February 2005

RPXML - Standardisierung von Reverse-Pricing-Mechanismen

Martin Bernhardt
Johann Wolfgang Goethe-Universität Frankfurt am Main

Oliver Hinz
Johann Wolfgang Goethe-Universität Frankfurt am Main

In: Ferstl, Otto K, u.a. (Hg) 2005. *Wirtschaftsinformatik 2005: eEconomy, eGovernment, eSociety*; 7. Internationale Tagung Wirtschaftsinformatik 2005. Heidelberg: Physica-Verlag

ISBN: 3-7908-1574-8

RPXML – Standardisierung von Reverse-Pricing-Mechanismen

Martin Bernhardt, Oliver Hinz
Johann Wolfgang Goethe-Universität Frankfurt am Main

Zusammenfassung: Reverse Pricing ist ein dynamischer Preismechanismus, der sowohl dem Käufer als auch dem Verkäufer eines Produktes die Beeinflussung des zu zahlenden Produktpreises gestattet. Eine formale und einheitliche Abbildung des Reverse Pricing ist mit bestehenden Standards im E-Business kaum möglich, da diese die Berücksichtigung von Interaktion zwischen Käufer und Verkäufer bei der Preisfindung häufig nicht zulassen. Mit der Darstellung des Design-Raums von Reverse Pricing und dessen Modellierung wird diese Lücke durch RPXML, einen Standardisierungsvorschlag des Reverse Pricing in XML, geschlossen. Darauf aufbauend demonstriert dieser Beitrag durch den Einsatz der Mathematical Markup Language die Flexibilität und Erweiterbarkeit von RPXML.

Schlüsselworte: Reverse Pricing, Preismechanismen, B2B, XML, Standardisierung, E-Business

1 Einleitung

Gesunkene Transaktions- und Prozesskosten im Internet haben eine Vielzahl dynamischer Preismechanismen ermöglicht [Bako97]. So wies die OECD bereits vor fünf Jahren darauf hin, dass uniforme Fixpreise zunehmend von differenzierten Preisen abgelöst werden: „It is clear that electronic commerce will change the structure, if not the level, of pricing, as more and more products are subject to the differential pricing associated with customised products, fine market segmentation and auctions, and as the ease of changing prices increases" [OECD99].

Neben den in Online-Shops üblichen Fixpreisen erfreuen sich insbesondere Auktionen wachsender Beliebtheit. Während bei Fixpreisen nur relativ wenige Modifikatoren wie z. B. Rabattierungen oder Intervallstaffelungen berücksichtigt werden müssen [Kelk+02], können Auktionen in vielfältigen Ausgestaltungen durchgeführt werden [Wurm+01]. Zu den bekanntesten Verfahren zählen die englische, die holländische, die Höchstpreis- und die Vickrey-Auktion [MiWe82].

Ein weiterer dynamischer Preismechanismus, der verstärkt ins Interesse wissenschaftlicher Betrachtungen rückt, ist das Reverse Pricing [Span+03; HaTe03; Fay04]. Dabei bestimmt der Käufer den letztendlich zu zahlenden Preis, indem er

dem Verkäufer ein Gebot in Höhe des Preises nennt, den er zu zahlen bereit ist. Liegt dieses Gebot über einer geheimen und vorab vom Verkäufer festgelegten Preisschwelle, kommt es zur Transaktion in Höhe des Käufergebots. Der Mechanismus kann vielfältig ausgestaltet werden: Zwei wesentliche Grundformen, die sich anhand einmaliger oder mehrfacher Gebotsabgabe unterscheiden, können durch weitere Design-Variablen in ihrer Ausgestaltung variiert werden.

Im Unterschied zu Auktionen stehen (potenzielle) Käufer beim Reverse Pricing nicht in unmittelbarer Preiskonkurrenz zueinander, da sie lediglich die geheime Preisschwelle eines Verkäufers übertreffen müssen, um eine Transaktion abzuschließen. Aufgrund der durch die Geheimhaltung der Preisschwelle hervorgerufenen Preisintransparenz kann das Reverse Pricing als zusätzlicher Online-Vertriebskanal eingesetzt werden, ohne die Preisstrukturen bestehender Vertriebskanäle zu gefährden. Die somit denkbare Ansprache neuer Kundensegmente kann schließlich eine Umsatzsteigerung nach sich ziehen. Aufgrund der Preisintransparenz sowie der Möglichkeit, differenzierte Preise mit einem Angebot zu erzielen, eignet sich Reverse Pricing insbesondere als Vertriebskanal für mehrere Einheiten eines Produktes oder für Produkte mit hoher individueller Wertschätzung (z. B. Restkapazitäten).

Bekanntester und erfolgreichster Anbieter in diesem Bereich ist Priceline (http://www.priceline.com), der seit 1998 auf dem Markt ist und auf seiner Plattform vorwiegend Flüge über einen Reverse-Pricing-Mechanismus verkauft. Das Angebot wurde aber auch um Mietwagen- und Hotelnutzungen, Urlaubsreisen und Hypothekenzinssätze erweitert. Daneben setzen unter anderem Expedia (http://www.expedia.com), Ebay Travel (http://www.ebay.com/travel) und in Deutschland LTU Biet & Flieg (http://www.ltu.de) einen entsprechenden Preismechanismus ein.

Verkäufer und Betreiber einer zum Verkauf eingesetzten Transaktionsplattform sind in der Regel verschiedene Personen. Die Übermittlung umfangreicher Produkt- und Angebotsdaten zwischen den beteiligten Parteien ist daher notwendig. Über eine Web-Oberfläche gestaltet sich dieser Vorgang insbesondere bei einer großen Anzahl an Produkten äußerst zeitaufwändig. Daher bietet sich die Weitergabe von Daten in einem standardisierten Austauschformat an. Gleichzeitig stellt sich die Frage, wie entsprechende Produkt- und Angebotsdaten formal abgebildet werden können. In der Praxis hat sich hierfür XML [Bray+04] bewährt.

Während uniforme Fixpreise mittlerweile in gängigen XML-Standards im Bereich E-Business wie der *Electronic Business using eXtensible Markup Language (ebXML)*, *BMEcat* und *Common Business Library (xCBL)* abgebildet werden können, sind dynamische Preismechanismen in der Standardisierung bisher wenig oder gar nicht berücksichtigt [Wagn03]. Gerade im Bereich Reverse Pricing existiert keine Möglichkeit, die verschiedenartigen Ausgestaltungen formal abzubilden.

Ziel dieser Arbeit ist es daher, einen entsprechenden Ansatz zur standardisierten Abbildung von Reverse-Pricing-Mechanismen in XML vorzustellen. Hierdurch wird gezeigt, wie bestehende Standards um dynamische Preismechanismen ergänzt werden können. Eine Erweiterung des vorgestellten Ansatzes ermöglicht zudem die Berücksichtigung verschiedener Einflussgrößen innerhalb eines dynamischen Preismechanismus. Dieser könnte somit eigenständig auf das Gebotsverhalten eines Käufers reagieren und dessen Zahlungsbereitschaft stärker abschöpfen.

Die Arbeit ist wie folgt aufgebaut: Abschnitt 2 stellt die Funktionsweise eines Reverse-Pricing-Mechanismus dar und beschreibt mögliche Design-Variablen. Eine Untersuchung bestehender XML-Standards im Bereich E-Business in Abschnitt 3 geht auf die Abbildungsmöglichkeiten dynamischer Preismechanismen ein, um dann in Abschnitt 4 ein entsprechendes Modell für Reverse Pricing zu erarbeiten. Im darauf folgenden Abschnitt wird das Modell mit Hilfe der Mathematical Markup Language (MathML) erweitert. Abschnitt 6 zieht abschließend ein Fazit.

