<*DocumentEnvelope*>). Um die Rechtsverbindlichkeit des Kontrakts herzustellen, muss „BOAcceptance“ dagegen aber den vollen, bereits von „ContractRequester“ unterzeichneten Inhalt aus „BindingOffer“ sowie die digitale Signatur von „ContractResponder“ enthalten.

Abbildung 15 zeigt das dazugehörige *State*-Diagramm. Vom erfolgreichen Abschluss der vorangegangenen Kollaboration („RequesterNegotiation“) initiiert, startet dieser Teilprozess mit der einzigen Transaktion „RequesterOffer“. Nur ein ebenfalls erfolgreiches Ende dieser Transaktion führt auch die übergeordnete Kollaboration („ReachAgreement“) in einen positiven Endzustand (siehe Abbildung 4).

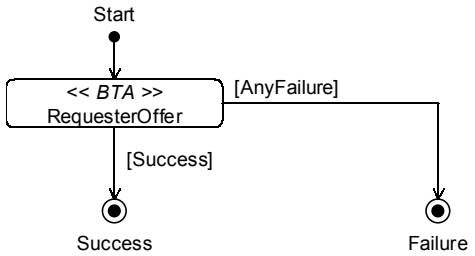


Fig. 15. State-Diagramm der Kollaboration „CloseContract“

3 Datenmodellierung

Das oben zitierte *Simple Negotiation Pattern* beschreibt lediglich den Prozess, nicht jedoch die ausgetauschten Daten. In diesem Abschnitt erfolgt eine Beschreibung der Dokumente „NegotiableProposal“ und „BindingOffer“, welche die Nutzdaten innerhalb der in den Kapiteln 2.2 und 2.3 definierten Kollaborationen transportieren. Da die übrigen Dokumente Informationen lediglich über ihren Typ weitergeben, kann hier auf ihre Darstellung verzichtet werden. Gleiches gilt für das Dokument „BOAcceptance“, da dieses den Inhalt von „BindingOffer“ lediglich signiert wiederholt. Besonderes Ziel bei der Modellierung ist eine möglichst flexible Datenstruktur, um verschiedenste Vertragsarten abbilden zu können.

3.1 Vertragsdokument

Ein Vertrag setzt sich aus einer beliebigen Anzahl von Attributen („Attribute“) und Positionen („Item“) zusammen (Abbildung 16). Vertragsbezogene Attribute beinhalten globale Informationen wie Vertragswährung, allgemeine Liefer- oder Zahlungsbedingungen. Positionen definieren i.d.R. Artikel oder Leistungen, welche mittels positionsbezogener Attribute beschrieben werden. Diese sind etwa Positionsbezeichnung, Mengen- oder Preisangabe. Neben einem Namen, einem Datenformat (*String*, *Integer* etc.) sowie dem eigentlichen Inhalt besitzt jedes Attribut einen optionalen

Verweis auf eine *Core Component* (vgl. [25]). Über die Referenz zu einer solchen kontextbefreiten Informationseinheit kann einem Attribut eine standardisierte Bedeutung zugewiesen werden. ebXML stellt die Eindeutigkeit dieser Komponenten sicher. Eine *Core Library* ermöglicht das Auffinden der *Core Components*.

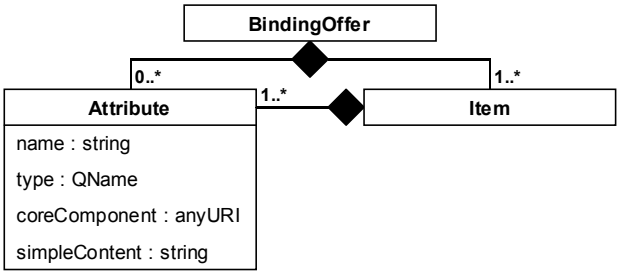


Fig. 16. Class-Diagramm von „BindingOffer“

```
<BindingOffer
  ... >
    <Attribute name="currency"
      type="xsd:string"
      coreComponent="http://www.ebxml.org/cc/000133">
        EUR
    </Attribute>
    <Item>
       <Attribute name="productNumber"
         type="xsd:string">
           1234-5678
       </Attribute>
       <Attribute name="productName"
         type="xsd:string">
           Notebook Computer
       </Attribute>
       <Attribute name="orderQuantity"
         type="xsd:integer">
           12
       </Attribute>
       <Attribute name="orderPrice"
         type="xsd:decimal">
           3456.78
       </Attribute>
    </Item>
</BindingOffer>
```

Fig. 17. Beispiel eines konkreten Vertragsdokuments

Die Umsetzung des Datenmodells erfolgt in Form eines XML *Schema*-Dokuments. Aus diesem Grund kommen im obigen Diagramm XML *Schema*-Datentypen zum Einsatz, welche ebenfalls in Instanzdokumenten benutzt werden, dort für die Beschreibung des Datenformats eines Attributs.

Abbildung 17 zeigt beispielhaft eine Instanz des „BindingOffer“-Dokuments. Der Beispielvertrag weist u.a. ein globales Währungsattribut („currency“) auf, welches einen Verweis auf *Core Component* Nummer 133 enthält (vgl. [26]). Einer Applikation wird so ermöglicht, das Attribut zu *verstehen* und beispielsweise eine Währungsumrechnung anzubieten.

