cXML-Benutzer-handbuch

VERSION 1.1

APRIL 2000

Inhaltsverzeichnis

Anhang B
Neue Funktionen in cXML 1.1 105

Vorwort

Im vorliegenden Dokument wird die Verwendung von cXML (commerce eXtensible Markup Language) für die Übermittlung von Daten im elektronischen Handel beschrieben.

Zielgruppe und Voraussetzungen

Es richtet sich in erster Linie an Programmierer cXML-fähiger Anwendungen und orientiert sich dabei an den Bedürfnissen von Lieferanten, die ihre E-Commerce-Websites mit Punchoutfunktionen versehen möchten.

cXML ist eine offene und flexible Programmiersprache für die Transaktionserfordernisse von:

- elektronischen Produktkatalogen
- cXML-Punchoutkatalogen
- Beschaffungsanwendungen
- Käufergemeinschaften

Die Leser sollten bereits über Erfahrungen im Umgang mit dem E-Commerce-Konzept und dem HTTP-Webkommunikationsstandard verfügen.

Spezifische Beschaffungsanwendungen oder E-Commerce-Netzknoten werden in diesem Dokument nicht näher beschrieben.

Empfohlene Kapitel

- **E-Commerce-Manager:** Einen Überblick über die cXML-Funktionen finden Sie in Kapitel 1, „Einführung in cXML“.
- **Webdesigner:** Webdesigner, die E-Commerce-Sites implementieren, sollten alle Kapitel lesen.
- **Administratoren von Punchoutsites:** Webdesignern, die über Erfahrungen mit Punchoutwebsites verfügen, empfehlen wir die Lektüre von Anhang B, „Neue Funktionen in cXML 1.1“.

Kapitel 1
Einführung in cXML

Im folgenden Kapitel erhalten Sie eine Einführung in die Struktursprache cXML (commerce eXtensible Markup Language), die vorrangig für E-Commerce-Transaktionen Verwendung findet.

Im Rahmen dieser cXML-Übersicht werden folgende Themen behandelt:

- Was cXML leisten kann
- Anwendungen, die cXML verwenden
- Angebotsstrategie
- Prüfung anhand von DTDs
- Profiltransaktion
- XML-Dienstprogramme

Was cXML leisten kann

Mit cXML ist es möglich, dass Käufer, Verkäufer, informationserfassende Unternehmen und Vermittler über eine einzige standardisierte und gleichzeitig systemoffene Sprache miteinander kommunizieren.

Erfolgreiche Portale für den elektronischen Handel zwischen Unternehmen (B2B E-Commerce) benötigen ein flexibles, weit verbreitetes Protokoll. Als sauber definierte, leistungsfähige Sprache, die speziell für den B2B E-Commerce entwickelt wurde, wird cXML bevorzugt von Großkäufern und -verkäufern verwendet und ist somit der Schlüssel für einen breiten Zugriff auf Ihre Produkte und Dienstleistungen.

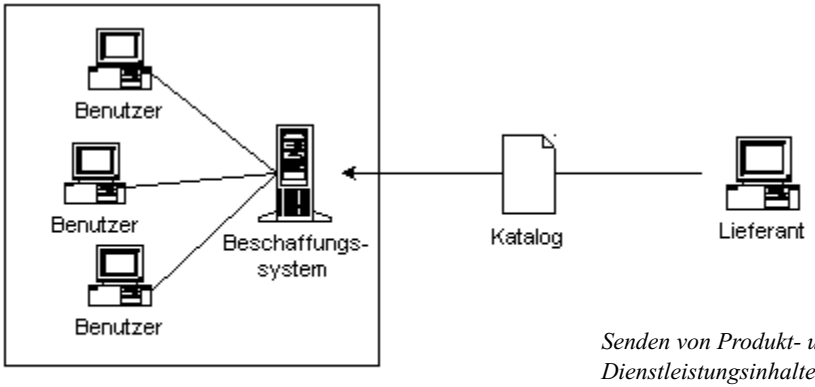
cXML-Transaktionen werden über *Dokumente* abgewickelt, bei denen es sich um einfache Textdateien mit klar definierten Formaten und Inhalten handelt. Die meisten Typen von cXML-Dokumenten lassen sich analog zu Papierdokumenten verwenden, die im Geschäftsverkehr traditionell zum Einsatz kommen.

In den nachfolgenden Abschnitten werden die Haupttypen von cXML-Dokumenten genauer beschrieben.

Kataloge

Kataloge sind Dateien, in denen Produkt- und Dienstleistungsinhalte an Käuferunternehmen herangetragen werden. Neben den angebotenen Produkten und Dienstleistungen enthalten Kataloge auch Informationen zu den jeweiligen Preisen. Kataloge stellen den wichtigsten Kommunikationskanal von Ihnen zu Ihren Kunden dar.

Kataloge werden erstellt, damit Käuferunternehmen Ihre Produkt- und Dienstleistungsangebote mit Hilfe von Beschaffungsanwendungen einsehen und von Ihnen erwerben können. Die Beschaffungsanwendungen lesen Ihre Kataloge und speichern deren Inhalt in internen Datenbanken. Nachdem ein Käuferunternehmen Ihre Kataloge genehmigt hat, wird deren Inhalt für Benutzer sichtbar, die bestimmte Artikel auswählen und einer Kaufanforderung hinzufügen können.



Senden von Produkt- und Dienstleistungsinhalten an Käuferunternehmen

Kataloge lassen sich unabhängig von Menge, Preis und Lieferart grundsätzlich für alle Produkte oder Dienstleistungen erstellen.

Neben obligatorischen Grundinformationen zu jedem Artikel können auch optionale Informationen (wie z. B. mehrsprachige Beschreibungen) hinzugefügt werden, die die Katalogfunktionen erweitern.

Punchout

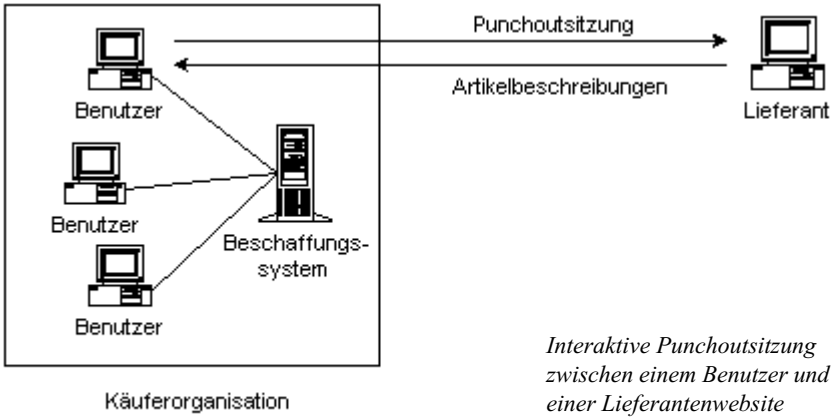
Punchouts stellen als dynamische, interaktive Kataloge auf Ihrer Website eine Alternative zu statischen Katalogdateien dar.

Wenn Sie über eine E-Commerce-Website verfügen, können Sie diese zusätzlich mit Punchoutunterstützung versehen. Punchoutsites sind in der Lage, auf cXML-Basis über das Internet selbständig mit Beschaffungssystemen zu kommunizieren.

Weitere Informationen:

Kapitel 2, „Implementieren einer Punchoutsite“

Bei Punchoutsites wird durch die Beschaffungsanwendung an Stelle von Produkt- oder Preisinformationen eine Schaltfläche angezeigt. Wenn der Benutzer auf diese Schaltfläche klickt, werden in seinem Webbrowser Seiten Ihrer lokalen Website angezeigt. Je nachdem, wie Sie Ihre Seiten gestalten, können die Benutzer dann verschiedene Produktoptionen anzeigen, Konfigurationen festlegen und Liefermethoden auswählen. Sobald die Artikelauswahl abgeschlossen ist, klickt der Benutzer wieder auf eine Schaltfläche, woraufhin die Bestellinformationen an die Beschaffungsanwendung übertragen werden. Die vollständig konfigurierten Produkte und die dazugehörigen Preise werden dann innerhalb der Kaufanforderung des Benutzers angezeigt.



Interaktive Punchoutsitzung zwischen einem Benutzer und einer Lieferantenwebsite

Auf Ihrer Website können zuvor vereinbarte Vertragsprodukte und -preise angeboten werden.

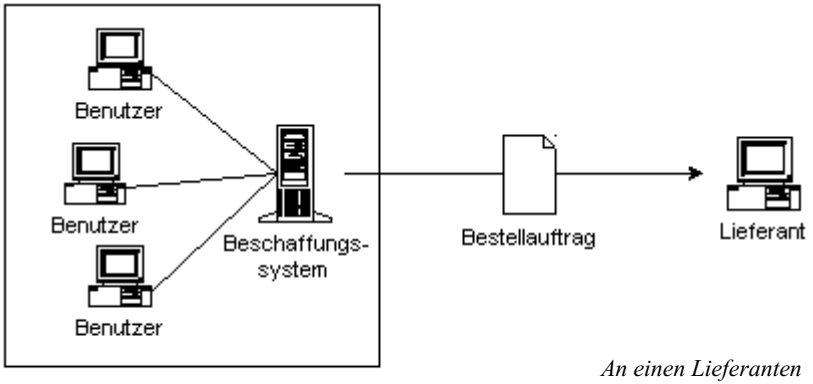
Bestellaufträge

Käuferunternehmen senden an Lieferanten einen Bestellauftrag, in dem eine bestimmte Leistung vertraglich angefordert wird.

Weitere Informationen:

Kapitel 3, „Empfangen von cXML-Bestellaufträgen“

Die Bestellaufträge können über eine E-Commerce-Netzwerkplattform wie Ariba Network geleitet werden.



An einen Lieferanten übermittelter Bestellauftrag

cXML ist nur eines von mehreren Formaten für Bestellaufträge. Weitere mögliche Formate sind E-Mail, Fax und EDI (X.12 Electronic Data Interchange; Elektronischer Datenaustausch). Da cXML im Vergleich zu den anderen Formaten jedoch äußerst flexibel und kostengünstig zu implementieren ist, die umfassendste Bandbreite von Daten und Anlagen unterstützt und darüber hinaus von automatisierten Systemen genutzt werden kann, stellt es zweifellos das ideale Format für die Verwaltung von Bestellaufträgen dar.

Anwendungen, die cXML verwenden

cXML kann von jeder beliebigen E-Commerce-Anwendung eingesetzt werden. Es wird derzeit von Käuferunternehmen, vertikal und horizontal strukturierten Käufergemeinschaften, Lieferanten und Anbietern von Anwendungen genutzt.

In den nachfolgenden Abschnitten werden die wichtigsten Anwendungen beschrieben, die cXML verwenden.

Beschaffungsanwendungen

In Beschaffungsanwendungen wie Ariba ORMS (Operating Resource Management System) und Ariba IBX (Internet Business eXchange) kommt cXML für externe Transaktionen zum Einsatz.

Ariba ORMS ist eine Unternehmensanwendung, die den Mitarbeitern großer Unternehmen über ein Intranet zur Verfügung gestellt wird.

Ariba IBX ist ein internetbasierter Dienst, der die Schaffung von Käufergemeinschaften aus kleinen und mittleren Unternehmen ermöglicht.

Mit den genannten Anwendungen können Benutzergemeinschaften auf vertraglicher Grundlage Produkte und Leistungen von Anbietern erwerben, die von den zuständigen Einkaufsdirektoren genehmigt wurden. Nachdem Bestellaufträge von den Leitern in den Gemeinschaften genehmigt wurden, können sie den Lieferanten über verschiedene Kanäle zugestellt werden, beispielsweise per cXML über das Internet.

E-Commerce-Netzwerkplattformen

E-Commerce-Netzwerkplattformen wie Ariba Network sind webbasierte Dienste, die eine Verbindung zwischen Käufern und Lieferanten ermöglichen.

Über solche Webdienste stehen Funktionen wie Katalogprüfung und Dateiverwaltung, Veröffentlichung und Abonnements von Katalogen, automatische Weiterleitung von Bestellaufträgen und Bestellauftragshistorie zur Verfügung.

Die Kommunikation zwischen diesen Webdiensten, Käufer- und Lieferantenanwendungen kann vollständig mit cXML über das Internet abgewickelt werden.

Punchoutkataloge

Wie bereits beschrieben, handelt es sich bei Punchouts um interaktive Kataloge auf den Websites von Lieferanten. Punchoutkataloge sind Webserveranwendungen, die in einer für Punchoutsitzungen geeigneten Programmiersprache wie ASP (Active Server Pages), JavaScript oder CGI geschrieben sind.

Weitere Informationen:

Kapitel 2, „Implementieren einer Punchoutsite“

Punchoutkataloge akzeptieren Punchoutanforderungen von Beschaffungsanwendungen, erkennen das Käuferunternehmen und zeigen automatisch die richtigen Produkte und Preise im HTML-Format an. Die Benutzer können dann die gewünschten Artikel auswählen, selbst Konfigurationen zusammenstellen oder gegebenenfalls diverse Optionen festlegen.

Am Ende der Punchoutsitzung sendet die Punchoutsite eine Beschreibung der Benutzerauswahl im cXML-Format an die Beschaffungsanwendungen.

Auftragseingangssysteme

Auftragseingangssysteme sind Anwendungen beim Lieferanten, die von Käuferunternehmen gesendete Bestellaufträge annehmen und verarbeiten. Als Auftragseingangssystem können beliebige automatische Systeme verwendet werden, beispielsweise Bestandsverwaltungssysteme, Auftragserfüllungssysteme oder Auftragsverarbeitungssysteme.

Weitere Informationen:

Kapitel 3, „Empfangen von cXML-Bestellaufträgen“

Da sich die Informationen aus cXML-Bestellaufträgen problemlos extrahieren lassen, ist es auch relativ einfach möglich, für ein vorhandenes Auftragseingangssystem einen cXML-Adapter zu entwickeln.

Angebotsstrategie

Über Beschaffungsanwendungen werden den Benutzern Produkte und Dienstleistungen angeboten. Lieferanten sind daran interessiert, die Präsentation ihrer Produkte und Dienstleistungen für den Kunden möglichst genau zu steuern, da die richtige Präsentation einen wesentlichen Faktor im Verkaufsprozess darstellt. Käuferunternehmen wiederum möchten, dass der Angebotsinhalt leicht zugänglich und gut zu durchsuchen ist, damit sich ein hohes Maß an Vertragstreue gewährleisten lässt.

Käuferunternehmen und Lieferanten können aus mehreren Methoden des Angebots von Produkt- und Leistungsinhalten auswählen. Die jeweils zu verwendende Methode richtet sich nach der Vereinbarung zwischen Käuferunternehmen und Lieferanten sowie nach dem Charakter der gehandelten Produkte oder Leistungen.

In der nachstehenden Tabelle sind Beispielkategorien typischer Produkt- und Leistungsangebote und deren bevorzugte Präsentationsart aufgeführt.

Waren	Eigenschaften	Präsentationsart
Büroartikel, interner Bedarf	Statischer Inhalt, stabile Preise	Statische Kataloge
Laborbedarf, Wartung, Reparatur und Betrieb, Elektronische Bauteile	Normierung unbedingt erforderlich	Punchout an ein vertikal strukturiertes Warenportal
Bücher, Chemikalien	Große Anzahl verschiedener Artikel	Punchout an eine Lieferantensite
Computer, Netzwerkartikel, Peripheriegeräte	Zahlreiche Konfigurationsmöglichkeiten	Punchout an ein Konfigurationstool des Lieferanten
Dienstleistungen, Druckmaterialien	Inhalt mit stark variierenden Attributen	Punchout an ein elektronisches Formular beim Lieferanten

Käuferunternehmen können die Inhalte lokal im Unternehmen speichern oder über Punchouts aus dem Internet abrufen. Beide Verfahren werden durch cXML-Kataloge unterstützt.

Wie aus der oben stehenden Tabelle hervorgeht, steht durch Punchoutsysteme ein flexibler Grundrahmen bereit, über den Lieferanten waren- und kundenspezifische Inhalte anbieten können. Das Ziel dieser Präsentationsstrategie besteht darin, Käufern und Lieferanten gleichermaßen ein optimales Verfahren zum Austausch von Katalogdaten zu ermöglichen.

Prüfung anhand von DTDs

Da es sich bei cXML um ein XML-Derivat handelt, werden die Elemente durch einen Satz von Dokumenttyp-Definitionen (DTDs) umfassend definiert. DTDs sind Textdateien, die die genaue Syntax und Reihenfolge von cXML-Elementen beschreiben. Mit Hilfe von DTDs können Anwendungen die von ihnen geschriebenen oder gelesenen cXML-Daten überprüfen.

Das Überprüfen von cXML-Dokumenten durch die cXML-Anwendungen ist zwar nicht zwingend erforderlich, wird jedoch empfohlen.

Abrufen von cXML-DTDs

DTDs für alle Versionen von cXML können aus dem Internet unter cxml.org bezogen werden:

http://xml.cXML.org/schemas/cXML/<*Version*>/cXML.dtd

wobei <*Version*> für die vollständige cXML-Versionsnummer steht, z. B. 1.1.007.

Durchführen einer Prüfung

Mit diesen DTDs können Ihre Anwendungen alle empfangenen und gesendeten cXML-Dokumente überprüfen. XML-Prüfungsanwendungen stehen im Web zur Verfügung. Microsoft Internet Explorer 5 verfügt bereits über eine integrierte XML-Prüffunktion.

Den stabilsten Transaktionsverkehr erreichen Sie durch konsequente Überprüfung aller empfangenen cXML-Dokumente. Werden Fehler erkannt, geben Sie den entsprechenden Fehlercode aus, damit der Absender die Übertragung wiederholen kann.

Aus Leistungsgründen ist es sinnvoll, wenn die cXML-Clients nicht bei jedem Parsen von Dokumenten DTDs abrufen müssen. Stattdessen sollten sie die cXML-Version aus dem Dokumentenheader auslesen und nur die DTDs abrufen, die noch nicht lokal gespeichert sind.

Profiltransaktion

Bei der Profiltransaktion werden Basisinformationen zu cXML-Servern ausgetauscht. Dies ist die einzige Transaktion, die von allen cXML-Servern unterstützt werden muss.

Diese Transaktion, die sich aus den beiden Dokumenten ProfileRequest und ProfileResponse zusammensetzt, ruft verschiedene Servermerkmale ab, einschließlich der unterstützten cXML-Version, der unterstützten Transaktionen und der dazugehörigen Optionen.

> **Hinweis:** Alle Server ab cXML 1.1 **müssen** die Profiltransaktion unterstützen.

Ähnlich wie mit dem „Ping“-Befehl können Clients mit Hilfe der Profiltransaktion prüfen, ob der jeweilige Server verfügbar ist.

ProfileRequest

Das Dokument ProfileRequest hat keinen Inhalt. Es wird einfach zum angegebenen cXML-Server geleitet.

ProfileResponse

Der Server antwortet mit dem Dokument ProfileResponse, das die von ihm unterstützten Transaktionen und deren Positionen aufführt. Eventuelle vorhandene Optionen werden durch entsprechende Zeichenfolgenwerte angegeben.

XML-Dienstprogramme

Im Internet stehen verschiedene kostenlose und kommerzielle Dienstprogramme zum Bearbeiten und Prüfen von XML-Dateien zur Verfügung. Eine Auswahl dieser Dienstprogramme finden Sie in der folgenden Liste:

- **Internet Explorer 5** von Microsoft. XML-fähiger Webbrowser, der XML-Dateien anhand von DTDs überprüfen kann.

  www.microsoft.com/windows/ie/default.htm

- **XML Notepad** von Microsoft. Einfacher XML-Editor.

  msdn.microsoft.com/xml/notepad/intro.asp

- **XML Authority** von Extensibility. Javabasierter XML-DTD-Editor mit hierarchisch aufgebauter grafischer Oberfläche.

  www.extensibility.com

- **XML Spy** von Icon Information Systems. Tool zur Pflege von DTDs und XML-Dateien mit Raster-, Quell- und Browseransicht.

  www.icon-is.com

- **XMetaL** von Softquad Software. Benutzerspezifisch anpassbares XML-Authoringtool.

  www.softquad.com

- **CLIP** von Techno2000 USA. Bedienerfreundliches XML-Authoringtool mit Bearbeitungsanleitung.

  www.t2000-usa.com

- **XMLwriter** von Wattle Software. Grafisches XML-Authoringtool zum Verwalten von XML-Projekten.

  www.xmlwriter.net

Auf folgenden Websites finden Sie weitere XML-Tools:

www.xmlsoftware.com
www.xml.com/pub/pt/Editors

Kapitel 2
Implementieren einer Punchoutsite

Mithilfe von Punchoutfunktionen können Benutzer von Beschaffungsanwendungen auf Lieferantenverträge für Produkte oder Leistungen zugreifen, die sich auf der Website des jeweiligen Lieferanten befinden. Anstelle ganzer Kataloge brauchen Sie daher an die interessierten Käuferorganisationen nur noch kurze Dateien mit einer Beschreibung Ihrer Angebotstruktur, Produktkategorien oder Produkte zu senden.

In diesem Kapitel erfahren Sie, wie Sie eine Website punchoutfähig einrichten können. Folgende Themen werden behandelt:

- Punchoutanforderungen
- Punchoutereignisfolge
- Punchoutdokumente
- Änderungen an Ihren Webseiten
- Anregungen für Ihre Punchoutwebsite

Punchoutanforderungen

Bevor Käuferorganisationen ihre Beschaffungsanwendungen für Punchout konfigurieren oder Lieferanten punchoutfähige Websites implementieren, müssen beide Seiten über die Anforderungen und Vorteile von Punchoutsites nachdenken.

Käuferorganisationen

cXML-kompatible Beschaffungsanwendungen lassen sich bei punchoutfähigen Lieferanten in weniger als einem Tag installieren und testen.

Da die Grenzen für die technische Integration bei Käuferorganisationen in der Regel relativ gering sind, sollte die Entscheidung über die Einführung einer punchout-fähigen Site vorrangig im Hinblick auf die Geschäftsabwicklungspraxis und die erworbenen Waren getroffen werden. (Eine Liste der für Punchout geeigneten Waren finden Sie im Abschnitt „Angebotsstrategie“ auf Seite 6.)

Geschäftliche Aspekte

Käuferorganisationen sollten folgende Fragen in Betracht ziehen:

- Verfügen alle Mitarbeiter, die Bestellanforderungen erstellen und genehmigen, über einen Zugang zum Internet? Wenn nicht, würde ihnen ein kontrollierter Zugang zum Internet erlaubt werden?
- Wünscht die Käuferorganisation, dass ihre Lieferanten Katalogangebote (einschließlich der Preise) erstellen und pflegen?
- Kaufen die Anforderer zur Zeit schon Waren im Internet? Wenn das der Fall ist, erfordern diese Waren auf Lieferantenseite ein Konfigurationswerkzeug oder enthalten sie besondere Attribute, die sich nicht mit einem statischen Inhaltsmodell vereinbaren lassen?
- Verwendet die Käuferorganisation Katalogprogramme zum Zusammenstellen von Angeboten (z. B. Aspect, TPN-Register oder Harbinger)?
- Bezieht die Käuferorganisation derzeit Leistungen (z. B. Berater, Zeitarbeit oder Wartung) über das Internet?
- Nutzt die Käuferorganisation derzeit Onlinequellen?

Wenn Sie eine der gestellten Fragen mit Ja beantworten können, empfiehlt sich für Ihre Käuferorganisation die Nutzung der Punchoutfunktionen.

Technische Aspekte

Folgende technische Voraussetzungen müssen durch die Käuferorganisationen erfüllt werden:

- Direkter Internetzugang: Die Benutzer in der Käuferorganisation müssen über einen direkten Zugang zum Internet verfügen. Punchout stützt sich auf regelmäßige Webbrowsersitzungen, in denen der Benutzer mit den aktiven Websites des Lieferanten kommuniziert. Diese Kommunikation wird über eine internetbasierte Infrastruktur und nicht über die Beschaffungsanwendung realisiert.
- Zuverlässige Internetanbindung: Der Internetzugang muss permanent verfügbar und absolut zuverlässig sein. Wenn Benutzer infolge von Internetausfällen keine Produkte erwerben können, kaufen sie wahrscheinlich anderswo.
- Verträge mit Punchoutlieferanten: Die Einkäufer müssen vertraglich mit punchoutfähigen Lieferanten verbunden sein. Punchoutfähige Websites gewähren nur entsprechend ausgewiesenen und authentifizierten Käuferunternehmen Zugang.

Lieferanten

Im Zusammenhang mit Punchoutfunktionen umfasst der Begriff *Lieferant* weit mehr als in seiner herkömmlichen Definition. Das Punchoutprotokoll ist als flexibler Rahmen konzipiert, über den Daten zu praktisch jedem Produkt und jeder Dienstleistung von beliebigen Lieferanten, Verteilern, Zusammenstellern oder Herstellern übermittelt werden können.

Beispiele für Produkte und Dienstleistungen:

- Computer direkt vom Hersteller oder Händler
- Chemikalien und Reagenzien von einem Sammellieferanten
- Büroartikel von einem Verteiler
- Vertragsleistungen von einer Zeitarbeitsagentur

Möglicherweise verfügt Ihr Unternehmen bereits über eine transaktive Website, die Angebote anzeigen und Bestellaufträge entgegennehmen kann. Ist diese Möglichkeit gegeben, müssen Sie die Art Ihrer Geschäftsabwicklung und technischen Ressourcen in Betracht ziehen, um über die Nutzung der Punchoutfunktionen zu entscheiden.

Geschäftliche Aspekte

Lieferanten sollten folgende Fragen in Betracht ziehen:

- Verkaufen Sie Ihre Waren bzw. Dienstleistungen bereits jetzt über das Internet? Wenn ja, bieten Sie kundenspezifische Inhalte (vertragliche Preisfestsetzung) über Ihre Website an?

- Fallen Ihre Produkte bzw. Dienstleistungen unter eine der im Abschnitt „Angebotsstrategie“ auf Seite 6 dargestellten Punchoutkategorien? Zu diesen Kategorien gehören:

  Stark konfigurierbare Produkte (wie Computer)
  Große Zahl von Einzelpositionen (z. B. Bücher)
  Einzigartige Produktattribute (z. B. Chemikalien)
  Normierte Daten (z. B. Ersatzteile und Verbrauchsmittel)

- Möchten Sie Bestellaufträge und/oder Zahlungen lieber über Ihre Website erhalten?

Wenn Sie eine der gestellten Fragen mit Ja beantworten können, empfiehlt sich für Ihre Organisation die Nutzung der Punchoutfunktionen.

Technische Aspekte

Folgende technische Voraussetzungen müssen durch die Lieferanten erfüllt werden:

- Zuverlässige Internetanbindung: Die Infrastruktur Ihres Webservers und Ihre Internetanbindung müssen äußerst zuverlässig sein. Wenn Benutzer nicht auf Angebote zugreifen können, wenden sie sich wahrscheinlich an einen anderen Lieferanten.
- Kompetente Websiteadministratoren: Die Punchoutwebsite und die dazugehörigen Hilfsanwendungen müssen regelmäßig gewartet und angepasst werden. Die Bedürfnisse der Benutzer und Ihre Produktangebote ändern sich, also brauchen Sie auch Mitarbeiter, die Ihre Punchoutinfrastruktur weiterentwickeln.
- Unterstützung für Basistransaktionen: Grundsätzlich ist es nicht erforderlich, dass Punchoutwebsites alle verfügbaren cXML-Funktionen unterstützen. Folgende Transaktionen müssen jedoch in jedem Fall unterstützt werden:

  Profiltransaktion
  PunchOutSetupRequest
  PunchOutSetupResponse
  PunchOutOrderMessage

Aufwandsschätzung

In der nachstehenden Tabelle finden Sie den geschätzten Aufwand für die Punchoutintegration auf cXML-Basis. Die Angaben basieren auf Lieferantenschätzungen:

Vorhandene Infrastruktur	**Voraussichtliche Fertigstellungsdauer**
Transaktive Site mit XML-Infrastruktur	3 Wochen mit eigenem EDV-Personal 3–4 Wochen mit anderen Auftragnehmern
Transaktive Site ohne XML-Infrastruktur	4 Wochen mit eigenem EDV-Personal 4–5 Wochen mit anderen Auftragnehmern

XML-Kenntnisse

Info zu XML

XML (eXtensible Markup Language) ist ein Standard für die Datenübertragung zwischen Internetanwendungen. XML-Dokumente enthalten Daten in Form von Tag-Wert-Paaren. Bei einer grundsätzlich ähnlichen Struktur ermöglichen XML-Dokumente eine wesentlich einfachere Datenextraktion und -nutzung durch Internetanwendungen als einfache HTML-Dokumente (Hypertext Markup Language).