2 Darstellung des Reverse-Pricing-Mechanismus

Reverse Pricing ist ein dynamischer Preismechanismus, bei dem sowohl Käufer als auch Verkäufer eines Produktes Einfluss auf den zu zahlenden Produktpreis nehmen können. Durch einen vorab festgelegten und geheim gehaltenen Mindestpreis bestimmt der Verkäufer eine untere Preisschwelle, während ein Käufer durch die Abgabe eines Gebotes den zu zahlenden Produktpreis determiniert, sofern sich dieses Gebot oberhalb des geheimen Mindestpreises des Verkäufers befindet.

2.1 Funktionsweise

Zunächst kann ein Reverse-Pricing-Mechanismus nach der Anzahl möglicher Gebote eines Käufers in zwei Grundformen untergliedert werden [Span+03]. Während Priceline Käufern für einen Flug lediglich ein Gebot innerhalb eines Zeitraumes von sieben Tagen gestattet, ist es Käufern in einem erweiterten Modell möglich, mehrere Gebote abzugeben. Ein ursprünglich erfolgloses Gebot unterhalb der unbekannten Preisschwelle kann somit erhöht werden und schließlich doch noch zu einer Transaktion führen. Dieses Verhalten könnte folglich eine für Verkäufer wünschenswerte Umsatzsteigerung herbeiführen [Span+03].

Hat ein Käufer die Möglichkeit, beliebig viele Gebote auf ein Produkt abzugeben, so könnte er durch schrittweise Erhöhung eines ursprünglichen Gebotes um minimale Beträge versuchen, die geheime Preisschwelle des Verkäufers exakt zu treffen und dadurch möglichst billig an das Produkt zu gelangen [HaTe03]. Allerdings muss ein Käufer bei jeder Gebotsabgabe verschiedene Bietkosten berücksichtigen.

Diese werden definiert als Summe aus Such- und Strafkosten bei der Abgabe eines Gebotes. Suchkosten entstehen dabei durch den mentalen Aufwand zur Bestimmung eines geeigneten Gebotes (z. B. durch Preisvergleich bei verschiedenen Anbietern [Stig61]) sowie die Zeitdauer bis zu einer möglichen Gebotsabgabe. Als Strafkosten hingegen werden monetäre Kosten verstanden, die ein Käufer bei der Abgabe eines Gebotes berücksichtigen muss (z. B. eine Gebühr auf zusätzliche Gebote). Logistikkosten werden hier nicht berücksichtigt, da sie der eigentlichen Transaktion nachgelagert sind und bei der Betrachtung des Preismechanismus folglich keine Rolle spielen.

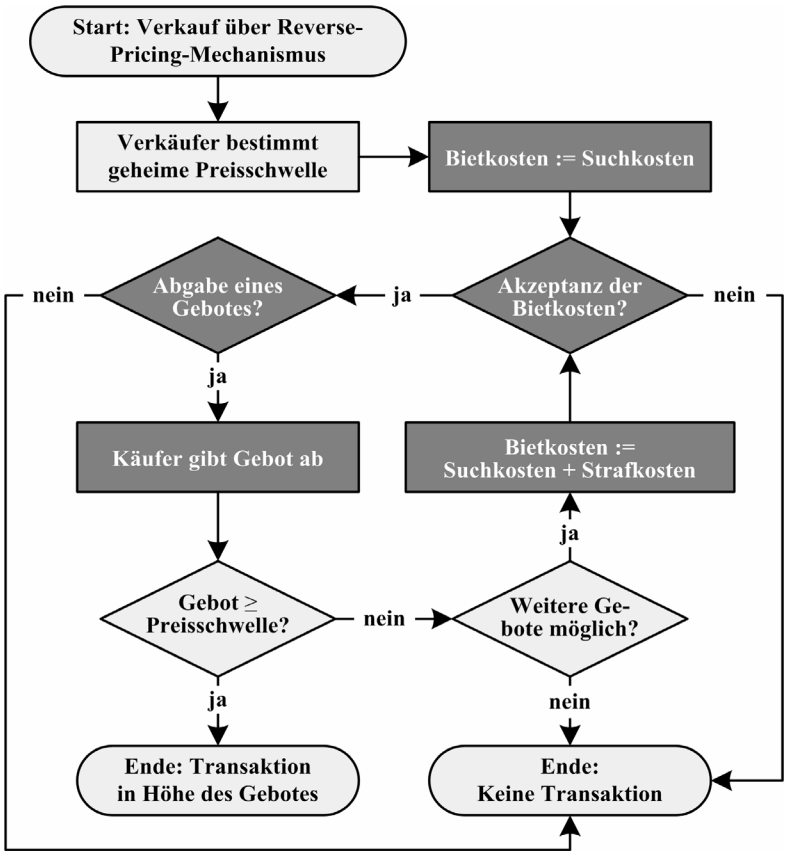

Abbildung 1: Ablauf des Reverse-Pricing-Mechanismus

Durch den Reverse-Pricing-Mechanismus wird garantiert, dass eine Transaktion nur oberhalb der vom Verkäufer festgelegten Preisschwelle stattfindet. Dies verdeutlicht auch Abbildung 1, in der Entscheidungen und Aktionen des Käufers beim Ablauf eines Reverse-Pricing-Mechanismus mit dunklem Hintergrund her-

vorgehoben sind. Ein auch nach der maximal möglichen Anzahl an Geboten nicht erfolgreicher Käufer hat demnach keine Möglichkeit mehr, das gewünschte Produkt über diesen Mechanismus zu erwerben.

2.2 Design-Variablen

Design-Variablen beschreiben Eigenschaften, anhand derer sich ein Reverse-Pricing-Mechanismus von einem Verkäufer auf die von ihm gewünschte Form einstellen lässt [Bern04]. Mit ihrer Hilfe ist es möglich, eine konkrete Ausprägung eines solchen Mechanismus eindeutig zu beschreiben.

Neben der Bestimmung einer maximalen Anzahl an Geboten ermöglichen Design-Variablen insbesondere die Variation des Preisauswahlverfahrens sowie verschiedener Restriktionen für die Gebotsabgabe eines Käufers. Restriktionen äußern sich für einen Käufer in Form erhöhter Such- oder Strafkosten bei der Abgabe zusätzlicher Gebote. Verkäufer können sie einsetzen, um die Attraktivität eines schrittweisen Gebotsverhaltens einzuschränken. Im Folgenden soll kurz auf mögliche Design-Variablen und ihre Ausprägungen eingegangen werden (vgl. [Bern04]).

Als zentraler Bestandteil des Reverse-Pricing-Mechanismus lässt sich zunächst die **Preisschwelle** über eine Design-Variable festlegen. Der Verkäufer hat dabei die Wahl zwischen einer fixen Preisschwelle oder einer von verschiedenen Einflussgrößen abhängigen, dynamischen Preisschwelle. Als mögliche abhängige Werte kommen z. B. die abgelaufene oder verbleibende Laufzeit eines Produktangebotes, die bisherige Anzahl an Geboten eines (oder aller) Käufer(s) oder etwa die Anzahl verbleibender Produkteinheiten in Frage.