3.2 Verhandlungsdokument

Der grundlegende Aufbau von „NegotiableProposal“ ähnelt dem oben gezeigten Muster, weist allerdings eine komplexere Struktur auf (Abbildung 18).

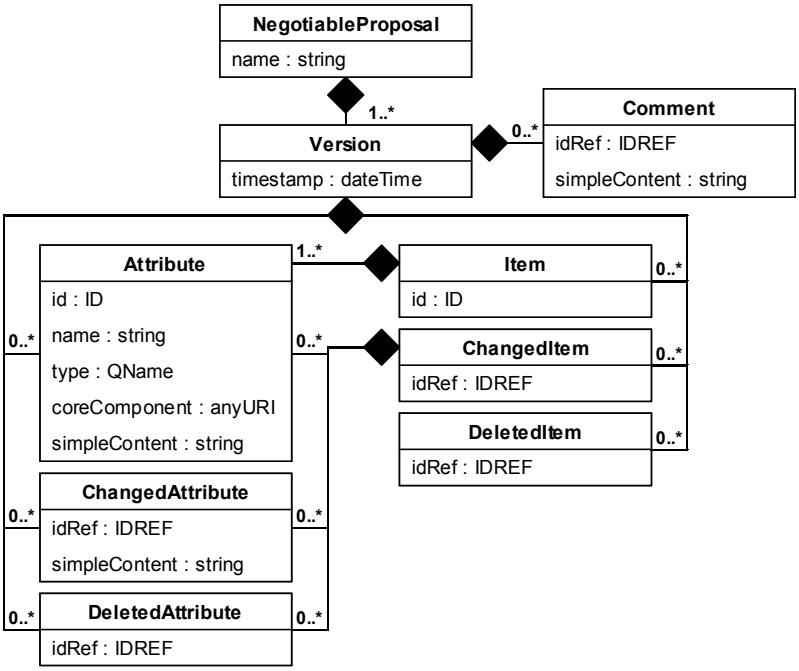


Fig. 18. Class-Diagramm von „NegotiableProposal“

Das Wurzelelement („NegotiableProposal“) beinhaltet ein „name“-Attribut, welches dem empfangenden Geschäftspartner eine schnellere Beurteilung des hereinkommenden Gebots ermöglicht. Daneben besitzt es ein Unterelement „Version“, welches mit einem genauen Zeitpunkt („timestamp“) versehen ist. Hier wird ein Unterschied zum Aufbau des Vertragsdokuments deutlich: „NegotiableProposal“ speichert die gesamte Historie der Verhandlung, so dass diese Aufgabe nicht von einer auf diesem Konzept aufbauenden Applikation übernommen werden muss. Eine möglichst generische und flexible Verhandlungskonzeption erlaubt nicht nur das Hinzufügen, Ändern und Löschen von Positionen, sondern auch von einzelnen Attributen. Die Substruktur des „Version“-Elements erweitert daher den von „BindingOffer“ her

bekannten Aufbau. Zusätzliche Elemente ermöglichen es, zuvor definierte Attribute oder Positionen als gelöscht zu deklarieren oder deren Änderungen zu verfolgen. Kommentare zu einzelnen Elementen können ebenfalls erfasst werden („Comment“). Ein Kommentar verweist über ein XML IDREF-Attribut auf ein anderes Element (Attribut oder Position), so dass gezielt Bemerkungen dazu hinzugefügt werden können. Durch Auslassen des optionalen IDREF-Attributs werden auch globale Kommentare möglich. Wandelt eine Applikation ein Verhandlungs- in ein Vertragsdokument um, können die „Comment“-Elemente somit einfach entfernt werden.

4 Zusammenfassung und Ausblick

Der vorliegende Beitrag entwickelt ein generisches Modell eines Verhandlungsprozesses als *ebXML-Business Process Specification* und modelliert die auszutauschenden Geschäftsdaten. Es zeigt sich, dass ebXML für die Modellierung auch komplexerer bilateraler, multiattributiver Verhandlungen gut geeignet ist. Vorliegende Verhandlungsschemata in ebXML wurden aufgegriffen, konkretisiert und in einigen Punkten begründet verändert. Zusätzlich wurde ein Datenmodell erarbeitet, welches einen besonderen Fokus auf eine flexible semantische Struktur von Verhandlungs- und Vertragsdokumenten legt.

Auf Basis der vorliegenden Ergebnisse kann die Spezifikation des Datenaustauschs sowie die Implementierung eines Prototypen erfolgen. Eingesetzt werden dazu *Collaboration Protocol Profile* und *Collaboration Protocol Agreement* (vgl. [27]). Zur prototypischen Implementierung wird Java verwendet. Für den Austausch der ebXML-Nachrichten steht die *Java API for XML Messaging* (JAXM; vgl. [28]) zur Verfügung, deren Referenzimplementierung grundlegende ebXML-Funktionalität bietet.