Mit der weiteren Verbreitung von Internetanwendungen wird XML immer mehr an Bedeutung gewinnen.

Der erste Schritt beim Erarbeiten einer Punchoutlösung besteht im Erwerb von Kenntnissen über XML. XML ist eine Sprache zur Beschreibung von Sprachen. cXML-Dokumente beruhen auf den Dokumenttypdefinitionen (DTDs) von XML. Diese DTDs dienen zum Definieren von Vorlagen für Inhaltsmodelle in einem cXML-Dokument (z. B. die gültige Reihenfolge und Verschachtelung von Elementen) und zum Festlegen der Datentypen für Attribute.

Zum Implementieren einer Punchoutwebsite müssen Sie genau wissen, wie Sie XML-Daten erstellen, parsen, abfragen, senden und von einer entfernten Quelle empfangen können.

Als grundlegende Werkzeuge zum Verarbeiten von XML-Dokumenten benötigen Sie XML-Parser. Diese Parser können von Microsoft und weiteren Firmen kostenlos bezogen werden. (Beispielsweise ist in Microsoft Internet Explorer 5 standardmäßig ein XML-Parser integriert.) Eine Liste von XML-Tools finden Sie im Abschnitt „XML-Dienstprogramme“ auf Seite 9.

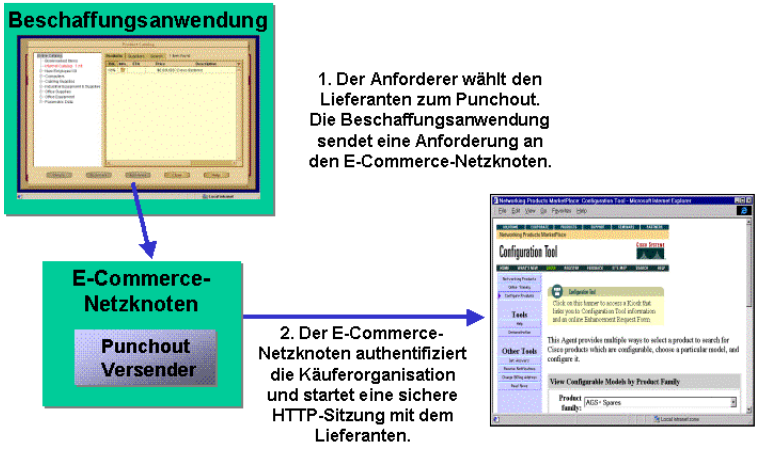
Punchoutereignisfolge

Jede Punchoutsitzung besteht aus mehreren klar abgrenzbaren Schritten.

Schritte 1 und 2: Punchoutanforderung

Die Benutzer melden sich bei einer Beschaffungsanwendung an und erstellen neue Bestellanforderungen. Sie ermitteln die gewünschten Artikel, indem Sie die lokalen Kataloge nach Ware, Lieferant oder Produktbeschreibung durchsuchen. Wenn sie einen Punchoutartikel wählen, öffnet die Beschaffungsanwendung ein neues Browserfenster und meldet die Benutzer bei deren Konten auf Ihrer Website an.

Die folgende Abbildung veranschaulicht die Schritte zur Punchoutanforderung:



Funktionsweise: Wenn ein Benutzer auf einen Punchoutartikel klickt, sendet die Beschaffungsanwendung das cXML-Dokument PunchOutSetupRequest an einen E-Commerce-Netzknoten. Der Netzknoten fungiert als vertrauenswürdige dritte Partei, die die Anforderung entgegennimmt, die Käuferorganisation prüft und die Anforderung dann an Ihre Punchoutwebsite weiterleitet.

> **Hinweis:** Alle über das Internet versandten cXML-Dokumente können über sichere HTTPS-Verbindungen geleitet und mit SSL (Secure Socket Layer) 3.0 verschlüsselt werden.

Der Zweck dieser Anforderung ist die Information Ihrer Website über die Identität des Käufers und eine Mitteilung über den auszuführenden Vorgang. Zu den unterstützten Vorgängen gehören:

- **Create:** Startet eine neue Punchoutsitzung.
- **Edit:** Öffnet eine vorhandene Punchoutsitzung zur Bearbeitung.
- **Inspect:** Öffnet eine vorhandene Punchoutsitzung zur Überprüfung (die Daten können nicht geändert werden).

Wenn Ihre Website eine Anforderung erhält, gibt sie eine PunchOutSetupResponse-Meldung mit einem URL zurück. Diesem URL kann die Beschaffungsanwendung entnehmen, wo sie eine Sitzung zum Durchsuchen Ihrer Website starten soll.

Die Beschaffungsanwendung öffnet ein neues Browserfenster, in dem eine bei einem Konto Ihrer Website angemeldete Sitzung angezeigt wird. Dieses Konto kann zu einer bestimmten Region, einem Unternehmen, einer Abteilung oder einem Benutzer gehören.

Schritt 3: Produktauswahl

Die Benutzer wählen Artikel aus Ihrem Bestand aus, wobei sie die auf Ihrer Website angebotenen Funktionen und Leistungen nutzen.

3. Der Anforderer sucht und konfiguriert die gewünschten Produkte auf der Lieferantensite.

Je nach Produkt bzw. Dienstleistung können dies beispielsweise folgende Funktionen sein:

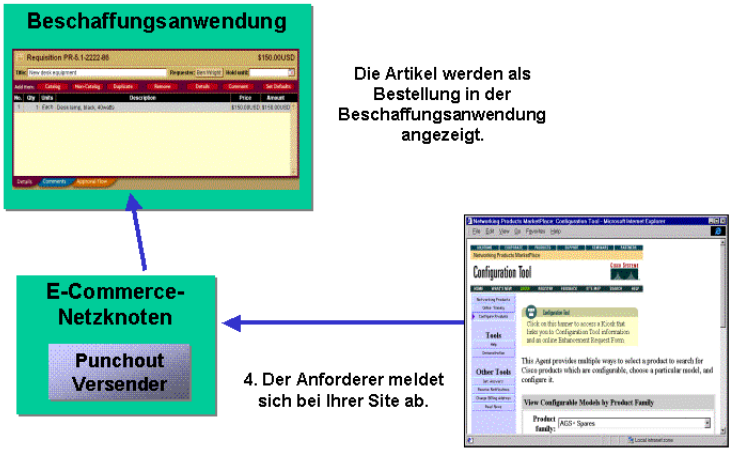
- Konfigurationswerkzeuge zum Zusammenstellen benutzerspezifischer Produkte (z. B. Computer, organische Verbindungen oder personalisierte Produkte)
- Suchmaschinen zum Auffinden der gewünschten Artikel aus umfangreichen Katalogen
- Ansichten normierter Daten zum Vergleich von Produkten anhand von Preisen, Funktionen oder deren Verfügbarkeit (z. B. Ersatzteile und Verbrauchsmittel)
- Ansichten von Attributen, die eine bestimmte Ware einzigartig machen (z. B. Druckmaterialien, Chemikalien und Reagenzien oder Dienstleistungen)
- Preisfestsetzung, Bestands- und Verfügbarkeitsprüfung in Echtzeit
- Automatische Steuer- und Frachtkostenberechnung anhand der Lieferadresse, Größe oder Menge der Artikel (muss nicht während der Punchoutsitzung errechnet werden)

Funktionsweise: Wenn die Beschaffungsanwendung die Benutzer zu Ihrer Website geführt hat, ist deren Einkaufserlebnis dasselbe wie bei direkter Anmeldung auf Ihrer Website. Somit muss keine der zuvor aufgeführten Funktionen und Leistungen geändert werden.

Schritt 4: Kasse

Ihre Website berechnet die Gesamtkosten für die Auswahl des Benutzers einschließlich Steuern, Frachtkosten und kundenspezifischer Rabatte. Die Benutzer klicken dann auf die Schaltfläche **Kasse** Ihrer Website und senden den Inhalt ihres Einkaufskorbes an die Bestellanforderungen in ihrer Beschaffungsanwendung.

Diese Schritte werden durch die folgende Abbildung veranschaulicht:



Funktionsweise: Wenn Benutzer auf die Schaltfläche **Kasse** auf Ihrer Website klicken, sendet die Website eine PunchOutOrderMessage-Meldung (im cXML-Format) mit Produktdetails und Preisen an die Beschaffungsanwendung. Zusätzlich können verborgene Lieferantencookies gesendet werden, über die sich später Artikel zu bestimmten Einkaufssitzungen zuordnen lassen.

Genau genommen haben Sie damit ein Preisangebot für die angefragten Artikel bereitgestellt. Da Sie noch keinen Bestellauftrag erhalten haben, können Sie den Auftrag auch noch nicht buchen.

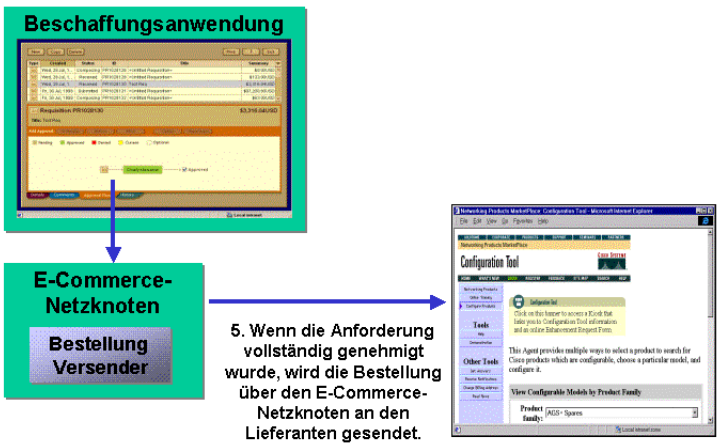
Wenn Benutzer die Artikel in einer Bestellanforderung nachträglich bearbeiten müssen, können Sie ihnen einen „Rückpunchout“ zu Ihrer Website gestatten. Die Beschaffungsanwendung sendet den Inhalt des ursprünglichen Einkaufswagens an Ihre Website zurück, und der Benutzer nimmt seine Änderungen dort vor. Beim Abmelden sendet Ihre Website die Artikel an die Beschaffungsanwendung zurück.

Ihre Website fungiert als Informationsquelle für sämtliche Punchoutartikel. Mengenänderungen oder das Hinzufügen neuer Artikel zur Bestellanforderung können sich auch auf Steuern und Frachtgebühren auswirken, was eine Neuberechnung auf Ihrer Website erfordert. Daher müssen Änderungen an den ursprünglichen Artikeln auf Ihrer Website und nicht in der Beschaffungsanwendung erfolgen. Dies wiederum macht einen „Rückpunchout“ erforderlich, bei dem es sich genau genommen um eine PunchOutSetupRequest-Anforderung mit dem Vorgang „Edit“ handelt.

Schritt 5: Übermittlung der Bestellanforderung

Nachdem der Inhalt des Einkaufskorbs von Ihrer Website zur Bestellanforderung des Benutzers weitergeleitet wurde, werden die Genehmigungsprozesse der Beschaffungsanwendung in Gang gesetzt. Wenn die Bestellanforderung genehmigt wird, wandelt die Beschaffungsanwendung sie in einen Bestellauftrag um und sendet sie zur Erledigung an die Website zurück. Sofern bei der Bestellung keine Kundenkartendaten übermittelt werden, lässt sich der Auftrag auch bei Lieferung berechnen.

In der folgenden Abbildung wird die Übermittlung von Bestellanforderungen veranschaulicht:



Funktionsweise: Die Beschaffungsanwendung sendet sämtliche Bestellaufträge im cXML-Format an den E-Commerce-Netzknoten. Der Knoten leitet die Bestellaufträge dann unter Nutzung des angegebenen Weiterleitungsverfahrens an Sie weiter. Im Zuge der Eingangsbestätigung eines Bestellauftrags wird der Auftrag erfolgreich gebucht.

Punchoutfähige Lieferanten sollten aus folgenden Gründen die Auftragsweiterleitung per cXML bevorzugen:

- cXML-Bestellaufträge gestatten die Rücksendung von Lieferantencookies an Sie. Da Lieferantencookies den Datentyp „beliebig“ aufweisen, ist eine Zuordnung zu anderen Weiterleitungsverfahren, wie Fax, E-Mail oder EDI, nur schwer möglich.
- Punchoutfähige Lieferanten sind ohnehin auf cXML eingestellt, sodass die Einrichtung des Empfangs von cXML-Bestellaufträgen nur geringen zusätzlichen Aufwand bedeutet.

Bestellaufträge werden ausführlich in Kapitel 3, „Empfangen von cXML-Bestellaufträgen“ behandelt.

Punchoutdokumente

Es gibt vier Arten von cXML-Dokumenten:

- Punchoutindexkatalog
- PunchOutSetupRequest
- PunchOutSetupResponse
- PunchOutOrderMessage

Punchoutindexkatalog

Punchoutindexkataloge sind Dateien, die Punchoutartikel auflisten und auf Ihre Punchoutwebsite verweisen.

Nachstehend finden Sie ein Beispiel für einen Punchoutindexkatalog:

Art des cXML-Dokuments und URL der DTD

Ihr Bezeichner für den Punchoutartikel

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE Index SYSTEM "http://xml.cxml.org/schemas/cXML/1.1.007/cXML.dtd">
<Index>
    <SupplierID domain="DUNS">83528721</SupplierID>
    <IndexItem>
        <IndexItemPunchout>
            <ItemID>
                <SupplierPartID>5555</SupplierPartID>
            </ItemID>
             <PunchoutDetail>
                <Description xml:lang="en-US">Desk Chairs</Description>
                <Description xml:lang="fr-FR">Chaises de Bureau</Description>
```

URL Ihrer Punchoutwebsite

```
                <URL>http://www.workchairs.com/punchout.asp</URL>
                <Classification domain="UNSPSC">5136030000</Classification>
            </PunchoutDetail>
        </IndexItemPunchout>
    </IndexItem>
</Index>
```


Durch SupplierID wird die Lieferantenorganisation gekennzeichnet. Obwohl Sie grundsätzlich jede beliebige Kennzeichnungsdomäne verwenden können, werden DUNS (Dun & Bradstreet Universal Naming System) und NetworkID empfohlen. Weitere Informationen zu DUNS-Nummern finden Sie unter www.dnb.com.

Unter Description ist der Text angegeben, den die Beschaffungsanwendung in Produktkatalogen anzeigt. Sie können Beschreibungen in mehreren Sprachen bereitstellen, wobei die Beschaffungsanwendung jeweils nur die für die Gebietsangaben des Benutzers zutreffende Sprache anzeigt.

Classification gibt die Warengruppe der Position für den Käufer an. Alle Produkte und Dienstleistungen müssen im UNSPSC-Schema standardisiert und zugeordnet sein. Bei Punchoutindexkatalogen gibt diese Klassifizierung die Position des Punchoutartikels im Katalog an. Eine Liste der UNSPSC-Codes finden Sie unter www.unspsc.com.

Erstellen und Veröffentlichen von Indexkatalogen

Erstellen Sie diese Kataloge, und veröffentlichen Sie sie auf einem E-Commerce-Netzknoten für Ihre Kunden. Der Katalogmanager in der Käuferorganisation lädt diese dann herunter und speichert sie zur Verwendung in Beschaffungsanwendungen.

Die Benutzern sehen den Inhalt Ihrer Punchoutindexkataloge neben herkömmlichen, statischen Katalogen.

Auflösungstiefe von Punchoutartikeln

Sie können Kataloge auf Lagerebene, Gangebene oder Produktebene erstellen.

- In Katalogen auf Lagerebene werden alle Ihre Produkte und Dienstleistungen aufgelistet. Die Benutzer müssen nach dem gewünschten Artikel suchen.
- In Katalogen auf Gangebene werden jeweils verwandte Produkte und Dienstleistungen aufgeführt.
- Kataloge auf Produktebene dagegen enthalten jeweils nur ein Produkt bzw. eine Dienstleistung. Daher brauchen die Benutzer keine weiteren Suchläufe auszuführen.

Bei den Überlegungen dazu, wie umfassend Sie die Punchoutartikel fassen möchten, sollten Ihr Geschäftsmodell, die Gestaltung Ihres Produkt- und Dienstleistungsangebots und die Struktur Ihrer Punchoutwebsite eine wichtige Rolle spielen.

Je mehr Such- und Konfigurationswerkzeuge Sie auf Ihrer Website anbieten, desto breiter können Sie die Punchoutartikel in Ihren Indexkatalogen anlegen.

PunchOutSetupRequest

Zum Starten einer Punchoutsitzung wählt der Benutzer Ihren Punchoutartikel aus. Die Beschaffungsanwendung erzeugt ein PunchOutSetupRequest-Dokument und sendet es an den E-Commerce-Netzknoten, der es wiederum an Ihre Punchoutwebsite weiterleitet.

Nachstehend finden Sie ein Beispiel für ein PunchOutSetupRequest-Dokument:

```
<?xml version="1.0"?>
<!DOCTYPE cXML SYSTEM "http://xml.cxml.org/schemas/cXML/1.1.007/cXML.dtd">
<cXML version="1.1.007" xml:lang="en-US"
payloadID="933694607118.1869318421@jlee" timestamp="2000-08-15T08:36:47-
07:00">
    <Header>
        <From>                                          <-- Erzeuger (Käuferorganisation)
            <Credential domain="DUNS">
                <Identity>65652314</Identity>
            </Credential>
        </From>
        <To>                                            <-- Bestimmungsort (Lieferant)
            <Credential domain="DUNS">
                <Identity>83528721</Identity>
            </Credential>
        </To>
        <Sender>                                        <-- Vorhergehende Übertragungseinheit (im vorliegenden Fall Ariba Network)
            <Credential domain="AribaNetworkUserId">
                <Identity>sysadmin@ariba.com</Identity>
                <SharedSecret>abracadabra</SharedSecret>
            </Credential>
            <UserAgent>Ariba ORMS 6.1</UserAgent>
        </Sender>
    </Header>
    <Request>
        <PunchOutSetupRequest operation="create">   <-- Art der Anforderung
            <BuyerCookie>1CX3L4843PPZO</BuyerCookie>
            <Extrinsic name="CostCenter">610</Extrinsic>
            <Extrinsic name="User">john_smith</Extrinsic>
            <BrowserFormPost>                       <-- Bestimmungsort für endgültige PunchOutOrderMessage
                <URL>https://aribaorms:26000/punchout.asp</URL>
            </BrowserFormPost>
```

```
                    <SupplierSetup>
                        <URL>http://www.workchairs.com/punchout.asp</URL>
                    </SupplierSetup>
Vom Benutzer        <SelectedItem>
ausgewählter Artikel    <ItemOut quantity="1">
                            <ItemID>
                                <SupplierPartID>5555</SupplierPartID>
                            </ItemID>
                        </ItemOut>
                    </SelectedItem>
                </PunchOutSetupRequest>
            </Request>
        </cXML>
```

Die Attribute payloadID und timestamp am Anfang werden von den cXML-Clients zum Verfolgen von Dokumenten und Erkennen doppelter Dokumente genutzt.

Die Elemente From, To und Sender ermöglichen den empfangenden Systemen, die jeweiligen Parteien zu erkennen und zu autorisieren. Die Elemente From und To sind innerhalb eines Dokuments eindeutig. Während das Dokument jedoch zu seinem Bestimmungsort unterwegs ist, wird das Element Sender durch Zwischenknoten (wie Ariba Network) verändert.

Die Vorgänge „Create“, „Edit“ und „Inspect“

Das Attribut Operation gibt die Art der vom Käufer gestarteten Sitzung an. Es kann den Wert create, edit oder inspect haben.

- In Create-Sitzungen werden neue Einkaufswagen erstellt, die neuen Bestellanforderungen entsprechen.
- Edit-Sitzungen öffnen bereits erstellte Einkaufswagen erneut, damit Änderungen vorgenommen werden können. Die Beschaffungsanwendung sendet Positionsdaten als Teil des PunchOutSetupRequest-Dokuments. Die Punchoutwebsite kann mit diesen Daten den in der ursprünglichen Sitzung erstellten Einkaufswagen erneut instanziieren.
- In Inspect-Sitzungen werden zuvor erstellte Einkaufswagen nur zur Ansicht neu geöffnet. Wie beim Vorgang Edit sendet die Beschaffungsanwendung Positionsdaten im Rahmen des PunchOutSetupRequest-Dokuments. Allerdings sind in diesem Fall nach der erneuten Instanziierung des Einkaufswagens keine Inhaltsänderungen mehr möglich.

Im folgenden Beispiel ist eine Edit-Anforderung dargestellt:

```
<?xml version="1.0"?>
<!DOCTYPE cXML SYSTEM "http://xml.cxml.org/schemas/cXML/1.1.007/cXML.dtd">
<cXML version="1.1.007" xml:lang="en-US" payloadID="933695135608.677295401@jlee"
timestamp="2000-08-15T08:45:35-07:00">
    <Header>
        <From>
            <Credential domain="DUNS">
                <Identity>65652314</Identity>
            </Credential>
        </From>
        <To>
            <Credential domain="DUNS">
                <Identity>83528721</Identity>
            </Credential>
        </To>
        <Sender>
            <Credential domain="AribaNetworkUserId">
                <Identity>sysadmin@ariba.com</Identity>
                <SharedSecret>abracadabra</SharedSecret>
            </Credential>
            <UserAgent>Ariba ORMS 6.1</UserAgent>
        </Sender>
    </Header>
    <Request>
        <PunchOutSetupRequest operation="edit">
            <BuyerCookie>1CX3L4843PPZO</BuyerCookie>
            <Extrinsic name="CostCenter">610</Extrinsic>
            <Extrinsic name="User">john_smith</Extrinsic>
            <BrowserFormPost>
                <URL>https://aribaorms:26000/punchout.asp</URL>
            </BrowserFormPost>
            <SupplierSetup>
                <URL>http://www.workchairs.com/punchout.asp</URL>
            </SupplierSetup>
            <ItemOut quantity="2">
                <ItemID>
                    <SupplierPartID>220-6338</SupplierPartID>
                    <SupplierPartAuxiliaryID>E000028901
                    </SupplierPartAuxiliaryID>
                </ItemID>
            </ItemOut>
        </PunchOutSetupRequest>
    </Request>
</cXML>
```

Wenn der Benutzer die Edit-Sitzung durch Auswahl eines Katalogartikels einleitet, enthält das PunchOutSetupRequest-Dokument wie bei einer Create-Sitzung das Element SelectedItem.

Authentifizierung durch einen E-Commerce-Netzknoten

Alle PunchOutSetupRequest-Dokumente werden zur Authentifizierung und zum Nachschlagen des URL Ihrer Punchoutwebsite durch einen E-Commerce-Netzknoten geleitet. Dabei werden folgende Schritte ausgeführt:

1. Der Netzknoten empfängt das Dokument PunchOutSetupRequest vom Benutzer.
2. Der Netzknoten prüft die Käufer-ID (From und Shared Secret) anhand des E-Commerce-Kontos des Käufers. Darüber hinaus wird auch der angeforderte Lieferant (To) erkannt.
3. Der Netzknoten ruft Ihr gemeinsames Passwort aus Ihrem Konto (Shared Secret) ab und fügt es in das Element Sender ein.
4. Der Netzknoten sucht in Ihrem Konto die URL-Adresse Ihrer Punchoutwebsite und sendet das PunchOutSetupRequest-Dokument an diese Adresse.
5. Ihre Website empfängt das cXML-Dokument und erkennt am gemeinsamen Passwort, dass das Dokument bereits authentifiziert wurde.
6. Anhand der Angaben im Element From erkennt Ihre Website das anfordernde Unternehmen (Beispiel: acme.com).
7. Zur eindeutigen Bestimmung des Benutzers (Beispiel: John Smith, Finanzen, acme.com) können Sie das Element Contact und die extrinsischen Daten im Hauptteil der Anforderung verwenden.

Die Dokumente PunchOutSetupRequest und PunchOutSetupResponse durchlaufen den E-Commerce-Netzknoten zur Authentifizierung. Das Dokument PunchOutOrderMessage (das den Inhalt des Einkaufskorbs an die Beschaffungsanwendung zurücksendet) wird entweder mit dem HTTP-Standardprotokoll oder per HTTPS direkt zwischen Ihrer Website und dem Benutzer ausgetauscht.

SupplierSetup URL und SelectedItem

In älteren Versionen von cXML konnte die URL-Adresse einer Punchoutwebsite ausschließlich über das Element SupplierSetup angegeben werden. Ab cXML 1.1, ist die URL-Adresse Ihrer Punchoutwebsite dem E-Commerce-Netzknoten bereits bekannt.

Ebenfalls ab cXML 1.1 können Beschaffungsanwendungen mit dem Element SelectedItem angeben, ob der Punchout auf Lager-, Gang- oder Produktebene erfolgt.

Obwohl das Element SupplierSetup verworfen wurde, muss Ihre Punchoutwebsite dennoch mit beiden Verfahren umgehen können, bis alle Punchoutwebsites das Element SelectedItem erkennen.

Daten zur Kontaktperson für extrinsische Daten und Benutzererkennung

Das Dokument PunchoutSetupRequest kann im Element Contact detaillierte Benutzerangaben enthalten, mit denen Ihre Website Benutzer erkennt und führt. Beispiele:

- Benutzername und Rolle
- E-Mail-Adresse

Außerdem kann das PunchOutSetupRequest-Dokument zur genaueren Benutzererkennung noch *extrinsische* Daten enthalten. Beispiele:

- Kostenstelle und Unterkonto des Benutzers
- Region
- Supervisor
- Standardwährung

Käuferorganisationen konfigurieren ihre Beschaffungsanwendungen zum Einfügen des Elements Contact und extrinsischer Daten. Fragen Sie Ihre Kunden, mit welchen Angaben Sie rechnen dürfen.

PunchOutSetupResponse

Nach Erhalt des PunchOutSetupRequest-Dokuments sendet Ihre Website als Antwort ein PunchOutSetupResponse-Dokument, das zwei Funktionen erfüllt:

- Es gibt an, ob das PunchOutSetupRequest-Dokument erfolgreich war.
- Es stellt der Beschaffungsanwendung eine indirekte URL-Adresse für Ihre Startseite bereit.

Es enthält ein <URL>-Element, das den Startseiten-URL für die interaktive Browsersitzung an den Internetbrowser des Benutzers sendet. Dieser URL muss ausreichend Statusinformationen zur Anbindung an einen Sitzungskontext Ihrer Website enthalten, zum Beispiel die Identität des Anforderers und den Inhalt des BuyerCookie-Elements.

Beispiel für ein PunchOutSetupResponse-Dokument:

```
<?xml version="1.0"?>
<!DOCTYPE cXML SYSTEM "http://xml.cxml.org/schemas/cXML/1.1.007/cXML.dtd">
<cXML version="1.1.007" xml:lang="en-US" payloadID="933694607739"
timestamp="2000-08-15T08:46:00-07:00">
    <Response>
        <Status code="200" text="success"></Status>
        <PunchOutSetupResponse>
```