Weiterhin kann der Verkäufer anhand einer Design-Variable festlegen, welches **Preisauswahlverfahren** eingesetzt werden soll. Die Abgabe eines Gebotes kann entweder mittels einer durch eine bestimmte Anzahl an Preispunkten aufgespannten Referenzpreisspanne erleichtert werden oder ohne explizite Nennung geeigneter Preise völlig frei erfolgen. Das erstere, von [Cher03] als „Select-Your-Price“ bezeichnete Preisauswahlverfahren, reduziert dabei den mentalen Aufwand der Käufer bei der Suche nach einem geeigneten Preis und wird von diesen als erfolgsversprechender und einfacher beurteilt [Cher03]. Gleichzeitig schränkt dieses Preisauswahlverfahren allerdings die Freiheit bei der Auswahl eines Gebotes ein, da dieses lediglich aus der vorgegebenen Liste an Preispunkten und nicht wie bei dem als „Name-Your-Price“ bezeichneten Preisauswahlverfahren völlig frei bestimmt werden kann. Ein Käufer hat auch bei „Select-Your-Price“ keine absolute Gewissheit über den Ausgang eines Gebotes, da nicht alle der angebotenen Preispunkte oberhalb der geheimen Preisschwelle liegen müssen.

Die **maximale Anzahl** der durch einen Käufer auf ein Produktangebot möglichen Gebote kann ebenfalls durch eine Design-Variable dargestellt werden. Mit ihr ist es einem Verkäufer möglich, entweder den von Priceline eingesetzten Fall ledig-

lich einmaliger Gebotsabgabe nachzubilden oder mehrere Gebote zuzulassen. Über diese Design-Variable wird somit nicht nur die Unterscheidung zwischen bereits angesprochener Grund- und erweiterter Form erreicht, sondern auch direkt das Gebotsverhalten eines Käufers beeinflusst. So könnte eine geringe Anzahl verbleibender Gebote einen Käufer dazu veranlassen, Gebote in größeren Schritten zu erhöhen und somit näher an seine wahre Zahlungsbereitschaft zu gelangen.

Eine Reihe von Design-Variablen spiegeln **Restriktionen** wider, die einem Verkäufer ebenfalls erlauben, das Gebotsverhalten eines Käufers durch die explizite Beeinflussung von Such- und Strafkosten in die von ihm gewünschte Richtung zu modifizieren. Restriktionen können eingesetzt werden, um den Käufer zur Abgabe von Geboten in der Nähe seiner wahren Zahlungsbereitschaft zu veranlassen, ohne dabei viel über die geheime Preisschwelle des Verkäufers preiszugeben. Gemäß einer in [Bern04] vorgenommenen Klassifikation werden die Restriktionen durch kostenbasierte, zeitbasierte und stochastische Design-Variablen dargestellt. Kostenbasierte Restriktionen verursachen einem Käufer dabei Strafkosten, da zusätzliche Gebote nur durch Übernahme von Gebühren möglich sind. Zeitbasierte Restriktionen hingegen ändern direkt die Zeitdauer zwischen der Abgabe aufeinander folgender Gebote und beeinflussen somit die Suchkosten eines Käufers. Stochastische Restriktionen zielen darauf ab, den Käufer über die Möglichkeit weiterer Gebote im Unklaren zu lassen und ihn daher von der Abgabe beliebig vieler Gebote abzuhalten. Ähnlich der Festlegung der Preisschwelle können alle hier angesprochenen Design-Variablen sowohl fix als auch dynamisch gestaltet werden. Dynamik ließe sich beispielsweise durch Abhängigkeit der Werte von der bereits abgegebenen Anzahl an Geboten eines Käufers erreichen.

Mit Hilfe von Design-Variablen lassen sich Reverse-Pricing-Mechanismen eindeutig beschreiben und an die Eigenschaften von Produkten anpassen. Allerdings wird zu ihrer Darstellung ein einheitliches Format benötigt, um einen standardisierten Austausch von Daten zu Produkt und eingesetztem Preismechanismus zu ermöglichen. Eine Übersicht über Preismodelle in XML im folgenden Abschnitt verdeutlicht, dass der automatisierte Transfer von Daten über Reverse-Pricing-Mechanismen durch bestehende XML-Standards nicht unterstützt wird. Dementsprechend ist die Entwicklung von Erweiterungen bestehender Standards notwendige Voraussetzung, um den problemlosen Einsatz des Reverse-Pricing-Mechanismus auch in standardisierten E-Business-Prozessen zu ermöglichen.

3 Preismodelle in XML

In den letzten Jahren haben sich viele Initiativen zum Ziel gesetzt, typische Workflows im Bereich E-Business zu analysieren und einen Standard zu entwickeln. Einige Initiativen haben dabei ihren Fokus auf die Bedürfnisse einer bestimmten Zielgruppe gerichtet, andere dagegen verfolgen einen breiteren Ansatz.

Daher haben diese Standardmodelle durchaus ihre Berechtigung und können auch heute noch nebeneinander existieren. Untersucht werden die Preismodelle der verschiedenen Standards, die z. B. bei Artikellisten oder Faktura eingesetzt werden.

Ein Standard zur elektronischen Datenübertragung für Artikelkataloge ist *BMEcat*, das auf Initiative des Bundesverbandes Materialwirtschaft, Einkauf und Logistik e. V. (BME) entwickelt wurde. Jedem Artikel ist ein fester Preis zugeordnet, der z. B. anhand von Zeiträumen oder Absatzländern differenziert werden kann. Auch die Modifikation durch Faktoren ist möglich [BME03]. Abbildung 2 veranschaulicht *BMEcat* beispielhaft anhand eines entsprechenden XML-Fragmentes.

```
<ARTICLE>
...
  <ARTICLE_PRICE_DETAILS>
    <DATETIME type="valid_start_date">
      <DATE>2004-01-01</DATE>
    </DATETIME>
    <DATETIME type="valid_end_date">
      <DATE>2004-04-01</DATE>
    </DATETIME>
    <ARTICLE_PRICE price_type="net_list">
      <PRICE_AMOUNT>65.95</PRICE_AMOUNT>
      <PRICE_CURRENCY>EUR</PRICE_CURRENCY>
      <TAX>0.16</TAX>
      <PRICE_FACTOR>0.8</PRICE_FACTOR>
      <LOWER_BOUND>1</LOWER_BOUND>
      <TERRITORY>DE</TERRITORY>
      <TERRITORY>NL</TERRITORY>
    </ARTICLE_PRICE>
  </ARTICLE_PRICE_DETAILS>

  <ARTICLE_PRICE_DETAILS>
  ...
  </ARTICLE_PRICE_DETAILS>
...
</ARTICLE>
```

Abbildung 2: Beispiel in *BMEcat* mit zeitlich und regional beschränktem Artikelpreis

Ähnlich verhält es sich bei der *XML Common Business Library (xCBL)*, die von einem Unternehmenskonsortium um Commerce One entwickelt wurde. So ist eine Differenzierung nach Zeit oder Währungen möglich. Über ein Kurzbeschreibungsfeld können zusätzliche Informationen weitergereicht werden. Ferner können eine Reihe von Preistypen spezifiziert werden, z. B. `StandardPrice`, `PromotionalPrice` und `DiscountPrice` [XCBL03].

Die *commerce eXtensible Markup Language (cXML)* ist als Standard von E-Business-Unternehmen wie Ariba und Sterling Commerce sehr breit aufgestellt und bietet keine große Detailtiefe. Preise können in *cXML* nur als Stückpreise auf Produktebene definiert und auf einer höheren Ebene zu einer Gesamtsumme aggregiert werden [CXML04].

Auch der Standard von *RosettaNet* verfügt in der aktuell gültigen Version 1.3 lediglich über einen festen Preis für eine Einheit (`UnitPrice`) [Rose00]. Version

2.0, die sich gerade in der Validierung befindet, wird dagegen auch Preisgültigkeiten für bestimmte Zeiträume und Rabattierungen enthalten [Rose02].