Literaturverzeichnis

1. ENegotiations Working Group (2002): Negotiation Protocols. http://enegotiations.wu-wien.ac.at/, Abruf am 18.6.2002
2. Rebstock, M. (2001): Elektronische Unterstützung und Automatisierung von Verhandlungen, in: Wirtschaftsinformatik, Nr. 6 (2001), S. 609-617
3. Ariba, Inc. (2001): Dynamic Trade. http://www.ariba.com/com_plat/b2b_dynamic_trade.cfm, Abruf am 26.5.2001
4. CommerceOne, Inc. (2001): MarketSite. http://www.commerceone.com/download/cmrc_marketsite_ds.pdf, Abruf am 26.5.2001
5. IBM, Inc. (2002): SilkRoad project. http://www.zurich.ibm.com/csc/ebizz/silkroad.html, Abruf am 18.6.2002
6. Oracle, Inc. (2001): Exchange Marketplace. http://www.oracle.com/appsnet/products/exchanges/, Abruf am 26.5.2001
7. SAP, Inc. (2001): MarketSet. http://www.sapmarkets.com/collateral/brochures/marketset_overview.pdf, Abruf am 26.5.2001
8. Davis, R. / Smith, R. (1983): Negotiation as a Metaphor for Distributed Problem Solving, in: Artificial Intelligence, Nr. 1 (1983), S. 63-109

9. Zarnekow, R. (1999): Softwareagenten und elektronische Kaufprozesse. Gabler
10. Ito, T. u.a. (2001): BiddingBot: A Multiagent Support System for Cooperative Bidding in Multiple Auctions. http://www.cs.cmu.edu/~softagents/papers/itota-icmas00-poster.pdf, Abruf am 23.5.2001
11. Arpinar, S. / Dogac, A. / Tatbul, N. (2000): An Open Electronic Marketplace through Agent-based Workflows: MOPPET, in: International Journal on Digital Libraries, Nr. 1 (2000), S. 36-59
12. Ströbel, M. (2001): A Design and Implementation Framework for Symmetric Multi-Attribute Negotiation Support in Electronic Markets. Vorstudie zur Dissertation, Universität St. Gallen. http://www.zurich.ibm.com/~mis/publications/ThesisProposal.pdf, Abruf am 1.2.2002
13. ebXML Joint Coordination Committee (2002): ebXML - Enabling A Global Electronic Market. http://www.ebxml.org/, Abruf am 18.6.2002.
14. World Wide Web Consortium (2002): Extensible Markup Language. http://www.w3c.org/XML/, Abruf am 18.6.2002.
15. ebXML Business Process Project Team (2001): Catalog of Common Business Processes, Version 1.0. http://www.ebxml.org/specs/bpPROC.pdf, Abruf am 12.2.2002
16. Willaert, F. (2001): XML-based Frameworks and Standards for B2B eCommerce. Dissertation, Katholische Universität Leuven. http://www.ebxml.org/documents/ebxml-thesis.pdf, Abruf am 12.2.2002.
17. ebXML Business Process Project Team (2001): Business Process Specification Schema, Version 1.01. http://www.ebxml.org/specs/ebBPSS.pdf, Abruf am 12.2.2002.
18. World Wide Web Consortium (2002): XML Schema. http://www.w3c.org/XML/Schema, Abruf am 18.6.2002.
19. Schmid, B. (1999): Elektronische Märkte - Merkmale, Organisation und Potentiale, in: Management-Handbuch Electronic Commerce, S. 31-48. Vahlen
20. Rebstock, M. / Amirhamzeh Tafreschi, O. (2002): Secure Interactive Electronic Negotiations in Business-to-Business Marketplaces, in: ECIS 2002. Gdansk.
21. Object Management Group, Inc (2002): Unified Modeling Language. http://www.uml.org/, Abruf am 18.6.2002.
22. ebXML Business Process Project Team (2001): E-Commerce Patterns, Version 1.0. http://www.ebxml.org/specs/bpPATT.pdf, Abruf am 12.2.2002.
23. World Wide Web Consortium (1999): XML Path Language. http://www.w3c.org/TR/xpath, Abruf am 18.6.2002.
24. Sun Microsystems, Inc. (2002): The Source for Java Technology. http://java.sun.com/, Abruf am 18.6.2002.
25. ebXML Core Components Project Team (2001): Core Component Overview, Version 1.05. http://www.ebxml.org/specs/ccOver.pdf, Abruf am 12.2.2002.
26. ebXML Core Components Project Team (2001): Core Component Dictionary, Version 1.04. http://www.ebxml.org/specs/ccDICT.pdf, Abruf am 12.2.2002.
27. ebXML Trading Partners Project Team (2001): Collaboration-Protocol Profile and Agreement Specification, Version 1.0. http://www.ebxml.org/specs/ebCPP.pdf, Abruf am 12.2.2002.
28. Sun Microsystems, Inc. (2002): Java API for XML Messaging. http://java.sun.com/xml/jaxm/, Abruf am 18.6.2002.
29. ebXML Technical Architecture Project Team (2001): Technical Architecture Specification, Version 1.04. http://www.ebxml.org/specs/ebTA.pdf, Abruf am 12.2.2002.