```
                <StartPage>
                    <URL>
                    http://xml.workchairs.com/retrieve?reqUrl=20626;Initial=TRUE
                    </URL>
                </StartPage>
            </PunchOutSetupResponse>
        </Response>
</cXML>
```


PunchOutOrderMessage

Wenn der Benutzer auf Ihrer Website Artikel ausgewählt und konfiguriert hat und auf die Schaltfläche **Weiter** klickt, senden Sie ein PunchOutOrderMessage-Dokument an die Beschaffungsanwendung, um den Inhalt des Einkaufskorbs zu übermitteln. Dieses Dokument kann viel mehr Daten als die anderen Elemente enthalten, da es geeignet sein muss, jeden nur denkbaren Inhalt eines Einkaufskorbs wiederzugeben. Es folgt auch nicht strikt dem Anforderung-Antwort-Paradigma. Es folgen genauere Erläuterungen zu seiner Verwendung.

Beispiel für ein PunchOutOrderMessage-Dokument:

```
<?xml version="1.0"?>
<!DOCTYPE cXML SYSTEM "http://xml.cxml.org/schemas/cXML/1.1.007/cXML.dtd">
<cXML version="1.1.007" xml:lang="en-US" payloadID="933695160894"
timestamp="2000-08-15T08:47:00-07:00">
    <Header>
        <From>
            <Credential domain="DUNS">
                <Identity>83528721</Identity>
            </Credential>
        </From>
        <To>
            <Credential domain="DUNS">
                <Identity>65652314</Identity>
            </Credential>
        </To>
        <Sender>
            <Credential domain="workchairs.com">
                <Identity> website 1</Identity>
            </Credential>
            <UserAgent>Workchairs cXML Application</UserAgent>
        </Sender>
    </Header>
    <Message>
        <PunchOutOrderMessage>
            <BuyerCookie>1CX3L4843PPZO</BuyerCookie>
            <PunchOutOrderMessageHeader operationAllowed="edit">
                <Total>
```

```
                    <Money currency="USD">763.20</Money>
                </Total>
            </PunchOutOrderMessageHeader>
            <ItemIn quantity="3">
                <ItemID>
                    <SupplierPartID>5555</SupplierPartID>
                    <SupplierPartAuxiliaryID>E000028901
                    </SupplierPartAuxiliaryID>
                </ItemID>
                <ItemDetail>
                    <UnitPrice>
                        <Money currency="USD">763.20</Money>
                    </UnitPrice>
                    <Description xml:lang="en">
                        <ShortName>Excelsior Desk Chair</ShortName>
                        Leather Reclining Desk Chair with Padded Arms
                    </Description>
                    <UnitOfMeasure>EA</UnitOfMeasure>
                    <Classification domain="UNSPSC">5136030000
                    </Classification>
                </ItemDetail>
            </ItemIn>
        </PunchOutOrderMessage>
    </Message>
</cXML>
```

Über das Element BuyerCookie kann die Beschaffungsanwendung der ursprünglichen PunchOutSetupRequest eine bestimmte PunchOutOrderMessage zuordnen. Daher muss Ihre Website dieses Element bei jedem Auftreten unbedingt zurücksenden. Versuchen Sie nicht, Punchoutsitzungen mithilfe des Elements BuyerCookie zu verfolgen, da sich der Wert dieses Cookies („Create“, „Inspect“ oder „Edit“) in jeder Sitzung ändert.

Das Element SupplierPartAuxiliaryID fungiert als Lieferantencookie. In diesem Feld können Sie zusätzliche Daten senden, z. B. eine Angebotsnummer oder ein weiteres cXML-Dokument. Die Beschaffungsanwendung sendet es in allen nachfolgenden PunchOutSetupRequest-Sitzungen mit dem Wert Edit oder Inspect sowie im resultierenden cXML-Bestellauftrag zurück. Mithilfe des Lieferantencookies kann eine Zuordnung zwischen den Artikeln einer Bestellanforderung und den entsprechenden Artikeln im Einkaufswagen direkt auf Ihrer Website erfolgen.

UnitOfMeasure beschreibt, in welcher Form das Produkt verpackt oder versandt wird. Dieses Element muss den Standard-Maßeinheitencodes UN/CEFACT entsprechen. Eine Liste der UN/CEFACT-Codes finden Sie unter www.unece.org/cefact.

Das Element Classification führt für jeden ausgewählten Artikel den UNSPSC-Warencode (United Nations Standard Product and Service Code). Diese Codes verwenden die Backendsysteme in den Käufer- und Lieferunternehmen für die Buchhaltung und Berichtserstellung. Eine Liste der UNSPSC-Codes finden Sie unter www.unspsc.org.

Änderungen an Ihren Webseiten

Um die drei Punchoutdokumente im cXML-Format senden und empfangen zu können, müssen Sie unter Umständen auf Ihrer Website vier Seiten erstellen oder ändern:

- Aufrufseite
- Startseite
- Absenderseite
- Bestellannahmeseite

Zur Veranschaulichung des Implementierungsvorgangs für diese Seiten sollen einfache ASP-Codebeispiele (Active Server Page) und der XML-Parser von Microsoft Internet Explorer 5 dienen. Die tatsächliche Implementierung dieser Seiten richtet sich nach der konkreten Entwicklungsumgebung des Lieferanten (CGI, JavaScript oder WebObjects).

Aufrufseite

Die Aufrufseite empfängt sämtliche authentifizierten PunchOutSetupRequest-Dokumente vom E-Commerce-Netzknoten. Sie liest den HTTP-Datenstrom vom Netzknoten und prüft die darin eingebettete cXML-Anforderung anhand der cXML-DTD (bei ASP durch Methodenaufrufe an den XML-Parser von Internet Explorer 5).

Nach der Prüfung extrahiert Ihre Aufrufseite die Elemente aus dem Dokument. Dies geschieht mit folgendem Ziel:

1. Erkennen des Benutzers und Festlegen, wohin dieser geleitet werden soll.
2. Erstellen eines PunchOutSetupResponse-Dokuments und dessen Rücksendung an den Absender.

Ihre Aufrufseite muss folgende Daten für die Startseite bereitstellen:

- Identität des Anforderers (Sender)
- Benutzersprache (xml:lang) zum Bereitstellen lokalisierter Inhalte

- Anforderungsart (Create, Edit oder Inspect)
- Eventuelle extrinsische Daten zur weiteren Spezifizierung des Benutzers und seines Standortes

Nachstehend ist ein Beispiel für eine Aufrufseite dargestellt. Dieser Code verwendet zum dynamischen Erstellen des PunchOutSetupResponse-Dokuments keinen XML-Parser sondern eine statische XML-Vorlage, in die Positionsdaten eingetragen sind. **Dieses Codebeispiel dient lediglich zur Veranschaulichung.**

```
script language=JScript RUNAT=Server>
function elementValue(xml, elem)
{
	var begidx;
	var endidx;
	var retStr;

	begidx = xml.indexOf(elem);
	if (begidx > 0) {
		endidx = xml.indexOf('</',begidx);
		if (endidx > 0)
			retStr =  xml.slice(begidx+elem.length,
			endidx);
			return retStr;
		}
		return null;
}

function twoChar( str )
{
	var retStr;
	str = str.toString();
	if ( 1 == str.length ) {
		retStr = "0" + str;
	} else {
	retStr = str;
	}
	return retStr;
	}

function timestamp( dt )
	{
	var str;
	var milli;
	str = dt.getFullYear() + "-" + twoChar( 1 + dt.getMonth() ) + "-";
	str += twoChar( dt.getDate() ) + "T" + twoChar( dt.getHours() ) + ":";
	str += twoChar( dt.getMinutes() ) + ":" + twoChar( dt.getSeconds() ) + ".";
	milli = dt.getMilliseconds();
	milli = milli.toString();
	if ( 3 == milli.length ) {
```

```
            str += milli;
        } else {
            str += "0" + twoChar( milli );
        }
        str += "-08:00";
        return str;
        }

function genProlog( cXMLvers, randStr )
        {
        var dt;
        var str;
        var vers, sysID;
        var nowNum, timeStr;
        if ( 1.1 > parseFloat( cXMLvers )) {
            vers = "1.0";
            sysID = "cXML.dtd";
        } else {
            vers = "1.1.007";
        sysID = "http://xml.cXML.org/schemas/cXML/" + vers + "/cXML.dtd";
        }
        dt = new Date();
        nowNum = dt.getTime();
        timeStr = timestamp( dt );
        str = '<?xml version="1.1.007" encoding="UTF-8"?>\n';
        str += '<!DOCTYPE cXML SYSTEM "' + sysID + '">\n';
        str += '<cXML version="' + vers + '" payloadID="' + nowNum + ".";
        str += randStr + '@' + Request.ServerVariables("LOCAL_ADDR");
        str += '" timestamp="' + timeStr + '">';
        return str;
}
</script>
REM Create data needed in prolog.
%<
Randomize
randStr = Int( 100000001 * Rnd )
prologStr = genProlog( "1.0", randStr )
Response.ContentType = "text/xml"
Response.Charset = "UTF-8"
%>
<%
REM Empfängt die PunchOutSetup-Anforderung vom E-Commerce-Netzknoten.
REM ORMSURL und Käufercookie werden an den Startseiten-URL angehängt,
REM und die Antwort wird an den Anforderer gesendet.
REM punchoutredirect.asp?bc=2133hfefe&url="http://workchairs/com/..&redirect="
        Dim ret
        Dim punch
        Dim statusText
        Dim statusCode
        Dim cookie
```

```
        Dim url
        Dim xmlstr
        Dim fromUser
        Dim toUser
        cookie = ""
        url = ""
        xmlstr = ""
        dir = ""
        path = Request.ServerVariables("PATH_INFO")
        dir = Left(path, InstrRev(path, "/"))
        if IsEmpty(dir) then
              dir = "/"
        end if

REM Dieser Befehl liest die ankommende HTTP-cXML-Anforderung
        xml = Request.BinaryRead(Request.TotalBytes)
        for i = 1 to Request.TotalBytes
              xmlstr = xmlstr + String(1,AscB(MidB(xml, i, 1)))
        Next
        cookie = elementValue(xmlstr, "<BuyerCookie>")
        url = elementValue(xmlstr, "<URL>")
        fromUser = elementValue(xmlstr, "<Identity>")
        newXMLStr = Right(xmlstr, Len(xmlstr) - (InStr(xmlstr,"<Identity>") +
Len("<Identity>")))
        toUser = elementValue(newXMLStr, "<Identity>")
%>
REM cXML-PunchOutSetupReponse wird formatiert
<% if IsEmpty(cookie) then %>
<%= prologStr %>
        <Response>
              <Status code="400" Text="Bad Request">Invalid Document. Unable to extract
              BuyerCookie.</Status>
        </Response>
</cXML>
<% else %>
<%= prologStr %>
        <Response>
              <Status code="200" text="OK"/>
              <PunchOutSetupResponse>
                    <StartPage>
                          <URL>http://<%=
Request.ServerVariables("LOCAL_ADDR")%>/<%= dir%>/punchoutredirect.asp?bc=<%=
cookie%>&amp;url="<%= url%>"&amp;from=<%= fromUser%>&amp;to=<%=
toUser%>&amp;redirect=<%= StartPage%></URL>
                    </StartPage>
              </PunchOutSetupResponse>
        </Response>
</cXML>
<%end if%>
```

Durch Ihre Aufrufseite sollte für die jeweilige Punchoutsitzung ein eindeutiger StartPage-URL zurückgegeben werden, der außerdem nur für eine begrenzte Zeit gültig ist. Wenn Sie diesen URL deaktivieren, habe es unbefugte Benutzer schwerer, auf Ihre Startseite zuzugreifen.

Denken Sie daran, die Funktionen für spätere Edit- und Inspect-Sitzungen zu implementieren. Die Benutzer sind innerhalb ihrer Beschaffungsanwendung nicht in der Lage, Bestelldetails (z. B. die Bestellmenge) für Punchoutartikel zu ändern. Stattdessen müssen Sie einen erneuten Punchout mit einer Edit-Sitzung durchführen. Im Interesse Ihrer Kunden ist es sinnvoll, wenn bei allen nach Auftragseingang stattfindenden Inspect-Sitzungen der Bestellstatus mit angezeigt wird.

Startseite

Über Ihre Startseite wird der Anforderer bei einem Konto auf Ihrer Website angemeldet. Die Benutzer beginnen ihren Einkauf auf der Startseite. Möglicherweise ist eine solche Seite auf Ihrer Website bereits vorhanden. Ändern Sie sie so, dass Benutzername und Passwort aus dem PunchOutSetupRequest-Dokument abgerufen werden.

Gewähren Sie nur autorisierten Benutzern Zugang zu Ihrer Startseite. Wenn Sie die Authentifizierung erst an der Kasse vornehmen, sind Ihre vertraulichen Preisangaben oder Geschäftsbedingungen nicht geschützt.

Wenn Sie mithilfe von HTTP-Cookies die Benutzerpräferenzen erfassen, sollten Sie diese nach Absenden des PunchOutOrderMessage-Dokuments an die Käufer vernichten. Dadurch verhindern Sie, dass möglicherweise unbefugte Benutzer Zugriff auf geschützte Funktionen erhalten.

Absenderseite

Über die Absenderseite wird der Inhalt des Einkaufswagens an den jeweiligen Benutzer gesendet. Wie weiter vorn beschrieben, müssen die Benutzer dazu auf die Schaltfläche **Kasse** klicken, sobald sie ihren Einkaufswagen gefüllt haben.

Nachstehend ist ein Beispiel für eine einfache ASP-Implementierung dieser Funktion dargestellt. Dieser Code verwendet zum dynamischen Erstellen des PunchOutOrderMessage-Dokuments keinen XML-Parser sondern eine statische XML-Vorlage, in die Positionsdaten eingetragen sind. **Dieses Codebeispiel dient lediglich zur Veranschaulichung.**

Im folgenden Beispiel handelt es sich um einen Auszug aus der Produktseite einer Lieferantenwebsite:

```
<!--#include file="punchoutitem.inc"-->
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<!-- saved from
url=(0093)https://secure1.shore.net/wbird/cgi/vsc.cgi/wbird/houses/urban.htm?L+wbird+w
adt4101+928011405 -->

<TABLE border=0>
      <TBODY>
            <TR>
                  <TD><IMG src="UrbanHouses_files/uhjm.gif"> </TD>
                  <TD><STRONG>Jefferson Memorial</STRONG>- A birdfeeder with a
rotunda! This famous American monument will be a unique addition to any garden or yard.
It attracts small to medium sized birds and its dimensions are 11" x 9 1/2" x 8" H.
                  </TD>
            </TR>
      </TBODY>
</TABLE><BR>
-Jefferson Memorial<STRONG>
$139.95</STRONG><BR>
<% AddBuyButton 139.95,101,"Bird Feeder, Jefferson Memorial",5 %>
<BR>
<HR>
```

Die AddBuyButton-Funktion sendet die PunchOutOrderMessage-Meldung zurück an den Benutzer.

Der folgende Code entspricht der oben referenzierten Includedatei (punchoutitem.inc):

```
<%
REM Diese asp entstammt der Datei items.asp, die Artikelparameter festlegt, ein
REM cXML-Dokument formatiert und einem Benutzer ermöglicht, mit dem Artikel zur
Kasse zu gehen.
function CreateCXML(toUser, fromUser, buyerCookie, unitPrice, supPartId, desc)
%>
<?xml version=&quot;1.0&quot; encoding=&quot;UTF-8&quot;?>
<!DOCTYPE cXML SYSTEM
&quot;http://xml.cxml.org/schemas/cXML/1.1.007/cXML.dtd&quot;>
<cXML version=&quot;1.1.007&quot; payloadID=&quot;<%= Now &"@"&
Request.ServerVariables("LOCAL_ADDR")%>&quot; timestamp=&quot;<%= Now
%>&quot;>
      <Header>
            <From>
                  <Credential domain=&quot;ariba.com&quot;>
                        <Identity><%= toUser%></Identity>
                  </Credential>
            </From>
```

```
            <To>
                <Credential domain=&quot;ariba.com&quot;>
                    <Identity><%= fromUser%></Identity>
                </Credential>
            </To>
            <Sender>
                <Credential domain=&quot;ariba.com&quot;>
                    <Identity><%= toUser%></Identity>
                </Credential>
                <UserAgent>PunchoutSite</UserAgent>
            </Sender>
        </Header>
        <Message>
            <PunchOutOrderMessage>
                <BuyerCookie><%= buyerCookie%></BuyerCookie>
                <PunchOutOrderMessageHeader
                operationAllowed=&quot;edit&quot;>
                    <Total>
                        <Money currency=&quot;USD&quot;><%=
                        unitPrice%></Money>
                    </Total>
                </PunchOutOrderMessageHeader>
                <ItemIn quantity=&quot;1&quot;>
                    <ItemID>
                        <SupplierPartID><%= supPartId%></SupplierPartID>
                        <SupplierPartAuxiliaryID><%= supPartAuxId%>
                        </SupplierPartAuxiliaryID>
                    </ItemID>
                    <ItemDetail>
                        <UnitPrice>
                            <Money currency=&quot;USD&quot;><%= unitPrice%>
                            </Money>
                        </UnitPrice>
                        <Description xml:lang=&quot;en&quot;><%= desc%>
                        </Description>
                        <UnitOfMeasure>EA</UnitOfMeasure>
                        <Classification
                            domain=&quot;SupplierPartID&quot;><%= supPartId%>
                        </Classification>
                    </ItemDetail>
                </ItemIn>
            </PunchOutOrderMessage>
        </Message>
    </cXML>
    <% end function

    function AddBuyButton(unitPrice, supPartId, supPartAuxId, desc)

    toUser = Session("toUser")
    fromUser = Session("fromUser")
```

```
buyerCookie = Session("buyercookie")
url = Session("urlToPost")
if not IsEmpty(buyerCookie) then
      %>
      <FORM METHOD=POST ACTION=<%= url%>>
      <INPUT TYPE=HIDDEN NAME="cxml-urlencoded" VALUE="<% CreateCXML
toUser, fromUser, buyerCookie, unitPrice, supPartId, supPartAuxId, desc%>">
      <INPUT TYPE=SUBMIT  value=BUY>
      </FORM>
<%else%>
      </p>
      <%
end if
end function
%>
```

Die AddBuyButton-Funktion enthält das FORM POST-Element, das die URL-codierte PunchOutOrderMessage an den Benutzer zurücksendet.

HTTP-Formularcodierung

Beim Senden einer PunchOutOrderMessage kommt die *HTML-Formularcodierung* zum Einsatz. Hierbei handelt es sich um ein Übertragungsmodell, das sich vom herkömmlichen HTTP-basierten Anforderung/Antwort-Modell unterscheidet und die Integration zwischen Ihrer Website und der Beschaffungsanwendung wesentlich erleichtert. Mithilfe der HTML-Formularcodierung können Käuferorganisationen XML-Daten empfangen, ohne dass sie dazu einen firewallgeschützten Webserver benötigen.

Statt die PunchOutOrderMessage direkt an die Beschaffungsanwendung zu senden, verschlüsselt Ihre Website sie als verborgenes HTML-Formularfeld und sendet sie an den im BrowserFormPost-Element des PunchOutSetupRequest-Dokuments angegebene URL. Das verborgene HTML-Formularfeld muss entweder mit cxml-urlencoded oder cxml-base64 bezeichnet werden. (Bei diesen Namen wird zwischen Groß- und Kleinschreibung unterschieden.)

Diese Codierung ermöglicht Ihnen die Gestaltung einer Kassenseite, die das cXML-Dokument enthält. Wenn Benutzer auf Ihre Schaltfläche **Kasse** klicken, zeigt Ihre Website die Daten (für die Benutzer nicht sichtbar) als HTML-Einreichungsformular der Beschaffungsanwendung an.

Abbrechen der Punchoutfunktion

Es empfiehlt sich in jedem Fall auch eine Schaltfläche einzuplanen, mit der die Benutzer den Punchoutvorgang abbrechen können. Über die Schaltfläche **Abbrechen** wird eine leere PunchOutOrderMessage-Meldung an die Beschaffungsanwendung gesendet. Aus dieser Meldung entnimmt die Beschaffungsanwendung, dass keine Artikel zurückgesendet werden und die vorhandenen Punchoutartikel aus der Anforderung gelöscht werden sollen. Darüber hinaus lassen sich damit sämtliche Aufräumarbeiten für Ihre Website verknüpfen, wie das Leeren des Einkaufswagens und das Schließen der Benutzersitzung.

Bestellannahmeseite

Die Bestellannahmeseite nimmt die von Käuferorganisationen gesendeten cXML-Bestellaufträge an. Ihr Aufbau könnte mit der bereits beschriebenen Aufrufseite vergleichbar sein.

Weitere Informationen zur Annahme von Bestellungen finden Sie in Kapitel 3, „Empfangen von cXML-Bestellaufträgen“.

Anregungen für Ihre Punchoutwebsite

Beim Planen Ihrer Punchoutwebsite sollten Sie die nachstehenden Hinweise beachten.

Richtlinien für die Implementierung

Beachten Sie folgende Richtlinien beim Entwickeln Ihrer Punchoutwebsite:

- Informieren Sie sich über die cXML-Spezifikation.
- Verwenden Sie einen XML-Parser, und überprüfen Sie die Dokumente anhand der cXML-DTD.
- Wenn Sie die Benutzersprachen über die Eigenschaft xml:lang= festlegen, können Sie auch lokalisierte Inhalte bereitstellen.
- Kennzeichnen Sie die Käuferorganisationen mit der Anmeldeinformation From.
- Senden Sie eine eindeutige, temporäre URL-Adresse, zu der die Sitzung umgeleitet werden soll.
- Richten Sie keine permanenten Browsercookies ein.

- Setzen Sie extrinsische Datenanforderungen im Interesse Ihrer Kunden sparsam ein.
- Verwenden Sie für jede Position die UNUOM-Einheiten (United Nations Units of Measure) und UNSPSC-Codes (United Nations Standard Product and Service Codes).
- Stellen Sie Informationen zur Verfügung, mit denen Ihre Kunden wirklich etwas anfangen können. Machen Sie Angaben zur Produktverfügbarkeit, zum Auftragsstatus und zu Sonderangeboten.
- Die Kasse sollte einfach und intuitiv gestaltet sein. Im Idealfall brauchen Kunden zum Einkauf nur auf drei Schaltflächen zu klicken.
- Sehen Sie im Programmcode auch die Möglichkeit nachfolgender Sitzungen zum Bearbeiten und Anzeigen vor. Die Benutzer können Auftragsdetails für Punchoutartikel (z. B. Bestellmenge) in ihrer Beschaffungsanwendung nicht ändern sondern müssen einen weiteren Punchout mit einer Bearbeitungssitzung durchführen.
- Am besten ist es für die Benutzer, wenn in einer Überprüfungssitzung auch der Bestellstatus angezeigt wird.
- Testen Sie Ihre Punchoutwebsite. Planen Sie Zeit für komplexe Tests mit den Beschaffungsanwendungen Ihrer Kunden ein.
- Bei Punchouttransaktionen werden lediglich Angebote, jedoch keine Bestellaufträge erstellt. Implementieren Sie eine Bestellannahmeseite für die Annahme von cXML-Bestellaufträgen.

Käufer- und Lieferantencookies

Mit den Käufer- und Lieferantencookies können Käufer bzw. Lieferanten ihre eigenen Positionsdaten für ihre Backendsysteme neu instanziieren.

- Senden Sie das empfangene Käufercookie (BuyerCookie) unverändert zurück.
- Nutzen Sie das Lieferantencookie (SupplierPartAuxiliaryID).

Käufercookies sind mit einer Bestellanforderungsnummer vergleichbar. Sie übermitteln Statusangaben, mit denen das System der Käuferorganisation eine Beziehung zwischen Anforderung und Einkaufskorb herstellen kann.

Auf dieselbe Weise entspricht ein Lieferantencookie gewissermaßen einer Angebotsnummer. Es übermittelt Statusangaben, mit denen Ihr System eine Beziehung zwischen dem Einkaufskorb und der Anforderung bzw. dem Bestellauftrag der Käuferorganisation herstellt. Beschaffungsanwendungen senden das Lieferantencookie in den darauffolgenden Edit- oder Inspect-Punchoutsitzungen an Sie zurück und fügen es in den daraus hervorgehenden Bestellauftrag ein. Ihre Website sollte das Lieferantencookie nutzen, damit keine sichtbaren Lieferantendaten an den Käufer zurückgesendet werden müssen.

Personalisierung

Die Kopfzeile des PunchOutSetupRequest-Dokuments bezeichnet stets die Käuferorganisation, kann aber auch Elemente wie Contact und Extrinsic (Kostenstelle des Benutzers, Benutzerstandort oder Produktkategorie) enthalten, mit denen Sie eine dynamische URL-Adresse für den Benutzer festlegen können.

Auch wenn nicht alle Käuferorganisationen solche extrinsischen Daten senden, können Sie Ihr Internetgeschäft mit diesen Angaben über die einfache Organisationsebene hinaus kundenspezifisch ausrichten. Beispielsweise ist es möglich, eine separate Website für die einzelnen Kostenstellen (bzw. die einzelnen Produktkategorien oder Benutzer) in einer Käuferorganisation einzurichten.

Ebenso können Sie frühere Angebote an den Benutzer speichern und anzeigen. Sie können Benutzern gestatten, Angebote wiederzuverwenden, den Status von Aufträgen zu prüfen und Berichte über die Tätigkeit in der Vergangenheit zu erstellen. Zur Verhinderung von Sicherheitsproblemen sollten Sie die Angebotshistorie getrennt nach Benutzern speichern.

Eine wesentliche Überlegung in der Planungsphase ist der zum Erstellen einer hochdynamischen und benutzerspezifischen Punchoutwebsite erforderliche Aufwand. Dabei muss ein Kompromiss zwischen Benutzerausrichtung und Komplexität gefunden werden: eine komplexere Website erfordert mehr Zeit zum Implementieren und Pflegen, ist aber auch wertvoller für die Benutzer. Wir empfehlen, mit einer einfachen Punchoutwebsite zu beginnen und diese dann nach und nach zu perfektionieren.

Kapitel 3
Empfangen von cXML-Bestell-aufträgen

Im folgenden Kapitel wird die Einrichtung einer Website für den Empfang von Bestellaufträgen im cXML-Format beschrieben. Darüber hinaus finden Sie hier Informationen zum Senden von Bestellstatusmeldungen an Käuferorganisationen oder Märkte.

Bestellauftragsvorgang

Beschaffungsanwendungen wandeln Bestellanforderungen in einen oder mehrere Bestellaufträge um. Ein Bestellauftrag ist eine formelle Anforderung eines Käuferunternehmens an einen Lieferenten zur Erfüllung eines Vertrags.

cXML ist nur eine Art der Übertragung von Bestellaufträgen. Weitere gebräuchliche Formate sind E-Mail, Fax und EDI (X.12 Electronic Data Interchange). cXML stellt insofern das für Bestellaufträge am besten geeignete Format dar, weil es ohne weiteres eine automatische Auftragsverarbeitung gestattet. Seine klar definierte Struktur ermöglicht Auftragsverarbeitungssystemen, die Elemente eines Bestellauftrags problemlos zu interpretieren. Nahezu ohne menschliches Eingreifen werden die entsprechenden Daten in Bestellaufträgen bedarfsgerecht an Ihre Versand-, Abrechnungs- und Verkaufsabteilung geleitet.

Außerdem gestattet die Auftragsweiterleitung per cXML die Übertragung von Lieferantencookies (SupplierPartAuxiliaryID) und Anlagen zum Bestellauftrag.

Wenn Sie Ihr Konto auf einem E-Commerce-Netzknoten konfigurieren, geben Sie eine URL-Adresse an, an die alle cXML-Bestellaufträge gesendet werden. Bei Erhalt eines Bestellauftrags senden Sie diesen an Ihr internes Auftragsverwaltungssystem und erfüllen ihn wie gewöhnlich. Außerdem muss Ihre Website eine Bestellantwort an den E-Commerce-Netzknoten senden, die dem Käufer mitteilt, dass Sie den Bestellauftrag erfolgreich empfangen und geparst haben.

Zum Empfangen von Bestellaufträgen im cXML-Format ist eine Punchoutwebsite nicht unbedingt erforderlich. Punchout und cXML-Bestellannahme sind zwei verschiedene Funktionen. Allerdings sind die für Punchout benötigten Infrastrukturmerkmale und Anwendungen dieselben wie die zum Empfang von cXML-Bestellaufträgen.

Empfangen von Bestellaufträgen

Zum Übermitteln von Bestellaufträgen stehen zwei Arten von cXML-Dokumenten zur Verfügung. Die Beschaffungsanwendung sendet ein OrderRequest-Dokument, und Sie antworten mit einem OrderResponse-Dokument. Der Versand dieser Dokumente erfolgt über den E-Commerce-Netzknoten.

OrderRequest

Das OrderRequest-Dokument ist mit einem Bestellauftrag vergleichbar. Das folgende Beispiel zeigt ein OrderRequest-Dokument für einen Artikel:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE cXML SYSTEM "http://xml.cxml.org/schemas/cXML/1.1.007/cXML.dtd">
<cXML version="1.1.007" xml:lang="en-US" payloadID="93369535150910.10.57.136"
timestamp="2000-08-03T08:49:11+07:00">
<Header>
    <From>
        <Credential domain="AribaNetworkUserId">
            <Identity>admin@acme.com</Identity>
        </Credential>
    </From>
    <To>
        <Credential domain="DUNS">
            <Identity>114315195</Identity>
        </Credential>
    </To>
    <Sender>
        <Credential domain="AribaNetworkUserId">
            <Identity>sysadmin@ariba.com</Identity>
            <SharedSecret>abracadabra</SharedSecret>
        </Credential>
        <UserAgent>Ariba Network V1.1</UserAgent>
    </Sender>
</Header>
<Request>
    <OrderRequest>
        <OrderRequestHeader orderID="DO102880"
        orderDate="2000-08-03T08:49:09+07:00"  type="new">
```

```
<Total>
    <Money currency="USD">4688.00</Money>
</Total>
<ShipTo>
    <Address isoCountryCode="US" addressID="1000467">
        <Name xml:lang="en">Acme, Inc.</Name>
        <PostalAddress name="default">
            <DeliverTo>John Q. Smith</DeliverTo>
            <DeliverTo>Buyers Headquarters</DeliverTo>
            <Street>123 Main Street</Street>
            <City>Mountain View</City>
            <State>CA</State>
            <PostalCode>94089</PostalCode>
            <Country>United States</Country>
        </PostalAddress>
        <Email name="default">john_smith@acme.com</Email>
        <Phone name="work">
            <TelephoneNumber>
                <CountryCode isoCountryCode="US">1
                </CountryCode>
                <AreaOrCityCode>800</AreaOrCityCode>
                <Number>5555555</Number>
            </TelephoneNumber>
        </Phone>
    </Address>
</ShipTo>
<BillTo>
    <Address isoCountryCode="US" addressID="12">
        <Name xml:lang="en">Acme Accounts Payable</Name>
        <PostalAddress name="default">
            <Street>124 Union Street</Street>
            <City>San Francisco</City>
            <State>CA</State>
            <PostalCode>94128</PostalCode>
            <Country isoCountryCode="US">US</Country>
        </PostalAddress>
        <Phone name="work">
            <TelephoneNumber>
                <CountryCode isoCountryCode="US">1
                </CountryCode>
                <AreaOrCityCode>415</AreaOrCityCode>
                <Number>6666666</Number>
            </TelephoneNumber>
        </Phone>
    </Address>
</BillTo>
<Shipping>
    <Money currency="USD">12.34</Money>
    <Description xml:lang="en-us">FedEx 2-day</Description>
</Shipping>
```

```
                <Tax>
                    <Money currency="USD">10.74</Money>
                    <Description xml:lang="en">CA State Tax</Description>
                </Tax>
                <Payment>
                    <PCard number="1234567890123456" expiration="2002-03-12"/>
                </Payment>
            </OrderRequestHeader>
            <ItemOut quantity="2" >
                <ItemID>
                    <SupplierPartID>220-3165</SupplierPartID>
                    <SupplierPartAuxiliaryID>E000028901</SupplierPartAuxiliaryID>
                </ItemID>
                <ItemDetail>
                    <UnitPrice>
                        <Money currency="USD">2344.00</Money>
                    </UnitPrice>
                    <Description xml:lang="en">Laptop Computer Notebook Pentium® II
                    processor w/AGP, 300 MHz, with 12.1&amp;quot; TFT XGA Display
                    </Description>
                    <UnitOfMeasure>EA</UnitOfMeasure>
                    <Classification domain="UNSPSC">43171801</Classification>
                    <URL>http://www.supplier.com/Punchout.asp</URL>
                    <Extrinsic name="ExtDescription">Enhanced keyboard</Extrinsic>
                </ItemDetail>
                <Distribution>
                    <Accounting name="DistributionCharge">
                        <Segment type="Account" id="7720"
                            description="Office Supplies"/>
                        <Segment type="CostCenter" id="610"
                            description="Engineering Management"/>
                    </Accounting>
                    <Charge>
                        <Money currency="USD">4688.00</Money>
                    </Charge>
                </Distribution>
            </ItemOut>
        </OrderRequest>
</Request>
</cXML>
```