EAN.UCC, der XML-Standard von European Article Numbering (EAN) und dem Uniform Code Council (UCC), bietet unter anderem eine zeitliche Preisdifferenzierung an und erlaubt darüber hinaus die Abbildung verhältnismäßig komplexer Rabatt- und Kostenmodelle. Zudem lassen sich mit Hilfe von Attributen Preise als unikal und nicht öffentlich klassifizieren [EAN03].

Die *Electronic Business using eXtensible Markup Language (ebXML)* wurde von den Vereinten Nationen (UN/CEFACT) und der Organisation für die Förderung Strukturierter Informationsstandards (OASIS) entwickelt, der zahlreiche namhafte Unternehmen angehören. *ebXML* verfolgt einen sehr breiten Ansatz und versucht somit, ein umfassendes Framework für E-Business zu schaffen [OASIS04a]. Ein eigenes Preismodell in *ebXML* ist in der vorliegenden Version noch nicht spezifiziert. Beispiele gebrauchen nur ein sehr rudimentäres Preismodell [OASIS02].

Die *Universal Business Language (UBL)* wird ebenfalls von OASIS entwickelt. Sie ist im Rahmen von *ebXML* einsetzbar, kann aber auch losgelöst benutzt werden. Ebenso ist *ebXML* nicht auf das Vokabular von *UBL* angewiesen. *UBL* wurde als Austauschformat zwischen verschiedenen E-Business-Frameworks entwickelt und das Preismodell enthält deshalb viele der zuvor erwähnten Aspekte der anderen Standards. Allerdings geht auch *UBL* von einem Basispreis aus und bietet keine Möglichkeiten, den Preis als Ergebnis einer parametrisierten, dynamischen Preisfindung auszudrücken [OASIS04b].

Einen Schritt weiter geht das *XML Configuration & Pricing Format (XCPF)* aus dem Bereich der Geo-E-Business-Anwendungen, das den vollständigen Austausch eines gesamten Preismodells für ein unkonfektioniertes Produkt ermöglicht. *XCPF* repräsentiert Preismodelle über mathematische Formeln [Wagn03]. Die Ergebnisse in diesem Bereich sollen bei der Modellierung eines Standards für Reverse Pricing berücksichtigt werden.

Insgesamt gehen alle vorliegenden E-Business-Standards von einem Fixpreis aus, der durch verschiedene Parameter angepasst werden kann. Die Möglichkeit, dynamische und auf Interaktion beruhende Preismechanismen abzubilden, ist hingegen kaum vorhanden. Um die in Abschnitt 1 beschriebene Problemstellung lösen zu können, ist eine Erweiterung der vorhandenen Modelle folglich unerlässlich. Für gängige Auktionsmechanismen haben [Wurm+02] dieses Problem erkannt und die Designmöglichkeiten einer Vielzahl von Auktionen als Preismechanismen in Ihrem „Parametrized Auction Schema“ als XML Schema Definition (XSD) veröffentlicht [PAS00].

Eine entsprechende Standardisierung des Reverse Pricing und verschiedener Designmöglichkeiten in XML liegt bisher nicht vor. Nachfolgend sollen daher die in Abschnitt 2.2 vorgestellten Designmöglichkeiten unter Berücksichtigung von Erweiterbarkeit und Flexibilität in einer standardisierten Sprache modelliert werden.

4 Modellierung von Reverse Pricing in XML

Um den verbreiteten Einsatz eines Preismechanismus im E-Business zu ermöglichen, ist die Einigung aller am Datentransfer beteiligter Parteien auf einen gemeinsamen, den Preismechanismus definierenden Standard unerlässlich. Wie im vorangegangen Abschnitt gezeigt wurde, ist ein solcher Standard für das Reverse Pricing bisher nicht vorhanden. Die im Folgenden vorgestellte Beschreibungssprache Reverse Pricing in XML (RPXML) soll diese Lücke schließen und die standardisierte Verwendung des Reverse-Pricing-Mechanismus ermöglichen.

4.1 Reverse Pricing in XML (RPXML)

RPXML bietet die Möglichkeit, Reverse-Pricing-Mechanismen anhand eines standardisierten XML-Formats strukturiert und detailliert zu beschreiben. Mit Hilfe von RPXML ist es Verkäufern möglich, einen Reverse-Pricing-Mechanismus vollständig zu spezifizieren und ihn den Eigenschaften eines zu verkaufenden Produktes gemäß anzupassen. Dies soll hier aus Gründen der Anschaulichkeit mit beispielhaftem XML-Code geschehen. Eine XML Schema Definition (XSD) für RPXML steht zum Download bereit [RPXML04].

4.1.1 Grundlagen

Eingeleitet wird die strukturierte Darstellung eines Reverse-Pricing-Mechanismus mit einem `<reversePricing>`-Tag, das zugleich die Festlegung der Einheiten für Währung und Zeit über Attribute ermöglicht. Innerhalb dieses Tag gestatten eine Reihe von teilweise mehrstufigen Elementen die genaue Spezifikation des Mechanismus.

```
<?xml version="1.0" encoding="utf-8" ?>
<reversePricing currency="EUR" timeUnit="s">
  <thresholdPrice>10.40</thresholdPrice>
  <maxBids>4</maxBids>
  <lowerBound visible="true">8.50</lowerBound>
  <upperBound visible="true">13.50</upperBound>
  <regularPrice visible="true">16.00</regularPrice>
  ...
</reversePricing>
```

Abbildung 3: RPXML-Fragment mit grundlegenden Tags

Von zentraler Bedeutung ist zunächst die Bestimmung grundlegender Elemente wie der geheimen Preisschwelle eines Verkäufers oder der maximal möglichen Anzahl an Geboten eines Käufers. Die beiden RPXML-Tags `<thresholdPrice>` und `<maxBids>` in Abbildung 3 verdeutlichen dies beispielhaft.

Um die bereits erwähnte Unsicherheit eines Käufers bei der Abgabe eines Gebotes zu reduzieren, könnte ein Verkäufer diesem eine Referenzpreisspanne in Form einer Unter- und einer Obergrenze sinnvoller Preise anzeigen. Ein Käufer müsste dann lediglich entscheiden, wo er sein Gebot innerhalb dieser Preisspanne platzieren möchte, ohne jedoch über die Gewissheit eines erfolgreichen Gebotes zu verfügen. Wie aus Abbildung 3 hervorgeht, gestattet das `<regularPrice>`-Tag die Angabe eines Referenzpreises über einen Preispunkt. Käufer könnten dann selbst entscheiden, wie nahe ihr Gebot an diesem Referenzpreispunkt liegen soll.

4.1.2 Preisauswahlverfahren

Das Preisauswahlverfahren legt fest, auf welche Weise ein Käufer einen konkreten Preis im Rahmen eines Reverse-Pricing-Mechanismus wählen kann. Neben der völlig freien Auswahl eines Preises ist es möglich, Käufern durch die Wahl des von [Cher03] als „Select-Your-Price“ bezeichneten Preisauswahlverfahrens implizit eine Referenzpreisspanne vorzugeben. Käufer sind somit auf die Auswahl eines Preises aus einer Liste vorgegebener Preispunkte beschränkt. In Abbildung 4 wird ein solches Preisauswahlverfahren über den RPXML-Tag `<mechanism>` festgelegt. Wird dessen Attribut `type` auf `"selectYourPrice"` gesetzt, können innerhalb eines `<lists>`-Tag verschiedene Listen definiert werden. Für jede der nun folgenden Listen lässt sich anhand der Attribute `startBidNo` bzw. `endBidNo` eines `<list>`-Tags eine gültige Anfangs- und eine gültige Endgebotsnummer bestimmen. Im Beispiel aus Abbildung 4 liegt die definierte Preisliste allen Gebotsschritten eines Käufers zugrunde, da sie vom 1. bis zum 4. Gebot gilt und in dem Mechanismus pro Käufer nur maximal vier Gebote erlaubt sind (vgl. Abbildung 3).

```
<?xml version="1.0" encoding="utf-8" ?>
<reversePricing currency="EUR" timeUnit="s">
  ...
  <mechanism type="selectYourPrice">
    <lists>
      <list startBidNo="1" endBidNo="4" type="absolutePrices">
        <listItem id="1">8.50</listItem>
        <listItem id="2">9.75</listItem>
        <listItem id="3">11.00</listItem>
        <listItem id="4">12.25</listItem>
        <listItem id="5">13.50</listItem>
      </list>
    </lists>
    <listProgression>
      <listIncrement startBidNo="2" type="absolute">
        <constant>1.50</constant>
      </listIncrement>
    </listProgression>
  </mechanism>
  ...
</reversePricing>
```

Abbildung 4: RPXML-Fragment mit Detailinformationen zum Preisauswahlverfahren

Können durch einen Käufer mehrere Gebote abgegeben werden, so ist die Anpassung der Preispunkte in dem für den ersten Gebotsschritt gültigen `<list>`-Tag möglich. Abbildung 5 zeigt anhand einer Liste von fünf Preispunkten, dass neben der Wiederverwendung der Ausgangsliste (keine Modifikation) und der Festlegung eines neuen `<list>`-Tag (neue Liste) sowohl eine Transformation der ursprünglichen Liste als auch der Einsatz von Erhöhungsbeträgen zu einer solchen Anpassung führen können.

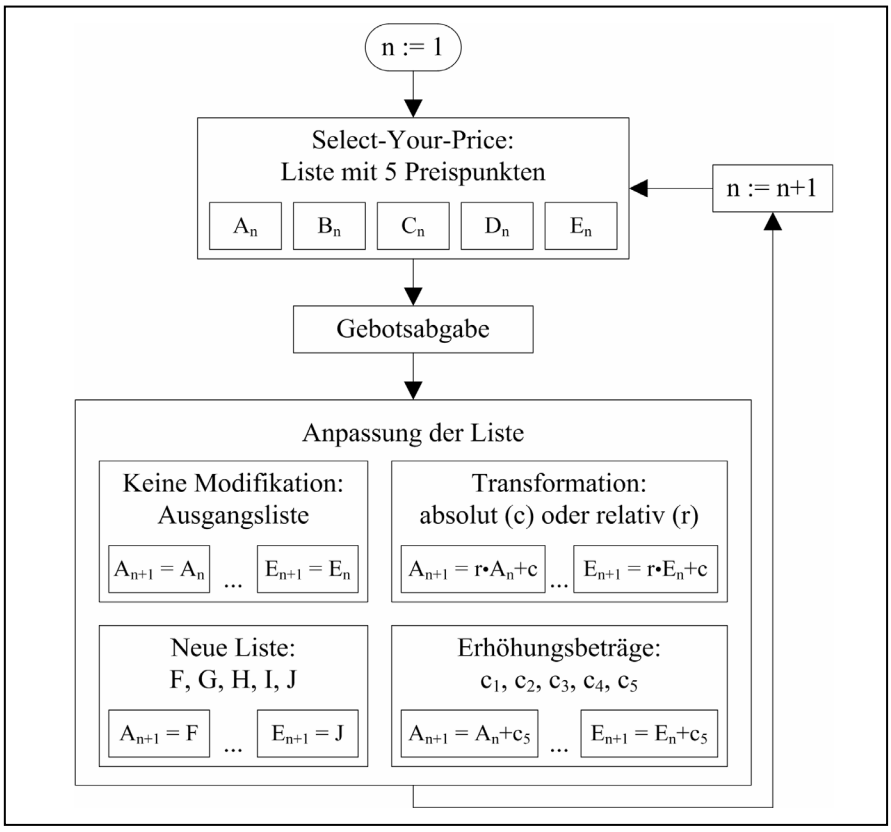

Abbildung 5: Mögliche Anpassung von Preislisten beim Select-Your-Price-Mechanismus

Die Transformation einer Liste ist mit Hilfe der innerhalb von `<list-Progression>` angesiedelten `<listIncrement>`-Tags möglich. Ist ein entsprechendes Tag spezifiziert, wird eine in einem bestimmten Gebotsschritt gültige Liste von Preispunkten um konstante oder dynamisch generierte Werte erhöht. Dabei lassen sich auch hierfür Anfangs- und gegebenenfalls Endgebotsnummern festlegen. Nach einem ersten erfolglosen Gebot unterhalb der Preisschwelle eines Anbieters würde ein potenzieller Käufer im Beispiel aus Abbildung 4 bei der Abgabe eines zweiten Gebotes fünf Preispunkte von 10 € (= 8,50 € + 1,50 €) bis

15 € (= 13.50 € + 1,50 €) angezeigt bekommen. Aus dieser modifizierten Liste könnte er dann sein zweites Gebot auswählen. Da in dem beispielhaften `<listIncrement>`-Tag kein Attribut `endBidNo` festgelegt ist, würde sich die Liste auch für nachfolgende Gebotsschritte um jeweils 1,50 € erhöhen. Ab dem dritten Gebot wäre somit jeder Preis in der Liste erfolgreich, da bereits das geringst-mögliche Gebot von 11,50 € oberhalb der geheimen Preisschwelle des Verkäufers von 10,40 € liegen würde.

Im Gegensatz zur Bestimmung von konstanten oder dynamisch generierten Werten für die Transformation einer Ausgangsliste benötigt die Verwendung von Erhöhungsbeträgen eigene Werte c_i (i = 1, ..., 5) für jeden der anzupassenden Preispunkte. Wird das `type`-Attribut des `<list>`-Tag nicht wie bei der Definition einer Ausgangsliste auf `absolutePrices` sondern auf `increments` gesetzt (vgl. Abbildung 4), kann eine solche Liste von Erhöhungsbeträgen gesetzt werden. Werden diese Beträge zu den Preispunkten der Ausgangsliste addiert, bildet sich eine modifizierte Preisliste, die dem Käufer im folgenden Gebotsschritt angezeigt wird. In einem weiteren Szenario könnten Käufer statt der modifizierten Preisliste die Liste der Erhöhungsbeträge direkt präsentiert bekommen. Ein Gebot würde sich dann als Summe aus zuletzt genanntem Preis und dem gewählten Erhöhungsbetrag ergeben. Die Berechnung dieser Gebotshöhe wäre in diesem Fall neben der Auswahl eines geeigneten Betrages zusätzlich zu erbringender mentaler Aufwand eines Käufers.

4.1.3 Restriktionen

Der Einsatz verschiedener Restriktionen gestattet einem Verkäufer, Gebote eines Käufers mit Strafkosten zu belegen sowie dessen Suchkosten explizit zu beeinflussen. Die somit mögliche Modifikation der Bietkosten eines Käufers lässt sich wie in Abschnitt 2.2 beschrieben in kostenbasierte, zeitbasierte und stochastische Restriktionen untergliedern.

Abbildung 6 stellt die Ausgestaltung der Restriktionen in beispielhaftem RPXML-Code dar. Über das RPXML-Tag `<fee>` kann der Verkäufer die Höhe der Gebühren festlegen, die ein Käufer für zusätzliche Gebote auf ein Produkt übernehmen muss. Dabei lassen sich wie schon zuvor gezeigt auch hier wieder ein Anfangs- und ein Endgebot über die Attribute `startBidNo` und `endBidNo` definieren. Im Beispielcode hätte ein Käufer nach einem freien ersten Gebot also genau 1,00 € für weitere Gebote zu bezahlen. Dabei deutet die Verwendung des `<constant>`-Tag darauf hin, dass sowohl unveränderliche als auch während des Gebotsvorgangs berechnete, dynamische Werte verwendet werden können. Letztgenannte Möglichkeit ist Gegenstand von Abschnitt 5.