OrderResponse

Das OrderResponse-Dokument bestätigt, dass Sie den Bestellauftrag erhalten und fehlerfrei geparst haben. Dies ist keine Verpflichtung zur Ausführung des Bestellauftrags, sondern bestätigt nur dessen Eingang als gültiges cXML-Dokument.

```
<?xml version="1.0"?>
<!DOCTYPE cXML SYSTEM "http://xml.cxml.org/schemas/cXML/1.1.007/cXML.dtd">
<cXML version="1.1.007" payloadID="8/3/2000 8:49:30 PM@205.180.14.45"
timestamp="2000-08-03T08:49:30+07:00">
      <Response>
            <Status code="200" text="OK"/>
       </Response>
</cXML>
```


Annehmen von Auftragsanlagen

Häufig ist es erforderlich, dass ein Bestellauftrag durch den Käufer mit Memos, Zeichnungen oder Faxnachrichten näher erläutert werden muss. MIME-codierte Dateien (Multipurpose Internet Mail Extensions) beliebigen Typs können an cXML-Bestellaufträge angehängt werden.

cXML enthält lediglich Verweise auf externe MIME-Teile, die in einem mehrteiligen MIME-Container übertragen werden (mit dem cXML-Dokument in einer E-Mail oder zusammen als Fax).

E-Commerce-Netzwerkhubs empfangen die Anlagen und leiten diese an den Lieferanten weiter oder speichern sie für den Onlineabruf.

Weitere Informationen über Anlagen zu Bestellaufträgen finden Sie im Abschnitt „Übertragen von Anlagen“ auf Seite 53.

Weitere Informationen zum MIME-Standard finden Sie auf folgenden Websites:

www.hunnysoft.com/mime
www.rad.com/networks/1995/mime/mime.htm

Anhang A
cXML-Sprachspezifikation

Im folgenden Anhang werden die Protokoll- und Datenformate von cXML (commerce eXtensible Markup Language) beschrieben. Er enthält alle Angaben, die Sie aus Sicht des Client- oder des Serversystems zum Implementieren der unterstützten Transaktionen brauchen. Sowohl die in den Transaktionen enthaltenen Protokollinteraktionen als auch die Geschäftsdokumente werden ausführlich behandelt.

Darüber hinaus werden anhand von Beispielen einige echte Implementierungen von cXML und deren Verwendung erläutert.

Dieser Anhang enthält folgende Abschnitte:

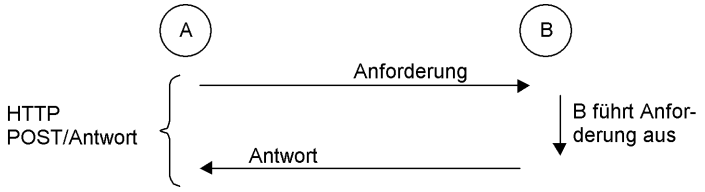
- Protokollspezifikation
- Grundelemente
- Profiltransaktion
- Auftragsdefinitionen
- Punchouttransaktion
- Spätere Statusänderungen
- Katalogdefinitionen
- Definitionen der Abonnementsverwaltung
- Nachrichtenabrufdefinitionen

Protokollspezifikation

Es gibt zwei Kommunikationsmodelle für cXML-Transaktionen: „Anforderung-Antwort“ und „Unidirektional“. Diese beiden Modelle ermöglichen eine unkomplizierte Implementierung, weil die erforderlichen Vorgänge mit großer Genauigkeit beschrieben sind. Da in bestimmten Situationen jeweils nur eines von beiden Modellen anwendbar ist, werden insgesamt beide Modelle benötigt.

Anforderung-Antwort-Modell

Anforderung-Antwort-Transaktionen können nur über eine HTTP-Verbindung abgewickelt werden. Die folgende Abbildung zeigt die Schritte zwischen den Kommunikationspartnern A und B bei einer Anforderung-Antwort-Interaktion:



Anforderung-Antwort-Transaktion

Diese Transaktion umfasst folgende Schritte:

1. A baut über einen festgelegten URL (die Adresse von B) eine HTTP/1.x-Verbindung zu B auf.
2. A versendet das cXML-Dokument mit einem POST-Vorgang über die HTTP-Verbindung.
3. A wartet auf eine über die HTTP-Verbindung zurückgesendete Antwort.
4. B verfügt über einen HTTP/1.x-kompatiblen Server, der die HTTP-Anforderung an die durch den URL aus Schritt 1 bezeichnete Ressource leitet. Dies kann eine beliebige, dem Server von B bekannte Ressource sein, zum Beispiel ein CGI-Programm oder eine ASP-Seite.
5. Die in Schritt 4 bezeichnete Ressource von B liest das cXML-Dokument und weist die Anforderung dem dafür zuständigen Handler zu.
6. Der von B verwendete Handler für cXML-Anforderungen führt die in der Anforderung angegebenen Schritte aus und formatiert ein cXML-Antwortdokument.

7. B sendet die cXML-Antwort über die in Schritt 1 aufgebaute HTTP-Verbindung an A.

8. A liest die cXML-Antwort und leitet sie an den Vorgang weiter, durch den die Anforderung ausgelöst wurde.

9. A beendet die in Schritt 1 aufgebaute HTTP-Verbindung.

Dieser Vorgang wird dann für weitere Anforderung-Antwort-Zyklen wiederholt.

Um die genannten Schritte zu vereinfachen, gliedern sich cXML-Dokumente in zwei deutlich zu unterscheidende Teile:

- Kopfzeile: Die Kopfzeile enthält Authentifizierungs- und Adressangaben.
- Anforderungs- bzw. Antwortdaten: Dieser Teil enthält eine bestimmte Anforderung bzw. Antwort und die zu übermittelnden Informationen.

Beide Elemente werden in einem übergeordneten Containerelement übertragen. Das folgende Beispiel zeigt den Aufbau eines cXML-Anforderungsdokuments:

```
<cXML>
    <Header>
        Kopfzeileninformationen…
    </Header>
    <Request>
        Anforderungsinformationen…
    </Request>
</cXML>
```

Das folgende Beispiel zeigt den Aufbau eines cXML-Antwortdokuments:

```
<cXML>
    <Response>
        Antwortinformationen…
    </Response>
</cXML>
```

In der Antwortstruktur ist kein Kopfzeilenelement erforderlich, da die Antwort stets in derselben HTTP-Verbindung übertragen wird wie die Anforderung.

XML-Konventionen

cXML beschreibt diskrete Artikel, häufig Eigenschaften in herkömmlichen Geschäftsunterlagen, mithilfe von Elementen. Auch Informationen mit offensichtlichen Untergliederungen und Beziehungen zwischen Untergliederungen, wie etwa eine Adresse, werden mit Elementen beschrieben.

Sehr häufig werden in cXML Attribute verwendet.

In cXML sind alle Element- und Attributnamen vollständige Wörter, wobei Großbuchstaben (keine Bindestriche) die Wörter voneinander trennen. Elementnamen beginnen mit einem Großbuchstaben, Attributnamen mit einem Kleinbuchstaben. Beispiel:

Elemente:	Sender, Credential, Payment, ItemDetail
Attribute:	version, payloadID, lineNumber, domain

cXML-Container

Das Containerelement cXML bildet die Wurzel der cXML-Dokumentstruktur, die sämtliche anderen Elemente enthält. Es ist in allen cXML-Transaktionen vorhanden.

Im folgenden Beispiel ist das cXML-Element vollständig spezifiziert:

```
<cXML version="1.1.007" xml:lang="en-US"
    payloadID=1234567.4567.5678@test.ariba.com
    timestamp="1999-03-31T18:39:09-08:00">
```

Folgende Attribute stehen für das cXML-Element zur Verfügung:

version (optional)	Gibt die Version des cXML-Protokolls an. Obwohl ein prüfender XML-Parser den Wert des Attributs „version“ auch aus der Referenz-DTD ermitteln kann, sollte die Version in allen cXML-Dokumenten explizit angegeben werden. Dadurch ist gewährleistet, dass die Dokumente auch von Anwendungen mit nicht prüfenden Parsern verarbeitet werden können.
xml:lang (optional)	Gebietsangaben, die für sämtliche freien Textteile im Dokument verwendet werden. Der Empfänger sollte diese Informationen mit denselben oder ähnlichen Gebietsangaben anzeigen bzw. beantworten. Beispiel: Ein Client, der in einer Anforderung xml:lang="en-UK" angegeben hat, erhält eine Rückmeldung in Form von "en"-Daten.
payloadID	Eine in Bezug auf Ort und Zeit eindeutige Nummer, mit der verloren gegangene oder problematische Dokumente protokolliert werden. Dieser Wert darf sich bei Wiederholversuchen nicht ändern. Empfohlene Implementierung: datetime.process id.random number@hostname
timestamp	Datum und Uhrzeit (im ISO 8601-Format) des Meldungsversands. Dieser Wert darf sich bei Wiederholversuchen nicht ändern. Das Angabe erfolgt im Format JJJJ-MM-TTThh:mm:ss-hh:mm (Beispiel: 1997-07-16T19:20:30+01:00).

Durch „xml:lang“ definierte Gebietsangaben

Das Attribut xml:lang wird auch bei den meisten freien Textelementen (z. B. Description und Comments) verwendet. Während es nach der XML-Spezifikation zulässig ist, dass ein Element standardmäßig die Gebietsangaben des jeweils übergeordneten Elements übernimmt, führt dies zu Leistungseinbußen bei Abfragen des Dokumentenbaums. Daher sollte der Gebietsangabenbezeichner in cXML zusammen mit der betroffenen Zeichenkette verwaltet werden.

Die im cXML-Protokoll auftretenden xml:lang-Attribute haben keinen Einfluss auf formatierte Daten wie Zahlen, Datumsangaben und Uhrzeiten. Wie im Folgenden für das Attribut timestamp beschrieben, werden solche diskreten Werte entsprechend ihrem Datentyp formatiert. Längere Zeichenketten (und referenzierte Webseiten), die nicht zur maschinellen Verarbeitung gedacht sind, enthalten unter Umständen ein gebietsspezifisches Ziffern- oder Datumsformat, das einem verwandten xml:lang-Attribut entspricht.

Uhrzeit und andere Datentypen

Das Attribut timestamp (und alle sonstigen Datums- und Uhrzeitangaben in cXML) müssen in dem eingeschränkten ISO-8601-Derivat formatiert werden, das in einer Word Wide Web Consortium (W3C)-Note mit dem Titel „Date and Time Formats“ beschrieben ist. Dieses Dokument kann unter www.w3.org/TR/NOTE-datetime-970915.html heruntergeladen werden.

Für Timestamps sind neben den Mindestangaben für ein vollständiges Datum auch Stunden-, Minuten- und Sekundenangaben erforderlich. Bruchteile von Sekunden sind optional. Dieses Protokoll verlangt Zeitangaben in Ortszeit mit Angabe des Zeitzonenunterschieds in Bezug auf UTC (Coordinated Universal Time, auch als Greenwich Mean Time bezeichnet). Der Zeitzonenbezeichner „Z“ ist nicht zulässig.

Beispiel: 2000-04-14T013:36:00-08:00 entspricht dem 14. April 2000, 13:36 Uhr (Pacific Standard Time).

Weitere Verweise auf die in cXML verwendeten Datums-, Uhrzeit- und sonstigen Datentypformate finden Sie:

- auf der Referenzseite für XML-Datentypen von Microsoft unter msdn.microsoft.com/xml/reference/schema/datatypes.asp
- in der ursprünglichen XML-Empfehlung des Word Wide Web Consortium (W3C) unter www.w3c.org/TR/1998/NOTE-XML-data-0105

Containerschichten

Das cXML-Element umgrenzt den Haupttext eines normalen XML-Dokuments. So könnte ein Dokument beispielsweise wie folgt beginnen:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE cXML SYSTEM "http://xml.cxml.org/schemas/cXML/1.1.007/cXML.dtd">
<cXML version="1.1.007" xml:lang="en-US"
      payloadID="0c300508b7863dcclb_13550"
      timestamp="2000-01-09T01:36:05-08:00">
      …
```

Dieses Dokument wird normalerweise mit dem MIME-Medientyp text/xml über HTTP gesendet, wobei der Parameter charset der Codierung im Dokument entspricht. Da HTTP 8-Bit-rein ist, kann jede vom Empfangsparser unterstützte Codierung ohne Inhaltsübertragungscodierung wie „base64“ oder „quoted-printable“ verwendet werden. Alle XML-Parser unterstützen die UTF-8-Codierung, die alle Unicode-zeichen umfasst. Daher ist es sinnvoll, wenn Anwendungen beim Übertragen von cXML-Dokumenten diese Codierung nutzen.

Hinweis: Nach den XML-Medientypen in RFC 2376 setzt der MIME-Parameter charset jede in der XML-Deklaration angegebene Codierung außer Kraft. Außerdem ist die Standardcodierung für den Medientyp text/xml us-ascii, nicht UTF-8, wie in Abschnitt 4.3.3 der XML-Spezifikation angegeben. Daher sollten cXML-Dokumente aus Gründen der Übersichtlichkeit in der XML-Deklaration eine explizite Codierungsangabe enthalten. MIME-Container müssen für die Medientypen text/xml oder application/xml einen passenden charset-Parameter verwenden.

Bei der HTTP-Übertragung eines cXML-Dokuments können beispielsweise die folgenden MIME- und HTTP-Kopfzeilen verwendet werden:

```
POST /cXML HTTP/1.0
Content-type: text/xml; charset="UTF-8"
Content-length: 1862
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
User-Agent: Java1.1
Host: localhost:8080
Connection: Keep-Alive

<?xml version="1.0" encoding="UTF-8"?>
…
```

Übertragen von Anlagen

Beim Senden eines OrderRequest-Dokuments, das auf externe Dateien verweist, können sich die referenzierten Dateien entweder auf einem für den Lieferanten zugänglichen Server befinden oder zusammen mit dem cXML-Dokument übertragen werden. Diese zweite Option erfordert die Verwendung eines MIME-Containers für mehrteilige Nachrichten. Eine cXML-Anforderung an diesen Container (eine Beschreibung seiner Grundlagen finden Sie in RFC 2046 „Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types“) ist der Einschluss von Inhalts-ID-Kopfzeilen für jede angehängte Datei.

Das folgende Beispiel zeigt das obligatorische Grundgerüst eines cXML-Dokuments mit einem angehängten JPEG-Bild (ohne die bereits angeführten HTTP-Kopfzeilen):

```
POST /cXML HTTP/1.0
Content-type: multipart/mixed; boundary=eindeutige Angabe

--eindeutige Angabe
Content-type: text/xml; charset="UTF-8"

<?xml version="1.0" encoding="UTF-8"?>
…
--eindeutige Angabe
Content-type: image/jpeg
Content-ID: <uniqueCID@cxml.org>
…
--eindeutige Angabe--
```

Dieses Grundgerüst enthält alle Elemente, die der empfangende MIME-Parser mindestens verarbeiten können muss. Anwendungen, die den in RFC 2387 „The MIME Multipart/ Related Content-type“ beschriebenen Medientyp verwenden, erhalten zusätzlich viele weitere Informationen, wenn das Grundgerüst erweitert wird:

```
POST /cXML HTTP/1.0
Content-type: multipart/related; boundary=eindeutige Angabe;
      type="text/xml"; start=<uniqueCIDmain@cxml.org>

--eindeutige Angabe
Content-type: text/xml; charset="UTF-8"
Content-ID: <uniqueCIDmain@cxml.org>

<?xml version="1.0" encoding="UTF-8"?>
…
--eindeutige Angabe
Content-type: image/jpeg
Content-ID: <uniqueCID@cxml.org>
…
--eindeutige Angabe--
```

Empfangende MIME-Parser, die den Medientyp multipart/related nicht verstehen, müssen die beiden dargestellten Beispiele gleich behandeln. Jeder Teil der MIME-Übertragung kann zusätzlich eine Inhaltsübertragungscodierung aufweisen und diese auch verwenden. Für die HTTP-Übertragung ist dieser Zusatz nicht erforderlich. Kopfzeilen mit Inhaltsbeschreibung und Inhaltsaufstellung liefern zwar nützliche Daten, sind aber im cXML-Protokoll optional.

Weitere Informationen über externe Dateien als Anlagen zu Bestellaufträgen finden Sie im Abschnitt „Attachment“ auf Seite 74.

Header

Das Header-Element enthält Angaben zur Adresse und zur Authentifizierung. Der Wert dieses Elements bleibt unabhängig von den Request- oder Response-Elementen im Haupttext der cXML-Nachricht unverändert. Die Anwendungen benötigen lediglich die Identität des Anforderers, ohne dass die zur Identifizierung angegebenen Daten auf ihre Richtigkeit geprüft werden müssen.

Im folgenden Beispiel ist das Header-Element dargestellt:

```
<Header>
      <From>
            <Credential domain="AribaNetworkUserId">
                  <Identity>admin@acme.com</Identity>
            </Credential>
      </From>
```