Mit Hilfe des `<raiseOfThreshold>`-Elementes kann der Verkäufer bestimmen, um welchen Betrag sich seine geheime Preisschwelle pro zusätzlichem Gebot eines Käufers vergrößert. Dieser Vorgang ist für den Käufer mit Strafkosten verbunden, da dieser nach erfolgter Anhebung der Preisschwelle einen höheren Be-

trag überbieten muss als in vorherigen Gebotsschritten. Wie alle anderen Restriktionen wirkt sich diese Anhebung nur auf einen bestimmten Käufer aus; das Gebotsverhalten anderer (potenzieller) Käufer hat keinen Einfluss auf die Modifikation der Werte. Auch der mittels des <minimumIncrement>-Tag definierte, minimale Erhöhungsbetrag über ein vorheriges Gebot ist für einen Käufer mit Strafkosten verbunden, da dieser ein eventuell höher als ursprünglich gewünschtes Gebot abgeben muss, um erfolgreich zu bieten.

```
<?xml version="1.0" encoding="utf-8" ?>
<reversePricing currency="EUR" timeUnit="s">
  ...
  <notificationDelay>900</notificationDelay>
  <startingProbability>100</startingProbability>
  <constraints>
    <fee startBidNo="2">
      <constant>1.0</constant>
    </fee>
    <raiseOfThreshold startBidNo="2" endBidNo="3">
      <constant>0.5</constant>
    </raiseOfThreshold>
    <minimumIncrement startBidNo="2">
      <constant>2.0</constant>
    </minimumIncrement>
    <timeDelay startBidNo="2">
      <constant>120</constant>
    </timeDelay>
    <decreaseProbability startBidNo="2">
      <constant>20</constant>
    </decreaseProbability>
  </constraints>
  ...
</reversePricing>
```

Abbildung 6: Festlegung von Restriktionen in RPXML

Die explizite Modifikation der einem Käufer bei der Abgabe eines zusätzlichen Gebotes entstehenden Suchkosten kann ein Verkäufer durch die RPXML-Tags <notificationDelay> und <timeDelay> erreichen. Die gleichzeitige Festlegung beider Elemente ist dabei sicherlich wenig sinnvoll: Ersteres Element definiert eine Zeitspanne bis zur Benachrichtigung eines Käufers über den Ausgang eines Gebotes (z. B. durch Verschicken einer Benachrichtigung per E-Mail), letzteres hingegen verzögert die frühestmögliche Abgabe eines Gebotes um eine bestimmte Anzahl von Zeiteinheiten (im Beispiel zwei Minuten).

Schließlich kann ein Verkäufer stochastische Restriktionen dazu verwenden, einem Käufer die Abgabe weiterer Gebote nur mit einer um eine bestimmte Prozentzahl verringerten Wahrscheinlichkeit zu ermöglichen. Die Festlegung eines solchen Prozentbetrags über das Element <decreaseProbability> reduziert eine über <startingProbability> festgelegte Ausgangswahrscheinlichkeit. Im Beispiel aus Abbildung 6 wären einem Käufer die Abgabe eines zweiten Gebots also nur mit 80%-Wahrscheinlichkeit, eines dritten Gebots nur mit 60%-Wahrscheinlichkeit etc. gestattet.

4.2 Einsatzgebiete

In Ergänzung bestehender Vertriebskanäle bietet Reverse Pricing die Möglichkeit, neue Kundensegmente anzusprechen und den Umsatz eines Verkäufers positiv zu beeinflussen. Die vorgestellte Beschreibungssprache RPXML stellt den Designraum des Reverse Pricing in einer standardisierten XML-Struktur dar und bildet somit die Grundlage für dessen Einsatz im E-Business.

Um Reverse Pricing in bestehenden E-Business Anwendungen zu integrieren, ist es lediglich notwendig, den `<reversePricing>`-Tag anstelle eines in bestehenden Standards üblichen Fixpreis-Tags einzusetzen. Zwei Einsatzgebiete illustrieren die Notwendigkeit dieser Vorgehensweise:

Zunächst können Katalogdaten mit Hilfe des `<reversePricing>`-Tag zwischen Verkäufer und dem Betreiber einer Reverse-Pricing-Plattform transferiert werden. Nach der Spezifikation eines Angebots in RPXML lässt ein Verkäufer die transferierten Daten dann von einem Plattformbetreiber in der gewünschten Form präsentieren. Durch die Trennung von Daten und Layout wäre eine Anpassung an das Design einer Plattform leicht möglich.

Des Weiteren gestattet RPXML im Bereich des automatisierten Handels den Einsatz von Reverse Pricing. Kauf-Agenten erhalten bei der Angebotsanfrage keinen Fixpreis, sondern die XML-Struktur, die der Aufforderung einer Gebotsabgabe entspricht. Aus der XML-Struktur erkennt der Kauf-Agent, welche Regeln in dem Verhandlungsprozess angewandt werden. Die geheime Preisschwelle wird dem Kauf-Agenten in diesem Szenario nicht mitgegeben.

5 Erweiterung mit MathML

Mit der in Abschnitt 4 vorgestellten XML-Struktur ist die Abbildung einer Grundfunktionalität des Reverse Pricing möglich. Der Einfluss externer Ereignisse hingegen kann durch das bisherige Modell nicht berücksichtigt werden. Möchte ein Verkäufer auf verfügbare Einflussgrößen wie Kundennachfrage oder den Abverkauf der Produkte dynamisch reagieren können, müssen verschiedene, zur Laufzeit berechenbare Funktionen in RPXML enthalten sein. Zur Beschreibung solcher Funktionen kann die Mathematical Markup Language (MathML) eingesetzt werden.

5.1 Grundlagen von MathML

MathML ist eine Empfehlung des W3 Konsortiums und liegt derzeit in Version 2 vor [Carl+03]. MathML basiert auf XML und wird für die Kodierung mathemati-

scher Strukturen sowohl zur Darstellung und Präsentation (z. B. Einsatz im Web als Meta-Sprache für den Browser) als auch für die Kommunikation zwischen Maschine und Maschine eingesetzt.

Für das vorliegende Problem der Erweiterung von RPXML um dynamisch zur Laufzeit berechnete Variablen wird die Kodierung mathematischer Inhalte benötigt. Dies ermöglicht MathML durch so genannte contents-Tags, die in Prefix-Notation angegeben werden. Hiermit wird eine Notation mathematischer Ausdrücke bezeichnet, bei der jeder Operator den zugehörigen Operanden vorangeht. Dadurch wird die Klammerung der Ausdrücke überflüssig und eine automatisierte Weiterverarbeitung vereinfacht. Abbildung 7 verdeutlicht diese Notation in MathML für den Ausdruck x^2+4. Das `<apply>`-Tag kapselt dabei einen eigenständigen mathematischen Ausdruck, der seinerseits wiederum in einem `<apply>`-Tag mit einem vorangehenden Operator logisch verknüpft werden kann.

```
<apply>
  <plus/>
  <apply>
    <power/>
    <ci>x</ci>
    <cn>2</cn>
  </apply>
  <cn>4</cn>
</apply>
```

Abbildung 7: Der Ausdruck x^2+4 in MathML

5.2 MathML als Erweiterung von RPXML

In der Praxis sind weitere Verfeinerungen des in Abschnitt 4 dargestellten Reverse-Pricing-Mechanismus denkbar. So könnte die Preisschwelle in Abhängigkeit der Angebotslaufzeit oder der Käufernachfrage angepasst werden.