```
        <To>
            <Credential domain="DUNS">
                <Identity>012345678</Identity>
            </Credential>
        </To>
        <Sender>
            <Credential domain="AribaNetworkUserId">
                <Identity>sysadmin@ariba.com</Identity>
                <SharedSecret>abracadabra</SharedSecret>
            </Credential>
            <UserAgent>Ariba Network 1.1</UserAgent>
        </Sender>
    </Header>
```

Die Elemente From und To sind als logischer Ursprung und Bestimmungsort der Nachrichten gleichbedeutend mit Absender und Empfänger in SMTP-Mails. Sender ist derjenige Kommunikationspartner, der die HTTP-Verbindung aufbaut und das cXML-Dokument sendet.

Sender enthält das Credential-Element, das dem Empfänger die Authentifizierung des Absenders gestattet. Es ermöglicht eine starke Authentifizierung, ohne eine Ende-zu-Ende-Digitalzertifikatsinfrastruktur mit öffentlichem Schlüssel erforderlich zu machen. Der Empfänger braucht nur einen Benutzernamen und ein Passwort auszugeben, damit der Absender Requests durchführen kann.

Zu Beginn sind die Werte für Sender und From identisch. Wenn das cXML-Dokument jedoch durch E-Commerce-Netzknoten geleitet wird, ändert sich das Element Sender und gibt jeweils den aktuell sendenden Partner wieder.

From

Dieses Element kennzeichnet den Urheber der cXML-Anforderung. Es kann optional mehrere Credential-Elemente enthalten, sodass sich die Anforderer mit mehreren Identifizierungsverfahren zu erkennen geben können. Dieser Gebrauch mehrerer Arten von Anmeldeinformationen erfolgt analog zum Senden sowohl der SMTP- als auch der X.400-Adresse in einer E-Mail-Nachricht.

To

Dieses Element kennzeichnet den Bestimmungsort der cXML-Anforderung. Wie das Element From kann es mehrere Credential-Elemente enthalten, um das Ziel genauer zu spezifizieren.

Sender

Über dieses Element kann der Empfänger, denjenigen Partner, der die HTTP-Verbindung aufgebaut hat, erkennen und authentifizieren. Da der Empfänger authentifizieren muss, wer die Ausführung von Arbeiten anfordert, enthält es aussagekräftigere Credential-Angaben als die Elemente From und To.

Credential

Dieses Element enthält Identifizierungs- und Authentifizierungswerte, die in cXML-Nachrichten verwendet werden.

Für das Credential-Element stehen folgende Attribute zur Verfügung:

domain	Gibt die Art der Anmeldeinformationen an. Durch dieses Attribut können Dokumente mehrere Arten von Informationen für mehrere Authentifizierungsdomänen enthalten. Für über Ariba Network gesendete Nachrichten ist die Domäne normalerweise AribaNetworkUserId oder DUNS.
type (optional)	Bei Anforderungen von bzw. zu einem Markt wird sowohl der Markt als auch das Mitgliedsunternehmen durch die Credential-Elemente From und To angegeben. In diesem Fall werden die Anmeldeinformationen für den Markt im Attribut type definiert, dessen Wert „marketplace“ lautet.

Credential enthält neben einem Identity-Element optional auch ein SharedSecret- oder DigitalSignature-Element. Während durch das Identity-Element angegeben wird, wen das Credential-Element darstellt, wird durch die optionalen Authentifizierungselemente die Identität des Partners geprüft.

Das SharedSecret-Element wird verwendet, wenn dem Sender-Element eine Benutzernamen-/Passwortkombination zugeordnet ist, die der Anforderer erkennt.

Das DigitalSignature-Element kann verwendet werden, wenn beide Parteien ein gemeinsames Zertifikatsformat und eine Zuständigkeit vereinbart haben. Das Attribut type eines DigitalSignature-Elements gibt die Art des verwendeten Zertifikats an.

> **Hinweis:** Die Verwendung von Authentifizierungselementen in Dokumenten, die über unidirektionale Verbindungen übertragen werden, ist zu vermeiden. Solche Dokumente werden durch die Browser der Benutzer geleitet, die somit die Dokumentherkunft (einschließlich der Credential-Elemente) sehen können.

Request

Clients senden Anforderungen für Vorgänge. Pro cXML-Container ist nur ein Request-Element zulässig. Dies vereinfacht die Serverimplementierungen entscheidend, da beim Lesen der cXML-Dokumente somit kein Demultiplexen erforderlich sind. Das Request-Element kann praktisch jeden Typ von XML-Daten enthalten.

Für das Request-Element stehen folgende Attribute zur Verfügung:

deploymentMode (optional)	Gibt an, ob es sich bei einer Anforderung um eine Testanforderung oder um eine Produktionsanforderung handelt. Zulässige Werte sind „Production“ (vorgeschlagener Wert) und „Test“.

Response

Server senden Antworten, um die Clients über das Ergebnis von Vorgängen zu informieren. Da das Ergebnis einer Anforderung manchmal keine Daten enthält, kann das Response-Element optional auch nur ein Status-Element enthalten. Darüber hinaus können Response-Elemente beliebige Daten auf Anwendungsebene umfassen, was in Punchoutkonfigurationen dem PunchOutSetupResponse-Element entsprechen würde.

Status

Dieses Element informiert über den Erfolg oder Misserfolg eines Anforderungsvorgangs.

Für das Status-Element stehen folgende Attribute zur Verfügung:

code	Statuscode der Anforderung. Richtet sich nach dem HTTP-Statuscodemodell. So steht 200 zum Beispiel für eine erfolgreiche Anforderung.
text	Text der Statusmeldung. Durch diesen Text, der aus kanonischen Zeichenketten in Englisch besteht, werden Protokolle für Benutzer leichter lesbar.
xml:lang (optional)	Sprache, in der die Daten im Status-Element angegeben sind. Aus Gründen der Kompatibilität mit cXML 1.1 ist dieses Element derzeit noch optional, kann aber in künftigen Versionen von cXML durchaus obligatorisch sein.

Durch die Attribute des Status-Elements wird angegeben, was mit der Anforderung geschehen ist.

Das Status-Element kann beliebige Daten enthalten, die der Anforderer benötigt. Beim Statuscode 200/OK sind unter Umständen überhaupt keine Daten zu übermitteln. Beim Statuscode 500/Internal Server Error ist hingegen dringend anzuraten, den eigentlichen XML-Parsingfehler oder Anwendungsfehler darzustellen. Dadurch werden die einseitige Fehlersuche und eventuelle Kompatibilitätstests wesentlich erleichtert.

Wenn der Statuscode nicht im 200er Bereich liegt (Beispiel: 200/OK), sollten durch den Server keine zusätzlichen Antwortelemente (z. B. PunchOutSetupResponse) eingefügt werden.

Die HTTP 1.1-Spezifikation enthält zahlreiche Statuscodes, die für cXML ungeeignet sind. Da cXML im Schichtenmodell zumeist über HTTP liegt, werden zahlreiche Fehler (z. B. 404/Not Found) durch die Übertragung behandelt. Die Statuscodes 200/OK und 500/Internal Server Error treten mit der höchsten Wahrscheinlichkeit auf. Prüfungsfehler beim Parsen eines Request-Dokuments würden normalerweise zu einem Übertragungsfehler führen, zum Beispiel zum HTTP-Fehler 400/Bad Request.

In der nachstehenden Tabelle finden Sie weitere HTTP-Codes, die verwendet werden können.

cXML enthält nur sehr wenige nicht-HTTP-konforme Statuscodes.

- 550: Der nächste cXML-Server zum Beenden einer Transaktion, die einen vorherigen Verbindungsaufbau erfordert, ist nicht erreichbar. Dieser Code wird eventuell durch einen Zwischenknoten zurückgesendet, wenn ein Lieferanten-standort nicht erreichbar ist. (Wenn eine Verbindung hergestellt wurde, melden Zwischenknoten Fehler direkt an den Client zurück.)
- 551: Die Anforderung kann nicht weitergeleitet werden, da beim Lieferanten eine falsche Konfiguration vorliegt. Beispielsweise ist es möglich, dass ein Zwischen-knoten durch einen Lieferanten nicht authentifiziert wird. Dieser Fehler kann zwar nicht durch den Client behoben werden, allerdings ist es möglich, dass er noch vor einem erneuten Versuch des Clients korrigiert wird.
- 560: Temporärer Serverfehler. Dieser Fehler tritt beispielsweise auf, wenn ein Server für Wartungsarbeiten heruntergefahren wurde. Der Client sollte den Versuch zu einem späteren Zeitpunkt wiederholen.

Status	Kanonischer Text	Bedeutung
200	OK	Der Server konnte diese Request ausführen, obwohl die zurückgesendete Response möglicherweise Warn- oder Fehlermeldungen von Anwendungen enthält.
201	Accepted	Einige Verarbeitungsschritte sind möglicherweise noch nicht abgeschlossen. Wie im Abschnitt „StatusUpdateRequest“ auf Seite 88 erwähnt, muss der Client mit späteren StatusUpdate-Transaktionen rechnen, wenn dieser Status als Antwort auf ein OrderRequest-Dokument zurückgesendet wird.
204	No Content	Sämtliche Request-Angaben waren gültig und sind erkannt worden. Der Server hat keinen Zugriff auf Response-Daten des angeforderten Typs. In einer PunchOutOrderMessage zeigt dieser Status an, dass die Punchoutsitzung ohne Änderungen am Einkaufswagen (oder der Kundenanforderung) beendet wurde.
400	Bad Request	Die Anforderung ist für den Server nicht akzeptabel, obwohl sie richtig geparst wurde.
401	Unauthorized	Die im Request-Element gelieferten Anmeldeinformationen (Sender) wurden vom Server nicht erkannt.
402	Payment Required	Diese Request muss ein vollständiges Payment-Element enthalten.
403	Forbidden	Der Benutzer ist zum Ausführen dieser Request nicht berechtigt.
406	Not Acceptable	Alias für Code 400: Das Request-Dokument ist für den Server nicht akzeptabel, obwohl es fehlerfrei geparst wurde.
409	Conflict	Der aktuelle Status des Servers oder dessen interne Daten haben die (aktualisierte) Vorgangsanforderung verhindert. Eine identische Anforderung kann später Erfolg haben, insbesondere nach Ausführung eines anderen Vorgangs.
412	Precondition Failed	Eine Voraussetzung für das Request-Dokument (z. B. eine für PunchOutSetupRequest edit geeignete Punchoutsitzung) wurde nicht erfüllt. Dieser Status impliziert in der Regel, dass der Client einen Teil einer früheren Übertragung vom Server ignoriert hat (z. B. das Attribut operationAllowed eines PunchOutOrderMessageHeader).

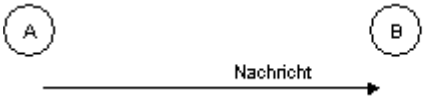
Status	Kanonischer Text	Bedeutung
417	Expectation Failed	Durch Request wurde eine Ressourcenbedingung impliziert, die nicht erfüllt wurde. Ein Beispiel hierfür wäre eine in ein SupplierDataRequest-Dokument eingebettete Frage nach den Daten eines Lieferanten, der dem Server nicht bekannt ist. Dieser Status kann bedeuten, dass beim Client oder Server Informationen verloren gegangen sind.
500	Internal Server Error	Der Server konnte die Request nicht beenden.
501	Not Implemented	Die jeweilige Request wird vom Server nicht implementiert. Dies ist zum Beispiel der Fall, wenn PunchOutSetupRequest oder der angeforderte Vorgang nicht unterstützt werden. Der Status bedeutet normalerweise, dass der Client das Profil des Servers nicht beachtet hat.

Wenn unerkannte Codes empfangen werden, müssen cXML-Clients diese nach ihrer Klasse behandeln. Ältere Clients sollten somit alle neuen 2*xx*-Codes wie 200, 4*xx*-Codes wie 400 und 5*xx*-Codes wie 500 behandeln. Dieses Verhalten ermöglicht spätere Erweiterungen zum cXML-Protokoll und Änderungen an serverspezifischen Codes, ohne dass es zu Kompatibilitätsproblemen kommt.

Unidirektionales Modell (asynchron)

Im Unterschied zu Anforderung-Antwort-Transaktionen sind unidirektionale Nachrichten nicht an die HTTP-Übertragung gebunden.

Sie sind daher für Situationen geeignet, in denen ein HTTP-Kanal (eine synchrone Anforderung-Antwort-Transaktion) nicht empfehlenswert ist. Die folgende Abbildung zeigt ein Beispiel dafür, wie A und B ohne die Vermittlung über eine Anforderung-Antwort-Transaktion mit Meldungen kommunizieren können.



Unidirektionale Nachricht (asynchron)

Ein möglicher Ablauf in in diesem Fall könnte wie folgt aussehen:

1. A formatiert und codiert ein cXML-Dokument für eine Übertragungsart, die B versteht.
2. A sendet das Dokument in der bekannten Übertragungsart. A wartet nicht aktiv auf eine Rückantwort von B (und kann das auch gar nicht).
3. B erhält das cXML-Dokument und decodiert es aus dem Übertragungs-datenstrom.
4. B verarbeitet das Dokument.

Im unidirektionalen Modell verfügen A und B *nicht* über einen expliziten Anforderung-Antwort-Zyklus. So können zwischen mehreren ankommenden unidirektionalen Nachrichten durchaus auch Nachrichten von anderen Partnern eingehen und weitere Gespräche stattfinden.

Um eine unidirektionale Transaktion vollständig anzugeben, muss das Übertragungsmedium für die Nachricht dokumentiert werden. Für cXML-Transaktionen im unidirektionalen Verfahren werden Übertragungsmedium und Codierung angegeben. Eine verbreitete Transaktion in diesem Modus ist die PunchOutOrderMessage.

Unidirektionale Nachrichten sind analog zum Anforderung-Antwort-Modell strukturiert.

```
<cXML>
    <Header>
        Kopfzeileninformationen…
    </Header>
    <Message>
        Nachrichteninformationen…
    </Message>
</cXML>
```

Das Header-Element wird wie beim Anforderung-Antwort-Modell behandelt. Auch das cXML-Element ist mit dem oben beschriebenen identisch. Die einfachste Möglichkeit, eine unidirektionale Nachricht von einer Anforderung-Antwort-Nachricht zu unterscheiden, stützt sich auf das Message-Element (nicht auf das Request- oder Response-Element). In den folgenden Abschnitten wird das Message-Element ausführlicher behandelt.

Message

Dieses Element enthält sämtliche Angaben auf Haupttextebene in Form einer cXML-Nachricht. Ähnlich wie das Response-Element kann es ein optionales Status-Element enthalten. Es wird in Nachrichten verwendet, die logische Antworten auf eine Anforderungsnachricht darstellen.

Für das Message-Element stehen folgende Attribute zur Verfügung:

deploymentMode (optional)	Gibt an, ob die Anforderung eine Testanforderung oder eine Produktionsanforderung ist. Zulässige Werte sind „Produktion“ (Standardwert) und „Test“.
inReplyTo (optional)	Gibt an, welche Message diese Message beantwortet. Der Inhalt des Attributs inReplyTo wäre die payloadID einer zuvor empfangenen Message. Daraus lässt sich dann ein bidirektionales Gespräch mit vielen Nachrichten aufbauen.

Das Attribut inReplyTo kann auch auf die payloadID eines früher gesendeten Request- oder Response-Dokuments verweisen. Wenn eine Anforderung-Antwort-Transaktion ein „Gespräch“ über mehrere unidirektionale Interaktionen auslöst, kann die erste Nachricht die payloadID des wichtigsten in die jeweils andere Richtung gesendeten Request- oder Response-Dokuments sein. Beispiel: eine Message mit einer PunchOutOrderMessage kann ein Attribut inReplyTo mit der payloadID der PunchOutSetupRequest enthalten, die die Punchoutsitzung gestartet hat. (Der in den Punchoutdokumenten enthaltene BuyerCookie hat eine ähnliche Funktion wie diese Verwendung des inReplyTo-Attributs.)

Transportoptionen

Es gibt zwei gebräuchliche Übertragungsmedien für unidirektionale Nachrichten: HTTP und URL-Formularcodierung. Dies sind nur zwei der gegenwärtig verfügbaren, ausführlich definierten Übertragungsarten. Es ist sehr wahrscheinlich, dass in der Zukunft weitere hinzukommen werden.

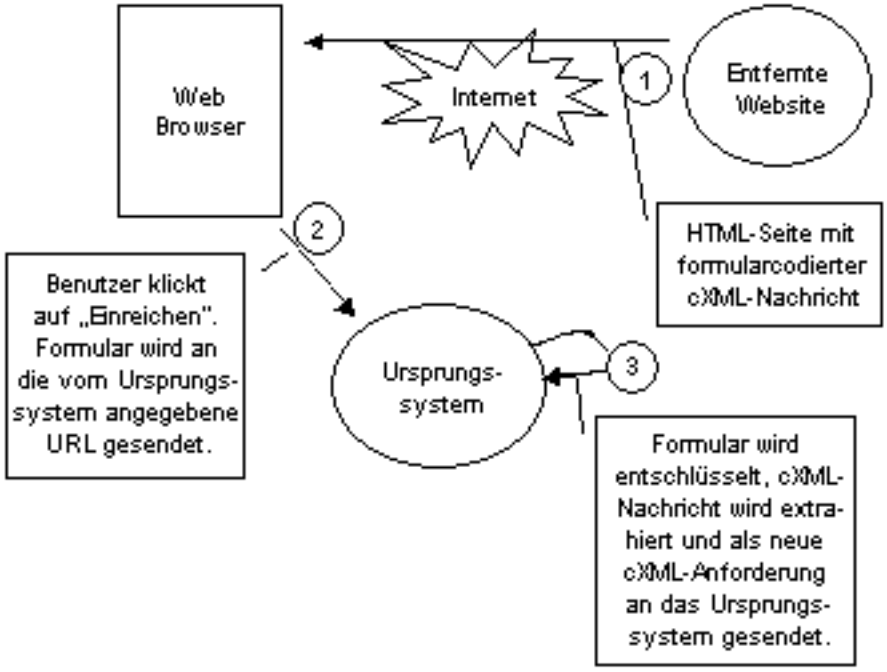
HTTP

HTTP wird für die unidirektionale Kommunikation genutzt, damit Beschaffungsanwendungen die erforderlichen Informationen laden können. Eine der Transaktionen über HTTP ist GetPendingRequest, die auf Seite 102 näher behandelt wird.

URL-Formularcodierung

Dieses Übertragungsverfahren lässt sich am besten anhand der PunchOutOrderMessage-Transaktion veranschaulichen. Die URL-Formularcodierung ermöglicht die Integration zwischen einer entfernten Website und einer Beschaffungsanwendung. Darüber hinaus macht Sie einen direkt über das Internet zugänglichen Server im System des Käufers überflüssig.

Die PunchOutOrderMessage-Nachricht im cXML-Format wird von der entfernten Website nicht direkt an die Beschaffungsanwendung gesendet, sondern als verborgenes HTML-Formularfeld codiert und zu dem im BrowserFormPost-Element von PunchOutSetupRequest angegebenen URL geschickt. Wenn der Benutzer auf **Kasse** klickt, sendet die Website die Daten als HTML-Formular an die Beschaffungsanwendung. Diese Vorgänge werden im nachstehenden Diagramm verdeutlicht:



Nachstehend wird die Semantik des Packens und Entpackens beschrieben.

Packen von Formularen

Das PunchOutOrderMessage-Dokument ist URL-codiert (gemäß der HTTP-Spezifikation) und einem verborgenen Feld in einem Formular mit der Bezeichnung cXML-urlencoded zugeordnet. Dem HTML-Formularelement ist ein VERFAHREN für POST und eine AKTION zugeordnet, die aus dem im BrowserFormPost-Element von PunchOutSetupRequest weitergeleiteten URL besteht. Beispiel:

```
<FORM METHOD=POST
        ACTION="http://workchairs.com:1616/punchoutexit">
    <INPUT TYPE=HIDDEN NAME="cXML-urlencoded"
        VALUE="URL-Encoded PunchOutOrderMessage document">
    <INPUT TYPE=SUBMIT VALUE="Proceed">
</FORM>
```

Weitere HTML-Tags auf der Seite können das dargestellte Fragment enthalten, um den Inhalt des Einkaufskorbes im Detail zu beschreiben.

> **Hinweis:** Wenn Webserver das Feld cXML-urlencoded versenden, ist es noch nicht URL-codiert. Diese Codierung wird nur benötigt, wenn das Formular von Webbrowsern eingereicht wird (wenn Benutzer beispielsweise oben auf **Kasse** klicken). Die Webbrowser selbst erfüllen diese Anforderung. Der Webserver muss nur den Feldwert HTML-codieren und die Anführungszeichen sowie andere Sonderzeichen unberührt lassen, damit das Formular dem Benutzer richtig angezeigt wird.

Bei der Bezeichnung cXML-urlencoded wird zwischen Groß- und Kleinschreibung unterschieden.

Für Daten mit der Codierung cXML-urlencoded darf der Parser keinen anderen charset-Parameter als den Standardzeichensatz für den Medientyp text/xml annehmen.
In einer HTTP POST sind keinerlei Zeichencodierungsangaben für gesendete Daten enthalten. Der Webserver auf der Empfangsseite kann die Codierung der HTML-Seite mit dem verborgenen Feld nicht ermitteln. Das in dieser Form weitergeleitete cXML-Dokument muss daher die Zeichencodierung us-ascii verwenden. Alle im XML-Ursprungsdokument enthaltenen Zeichen (einschließlich „URI encoded“ als „%XX“) müssen den Zeichensatz „us-ascii“ verwenden. Weitere Unicode-Symbole können mit Zeichenentitäten in diesem Ursprungsdokument codiert werden.

Base64-Codierung

Das verborgene Feld cXML-base64 erleichtert die Behandlung von fremdsprachigen Dokumenten. cXML-Dokumente mit Symbolen, die nicht im Zeichensatz „us-ascii“ enthalten sind, sollten dieses Feld dem verborgenen Feld cXML-urlencoded vorziehen. Diese Alternative weist nahezu dieselbe Semantik auf, wobei aber das Dokument über den gesamten Übertragungsweg die base64-Codierung aufweist und somit zum

Browser bzw. Webserver der Empfangsseite nicht HTML- bzw. URL-codiert ist. Die Base64-Codierung ist in RFC 2045, „Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies“ näher beschrieben.

Bei der Bezeichnung cXML-base64 wird zwischen Groß- und Kleinschreibung unterschieden.

Bei der Base64-Codierung von der Lieferantensite über den Browser zum Webserver auf der Clientseite bleibt die ursprüngliche Zeichencodierung des cXML-Dokuments erhalten. Obwohl mit den übermittelten Informationen kein charset-Parameter ankommt, kann das entschlüsselte Dokument (nach Entfernen der Übertragungscodierung) als Medientyp application/xml behandelt werden. Diese Codierung ermöglicht dem Parser auf der Empfangsseite, in der XML-Deklaration angegebene Encoding-Attribute zu berücksichtigen. Die Standardzeichencodierung für dieses Feld ist (wie bei allen application/xml-Dokumenten) UTF-8.

Eines der verborgenen Felder (cXML-urlencoded bzw. cXML-base64) muss in den an die Beschaffungsanwendung gesendeten Daten enthalten sein. Obwohl die Empfänger zuerst nach der Codierung cXML-base64 suchen sollten, wäre eine Übermittlung beider Felder ineffektiv.

Entpacken und Verarbeiten von Formularen

Die Beschaffungsanwendung, die zuvor die richtige URL-Adresse geliefert hat, erhält eine POST-Anweisung (HTML-Formular) mit den beschriebenen Formulardaten. Die POST-Verarbeitung sucht zuerst nach der Variablen cXML-base64, extrahiert deren Wert und entschlüsselt den Base64-Code des Inhalts. Ist dieses Feld bei den Daten nicht vorhanden, sucht die POST-Verarbeitung nach der Variablen für cXML-urlencoded, extrahiert die cXML-Nachricht mit der URL-Codierung und decodiert sie.
Der entschlüsselte Feldinhalt wird dann so verarbeitet wie beim Empfang über einen regulären Anforderung-Antwort-Zyklus.

Der implizierte Medientyp des Dokuments kann nach der Decodierung unterschiedlich sein, wobei verschiedene Zeichencodierungen möglich sind:

- Die Variable cXML-urlencoded gehört zum Medientyp text/xml und verfügt nicht über ein charset-Attribut. Sie ist somit auf die Zeichencodierung us-ascii beschränkt. Der Empfangsparser muss jegliches encoding-Attribut in der XML-Deklaration des cXML-Dokuments ignorieren, weil der Browser die Codierung geändert haben kann.
- Die Variable cXML-base64gehört zum Medientyp application/xml und kann daher eine beliebige Zeichencodierung aufweisen (angegeben durch das encoding-Attribut der eventuell enthaltenen XML-Deklaration).

Der Hauptunterschied zwischen dieser Transaktion und einer normalen Anforderung-Antwort-Transaktion besteht darin, dass keine Antwort erzeugt wird, da keine HTTP-Verbindung besteht, über die die Antwort gesendet werden könnte.

Grundelemente

Die folgenden Entitäten und Elemente werden in der gesamten cXML-Spezifikation verwendet. Bei den meisten Definitionen handelt es sich um Grundvokabular, mit dessen Hilfe Geschäftsunterlagen höherer Ordnung beschrieben werden. Zunächst folgt eine Darstellung allgemeiner Typentitäten und Elemente, die Objekte unterer Ebenen repräsentieren.

Typentitäten

Die Mehrzahl der Definitionen stammt aus der XML-Datennotierungsempfehlung an das World Wide Web Consortium (W3C). Einige wenige Entitäten höherer Ebene, die hier ebenfalls definiert sind, entstammen nicht den XML-Daten. Diese Typen werden im Abschnitt „cXML-Container“ auf Seite 50 behandelt.

isoLangCode

ISO-Sprachencode aus der ISO-Norm 639.

isoCountryCode

ISO-Ländercode aus der ISO-Norm 3166.

xmlLangCode

Sprachencode nach der XML 1.0-Spezifikation (unter www.w3.org/TR/1998/REC-xml-19980210.html). Im Normalfall ist dies ein Sprachencode nach ISO 639 und (optional) ein Ländercode nach ISO 3166, wobei beide Codes durch einen Bindestrich getrennt sind. Im Unterschied zur vollständigen XML-Empfehlung dürfen IANA-Codes oder private Sprachencodes nicht in cXML verwendet werden. IANA-Codes und private Untercodes sind zulässig, sollten aber hinter einem gültigen Ländercode nach ISO 3166 stehen.

Das empfohlene cXML-Sprachencodeformat lautet xx[-YY[-zzz]*]?, wobei xx ein Sprachencode nach ISO 639, YY ein Ländercode nach ISO 3166 und zzz ein IANA- oder privater Untercode für die jeweilige Sprache ist. Die Verwendung des Ländercodes wird in jedem Fall empfohlen. Der Konvention nach steht der Sprachencode in Kleinbuchstaben und der Ländercode in Großbuchstaben, wobei dies für die richtige Zuordnung der Codes nicht entscheidend ist.

unitOfMeasure

UnitOfMeasure beschreibt, in welcher Form das Produkt verpackt oder versandt wird. Dieses Element muss den Standard-Maßeinheitencodes UN/CEFACT entsprechen. Eine Liste der UN/CEFACT-Codes finden Sie unter www.unece.org/cefact.

URL

URL (Uniform Resource Locator) nach der Definition in der HTTP/1.1-Norm.

Grundelemente

Diese Elemente, die in der ganzen Spezifikation verwendet werden, reichen von allgemeinen, wie Name und Extrinsic, zu speziellen, wie Money.

Profiltransaktion

Die Dokumente ProfileRequest und ProfileResponse müssen von cXML 1.1-Server-implementierungen unterstützt werden. Mit dieser Transaktion rufen Sie Server-funktionen auf, darunter die unterstützte cXML-Version, Transaktionen und Optionen für diese Transaktionen.

In der Antwort sollten alle auf einer bestimmten Website unterstützten Anforderungen aufgelistet werden, jedoch nicht unbedingt alle, die vom Unternehmen unterstützt werden. Lieferanten, die OrderRequest-Dokumente empfangen und verschiedene Nachrichten senden oder Anforderung-Antwort-Transaktionen starten können, beschreiben in der Profiltransaktion, welche OrderRequest-Dokumente sie unterstützen.

Mit der Profiltransaktion können Sie innerhalb des cXML-Protokolls auch einen Ping-Befehl an den Server absenden.

ProfileRequest

Dieses Element hat keinen Inhalt. Es wird einfach mit der Kopfzeile zum entsprechenden cXML-Server geleitet. Der Server antwortet mit einer einzelnen ProfileResponse-Meldung, die im Folgenden näher beschrieben wird. Die einzigen dynamischen Teile dieser Antwort sind die Attribute payloadId und timestamp des cXML-Elements selbst. In diesem speziellen Fall muss der Lieferant keine Antworten mit verschiedenen Gebietsangaben senden.

Beispiel für ein Request-Dokument dieses Typs:

```
<Request>
    <ProfileRequest />
</Request>
```

ProfileResponse

Dieses Element enthält eine Liste der unterstützten Transaktionen mit deren Positionen und eventuell unterstützten Optionen. Obwohl noch keine Optionen definiert sind, ist Folgendes ein denkbares ProfileResponse-Dokument:

```
<ProfileResponse effectiveDate="2001-03-03T12:13:14-05:00">
    <Option name="Locale">1</Option>
    …
    <Transaction requestName="PunchOutSetupRequest">
        <URL>http://www.workchairs.com/cXML/PunchOut.asp</URL>
        <Option name="operationAllowed">create inspect</Option>
        <Option name="dynamic pricing">0</Option>
        …
    </Transaction>
    …
</ProfileResponse>
```

Ein ProfileResponse-Dokument von einem aktuellen Lieferanten könnte mit einiger Wahrscheinlichkeit wie folgt aussehen:

```
<ProfileResponse effectiveDate="2000-01-01T05:24:29-08:00">
    <Transaction requestName="OrderRequest">
        <URL>http://workchairs.com/cgi/orders.cgi</URL>
    </Transaction>
    <Transaction requestName="PunchOutSetupRequest">
        <URL>http://workchairs.com/cgi/PunchOut.cgi</URL>
    </Transaction>
</ProfileResponse>
```

Für das ProfileResponse-Element steht folgendes Attribut zur Verfügung:

effectiveDate	Datum und Uhrzeit des Beginns des Leistungsangebots. Die Datumsangaben dürfen nicht in der Zukunft liegen.

Option

Wert für eine definierte Option (entweder für den gesamten Dienst oder eine bestimmte Transaktion). Es sind noch keine Optionen definiert.

Für das Option-Element stehen folgende Attribute zur Verfügung:

name	Name der Option. Dieses Attribut sollte nicht direkt angezeigt werden, da das Profil für die Verarbeitung durch das System gedacht ist.

Transaction

Beschreibung einer von diesem Dienst unterstützten Transaktion. Gegenwärtig wird durch die Profildefinition angegeben, wohin bestimmte Anforderungen zu senden sind. Künftige Versionen von cXML werden weitere Definitionen von Option enthalten und ein erweitertes Profil mit detaillierteren Angaben zu den unterstützten Anforderungen aufweisen.

Jedes Transaction-Element muss ein URL-Element enthalten.

Für das Transaction-Element stehen folgende Attribute zur Verfügung:

requestName	Anforderung, die dieser Server an einer bestimmten URL akzeptiert. Folgende Werte sind möglich: ProfileRequest PunchOutSetupRequest GetPendingRequest SupplierListRequest SupplierDataRequest OrderRequest StatusUpdateRequest SubscriptionListRequest SubscriptionContentRequest

Auftragsdefinitionen

Die cXML-Auftragsdokumente sind OrderRequest und eine allgemeine Antwort. OrderRequest entspricht prinzipiell einem Bestellauftrag, und die Antwort ist eine Eingangsbestätigung des Lieferanten für den Bestellauftrag. Sie stellt an sich noch keine Verpflichtung dar, den Bestellauftrag auszuführen, bestätigt jedoch, dass dieser ordnungsgemäß eingegangen ist.

OrderRequest

Das folgende Beispiel veranschaulicht die Struktur des OrderRequest-Elements:

```
<OrderRequest>
    <OrderRequestHeader … >
        …
    </OrderRequestHeader>
    <ItemOut … >
        …
    </ItemOut>
    <ItemOut … >
        …
    </ItemOut>
</OrderRequest>
```

OrderRequestHeader

Das folgende Beispiel zeigt das OrderRequestHeader-Element in aller Ausführlichkeit:

```
<OrderRequestHeader orderID="DO1234"
        orderDate="1999-03-12T13:30:23+8.00"
        type="new"
        requisitionID="R1234">
    <Total>
        <Money currency="USD">12.34</Money>
    </Total>
    <ShipTo>
        <Address>
            <Name xml:lang="en">Acme Corporation</Name>
            <PostalAddress name="Headquarters">
                <DeliverTo>Joe Smith</DeliverTo>
                <DeliverTo>Mailstop M-543</DeliverTo>
                <Street>123 Anystreet</Street>
                <City>Sunnyvale</City>
                <State>CA</State>
                <PostalCode>90489</PostalCode>
                <Country isoCountryCode="US">USA</Country>
            </PostalAddress>
```