Auf diese Art könnte der Mechanismus durch zusätzliche Dynamik vorteilhafter gestaltet werden. Mit dem in Abschnitt 4 vorgestellten Modell ist das noch nicht möglich, da die Werte bereits beim Erstellen von RPXML festgelegt werden. Erst durch die Einführung von zur Laufzeit berechneten Variablen ist eine solche Dynamik erreichbar.

Die Möglichkeit der Erweiterung um dynamische Variablen könnte unter anderem für den gesamten `<constraints>`-Tag sinnvoll sein. Auch für die Berechnung der Erhöhungsschritte der `<listProgression>`-Tags bietet sich mehr Dynamik an, um auf externe Größen reagieren zu können. Abbildung 8 zeigt ein entsprechendes XSD-Fragment für das erweiterte RPXML.

Unterhalb des `<formula>`-Tag befinden sich MathML-Tags, die eine Beeinflussung der Variablen durch externe Größen ermöglichen. Denkbar sind z. B. Anzahl der bereits abgegebenen Gebote (`bidNo`), Restlaufzeit des Angebots (`remai-`

ningOfferTime), bisherige Laufzeit des Angebots (runningOfferTime), Anzahl der bereits verkauften Produkte (soldProducts), Anzahl der Gebote aller Kunden auf das Produkt (totalBids) und Anzahl der Zugriffe für das Produkt (totalClicks). Weitere Größen sind vorstellbar, über deren Relevanz und Auswirkungen liegen allerdings bisher keine empirischen Erkenntnisse vor, so dass in diesem Bereich zunächst weiterer Forschungsbedarf besteht.

```
...
<xsd:complexType name="constraintNode">
  <xsd:sequence maxOccurs="unbounded">
    <xsd:element name="raiseOfThreshold" type="constraint"
minOccurs="0"/>
    <xsd:element name="fee" type="constraint" minOccurs="0"/>
    <xsd:element name="minimumIncrement" type="constraint"
minOccurs="0"/>
    <xsd:element name="decreaseProbability" type="constraint"
minOccurs="0"/>
    <xsd:element name="timeDelay" type="timeConstraint"
minOccurs="0"/>
  </xsd:sequence>
</xsd:complexType>

<xsd:complexType name="constraint">
  <xsd:choice>
    <xsd:element name="constant" type="xsd:float"/>
    <xsd:element name="formula" type="formula"/>
  </xsd:choice>
  <xsd:attribute name="startBidNo" type="xsd:integer"
use="optional"/>
  <xsd:attribute name="endBidNo" type="xsd:integer"
use="optional"/>
</xsd:complexType>

<xsd:complexType name="formula">
  <xsd:sequence>
    <xsd:element ref="mml:math"/>
  </xsd:sequence>
</xsd:complexType>
...
```

Abbildung 8: XSD-Fragment für das um dynamische Einflussgrößen erweiterte RPXML

5.3 Implementierungsvorschlag

Bei der angestrebten Erweiterung mit MathML sind einige Praxisprobleme zu lösen, die hier anhand eines Implementierungsvorschlags angegangen werden sollen. Es wird gezeigt, wie die erweiterte XML-Struktur in die Grundform aus Abschnitt 4 überführt werden kann. Das MathML-Fragment aus Abbildung 9 demonstriert die Transformation.

```
<fee>
  <formula>
    <math>
      <apply>
        <times/>
        <cn>0.5</cn>
        <ci>BidNo</ci>
      </apply>
    </math>
  </formula>
</fee>
```

Abbildung 9: MathML-Fragment als Beispiel für die Festlegung von Gebotskosten

Um eine XML-Struktur in eine andere zu überführen, eignen sich eXtensible Stylesheet Language Transformations (XSLT) [Clar99]. Für die vorliegende Aufgabe empfehlen sich dabei zwei Zwischenschritte wie in Abbildung 10 dargestellt, um den Prozess einfach und übersichtlich zu gestalten.

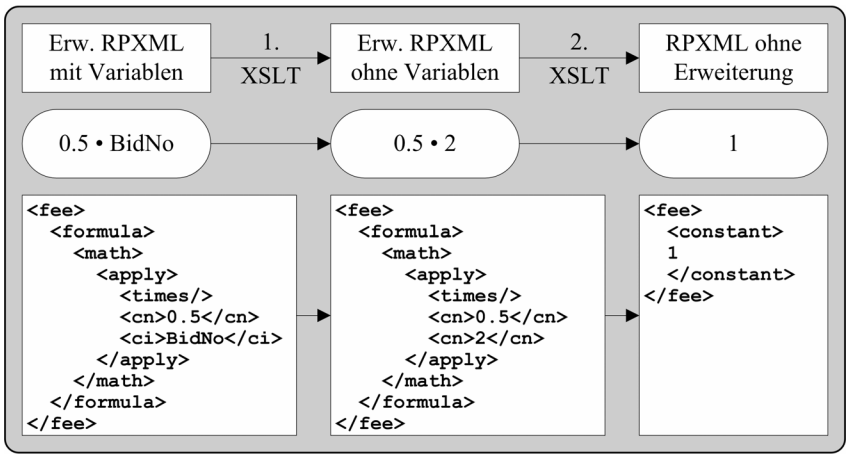

Abbildung 10: Schrittweise Auswertung eines erweiterten RPXML-Fragmentes durch XSL-Transformationen

Zunächst müssen alle Variablen in einer XSL-Transformation mit den im aktuellen Kontext zur Laufzeit gültigen Werten belegt werden. Diese Werte können innerhalb des eingesetzten Frameworks (z.B. ASP.NET oder cocoon) berechnet und als Parameter an die Transformation übergeben werden. Als global verfügbare Variablen können sie dort die als Platzhalter externer Einflussgrößen fungierenden `<ci>`-Tags der erweiterten RPXML-Struktur ersetzen.

Sind alle Variablen ersetzt, wird eine zweite XSL-Transformation benötigt, um die nun vorliegenden MathML-Ausdrücke zu berechnen. Alternativ wäre für diesen Schritt auch der Einsatz eines Webservice denkbar. Dieser würde die Ausdrücke

berechnen und eine entsprechend ausgewertete RPXML-Struktur als Ergebnis zurückliefern.

Nach der zweiten XSL-Transformation liegt schließlich eine RPXML-Struktur ohne Erweiterung vor, die wie gehabt verarbeitet werden kann. Somit stellt die Möglichkeit, RPXML um Funktionen zu erweitern, eine Option dar, die durch die angegebene Vorgehensweise lückenlos integriert werden kann.

Insgesamt betrachtet steht mit Erweiterung der RPXML-Struktur um MathML-Fragmente ein mächtiges und flexibles Werkzeug zur Verfügung. Durch die Verwendung entsprechender MathML-Ausdrücke könnte der Einfluss externer Größen in der Ausgestaltung eines Reverse-Pricing-Mechanismus berücksichtigt werden. Da der Wissenschaft insbesondere über den Einsatz solcher dynamischer Erweiterungen des Reverse Pricing kaum Erkenntnisse vorliegen, ist dies ein relevantes Problemfeld für die zukünftige Forschung.