```
            </Address>
        </ShipTo>
        <BillTo>
            <Address>
                <Name xml:lang="en">Acme Corporation</Name>
                <PostalAddress name="Finance Building">
                    <Street>124 Anystreet</Street>
                    <City>Sunnyvale</City>
                    <State>CA</State>
                    <PostalCode>90489</PostalCode>
                    <Country isoCountryCode="US">USA</Country>
                </PostalAddress>
            </Address>
        </BillTo>
        <Shipping>
            <Money currency="USD">12.34</Money>
            <Description xml:lang="en-US">FedEx 2-day</Description>
        </Shipping>
        <Tax>
            <Money currency="USD">12.34</Money>
            <Description xml:lang="en">CA State Tax</Description>
        </Tax>
        <Payment>
            <PCard number="1234567890123456" expiration="1999-03-12"/>
        </Payment>
        <Contact role="purchasingAgent">
            <Name xml:lang="en-US">Mr. Smart E. Pants</Name>
            <Email>sepants@acme.com</Email>
            <Phone name="Office">
                <TelephoneNumber>
                    <CountryCode isoCountryCode="US">1</CountryCode>
                    <AreaOrCityCode>800</AreaOrCityCode>
                    <Number>555-1212</Number>
                </TelephoneNumber>
            </Phone>
        </Contact>
        <Comments xml:lang="en-US">
            Jeder korrekt in XML-Syntax geformte Ausdruck kann hier stehen.
        </Comments>
        <Followup>
            <URL>http://acme.com/cgi/orders.cgi</URL>
        </Followup>
    </OrderRequestHeader>
```

Für das OrderRequestHeader-Element stehen folgende Attribute zur Verfügung:

orderID	Bezeichner für diesen Auftrag. Entspricht der Bestellauftragsnummer.
orderDate	Bestellauftragsdatum und -uhrzeit im ISO 8601-Format.
type (optional)	Anforderungsart (new (Standardwert), update oder delete)
requisitionID (optional)	Anforderungs-ID des Käufers für den gesamten Auftrag. Kann derselbe Bezeichner wie orderID sein oder auch ganz fehlen. Braucht nicht angegeben zu werden, wenn requisitionID in einem der ItemOut-Elemente angegeben ist.
shipComplete (optional)	Voreinstellung, durch die Teillieferungen verhindert werden. Der einzig zulässige Wert ist „yes“. Ohne dieses Attribut werden die Artikel standardmäßig sofort versandt, sobald sie verfügbar werden. Da Aufträge Artikel mit verschiedenen ShipTo-Elementen enthalten können, werden bei shipComplete="yes" nur Artikelgruppen mit gemeinsamem Versandort aufgehalten, bis die Lieferung komplett ist.

OrderRequestHeader und ItemOut (wenn durch ItemDetail erweitert) enthalten im Wesentlichen die gleichen Angaben. Während OrderRequestHeader Angaben zur Gesamtrechnungslegung (BillTo) und Zahlung (Payment) macht, beschreibt ItemOut an dieser Stelle die einzelnen Artikel (in ItemID, ItemDetail und Distribution).

Verwenden Sie die Angaben in OrderRequestHeader nicht als Vorschlagswert für artikelspezifische Elemente. Wenn vorhanden, müssen ShipTo, Shipping, Contact und alle benannten Extrinsic-Elemente entweder bei jedem ItemOut-Element oder im OrderRequestHeader stehen. Die Elemente Comments und Tax können auf beiden Ebenen gleichzeitig auftreten. Doch sollten die verschiedenen Comments-Elemente keine Informationen doppelt enthalten. Das Tax-Element auf Kopfzeilenebene enthält eine Gesamtsumme für den Auftrag.

Total

Dieses Element enthält die Gesamtauftragssumme. Es dient als Container für das Money-Element.

ShipTo/BillTo

Diese Elemente enthalten die Angaben zu den Entitäten Lieferadresse und Rechnungsadresse aus OrderRequest.

Da jeder Auftrag auf eine einzige Entität gebucht werden muss, tritt das BillTo-Element nur im Kopfzeilenelement OrderRequestHeader auf. Die Artikel eines Auftrags können an mehrere Orte versandt werden. Wie das Element Shipping (siehe nächsten Abschnitt), kann das Element ShipTo daher sowohl im OrderRequestHeader als auch in einzelnen ItemOut-Elementen auftreten.

Shipping

Dieses Element beschreibt, wie die Artikel im Auftrag zu versenden sind und welche Kosten dabei entstehen. Wenn das Shipping-Element im OrderRequestHeader-Element auftritt, darf es nicht auch in einzelnen ItemOut-Elementen stehen. Tritt es nicht im OrderRequestHeader auf, dann muss es in den ItemOut-Elementen vorkommen.

Tax

Dieses Element enthält auf den Auftrag erhobene Steuern. Es ist vorhanden, wenn die Käuferorganisation Steuern errechnet. Tritt es im OrderRequestHeader auf, gibt das Tax-Element die Gesamtsteuersumme eines Auftrags an. Tax-Elemente auf Artikelebene geben dagegen Einzelsteuerbeträge an.

Payment

Dieses Element beschreibt das zur Bezahlung der angeforderten Artikel verwendete Zahlungsmittel. Im dargestellten Beispiel enthält das Element Payment ein PCard-Element, das im cXML-Dokument für eine Standardkundenkarte steht. Zukünftig sollen weitere Zahlungsmöglichkeiten definiert und unterstützt werden.

Contact

Angaben zu Kontaktpersonen, die der Lieferant bei der Nachbereitung eines Auftrags verwenden kann. Dieses Element kennzeichnet eine Person und gibt eine Reihe von Möglichkeiten der Kontaktaufnahme mit dieser Person oder Entität an. Das einzige obligatorische Element ist der Name der Kontaktperson. Optionale und sich wiederholende Möglichkeiten sind PostalAddress (nicht empfohlen zur sofortigen Lösung von Auftragsproblemen), Email, Phone, Fax und URL.

Käuferorganisationen wählen dieses Element unter Umständen zur Kennzeichnung des ursprünglichen Anforderers, des Systemadministrators für die Beschaffungsanwendung oder einer anderen Kontaktperson, die zur Lösung von Auftragsproblemen befugt ist. Contact kann andere Angaben enthalten als die Elemente BillTo und ShipTo eines Auftrags.

Für das Contact-Element stehen folgende Attribute zur Verfügung:

role (optional)	Stellung dieser Person im Beschaffungsprozess. Mögliche Werte sind endUser, administrator, purchasingAgent, technicalSupport, customerService und sales.

Die Verwendung derselben Contact role-Werte in der Kopfzeile und auf Artikelebene ist unzulässig.

Wegen der verschiedenen Inhaltsoptionen für das Contact-Element ist keine Vorschlagsrolle definiert. Daher behandeln cXML-Anwendungen ein Contact-Element ohne role-Attribut als zusätzliche Rolle.

Comments

Beliebige für Menschen lesbare Angaben, die Käufer mit ihren Bestellaufträgen senden können. Diese Zeichenketten sind nicht für automatische Systeme auf Lieferantensites gedacht.

Das Element Comments kann ein Attachment-Element zum Einbinden externer Dateien enthalten.

Attachment

An Comments können externe Dateien angehängt werden, um Bestellaufträge mit zusätzlichen Informationen zu versehen. Das Attachment-Element tritt innerhalb von Comments auf und enthält nur einen Verweis auf den externen MIME-Teil der Anlage. Alle Anlagen müssen in einer einzigen mehrteiligen Übertragung mit dem Dokument OrderRequest gesendet werden. Selbst wenn dies nicht möglich ist, muss die mit dem Attachment-Element übermittelte contentID ein Aufrufen der Anlage ermöglichen.

Weitere Informationen zum Übertragen von Dateien als Anlagen finden Sie im Abschnitt „Übertragen von Anlagen“ auf Seite 53.

Attachment enthält einen einzigen URL mit dem Schema „cid:“. Eine Datei als Anlage zu einem cXML-Dokument kann wie folgt aussehen:

```
<Comments>
    <Attachment>
        <URL>cid: uniqueCID@cxml.org</URL>
    </Attachment>
    Das Bild in der Anlage gibt Ihnen eine Vorstellung davon, wie wir
    uns die Ausführung denken.
    </Comments>
```

Das Comments-Element tritt im cXML-Protokoll an vielen Stellen auf, kann aber nur in OrderRequest-Dokumenten ein Attachment-Element enthalten.

Followup

Gibt den URL an, an den nachfolgende StatusUpdateRequest-Dokumente gesendet werden sollen. Dies ist der Eingangsort für alle späteren Dokumente, die sich auf das aktuelle OrderRequest-Dokument beziehen.

Extrinsic

Dieses Element enthält maschinenlesbare Angaben zum Auftrag, die jedoch nicht im cXML-Protokoll definiert sind. Im Gegensatz dazu gibt das Comments-Element Informationen für Bearbeiter weiter. Extrinsic-Elemente enthalten Angaben, die höchstwahrscheinlich in späteren Dokumenten auftreten, während das Comments-Element keine solchen Angaben enthält. Auf dieser Ebene erweitert Extrinsic die Beschreibung aller im Bestellauftrag enthaltenen Artikel. Einige Extrinsic-Angaben können auch den gesamten Bestellauftrag beschreiben, ohne die Bedeutung eventuell enthaltener ItemOut-Elemente zu beeinflussen.

Jedes benannte Extrinsic-Element darf in den zu OrderRequestHeader und einzelnen ItemOut-Elementen (innerhalb der enthaltenen ItemDetail-Elemente) gehörenden Listen nur einmal auftreten. Kein Name darf sowohl in den Listen des OrderRequestHeader-Elements als auch in der Liste eines einzelnen ItemOut Elementen stehen. Wenn derselbe Extrinsic-Name und Wert in allen ItemOut-Listen vorkommt, sollte er in das OrderRequestHeader-Element verschoben werden.

Das Extrinsic-Element kann außerdem in den Elementen IndexItem, PunchOutSetupRequest und ContractItem stehen. Diese Kontexte werden an anderer Stelle in diesem Dokument beschrieben.

ItemOut

Das folgende Beispiel zeigt ein gültiges ItemOut-Element in Minimalversion.

```
<ItemOut quantity="1">
    <ItemID>
        <SupplierPartID>5555</SupplierPartID>
    </ItemID>
</ItemOut>
```

Für das ItemOut-Element stehen folgende Attribute zur Verfügung:

quantity	Anzahl der gewünschten Artikel. Bei einigen Maßeinheiten sind gebrochene Zahlen zulässig. Der Wert kann bereits vom Lieferanten während einer Punchoutsitzung geprüft worden sein. Negative Werte sind ungültig.
lineNumber (optional)	Position dieses Artikels in einem Auftrag. Dieser Ordnungszahlenwert erhöht sich einmal pro ItemOut in einem „neuen“ OrderRequest-Dokument. Kunden sollten dieses Attribut in einem OrderRequest-Dokument stets angeben, obwohl es in anderen ItemOut-Kontexten möglicherweise nicht sinnvoll ist.
requisitionID (optional)	Anforderungskennzeichnung des Käufers für diese Position. Muss hier nicht angegeben werden, wenn die requisitionID im OrderRequestHeader steht.
requestedDeliveryDate (optional)	Datum, an dem die Lieferung des Artikels angefordert wurde, wodurch in OrderRequest Lieferdaten auf Artikelebene möglich werden. Das Datum muss im ISO 8601-Format angegeben werden.

Das Attribut lineNumber bleibt bei allen Auftragsaktualisierungen konstant. Das Löschen von Artikeln aus einem Auftrag hat keine Änderung der lineNumber-Attribute der übrigen Artikel zur Folge. Neue Artikel haben höhere Nummern als die zuvor in den Auftrag eingetragenen. Änderungen an einem vorhandenen Artikel (z. B. eine Erhöhung der Bestellmenge) haben keine Auswirkung auf die lineNumber dieses Artikels.

Das folgende Beispiel zeigt ein komplexeres ItemOut-Element.

```
<ItemOut quantity="2" lineNumber="1"
    equestedDeliveryDate="1999-03-12">
    <ItemID>
        <SupplierPartID>1233244</SupplierPartID>
        <SupplierPartAuxiliaryID>ABC</SupplierPartAuxiliaryID>
    </ItemID>
    <ItemDetail>
        <UnitPrice>
            <Money currency="USD">1.34</Money>
        </UnitPrice>
        <Description xml:lang="en">hello</Description>
        <UnitOfMeasure>EA</UnitOfMeasure>
        <Classification domain="SPSC">12345</Classification>
        <ManufacturerPartID>234</ManufacturerPartID>
        <ManufacturerName xml:lang="en">foobar</ManufacturerName>
        <URL>www.bar.com</URL>
    </ItemDetail>
    <ShipTo>
        <Address>
            <Name xml:lang="en">Acme Corporation</Name>
            <PostalAddress name="Headquarters">
```

```
                        <Street>123 Anystreet</Street>
                        <City>Sunnyvale</City>
                        <State>CA</State>
                        <PostalCode>90489</PostalCode>
                        <Country isoCountryCode="US">USA</Country>
                    </PostalAddress>
                </Address>
            </ShipTo>
            <Shipping>
                <Money currency="USD">1.34</Money>
                <Description xml:lang="en-US">FedEx 2-day</Description>
            </Shipping>
            <Tax>
                <Money currency="USD">1.34</Money>
                <Description xml:lang="en">foo</Description>
            </Tax>
            <Distribution>
                <Accounting name="DistributionCharge">
                    <Segment type="G/L Account" id="23456"
                        description="Entertainment"/>
                    <Segment type="Cost Center" id="2323"
                        description="Western Region Sales"/>
                </Accounting>
                <Charge>
                    <Money currency="USD">.34</Money>
                </Charge>
            </Distribution>
            <Distribution>
                <Accounting name="DistributionCharge">
                    <Segment type="G/L Account" id="456"
                        description="Travel"/>
                    <Segment type="Cost Center" id="23"
                        description="Europe Implementation"/>
                </Accounting>
                <Charge>
                    <Money currency="USD">1</Money>
                </Charge>
            </Distribution>
            <Comments xml:lang="en-US">
                An dieser Stelle kann ein beliebiger gültiger XML-Ausdruck stehen.
            </Comments>
        </ItemOut>
```

Das ItemDetail-Element ermöglicht das Senden zusätzlicher Daten an den Lieferanten statt einfach nur die eindeutige Kennzeichnung des Artikels, wie sie durch ItemID repräsentiert wird.

Die Elemente ShipTo, Shipping, Tax, Contact, Comments und Extrinsic (einige davon eingebettet in ItemDetail) sind identisch mit denen, die im OrderRequestHeader stehen können. Diese Elemente gestatten Datenangaben auf Artikelebene zu Versand, Versandart und damit verbundenen Kosten. Verwenden Sie diese Elemente entweder auf OrderRequestHeader-Ebene oder auf ItemOut-Ebene, jedoch nicht auf beiden Ebenen gleichzeitig.

Distribution

Hierbei geht es um die Aufteilung der Kosten auf mehrere Parteien. Lieferanten senden das Element Distribution auf Rechnungen zurück, um die Abstimmung beim Käufer zu ermöglichen.

Accounting

Das Accounting-Element gruppiert Segments, um den Rechnungsempfänger zu ermitteln.

Für das Accounting-Element stehen folgende Attribute zur Verfügung:

name	Name dieser Buchführungskombination.

Für das Segment-Element stehen folgende Attribute zur Verfügung:

type	Bezeichnung für dieses Segment in Bezug auf andere Segmente im Accounting-Element.
id	Eindeutiger Bezeichner innerhalb dieses Segment-Typs. Dieser Wert kann die eigentliche Kontonummer darstellen, wenn das Attribut „Cost Center“ lautet.

Charge

Dieses Element gibt den Betrag auf an, der auf die durch das Accounting-Element repräsentierte Entität gebucht werden soll.

Antwort auf eine OrderRequest

Dies ist der Antwortteil der synchronen Anforderung-Antwort-Transaktion. Das folgende Beispiel zeigt ein Response-Dokument als Antwort auf ein OrderRequest-Dokument:

```
<cXML version="1.1.007" payloadID="9949494" xml:lang="en"
        timestamp="1999-03-12T18:39:09-08:00">
    <Response>
        <Status code="200" text="OK"/>
    </Response>
</cXML>
```

Wie ersichtlich, ist dieses Response-Dokument recht unkompliziert aufgebaut. Da sich die einzigen Daten, die im Beispiel an den Anforderer zurückgeschickt werden müssen, im Status-Teil von Response befinden, gibt es auch kein eigentliches „OrderResponse“-Element.

Das Response-Dokument informiert den Anforderer darüber, dass dessen OrderRequest erfolgreich geparst und vom entfernten Ende der HTTP-Verbindung verarbeitet wurde. Es stellt keine Auftragsbestätigung in dem Sinne dar, dass es angibt, welche Artikel lieferbar sind und welche nachbestellt werden müssen.

Punchouttransaktion

Bei den Punchoutnachrichtendefinitionen handelt es sich um Anforderung-Antwort-Nachrichten, die in den Elementen Request und Response übertragen werden. Lieferanten müssen alle nachfolgend genannten Nachrichten implementieren, wenn Punchout unterstützt werden soll.

PunchOutSetupRequest

PunchOutSetupRequest und PunchOutSetupResponse bilden das Anforderung-Antwort-Paar, mit dem eine Punchoutsitzung zu einem entfernten System eingerichtet wird. Der Client erkennt mit diesen Elementen die Beschaffungsanwendung, sendet Setupinformationen und erhält Informationen dazu, auf welcher entfernten Website eine HTML-Sitzung gestartet werden soll.

Ein PunchOutSetupRequest-Element wird im Request-Element übertragen. Das folgende Beispiel zeigt ein PunchOutSetupRequest-Element.

```
<PunchOutSetupRequest operation="create">
    <BuyerCookie>34234234ADFSDF234234</BuyerCookie>
    <Extrinsic name="department">Marketing</Extrinsic>
    <BrowserFormPost>
        <URL>http://orms.acme.com:1616/punchoutexit</URL>
    </BrowserFormPost>
    <SelectedItem>
        <ItemID>
            <SupplierPartID>54543</SupplierPartID>
        </ItemID>
    </SelectedItem>
    <SupplierSetup>
        <URL>http://workchairs.com/cxml</URL>
    </SupplierSetup>
</PunchOutSetupRequest>
```

Für das PunchOutSetupRequest-Element stehen folgende Attribute zur Verfügung:

operation	Gibt den Typ des PunchOutSetupRequest-Elements an: „create“, „inspect“, oder „edit“.

Dieses Element enthält folgende Elemente: BuyerCookie, Extrinsic, BrowserFormPost, Contact, ShipTo, SelectedItem, SupplierSetup und eine ItemOut-Liste. Nur das Element BuyerCookie ist obligatorisch. Die Struktur der Elemente Extrinsic, Contact und ShipTo wird ausführlicher im Abschnitt „OrderRequestHeader“ auf Seite 70 behandelt. Informationen zum ItemOut-Element finden Sie im Abschnitt „ItemOut“ auf Seite 75. In diesem Kontext (außerhalb einer OrderRequest) tragen die Elemente Distribution und Comments sowie die Attribute lineNumber, requisitionID und requestedDeliveryDate von ItemOut kaum wichtige Informationen bei und sollten weggelassen werden. Da die Punchoutsitzung vor der eigentlichen Bestellung stattfindet, sind diese Angaben in einem PunchOutSetupRequest-Dokument irrelevant.

Eine ItemOut-Liste beschreibt einen vorhandenen Einkaufswagen (Artikel einer früheren Punchoutsitzung). Der inspect-Vorgang startet eine schreibgeschützte Punchoutsitzung (sowohl vom Client als auch vom Server durchgesetzt) zum Anzeigen von Details der aufgeführten Artikel. Der edit-Vorgang wird auch von einem früheren Einkaufswagen aus gestartet (beschrieben durch die ItemOut-Liste), lässt jedoch Änderungen zu. Unterstützung des edit-Vorgangs impliziert inspect-Unterstützung. (Weitere Informationen hierzu finden Sie in den Abschnitten „PunchOutOrderMessageHeader“ auf Seite 83 und „Leere Einkaufswagen“ auf Seite 84.)

BuyerCookie

Dieses Element sendet für die entfernte Website nicht transparente Angaben, muss jedoch für alle späteren Punchoutsitzungen an den Absender zurückgeschickt werden. Mithilfe dieses Elements kann die Beschaffungsanwendung mehrere ausstehende Punchoutanforderungen zuordnen.

BrowserFormPost

Dieses Element stellt das Ziel für die Daten im PunchOutOrderMessage-Dokument dar. Es enthält ein URL-Element, dessen Verwendung in der Definition der PunchOutOrderMessage näher erläutert wird. Dieses Element kann ausgelassen werden, wenn an Stelle der URL-Formularcodierung ein anderes Verfahren verwendet wird.

Extrinsic

Dieses optionale Element enthält sämtliche zusätzlichen Daten, die der Anforderer an die externe Website übertragen möchte. Im Beispiel passiert das Element die Abteilung des Benutzers, der den Punchoutvorgang gestartet hat. Da der Inhalt von Extrinsic-Elementen durch die cXML-Spezifikation nicht vorgegeben wird, muss er zwischen dem Anforderer und der entfernten Website individuell vereinbart und entsprechend implementiert werden.

Extrinsic-Elemente sollen zusätzliche maschinenlesbare Informationen übermitteln. Sie erweitern das cXML-Protokoll zur Unterstützung von Funktionen, die nicht in allen Implementierungen benötigt werden. Im vorliegenden Kontext enthalten die neuen Daten eine nähere Beschreibung des Benutzers, der die Punchoutanforderung gestartet hat.

Das Extrinsic-Element kann außerdem in den Elementen OrderRequestHeader, ItemDetail und ContractItem stehen. Diese Kontexte werden an anderer Stelle in diesem Dokument beschrieben.

SelectedItem

Dieses optionale Element gibt die Artikel an, die die Benutzer unter Verwendung der Punchoutfunktion kaufen möchten. Es enthält eine einzige obligatorische ItemID, durch die die Benutzer von ihrem lokalen Katalog auf die Website des Lieferanten geführt werden.

Dieses Element ist normalerweise in create-Vorgängen vorhanden. Bei Beschaffungsanwendungen, die den Benutzern einen direkten Punchout von einer Lieferantenliste gestatten, sollte auf das SelectedItem-Element verzichtet werden.

Bei edit- und inspect-Vorgängen sollte das SelectedItem-Element nur dann auftreten, wenn der Benutzer nach Anzeige der Artikel im lokalen Katalog (nicht in einer vorhandenen Anforderung) zur Lieferantenwebsite zurückkehren möchte. In jedem Fall muss der aktuelle Einkaufswagen in der ItemOut-Liste stehen.

Lieferanten können ihre Kataloge so gestalten, dass das SelectedItem-Element zu einem Punchout auf Lager-, Gang- oder Produktebene führt. Je genauer ein Artikel im Katalog beschrieben ist, desto weniger müssen die Benutzer auf der Website des Lieferanten danach suchen.

SupplierSetup

Dieses optionale Element gibt die URL-Adresse an, an die das PunchOutSetupRequest-Dokument gesendet werden soll. Wenn der E-Commerce-Netzknoten den Punchout-URL des Lieferanten kennt, wird das Element nicht benötigt.

PunchOutSetupResponse

Nach Erhalt eines PunchOutSetupRequest-Dokuments antwortet die entfernte Webseite mit einem PunchOutSetupResponse-Dokument. Beispiel:

```
<PunchOutSetupResponse>
    <StartPage>
        <URL>
            http://premier.workchairs.com/store?23423SDFSDF23
        </URL>
    </StartPage>
</PunchOutSetupResponse>
```

StartPage

Dieses Element enthält ein URL-Element, das wiederum den für die Punchoutsitzung im PunchOutSetupRequest-Dokument angeforderten URL definiert, den an den Browser weitergegeben werden soll. Dieser URL muss über ausreichend Statusinformationen verfügen, um die Verbindung zu einem Sitzungskontext auf der entfernten Website herstellen zu können. Dies betrifft beispielsweise die Identität des Anforderers und das richtige BuyerCookie-Element.

Der Benutzer, durch den das PunchOutSetupRequest-Dokument gestartet wurde, durchsucht daraufhin die entfernte Website und wählt Artikel aus, die mit einer PunchOutOrderMessage an die Beschaffungsanwendung zurückgeschickt werden sollen.

PunchOutOrderMessage

Dieses Element sendet den Inhalt des entfernten Einkaufskorbes an den Absender einer PunchOutSetupMessage. Es kann wesentlich mehr Daten als andere Nachrichten enthalten, da es in der Lage sein muss, den Inhalt jedes nur denkbaren Einkaufskorbes auf der externen Website vollständig angeben zu können. Die Nachricht folgt dem Anforderung-Antwort-Modell daher nur bedingt.

Wenn der Benutzer zur Kasse geht, erstellt die entfernte Website eine PunchOutOrderMessage-Nachricht, durch die der Inhalt des entfernten Einkaufskorbes an die Beschaffungsanwendung übermittelt wird.

Beispiel:

```
<PunchOutOrderMessage>
    <BuyerCookie>34234234ADFSDF234234</BuyerCookie>
    <PunchOutOrderMessageHeader operationAllowed="create">
        <Total>
            <Money currency="USD">100.23</Money>
        </Total>
    </PunchOutOrderMessageHeader>
    <ItemIn quantity="1">
        <ItemID>
            <SupplierPartID>1234</SupplierPartID>
            <SupplierPartAuxiliaryID>
                zusätzliche Angaben zu diesem Artikel
            </SupplierPartAuxiliaryID>
        </ItemID>
        <ItemDetail>
            <UnitPrice>
                <Money currency="USD">10.23</Money>
            </UnitPrice>
            <Description xml:lang="en">
                Learn ASP in a Week!
            </Description>
            <UnitOfMeasure>EA</UnitOfMeasure>
            <Classification domain="SPSC">12345</Classification>
        </ItemDetail>
    </ItemIn>
</PunchOutOrderMessage>
```

Die folgenden Abschnitte behandeln die im Beispiel genannten Elemente.

BuyerCookie

Hierbei handelt es sich um dasselbe Element, das im ursprünglichen PunchOutSetupRequest-Dokument übermittelt wurde. Es muss zurückgesendet werden, damit die Beschaffungsanwendung die PunchOutOrderMessage-Nachricht einem früheren PunchOutSetupRequest-Dokument zuordnen kann.

PunchOutOrderMessageHeader

Dieses Element enthält Angaben zum gesamten übermittelten Inhalt des Einkaufskorbes. Das einzige obligatorische Element ist Total, die Gesamtkosten der der Anforderung hinzugefügten Artikel. Weitere zulässige Elemente sind Shipping und Tax, die Betrag und Beschreibung für auf der entfernten Website errechnete Versandkosten und Steuern darstellen. ShipTo ist ebenfalls optional und gibt die Lieferadresse an, die der Benutzer am entfernten Ort gewählt hat bzw. die mit dem ursprünglichen PunchOutSetupRequest-Dokument übermittelt wurde. Alle Geldbeträge befinden sich im Money-Element, das stets die Währung in einem Standardformat angibt.

Für das PunchOutOrderMessageHeader-Element stehen folgende Attribute zur Verfügung:

operationAllowed	Gibt die zulässigen PunchOutSetupRequest-Vorgänge an: create, inspect oder edit.

Dieses Attribut regelt, ob der Benutzer spätere PunchOutSetupRequest-Transaktionen starten kann, die Daten seiner PunchOutOrderMessage enthalten. Bei operationAllowed="create" können diese Artikel nur in einer späteren OrderRequest enthalten sein. Andernfalls kann die Beschaffungsanwendung den Einkaufswagen später prüfen oder bearbeiten (und nachfolgende PunchOutSetupRequest-Transaktionen mit den entsprechenden Operationen und ItemID-Elementen gemäß der in dieser PunchOutOrderMessage zurückgesendeten ItemIn-Liste starten). Unterstützung für die edit-Operation schließt Unterstützung für inspect ein. Die Beschaffungsanwendung kann die Artikel immer in einem nachfolgenden OrderRequest-Element verwenden.

Leere Einkaufswagen

Die PunchOutOrderMessage-Nachricht kann eine Artikelliste enthalten, die einem Einkaufswagen auf einer Lieferantenwebsite entspricht. Sie zeigt stets das Ende der interaktiven Punchoutsitzung an. In den nachstehenden Absätzen sind einige Fälle aufgeführt, bei denen die PunchOutOrderMessage-Nachricht keine Artikelliste enthält. Dadurch können die Clients, sofort fortfahren, wenn der Benutzer die Lieferantenwebsite verlässt.

Wenn der Vorgang im ursprünglichen PunchOutSetupRequest-Dokument inspect lautete, muss die Artikelliste der PunchOutOrderMessage von der Beschaffungsanwendung ignoriert werden. In diesem Fall darf die Lieferantensite keine ItemIn-Elemente zurückgeben. Wenn eine PunchOutOrderMessage keine ItemIn-Elemente enthält und der Vorgang create lautete, dürfen keine Artikel zur Anforderung hinzugefügt werden. Die Lieferantenwebsite oder der Benutzer hat die Punchoutsitzung beendet, ohne einen Einkaufswagen zu erstellen. Wenn der Vorgang edit lautete und die PunchOutOrderMessage keine ItemIn-Elemente enthält, müssen in der Punchoutsitzung vorhandene Artikel aus der Anforderung in der Beschaffungsanwendung gelöscht werden.

Der Statuscode „204/No Content“ zeigt das Ende einer Sitzung ohne Veränderung des Einkaufswagens an. Erneut darf die PunchOutOrderMessage (die stets wegen des BuyerCookie benötigt wird) keine ItemIn-Elemente enthalten. Dieser Code würde genauso wie die anderen „leeren“ Fälle behandelt, es sei denn, der Vorgang lautete edit. In diesem Fall hat der Benutzer die Sitzung beendet, ohne etwas zu ändern. Die Anforderung in der Beschaffungsanwendung darf nicht verändert werden.

ItemIn

Dieses Element fügt einen Artikel aus einem Einkaufswagen zu einer Anforderung in der Beschaffungsanwendung hinzu. Es kann verschiedene Elemente enthalten, von denen nur ItemID und ItemDetail obligatorisch sind.

Für das ItemIn-Element stehen folgende Attribute zur Verfügung:

quantity	Anzahl der Artikel, die der Benutzer auf der entfernten Website ausgewählt hat. Da die Lieferantenwebsite Regeln für Teileinheiten durchsetzen kann, lässt das Protokoll gebrochene Zahlen als Mengenangabe zu. Der Wert darf nie negativ sein.
lineNumber (optional)	Position dieses Artikels im Auftrag. Da Punchoutsitzungen normalerweise vor der Bestellung stattfinden und der Server die Platzierung von Artikeln in einem Auftrag ohnehin nicht steuern kann, ist dieses Attribut in einer PunchOutOrderMessage nicht erforderlich.

Optionale Elemente sind ShipTo, Shipping, und Tax, wobei es sich um dieselben Elemente wie in der oben beschriebenen PunchOutOrderMessage handelt.

Mit Ausnahme der im ItemOut-Element enthaltenen Elemente Distribution und Comments sowie der Attribute requisitionID und requestedDeliveryDate, sind die Strukturen der Elemente ItemIn und ItemOut identisch. Das Absendersystem des Käufers kann direkt zwischen ItemIn und ItemOut-Listen hin- und herschalten, wenn es einen inspect- oder edit-Vorgang startet. Auch Lieferanten können die eine Liste in die andere umwandeln (indem sie die aufgeführten Erweiterungen des ItemOut-Elements weglassen), wenn sie einen edit-Vorgang ausführen. Das Absendersystem des Käufers kann die direkte Umwandlung ausführen und weitere Versand- und Verteilungsangaben anfügen, wenn es eine OrderRequest-Transaktion startet. In ItemIn-Elementen enthaltene ItemDetail-Daten (außer eventuellen Extrinsic-Elementen) dürfen beim Umwandeln von ItemIn in ItemOut nicht gelöscht werden.

ItemID

Dieses Element kennzeichnet den Artikel für die entfernte Website eindeutig. Es ist das einzige obligatorische Element, das an die entfernte Website zurückgesendet werden muss, um den übertragenen Artikel erneut zu erkennen.

ItemID enthält zwei Elemente: SupplierPartID und SupplierPartAuxiliaryID. Nur SupplierPartID ist obligatorisch. SupplierPartAuxiliaryID hilft der entfernten Website beim Übertragen komplexer Konfigurationen oder Warenlistendaten zum Wiedererkennen des Artikels, wenn er der entfernten Website später zugesandt wird.

Wenn SupplierPartAuxiliaryID Sonderzeichen enthält (z. B. zusätzliche XML-Dokumente, die im XML-Protokoll nicht definiert sind), müssen diese ordnungsgemäß ausgeklammert werden. Da die Angaben der SupplierPartAuxiliaryID über verschiedene Anwendungen zum Absender zurückgeschickt werden, ist eine interne Teilmenge mit den zusätzlichen XML-Elementen nicht ausreichend.

ItemDetail

Dieses Element enthält beschreibende Angaben über den Artikel, den die Beschaffungsanwendungen den Benutzern anzeigen. Der Inhalt eines ItemDetail-Elements kann recht komplex sein, die Mindestanforderungen sind jedoch einfach: UnitPrice, Description, UnitOfMeasure und Classification.

Im Kontext eines ItemIn-Elements funktionieren die in ItemDetail enthaltenen Extrinsic-Elemente genauso wie die in einem Index (insbesondere IndexItemAdd) auftretenden.

Description

Dieses Element beschreibt den Artikel in Textform. Da der Text beim Überschreiten der Grenzen einer kurzen Positionsliste (oder andere Beschränkungen der Benutzeroberfläche) unvorhergesehen abgeschnitten werden kann, enthält das Description-Element ein optionales ShortName-Element. Sofern möglich, sollten die Clients an Stelle des eventuell abgeschnittenen Textes in Description das ShortName-Element anzeigen, wo immer Feldbeschränkungen auftreten. Wenn kein ShortName zur Verfügung steht, wird der Text in Description durch die Clients weiterhin gekürzt.

Beispiel:

```
<Description xml:lang="en-US">
    <ShortName>Big Computer</ShortName>
    This wonder contains three really big disks, four CD-Rom drives, two Zip drives, an
ethernet card or two, much more memory than you could ever use, four CPUs on two
motherboards.  We'll throw in two monitors, a keyboard and the cheapest mouse we can
find lying around.
</Description>
```

Das Beispiel wird, je nachdem wie viel Platz vorhanden ist, als „Big Computer“ oder als „Big Computer. This wonder ... lying around.“ angezeigt (bzw. vollständig in zwei separaten Feldern).

Spätere Statusänderungen

Nach Beendigung der OrderRequest-Transaktion müssen die Lieferanten und Zwischenserver unter Umständen weitere Informationen an das System des Käufers übermitteln. Diesem Zweck dienen die in diesem Abschnitt beschriebenen Transaktionen. Sie haben gemeinsame Bedeutungen und Elemente.

Wie die Antwort auf eine OrderRequest (siehe „Antwort auf eine OrderRequest“ auf Seite 78), enthält keine dieser Transaktionen ein spezielles Response-Element. Statt dessen enthält das zurückgesendete Dokument eine fast leere Response (lediglich einen Status). Alle zurückgesendeten Dokumente weisen folgende Form auf:

```
<cXML version="1.1.007" payloadID="9949494@supplier.com"
        timestamp="2000-01-12T18:39:09-08:00" xml:lang="en-US">
    <Response>
        <Status code="200" text="OK"/>
    </Response>
</cXML>
```

Der zurückgesendete Code lautet nur dann „200“, wenn der Vorgang erfolgreich abgeschlossen wurde.

DocumentReference

Das DocumentReference-Element enthält ausreichende Angaben, um die Aktualisierungsanforderung einem bestimmten Dokument zuzuordnen. Es wiederholt das ältere Dokument und fügt einen optionalen, vom Lieferanten hinzugefügten Bezeichner hinzu. Beispiel:

```
<DocumentReference
    payloadID="0c300508b7863dcclb_14999"
    orderID="DO4321" />
```

DocumentReference enthält keine Elemente, hat aber folgende Attribute:

payloadID	Eindeutige Nummer in Bezug auf Raum und Zeit, die für Anmeldezwecke zum Erkennen von Dokumenten verwendet wird. Empfohlene Implementierung: datetime.process id.random number@hostname Dieses Attribut wurde direkt aus dem cXML-Element des OrderRequest-Dokuments entnommen.
orderID	Bezeichner für diesen Auftrag. Entspricht der heutigen Bestellauftragsnummer. Dieses Attribut wurde direkt aus dem OrderRequestHeader-Element des OrderRequest-Dokuments entnommen.

StatusUpdateRequest

Diese Transaktion informiert einen vorgeschalteten Knoten über Änderungen im Verarbeitungsstatus eines Auftrags. Eine Änderung ist besonders wichtig: Wenn ein Zwischenknoten ein OrderRequest-Element erfolgreich weitergesendet hat, kann er den Absender oder einen vorgeschalteten Knoten über den Erfolg informieren. Das Passieren verschiedener Warteschlangen und Verarbeitungsschritte beim Lieferanten oder einem Netzknoten ist für den Käufer möglicherweise ebenfalls von Interesse.

Diese Anforderung aktualisiert den Verarbeitungsstatus eines einzelnen OrderRequest-Dokuments. Beispiel:

```
<cXML version="1.1.007" xml:lang="en-US"
        payloadID="0c30050@supplier.com"
        timestamp="2000-01-08T23:00:06-08:00">
    <Header>
        Angaben zur Weiterleitung, Identifizierung und Authentifizierung
    </Header>
    <Request>
        <StatusUpdateRequest>
            <DocumentReference
                payloadID="0c300508b7863dcclb_14999"
                orderID="DO4321" />
            <Status code="200" text="OK" xml:lang="en-US">Forwarded
                to supplier</Status>
        </StatusUpdateRequest>
    </Request>
</cXML>
```

Diese Anforderung enthält nur ein DocumentReference- und ein Status-Element. Beide sind obligatorisch. Der Status kann über einen späteren Übertragungsfehler informieren, der bei einem Zwischenknoten auftritt. Die Semantik dieses Elements ist identisch mit den Status-Elementen, die in der ursprünglichen HTTP-Antwort für ein OrderRequest-Dokument zurückgesendet wurden.

Der Code „200/OK“ ist besonders wichtig, wenn Dokumente gespeichert und weitergeleitet werden. Er gibt an, dass ein Lieferant mit der Verarbeitung der OrderRequest begonnen oder ein Netzknoten das Dokument weitergeleitet hat. Der Empfänger kann nach einem solchen Code mit keinen weiteren StatusUpdateRequest-Dokumenten rechnen.

Lieferanten und Netzknoten, die die StatusUpdate-Transaktion benutzen, müssen den Code „201/Accepted“ zurücksenden, wenn eine OrderRequest zur späteren Verarbeitung in einer Warteschlange geparkt wird. Wenn dann „200/OK“ (im unmittelbaren Response-Dokument auf eine OrderRequest oder eine spätere StatusUpdateRequest) gesendet wird, darf der Server keine weiteren StatusUpdate-Transaktionen für diesen Auftrag senden. Fehler bei der späteren Verarbeitung können diese Regel außer Kraft setzen.

Katalogdefinitionen

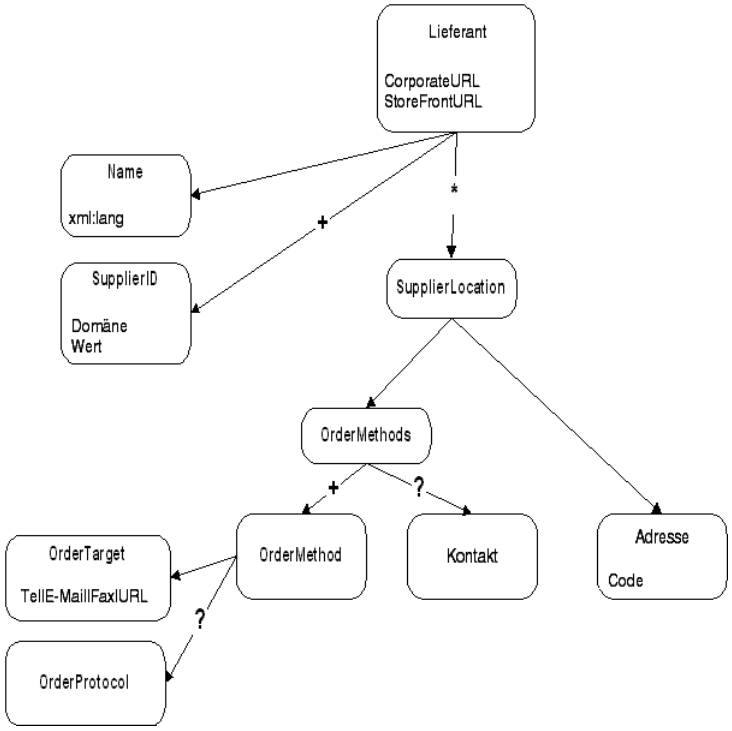
Die cXML-Katalogdefinitionen bestehen aus drei Hauptelementen: Supplier, Index und Contract. Alle drei Elemente beschreiben Daten, die für die Nutzung in einem Netzknoten oder einer Beschaffungsanwendung eines Käuferunternehmens dauerhaft oder zwischengespeichert werden.

- Supplier: Enthält Grunddaten zum Lieferanten wie Adresse, Kontaktperson und Bestellinformationen.
- Index: Beschreibt Daten über den Lieferantenbestand an Waren und Leistungen wie Beschreibung, Teilenummern und Klassifizierungscodes.
- Contract: Beschreibt Daten über flexible, zwischen Käufer und Lieferant ausgehandelte Bestandsaspekte, wie z. B. Preise.

Beachten Sie, dass Index verschiedene Unterelemente zur Beschreibung von Positionen im Bestand des Lieferanten verwendet. Lieferanten können Preisangaben entweder zur Zwischenspeicherung in den Systemen des Käufers oder als Punchoutangabe senden, die den Käufern ermöglicht, mit der Punchoutfunktion auf entfernte Websites zuzugreifen und dort Preise und andere Angaben einzusehen.

Supplier

Das Supplier-Element enthält einen benannten Lieferanten von Waren oder Dienstleistungen. Es muss über ein Name-Element und ein SupplierID-Element verfügen. Dazu beschreibt es optionale Adress- und Bestellangaben des Lieferanten:



Für das Supplier-Element stehen folgende Attribute zur Verfügung:

corporateURL (optional)	URL für die Website des Lieferanten.
storeFrontURL (optional)	URL der Website zum Einkaufen oder Durchsuchen.

Im folgenden Beispiel wird die Struktur des Supplier-Elements dargestellt:

```
<Supplier>
    <SupplierID domain="InternalSupplierID">29</SupplierID>
    <SupplierID domain="DUNS">76554545</SupplierID>
    <SupplierLocation>
        <Address>
```

```
                <Name xml:lang="en-US">Main Office</Name>
                <PostalAddress>
                    …
                </PostalAddress>
            <Email>bobw@workchairs.com</Email>
            <Phone name="Office">
                …
            </Phone>
            <Fax name="Order">
                …
            </Fax>
            <URL>http://www.workchairs.com/Support.htm</URL>
            </Address>
            <OrderMethods>
                <OrderMethod>
                    <OrderTarget>
                        <URL>http://www.workchairs.com/cxmlorders</URL>
                    </OrderTarget>
                </OrderMethod>
                <Contact>
                    <Name xml:lang="en-US">Mr. Smart E. Pants</Name>
                    <Email>sepants@workchairs.com</Email>
                    <Phone name="Office">
                        …
                    </Phone>
                </Contact>
            </OrderMethods>
        </SupplierLocation>
    </Supplier>
```

SupplierLocation

Einige Lieferanten wickeln ihre Geschäfte von mehreren Standorten aus ab. Für jeden Standort kann ein SupplierLocation-Element verwendet werden. Dieses Element umfasst auch Angaben dazu, wie an diesem Standort Geschäfte abgewickelt werden oder auf welche Weise Aufträge angenommen werden können. Ein SupplierLocation-Element enthält neben einem Address-Element auch mehrere OrderMethods-Elemente.

OrderMethods und OrderMethod

Das OrderMethods-Element fasst ein oder mehrere OrderMethod-Elemente für das gegebene SupplierLocation-Element zusammen. Die Position von OrderMethods in der Liste ist von Bedeutung: das erste Element nennt die bevorzugte Bestellmethode, das zweite das in der Priorität nächste usw. in absteigender Präferenzreihenfolge.

OrderMethod umfasst Bestellinformationen in Form eines Bestellziels (wie Telefon, Fax oder URL) und ein optionales Protokoll zur weiteren Klarstellung der erwarteten Bestellungen an einem bestimmten Zielpunkt, z. B. „cxml“ für eine URL-Adresse als Ziel.

Index

Dies ist das Grundelement für die Aktualisierung von Katalogen in den Beschaffungssystemen von Käuferorganisationen.

Ein Index-Element ist einem einzelnen Lieferanten zugeordnet. Das Index-Element lässt eine Liste von Lieferanten-IDs zu, von denen jede für diesen einen Lieferanten steht.

Das Index-Element enthält ein oder mehrere IndexItem-Elemente sowie eine optionale Menge von SearchGroup-Elementen zur Definition von Parametersuchdaten für Artikel. Das Element IndexItem enthält Elemente, die Artikel zum zwischengespeicherten Katalog des Unternehmens hinzufügen oder aus ihm löschen.
Im folgenden Beispiel ist die Struktur eines Index-Elements dargestellt:

```
<Index>
    <SupplierID> … </SupplierID>
    ...
    <IndexItem>
        <IndexItemAdd>
            <IndexItemDetail>
                …
            </IndexItemDetail>
        </IndexItemAdd>
        …
        <IndexItemDelete>
            …
        </IndexItemDelete>
        …
        <IndexItemPunchout>
            …
        </IndexItemPunchout>
    </IndexItem>
</Index>
```

IndexItem, IndexItemAdd, IndexItemDelete und IndexItemPunchout

Das IndexItem-Element dient als Container für die Artikelliste in einem Index.
Es enthält drei Arten von Elementen:

- IndexItemAdd: Fügt einen neuen Artikel ein oder aktualisiert einen im Index vorhandenen Artikel. Es enthält ein ItemID-Element, ein ItemDetail-Element und ein IndexItemDetail-Element.
- IndexItemDelete: Löscht einen Artikel aus dem Index. Es enthält ein ItemID-Element zur Kennzeichnung des Artikels.

- IndexItemPunchout: Fügt einen Artikel ein, mit dem ein Punchoutvorgang zur Lieferantenwebsite gestartet werden kann. Es enthält ein PunchoutDetail-Element und ein ItemID-Element. Äußerlich ähnelt es dem IndexItemAdd-Element, nur dass keine Preisangaben erforderlich sind. Die Käufer können sich auf der Lieferantenwebsite in Echtzeit über Artikeldetails informieren.

ItemID

Durch das ItemID-Element wird ein Artikel des Lieferanten eindeutig gekennzeichnet. Neben einem SupplierPartID-Element enthält es auch ein optionales SupplierPartAuxiliaryID-Element.

SupplierPartAuxiliaryID

Wenn der Artikel durch SupplierPartID nicht eindeutig gekennzeichnet ist, sollten die Lieferanten mit dem SupplierPartAuxiliaryID-Element einen Hilfsschlüssel angeben, der das Teil in Kombination mit den Elementen SupplierID und SupplierPartID eindeutig identifiziert. Beispiel: Ein Lieferant verwendet dasselbe SupplierPartID-Element für einen Artikel, hat aber unterschiedliche Preise für „Stück“ oder „Packung“. In diesem Fall wäre ein sinnvolles SupplierPartAuxiliaryID-Element für die beiden Artikel „Stück“ und „Packung“.

Das SupplierPartAuxiliaryID-Element kann auch als Lieferantencookie, verwendet werden, der dem Lieferanten die Bezugnahme auf komplexe Konfigurationen oder Teiledaten ermöglicht. Es könnte somit alle erforderlichen Daten enthalten, die der Lieferant zur Rekonstruktion des fraglichen Artikels in seinem Computersystem benötigt (eine Zusammenstellung von Daten in einem Korb oder Cookie, die nur für den Lieferanten sinnvoll ist). Weitere Informationen finden Sie im Abschnitt „Käufer- und Lieferantencookies“ auf Seite 38.

ItemDetail

Das ItemDetail-Element enthält Detailangaben über einen Artikel oder alle Daten, die der Benutzer zu einem Artikel kennen möchte und die über die im ItemID-Element dargestellten Grundangaben hinausgehen. Es muss ein UnitPrice-, ein UnitOfMeasure-, ein oder mehrere Description-Elemente sowie ein Classification-Element enthalten. Optional sind auch ein ManufacturerPartID-, ein ManufacturerName-, ein URL-, und eine beliebige Anzahl von Extrinsic-Elementen möglich. Weitere Informationen finden Sie im Abschnitt „ItemDetail“ auf Seite 86.

Im Kontext eines IndexItemAdd-Elements erweitern Extrinsic-Elemente die Angaben zu einem bestimmten Artikel. Diese Erweiterungen sollten nicht in einem OrderRequest-Dokument an den Lieferanten übermittelt werden, weil der Lieferant dieselben Daten mit der eindeutigen ItemID aufrufen kann.

IndexItemDetail

Das IndexItemDetail-Element enthält verzeichnisspezifische Elemente, die zusätzliche Aspekte eines Artikels wie LeadTime, ExpirationDate, EffectiveDate, SearchGroupData oder TerritoryAvailable definieren.

PunchoutDetail

PunchoutDetail gleicht dem ItemDetail-Element, macht darüber hinaus aber auch ein oder mehrere Description-Elemente und ein Classification-Element erforderlich. Es kann außerdem die Elemente URL, ManufacturerName, ManufacturerPartID, ExpirationDate, EffectiveDate, SearchGroupData, TerritoryAvailable und Extrinsic enthalten. Preisangaben und Angaben zu Bearbeitungszeiten oder Maßeinheiten sind dagegen nicht verfügbar.

Contract

Das Contract-Element stellt einen Vertrag zwischen einem Lieferanten und einem Käufer über Waren oder Dienstleistungen dar, die im Index des Lieferanten beschrieben sind. Es ermöglicht dem Lieferanten, Artikelattribute (wie den Preis) im Index mit den vertraglich vereinbarten Werten zu „überlagern“. Außerdem können Lieferant und Käufer diese Überlagerungen nach einem vereinbarten „Segmentschlüssel“ in Segmente zerlegen, die in der Käuferorganisation von Bedeutung sind, zum Beispiel der Name eines Werkes oder eine Kostenstelle.

Für das Contract-Element stehen folgende Attribute zur Verfügung:

effectiveDate	Gültigkeitsdatum und -uhrzeit des Vertrags im ISO 8601-Format.
expirationDate	Ablaufdatum und -uhrzeit des Vertrags im ISO 8601-Format.

Contract enthält ein oder mehrere ItemSegment-Elemente. Beispiel:

```
<Contract effectiveDate="2000-01-03T18:39:09-08:00"
          expirationDate="2000-07-03T18:39:09-08:00">
    <SupplierID domain="InternalSupplierID">29</SupplierID>
    <ItemSegment segmentKey=Plant12>
        <ContractItem>
            <ItemID>
                <SupplierPartID>pn12345</SupplierPartID>
            </ItemID>
            <UnitPrice>
                <Money currency=USD>40.00</Money>
            </UnitPrice>
        </ContractItem>
        …
    </ItemSegment>
</Contract>
```

ItemSegment

Das Element ItemSegment enthält eine Liste von ContractItem-Elementen für ein gegebenes „Segment“, wobei ein Segment eine beliebige Einteilung von Vertragsartikeln auf der Grundlage eines zwischen Lieferanten und Käufer vereinbarten Segmentschlüssels ist.

ItemSegment hat folgende Attribute:

segmentKey (optional)	Vereinbarte Zeichenkette zum Segmentieren benutzerspezifischer Preise.

ContractItem

Ein ContractItem-Element ist ein bestimmter Ersatzartikel für einen Listenartikel. Es enthält ein ItemID-Element, das den im Beschaffungssystem zu ersetzenden Listenartikel eindeutig kennzeichnet. Es kann eine beliebige Anzahl von Extrinsic-Elementen mit dem Überlagerungswert des benannten Listenartikelattributs enthalten.

Definitionen der Abonnementsverwaltung

Zwischenstationen, wie E-Commerce-Netzknoten, können die Lieferanten und die in den Beschaffungssystemen der Käuferunternehmen verwendeten Lieferantenkataloge verwalten. Über derartige Zwischenstationen lassen sich direkte Verbindungen zwischen Beschaffungssystemen und Lieferantensystemen herstellen. Dieser Abschnitt enthält die Elementdefinitionen zur Verwaltung von Lieferantendaten und Katalogінhalten. Diese Definitionen basieren auf vielen bereits erläuterten Definitionen für cXML-Anforderungen und -Antworten, unidirektionale Nachrichten und Kataloge.

Lieferantendaten

Die Definitionen für die Verwaltung von Lieferantendaten bestehen hauptsächlich aus den Elementen SupplierListRequest, SupplierListResponse, SupplierDataRequest, SupplierDataResponse und SupplierChangeMessage. Diese Elemente werden im Folgenden beschrieben, wobei die gegebenen Beispiele für den Fall gelten, dass die Zwischenstation Ariba Network ist.

SupplierListRequest

SupplierListRequest fordert eine Liste der Lieferanten an, mit denen der Käufer bereits geschäftliche Beziehungen unterhält.

```
<Request>
    <SupplierListRequest/>
</Request>
```

SupplierListResponse

SupplierListResponse listet die Lieferanten auf, mit denen der Käufer bereits geschäftliche Beziehungen unterhält.

```
<Response>
    <Status code="200" text="OK"/>
    <SupplierListResponse>
        <Supplier corporateURL=http://www.workchairs.com
                storeFrontURL="http://www.workchairs.com">
            <Name xml:lang="en-US">Workchairs, Inc.</Name>
            <Comments xml:lang="en-US">this is a cool company</Comments>
            <SupplierID domain="DUNS">123456</SupplierID>
        </Supplier>
        <Supplier corporateURL=http://www.computersRus.com
                storeFrontURL="http://www.computersRus.com">
            <Name xml:lang="en-US">Computers R us</Name>
            <Comments xml:lang="en-US">another cool company</Comments>
            <SupplierID domain="DUNS">123456789</SupplierID>
        </Supplier>
    </SupplierListResponse>
</Response>
```

SupplierDataRequest

Das SupplierDataRequest-Element fordert Daten zu einem Lieferanten an.

```
<Request>
    <SupplierDataRequest>
        <SupplierID domain="DUNS">123456789</SupplierID>
    </SupplierDataRequest>
</Request>
```

SupplierDataResponse

Das SupplierDataResponse-Element enthält Angaben zu einem Lieferanten.

```
<Response>
    <Status code="200" text="OK"/>
    <SupplierDataResponse>
        <Supplier corporateURL=http://www.workchairs.com
                storeFrontURL="http://www.workchairs.com">
            <Name xml:lang="en-US">Workchairs, Inc.</Name>
            <Comments xml:lang="en-US">this is a cool company</Comments>
            <SupplierID domain="DUNS">123456</SupplierID>
            <SupplierLocation>
                <Address>
                    <Name xml:lang="en-US">Main Office</Name>
                    <PostalAddress>
                        <DeliverTo>Bob A. Worker</DeliverTo>
                        <Street>123 Front Street</Street>
                        <City>Toosunny</City>
                        <State>CA</State>
                        <PostalCode>95000</PostalCode>
                        <Country isoCountryCode="US">USA</Country>
                     </PostalAddress>
                    <Email>bobw@workchairs.com</Email>
                    <Phone name="Office">
                        <TelephoneNumber>
                            <CountryCode
                                isoCountryCode="US">1</CountryCode>
                            <AreaOrCityCode>800</AreaOrCityCode>
                            <Number>5551212</Number>
                        </TelephoneNumber>
                    </Phone>
                    <Fax name="Order">
                        <TelephoneNumber>
                            <CountryCode
                                isoCountryCode="US">1</CountryCode>
                            <AreaOrCityCode>408</AreaOrCityCode>
                            <Number>5551234</Number>
                        </TelephoneNumber>
                    </Fax>
                    <URL>http://www.workchairs.com/Support.htm</URL>
                </Address>
                <OrderMethods>
                    <OrderMethod>
                        <OrderTarget>
                            <URL>http://www.workchairs.com/cxmlorder</URL>
                        </OrderTarget>
                        <OrderProtocol>cXML</OrderProtocol>
                    </OrderMethod>
                </OrderMethods>
            </SupplierLocation>
        </Supplier>
    </SupplierDataResponse>
</Response>
```

SupplierChangeMessage

Dieses Element dient zur Benachrichtigung über Änderungen an den Lieferantendaten.