6 Fazit

Das Internet bietet neue Möglichkeiten zum Einsatz dynamischer Preismechanismen. Am Beispiel von Reverse Pricing wird gezeigt, dass bisherige XML-Standards im Bereich E-Business den gewachsenen Anforderungen nicht gerecht werden. Mit ihrer Hilfe ist lediglich die Abbildung von Fixpreis-Mechanismen aus der klassischen Geschäftswelt möglich. Standardisierungen im Bereich dynamischer, auf Käufer-Käufer- oder Käufer-Verkäufer-Interaktion beruhender Preismechanismen sind bisher nur vereinzelt anzutreffen. Mit der Parametrisierung und Abbildung von Reverse Pricing in XML schließt diese Arbeit die in diesem Bereich bestehende Lücke und gibt einen Impuls für die Erweiterung bestehender Standards. Zusätzlich werden auch auf andere Mechanismen übertragbare Ansätze zur Modellierung dynamischer Preismechanismen entwickelt.

Die modellierte XML-Struktur RPXML bildet den Designraum von Reverse Pricing mit hohem Detaillierungsgrad ab. Durch den Einsatz der Mathematical Markup Language können darüber hinaus komplexe Funktionen spezifiziert und eine Abhängigkeit externer Einflussgrößen auf den Mechanismus modelliert werden.

Die vorgestellte Lösung kann zur Erweiterung von Katalogdaten benutzt oder aber im automatisierten Handel zwischen Agenten oder Agenten und Plattform eingesetzt werden. Aufgrund zunehmender Preisdifferenzierungen im Internet und der Abkehr von klassischen Fixpreisen besteht im Bereich der Standardisierung dynamischer Preismechanismen auch in Zukunft erheblicher Forschungsbedarf.

Literatur

[Bako97] Bakos, Y.: Reducing Buyer Search Costs: Implications for Electronic Marketplaces. Management Science 43 (12), 1997: S. 1676-1692.

[Bern04] Bernhardt, M.: Classification of Design Options in Reverse Pricing Mechanisms. In: Bichler, M.; Holtmann, C.; Kirn, S.; Müller, J. P.; Weinhardt, C. (Hrsg.): Coordination and Agent Technology in Value Networks. GITO: Berlin, 2004, S. 29-43.

[BME03] eBusiness Standardization Committee: BMEcat, Version 1.2, http://www.bmecat.org, 2003, Abruf am 13.05.2004.

[Bray+04] Bray, T.; Paoli, J.; Sperberg-McQueen, C. M.; Maler, E.; Yergeau, F.; Cowan, J.: Extensible Markup Language (XML) 1.1 W3C Recommendation 04 February 2004. http://www.w3.org/TR/xml11/, 2004, Abruf am 27.04.2004.

[Carl+03] Carlisle, D.; Ion, P.; Miner, R.; Poppelier, N.: Mathematical Markup Language (MathML) Version 2.0 (Second Edition) W3C Recommendation 21 October 2003. http://www.w3.org/TR/2003/REC-MathML2-20031021/, 2003, Abruf am 14.05.2004.

[Cher03] Chernev, A.: Reverse Pricing and Online Price Elicitation Strategies in Consumer Choice. Journal of Consumer Psychology 13 (1&2), 2003: S. 51-62.

[Clar99] Clark, J.: XSL Transformations (XSLT) Version 1.0 W3C Recommendation 16 November 1999. http://www.w3.org/TR/xslt, 1999, Abruf am 14.05.2004.

[CXML04] cXML.org: commerce eXtensible Markup Language Version 1.2.0.11. http://xml.cxml.org/current/cXML.zip, 2004, Abruf am 13.05.2004.

[EAN03] EAN International: EAN.UCC XML Version 1.3.1. http://www.ean-int.org/EAN.UCC%20XML%20Standard%20Schemas.html, 2003, Abruf am 13.05.2004.

[Fay04] Fay, S.: Partial Repeat Bidding in the Name-Your-Own-Price Channel. Marketing Science, zur Veröffentlichung angenommen.

[HaTe03] Hann, I.-H.; Terwiesch, C.: Measuring the Frictional Costs of Online Transactions: The Case of a Name-Your-Own-Price Channel. Management Science 49 (11), 2003: S. 1563-1579.

[Kelk+02] Kelkar, O.; Leukel, J.; Schmitz, V.: Towards Extended Price Models in XML: Standards for Electronic Product Catalogs. In: Piattini, M. G.; Filipe, J.; Braz, J. (Hrsg.): Enterprise Information Systems IV, Kluwer Academic Publishers: Dordrecht, 2002, S. 251-259.

[MiWe82] Milgrom, P. R.; Weber, R. J.: A Theory of Auctions and Competitive Bidding. Econometrica 50 (5), 1982: S. 1089-1122.

[OASIS02] OASIS: Message Service Specification Version 2.0. http://ebxml.org/specs/ebMS2.pdf, 2002, Abruf am 13.05.2004.

[OASIS04a] OASIS: Electronic Business using eXtensible Markup Language. http://www.ebxml.org/faq.htm, 2004, Abruf am 13.05.2004.

[OASIS04b] OASIS: Universal Business Language Version 1.0. http://docs.oasis-open.org/ubl/cd-UBL-1.0.zip, 2004, Abruf am 13.05.2004.

[OECD99] OECD: The Economic and Social Impact of Electronic Commerce – Preliminary Findings and Research Agenda. http://www.oecd.org/dataoecd/3/12/1944883.pdf, 1999, Abruf am 27.04.2004.

[PAS00] Parameterized Auction Schema Version 1.0. http://www.csc.ncsu.edu/faculty/wurman/Auction_xsd/ParamAuction.html, 2000, Abruf am 29.04.2004.

[Rose00] RosettaNet.org: Cluster 3: Order Management, Segment A: Quote and Order Entry, PIP3A2: Query Price and Availability, Version 1.3. http://www.rosettanet.org/RosettaNet/Doc/0/01ORFTF5L7A13FCB0282FOPTD8/3A2QueryPriceAndAvailability.zip, 2000, Abruf am 14.05.2004.

[Rose02] RosettaNet.org: Cluster 3: Order Management, Segment A: Quote and Order Entry, PIP3A2: Query Price and Availability, Version 02.01.00B. http://www.rosettanet.org/RosettaNet/Doc/0/7PPFDTIR7M4KD4QBG4944708BC/3A2_RequestPriceandAvailability_R02_01_00B.zip, 2002, Abruf am 14.05.2004.

[RPXML04] Reverse Pricing in XML (RPXML) Version 1.0. http://www.reverse-pricing.com/downloads/rpxml.xsd, 2004, Abruf am 10.06.2004.

[Span+03] Spann, M.; Skiera, B.; Schäfers, B.: Reverse-Pricing-Verfahren und Möglichkeiten zur Messung von individuellen Suchkosten und Zahlungsbereitschaften. Schmalenbachs Zeitschrift für betriebswirtschaftliche Forschung, zur Veröffentlichung angenommen.

[Stig61] Stigler, G. J.: The Economics of Information. Journal of Political Economy 69. 1961: S. 213-225.

[Wagn03] Wagner, R. M.: A Model For The Representation And Transaction Of Complex Pricing And Ordering For High-Value Spatial Products and Services. Dissertation an der Technischen Universtität Berlin, 2003.

[Wurm+01] Wurman, P. R.; Wellmann, M. P.; Walsh, W. E.: A Parametrization of the Auction Design Space. Games and Economic Behavior 35, 2001: S. 304-338.

[Wurm+02] Wurman, P. R.; Wellmann, M. P.; Walsh, W. E.: Specifying Rules for Electronic Auctions. AI Magazine 23 (3), 2002: S. 15-23.

[XCBL03] xCBL.org: XML Common Business Library Version 4.0, http://www.xcbl.org/xcbl40/xcbl40.html, 2003, Abruf am 13.05.2004.