```
<Message>
    <SupplierChangeMessage type="new">
        <Supplier corporateURL=http://www.workchairs.com
                storeFrontURL="http://www.workchairs.com">
            <Name xml:lang="en-US">Workchairs, Inc.</Name>
            <Comments xml:lang="en-US">this is a cool company</Comments>
            <SupplierID domain="DUNS">123456</SupplierID>
            <SupplierLocation>
                <Address>
                    <Name xml:lang="en-US">Main Office</Name>
                    <PostalAddress>
                        <DeliverTo>Bob A. Worker</DeliverTo>
                        <Street>123 Front Street</Street>
                        <City>Toosunny</City>
                        <State>CA</State>
                        <PostalCode>95000</PostalCode>
                        <Country isoCountryCode="US">USA</Country>
                    </PostalAddress>
                    <Email>bobw@workchairs.com</Email>
                    <Phone name="Office">
                        <TelephoneNumber>
                            <CountryCode
                            isoCountryCode="US">1</CountryCode>
                            <AreaOrCityCode>800</AreaOrCityCode>
                            <Number>5551212</Number>
                        </TelephoneNumber>
                    </Phone>
                    <Fax name="Order">
                        <TelephoneNumber>
                            <CountryCode
                            isoCountryCode="US">1</CountryCode>
                            <AreaOrCityCode>408</AreaOrCityCode>
                            <Number>5551234</Number>
                        </TelephoneNumber>
                    </Fax>
                    <URL>http://www.workchairs.com/Support.htm</URL>
                </Address>
                <OrderMethods>
                    <OrderMethod>
                        <OrderTarget>
                            <URL>http://www.workchairs.com/cxmlorder</URL>
                        </OrderTarget>
                        <OrderProtocol>cXML</OrderProtocol>
                    </OrderMethod>
                </OrderMethods>
            </SupplierLocation>
        </Supplier>
    </SupplierChangeMessage>
</Message>
```

Katalogabonnements

Im Folgenden werden die Definitionen für die Katalogabonnementsverwaltung beschrieben. Die Beispiele beziehen sich auf Ariba Network als Zwischenstation.

Subscription

Dieses Element erfasst Metadaten über ein einzelnes Katalogabonnement. Zu seinen Unterelementen gehören:

- InternalID: eindeutige zwischenstationsinterne ID
- Name: Name des Abonnements
- ChangeTime: Datum und Uhrzeit der letzten Abonnementsänderung
- SupplierID: ID des Lieferanten, der den Katalog liefert
- Format: Format des Katalogs
- Description: Beschreibung des Katalogs

```
<Subscription>
    <InternalID>1234</InternalID>
    <Name xml:lang="en-US">Q2 Prices</Name>
    <Changetime>1999-03-12T18:39:09-08:00</Changetime>
    <SupplierID domain="DUNS">123456789</SupplierID>
    <Format version="2.1">CIF</Format>
    <Description xml:lang="en-US">The best prices for software</Description>
</Subscription>
```

SubscriptionListRequest

Dieses Element fordert die aktuelle Liste der Katalogabonnements für diesen Käufer an.

```
<Request>
    <SubscriptionListRequest/>
</Request>
```

SubscriptionListResponse

Dieses Element listet die aktuellen Katalogsabonnements eines Käufers auf.

```
<Response>
    <Status code="200" text="OK"/>
    <SubscriptionListResponse>
        <Subscription>
            <InternalID>1234</InternalID>
            <Name xml:lang="en-US">Q2 Prices</Name>
            <Changetime>1999-03-12T18:39:09-08:00</Changetime>
            <SupplierID domain="DUNS">123456789</SupplierID>
            <Format version="2.1">CIF</Format>
            <Description xml:lang="en-US">The best prices for software
            </Description>
        </Subscription>
        <Subscription>
            <InternalID>1235</InternalID>
            <Name xml:lang="en-US">Q2 Software Prices</Name>
            <Changetime>1999-03-12T18:15:00-08:00</Changetime>
            <SupplierID domain="DUNS">555555555</SupplierID>
            <Format version="2.1">CIF</Format>
            <Description xml:lang="en-US">The best prices for software
            </Description>
        </Subscription>
    </SubscriptionListResponse>
</Response>
```

SubscriptionContentRequest

Dieses Element fordert den Inhalt eines abonnierten Katalogs an. Die Anforderung enthält die Elemente InternalID und SupplierID für den Katalog.

```
<Request>
    <SubscriptionContentRequest>
        <InternalID>1234</InternalID>
        <SupplierID domain="DUNS">123456789</SupplierID>
    </SubscriptionContentRequest>
</Request>
```

SubscriptionContentResponse

Dieses Element enthält den Kataloginhalt. Der Katalog kann im Format CIF (Catalog Interchange Format) oder cXML formatiert sein. CIF-formatierte Kataloge werden mit base64-Codierung als Teil eines CIFContent-Elements eingefügt. Bei Katalogen im cXML-Format sind die Elemente Index und Contract direkt eingebunden.

```
<Response>
    <Status code="200" text="OK"/>
    <SubscriptionContentResponse>
        <Subscription>
            <InternalID>1234</InternalID>
            <Name xml:lang="en-US">Q2 Prices</Name>
            <Changetime>1999-03-12T18:39:09-08:00</Changetime>
            <SupplierID domain="DUNS">123456789</SupplierID>
            <Format version="3.0">CIF</Format>
            <Description xml:lang="en-US">The best prices for software
            </Description>
        </Subscription>
        <SubscriptionContent filename="foobar.cif">
            <CIFContent>
                <!-- base64 encoded data -->
                ABCDBBDBDBDBDB
            </CIFContent>
        </SubscriptionContent>
    </SubscriptionContentResponse>
</Response>
```

SubscriptionChangeMessage

Über dieses Element wird dem Beschaffungssystem der Käuferorganisation mitgeteilt, dass an einem von ihm abonnierten Katalog Änderungen vorgenommen wurden.

```
<Message>
    <SubscriptionChangeMessage type="new">
        <Subscription>
            <InternalID>1234</InternalID>
            <Name xml:lang="en-US">Q2 Prices</Name>
            <Changetime>1999-03-12T18:39:09-08:00</Changetime>
            <SupplierID domain="DUNS">123456789</SupplierID>
            <Format version="2.1">CIF</Format>
        </Subscription>
    </SubscriptionChangeMessage>
</Message>
```

Für das SubscriptionChangeMessage-Element steht folgendes Attribut zur Verfügung:

type	Änderungsart: new, delete oder update

Nachrichtenabrufdefinitionen

Bei einigen Käuferorganisationen stehen keine HTTP-Eingangspunkte zur Verfügung, über die cXML-Nachrichten von außerhalb der Firewall empfangen werden können. Solche Umgebungen sind nach der cXML-Spezifikation zulässig.

Im folgenden Abschnitt erhalten Sie Informationen zu Definitionen, mit denen Absendersysteme Nachrichten gegebenenfalls in Warteschlangen parken, wenn die Zielsysteme keine direkten HTTP-Sendungen empfangen können. Die Nachrichten werden dann vom Zielsystem separat abgerufen.

GetPendingRequest

Dieses Element ruft einen Satz von Nachrichten auf, die auf den Anforderer warten. Das MessageType-Element und die Attribute lastReceivedTimestamp und maxMessages steuern Meldungstyp und Anzahl der abgeholten Nachrichten.

lastReceivedTimestamp (optional)	Zeitstempel der zuletzt erhaltenen Nachricht.
maxMessages (optional)	Maximale Anzahl von Nachrichten, die der Anforderer in einer einzelnen Antwort bearbeiten kann.

Bei Erhalt der Anforderung sendet der Empfänger die ältesten Nachrichten zurück, die den angegebenen Typen entsprechen, wobei deren Zeitstempel auf oder nach dem angegebenen Zeitstempel liegen muss. Erfüllen mehrere Nachrichten dieses Kriterium, können im Rahmen des Attributs maxMessages mehrere Nachrichten zurückgesendet werden. Das Warteschlangensystem verwirft alle anstehenden Nachrichten der angegebenen Typen, deren Zeitstempel vor dem angegebenen Wert liegt.

```
<Request>
    <GetPendingRequest lastReceivedTimestamp="1999-03-12T18:39:09-08:00"
            maxMessages="5">
        <MessageType>SubscriptionChangedMessage</MessageType>
    </GetPendingRequest>
</Request>
```

GetPendingResponse

Dieses Element enthält eine oder mehrere Nachrichten, die auf den Anforderer warten.

```
<Response>
    <Status code="200" text="OK"/>
    <GetPendingResponse>
        <cXML version="1.1.007" xml:lang="en-US"
                payloadID="456778@ariba.com"
                timestamp="1999-03-12T18:39:09-08:00">
            <Header>
                <From>
                    <Credential domain="AribaNetworkUserId">
                        <Identity>admin@ariba.com</Identity>
                    </Credential>
                </From>
                <To>
                    <Credential domain="AribaNetworkUserId">
                        <Identity>admin@acme.com</Identity>
                    </Credential>
                </To>
                <Sender>
                    <Credential domain="AribaNetworkUserId">
                        <Identity>admin@ariba.com</Identity>
                    </Credential>
                <UserAgent>Ariba.com</UserAgent>
                </Sender>
            </Header>
            <Message>
                <SubscriptionChangeMessage type="new">
                    <Subscription>
                        <InternalID>1234</InternalID>
                        <Name xml:lang="en-US">Q2 Prices</Name>
                        <Changetime>1999-03-12T18:39:09-08:00
                        </Changetime>
                        <SupplierID domain="DUNS">123456789
                        </SupplierID>
                        <Format version="2.1">CIF</Format>
                    </Subscription>
                </SubscriptionChangeMessage>
            </Message>
        </cXML>
    </GetPendingResponse>
</Response>
```

Anhang B
Neue Funktionen in cXML 1.1

Mit cXML 1.1 stehen folgende Kategorien neuer Funktionen zur Verfügung:

- Allgemeine Neuerungen in cXML
- Änderungen am Extrinsic-Elementen
- Verbesserungen bei Punchouttransaktionen
- Neue Funktionen für Bestellaufträge
- Neue Statustransaktion für Bestellaufträge

Eine ausführlichere Beschreibung aller genannten Elemente und Attribute finden Sie in Anhang A, „cXML-Sprachspezifikation“.

Allgemeine Neuerungen in cXML

Die folgenden Änderungen betreffen große Bereiche der cXML-Sprache. Durch die vorgenommenen Verbesserungen konnten die internationale Verwendung von cXML erleichtert und Kompatibilitätsprobleme zwischen verschiedenen cXML-Versionen weitgehend ausgeschlossen werden. Außerdem sind cXML-Clients über eine neue Profiltransaktion jetzt in der Lage, die Merkmale von cXML-Servern abzurufen.

Verbesserte Unterstützung mehrsprachiger Dokumente

Zur Erhöhung der Konsistenz und zur verbesserten Unterstützung mehrsprachiger Dokumente verfügen die Elemente cXML, Status und ManufacturerName jetzt über ein optionales Attribut mit der Bezeichnung xml:lang.

Weitere Informationen:

„Status“ auf Seite 57

Beispiel:

```
<Status
    xml:lang="en-US"
    code="200
    text="OK">
</Status>
```

Durch dieses Attribut wird die Sprache angegeben, in der die cXML-Clients ihre Antworten senden sollen und in der die Punchoutwebsites für die Benutzer angezeigt werden.

Zentralisierte DTDs

Da bisher für cXML-DTDs kein zentraler Speicherort zur Verfügung stand, waren cXML-Parser nicht in der Lage, die erforderlichen DTDs automatisch zu laden. Inzwischen können DTDs für alle cXML-Versionen dauerhaft von cxml.org heruntergeladen werden.

cXML-DTDs finden Sie im Internet unter:

http://xml.cXML.org/schemas/cXML/<*Version*>/cXML.dtd

Weitere Informationen:

„Prüfung anhand von DTDs“ auf Seite 8

wobei <*Version*> für die vollständige cXML-Versionsnummer steht, z. B. 1.1.007.

Aus Leistungsgründen ist es sinnvoll, wenn die cXML-Clients die entsprechenden DTDs nicht bei jedem Parsen eines cXML-Dokuments erneut abrufen müssen. Stattdessen sollten die DTDs im lokalen Cache abgelegt werden. Nachdem ein URL unterhalb von //xml.cxml.org/schemas/cXML angegeben wurde, muss diese Adresse nie mehr geändert werden. Die DTDs werden nicht überschrieben, sondern immer in neuen Unterverzeichnissen abgelegt.

Neue Profiltransaktion

Durch eine neue Profiltransaktion lassen sich grundlegende Informationen zu cXML-Servern abrufen. Diese Transaktion, die sich aus den beiden Dokumenten ProfileRequest und ProfileResponse zusammensetzt, ruft verschiedene Servermerkmale ab, einschließlich der unterstützten cXML-Version, der unterstützten Transaktionen und der dazugehörigen Optionen.

Weitere Informationen:

„Profiltransaktion“ auf Seite 67

> **Hinweis:** Alle cXML-Server ab Version 1.1 **müssen** diese Transaktion unterstützen.

Ähnlich wie mit dem „Ping“-Befehl können Clients mit Hilfe der Profiltransaktion prüfen, ob der jeweilige Server verfügbar ist.

ProfileRequest

Das Dokument ProfileRequest hat keinen Inhalt. Es wird einfach an den im Header angegebenen cXML-Server geleitet.

```
<cXML version="1.1.007" payloadID="9949494"
    xml:lang="en-US" timestamp="2000-03-12T18:39:09-08:00">
    <Header>
        Routing, identification, and authentication information.
    </Header>
    <ProfileRequest />
</cXML>
```

Der Server antwortet mit dem Dokument ProfileResponse, wie nachstehend beschrieben.

ProfileResponse

Das Dokument ProfileResponse listet die durch den cXML-Server unterstützten Transaktionen und deren Positionen auf. Eventuell vorhandene Optionen werden durch entsprechende Zeichenfolgenwerte angegeben.

```
<ProfileResponse effectiveDate="2000-01-01T05:24:29-08:00">
    <Transaction requestName="OrderRequest">
        <URL>http://workchairs.com/cgi/orders.cgi</URL>
    </Transaction>
    <Transaction requestName="PunchOutSetupRequest">
        <URL>http://workchairs.com/cgi/PunchOut.cgi</URL>
    </Transaction>
</ProfileResponse>
```

Neue Statuscodes

Weitere Informationen:

„Status“ auf Seite 57

Die cXML 1.1-Spezifikation beinhaltet neue Statuscodes für Transaktionen, durch die eine präzisere Kommunikation zwischen Client und Server gewährleistet wird. Darüber hinaus stehen in der neuen Spezifikation bessere Beschreibungen für vorhandene HTTP- und cXML-Statuscodes zur Verfügung.

Neues Typattribut für Marktmitglieder

Für das Element Credential ist ein neues Attribut mit der Bezeichnung type verfügbar, durch das angegeben wird, ob der Absender bzw. Empfänger Mitglied eines Marktes ist. Es können mehrere Märkte definiert sein, und für jeden Markt lassen sich unterschiedliche Anforderungen festlegen.

Weitere Informationen:

„Credential“ auf Seite 56

Der einzige mögliche Wert für das neue Attribut type ist marketplace. Mit Hilfe dieses Attributs können Anmeldeinformationen für Marktmitglieder von solchen für „normale“ Käufer- oder Lieferantenunternehmen abgegrenzt werden. Credential-Elemente ohne das Attribut type kennzeichnen Unternehmen, denen kein bestimmter Markt zugeordnet ist.

In Anforderungen an einen oder von einem Markt müssen im Credential-Element To bzw. From der Markt und das entsprechende Mitgliedsunternehmen angegeben werden.

Änderungen an Extrinsic-Elementen

Mit cXML 1.1 werden einige zusätzliche Elemente und Attribute für Daten eingeführt, die bisher in Extrinsic-Elementen enthalten waren. Durch diese Ergänzungen werden Informationen, die zuvor mit Extrinsic-Elementen gesendet wurden, in die Basisspezifikation übernommen.

Auf Headerebene steht aber auch in cXML 1.1 noch Unterstützung für Extrinsic zur Verfügung.

Neues Contact-Element

Die Elemente OrderRequestHeader, PunchOutSetupRequest und ItemOut können jetzt optionale Contact-Elemente mit einer Auflistung von Personen oder Gruppen enthalten, über die zusätzliche Informationen erhältlich sind.

Weitere Informationen:

„Contact“ auf Seite 73

```
<ItemOut quantity="1">
    <ItemID>
        <SupplierPartID>5555</SupplierPartID>
    </ItemID>
    <ItemDetail>
        <UnitPrice>
            <Money currency="USD">134.00</Money>
        </UnitPrice>
        <Description xml:lang="en">
            <ShortName>Office Chair</ShortName>
            Black leather, with adjustable arms, adjustable height and back angle.
        </Description>
        <UnitOfMeasure>EA</UnitOfMeasure>
        <Classification domain="UNSPSC">12345</Classification>
    </ItemDetail>
    <Contact role="customerService">
        <Address>
            <Name xml:lang="en-US">Joe Bob Emmet</Name>
            <Email>joebob@workchairs.com</Email>
```

```
                    <Phone name="Office">
                        <TelephoneNumber>
                            <CountryCode isoCountryCode="US">1</CountryCode>
                            <AreaOrCityCode>800</AreaOrCityCode>
                            <Number>5551212</Number>
                        </TelephoneNumber>
                    </Phone>
                    <Fax name="Order">
                        <TelephoneNumber>
                            <CountryCode isoCountryCode="US">1</CountryCode>
                            <AreaOrCityCode>408</AreaOrCityCode>
                            <Number>5551234</Number>
                        </TelephoneNumber>
                    </Fax>
                </Address>
            </Contact>
        </ItemOut>
```

Durch das Attribut role werden Position und Titel der Kontaktperson angegeben. Zulässige Werte sind endUser, administrator, purchasingAgent, technicalSupport, customerService und sales.

Unterstützung für requisitionID-Attribut

Das Attribut requisitionID wird jetzt vollständig unterstützt. Es handelt sich dabei um ein optionales Attribut für die Elemente OrderRequest und ItemOut, das die Anforderung des Käufers für eine Position kennzeichnet.

Weitere Informationen:

„OrderRequestHeader“ auf Seite 70

```
<OrderRequest>
    <OrderRequestHeader
            orderID="DO1234"
            orderDate="2000-03-12T13:30:23+8.00"
            type="new"
            requisitionID="R4321">
        <Total>
            <Money currency="USD">12.34</Money>
        </Total>
        <ShipTo>
        . . .
        </ShipTo>
    </OrderRequestHeader>
    <ItemOut>
    . . .
    </ItemOut>
</OrderRequest>
```

Wenn das Attribut requisitionID bereits auf der OrderRequestHeader-Ebene genutzt wird, sollte es auf der ItemOut-Ebene nicht verwendet werden.

Zusammenfassung der Änderungen für Extrinsic-Elemente

In der nachstehenden Tabelle finden Sie eine Liste der Änderungen, die an häufig verwendeten Extrinsic-Elementen vorgenommen wurden.

Altes Extrinsic-Element	Vorhandenes oder neues cXML-Element bzw. -Attribut
Angefordertes Lieferdatum	Attribut ItemOut requestedDeliveryDate
Lieferung abgeschlossen	neues Attribut shipComplete
ReqNumber	vorhandenes Attribut requisitionID
Anforderungsnummer	vorhandenes Attribut requisitionID
Name	neues Element Contact
Telefon	neues Element Contact
E-Mail-Adresse	neues Element Contact
Käufername	neues Element Contact
Käufertelefon	neues Element Contact
OriginalRequester	neues Element Contact
Telefonnummer des Anforderungsstellers	neues Element Contact
ETA	vorhandenes Attribut requestedDeliveryDate

Extrinsic-Elemente auf Headerebene

Bisher konnten Extrinsic-Elemente eines Bestellauftrags ausschließlich auf der Positionsebene des Produkts erscheinen. Diese Einschränkung gilt jetzt nicht mehr, d. h., sie können jetzt überall im OrderRequest-Dokument verwendet werden.

Verwenden Sie diese neue Funktion für Extrinsic-Daten, die für den gesamten Bestellauftrag gelten.

Beachten Sie jedoch, dass es nicht möglich ist, gleichnamige Extrinsic-Elemente sowohl im Header als auch auf der Positionsebene desselben OrderRequest-Dokuments zu verwenden.

Verbesserungen bei Punchouttransaktionen

Durch Verbesserungen an den Punchouttransaktionen konnte die Unterstützung von Einkäufen auf Gang- und Produktebene verbessert werden. cXML unterstützt inzwischen auch abgebrochene Punchoutsitzungen.

Verbesserungen am PunchOutSetupRequest-Dokument

Das von Käufern gesendete Dokument PunchOutSetupRequest wurde im Interesse einer verbesserten Flexibilität von Punchouttransaktionen geändert. Da der in PunchOutSetupRequest angegebene URL entwertet wurde, werden cXML-Server dieses Element in Zukunft ignorieren.

In dem neuen Verfahren wird die Lieferanten-ID (aus den Anmeldeinformationen) genutzt. E-Commerce-Netzwerkhubs empfangen das PunchOutSetupRequest-Dokument, lesen die Lieferanten-ID, suchen den URL der Punchoutwebsite anhand der Kontoinformationen des Lieferanten und senden das PunchOutSetupRequest-Dokument an diesen URL. Durch die Tatsache, dass der URL der Punchoutwebsite nicht durch den Käufer, sondern durch den E-Commerce-Netzwerkhub bestimmt wird, ergibt sich eine höhere Flexibilität.

E-Commerce-Netzwerkhubs ermöglichen den Lieferanten, die URLs ihrer Punchoutwebsites zu speichern.

SelectedItem-Element

Im Rahmen der PunchOutSetupRequest-Erweiterungen verfügt cXML jetzt über eine verbesserte Unterstützung für Punchouts auf Lager-, Gang- und Produktebene. Über ein neues optionales Element mit der Bezeichnung SelectedItem in diesem Dokument können Lieferanten ein Punchout für das gesamte Warenangebot oder nur einen bestimmten Teil davon definieren. Beschaffungsanwendungen können das SelectedItem-Element in PunchOutSetupRequest-Dokumente aufnehmen und somit Punchoutsites in die Lage versetzen, die für den Benutzer anzuzeigenden Produkte auszuwählen. Wenn das SelectedItem-Element nicht definiert ist, wird das gesamte Warenangebot des Lieferanten (Lagerebene) präsentiert.

Weitere Informationen:

„SelectedItem“ auf Seite 81

Das SelectedItem-Element enthält ein ItemOut-Element, das wiederum ein ItemID-Element enthält. Beispiel:

```
<SelectedItem>
    <ItemID>
        <SupplierPartID>5555</SupplierPartID>
    </ItemID>
</SelectedItem>
```

Für den Inhalt des Elements SelectedItem nutzen Beschaffungsanwendungen das Element ItemID (SupplierPartID und SupplierPartAuxiliaryID) aus dem Punchoutindexkatalog. Katalogänderungen sind nicht erforderlich.

In der Anfangsphase sollten die Beschaffungsanwendungen neben dem neuen SelectedItem-Element auch den alten Punchout-URL im PunchOutSetupRequest-Dokument übermitteln. Der alte URL wird durch die E-Commerce-Netzwerkhubs dann nur für die Lieferanten genutzt, die ihre Punchout-URLs noch nicht gespeichert haben.

Leere PunchOutOrderMessage-Dokumente

In cXML sind jetzt auch leere PunchOutOrderMessage-Dokumente zulässig, sodass Benutzer eine Punchoutsitzung beenden können, ohne einen Artikel ausgewählt zu haben. Bisher mussten alle zurückgegebenen PunchOutOrderMessage-Dokumente mindestens einen Artikel enthalten.

Die Lieferanten können eine Schaltfläche mit der Bezeichnung „Abbrechen“ einrichten, die ein leeres PunchOutOrderMessage-Dokument generiert. Bei Betätigung dieser Schaltfläche werden die Punchoutsite und die Beschaffungsanwendung darüber informiert, dass die Einkaufssitzung durch den Benutzer abgebrochen wurde. Daraufhin wird der Einkaufswagen geleert, Artikel werden aus der Anforderung gelöscht, und verschiedene andere Aufräumarbeiten werden ausgeführt.

Neues verborgenes cXML-base64-Feld

Das neue verborgene cXML-base64-Feld unterstützt innerhalb des von den Punchoutsites zurückgegebenen FORM POST-Elements auch fremdsprachige Dokumente. cXML-Dokumente mit in „us-ascii“ nicht enthaltenen Sonderzeichen sollten dieses Feld an Stelle des verborgenen Felds cXML-urlencoded verwenden. Bei nahezu identischer Semantik bietet diese Alternative den Vorteil, dass das gesamte Dokument während der Übertragung base64-codiert bleibt und nicht für den Browser HTML-codiert oder für den Webserver URL-codiert werden muss.

Weitere Informationen:

„URL-Formularcodierung“ auf Seite 63

Bei der Base64-Codierung bleibt die ursprüngliche Codierung des XML-Dokuments erhalten. Obwohl mit den gesendeten Daten kein „charset“-Parameter (Zeichensatz) übertragen wird, kann das entschlüsselte Dokument (nach Beseitigung der Übertragungscodierung) als Medientyp „application/xml“ behandelt werden. Dadurch wird der Parser auf Empfängerseite in die Lage versetzt, in der XML-Deklaration möglicherweise enthaltene Codierungsattribute zu beachten. Für dieses Feld und alle „application/xml“-Dokumente wird standardmäßig die UTF-8-Codierung verwendet.

Neue Funktionen für Bestellaufträge

Die cXML-Bestellauftragsdokumente wurden im Hinblick auf angeforderte Funktionen erweitert.

Neues lineNumber-Attribut

lineNumber ist ein neues optionales Attribut für das ItemOut-Element, durch das die Position eines Artikels innerhalb des Bestellauftrags eines Käufers bestimmt wird. Da es bei der Bestellungsaufgabe ausgewählt wird, ist es in der Regel für Punchout-sitzungen nicht relevant.

Weitere Informationen:

„ItemOut“ auf Seite 75

```
<ItemOut quantity="2" lineNumber="1"
        requestedDeliveryDate="2000-03-12">
    <ItemID>
        <SupplierPartID>11223344</SupplierPartID>
    </ItemID>
    <ItemDetail>
    . . .
    </ItemDetail>
</ItemOut>
```

Die Zeilennummer für bestimmte Artikel muss auch bei Auftragsänderungen erhalten bleiben, damit die Änderung erkennbar bleibt.

Anlagen zu Bestellaufträgen

Häufig ist es erforderlich, dass ein Bestellauftrag durch den Käufer mit Memos, Zeichnungen oder Faxnachrichten näher erläutert werden muss. Beschaffungs-anwendungen können jetzt MIME-codierte Dateien (Multipurpose Internet Mail Extensions) beliebigen Typs an cXML-Bestellaufträge anhängen.

cXML enthält lediglich Verweise auf externe MIME-Teile, die in einem mehrteiligen MIME-Container übertragen werden (mit dem cXML-Dokument in einer E-Mail oder zusammen als Fax).

Ein neues Element mit der Bezeichnung Attachment enthält Verweise auf die Anlagen:

Weitere Informationen:

„Übertragen von Anlagen“ auf Seite 53

```
<Comments>
    <Attachment><URL>cid: uniqueCID@buyer.com</Attachment>
    Please see attached image for my idea of what this should look like
</Comments>
```

E-Commerce-Netzwerkhubs empfangen die Anlagen und leiten diese an den Lieferanten weiter oder speichern sie für den Onlineabruf.

Weitere Informationen zum MIME-Standard finden Sie auf folgenden Websites:

www.hunnysoft.com/mime
www.rad.com/networks/1995/mime/mime.htm

Neues shipComplete-Attribut

shipComplete ist ein neues optionales Attribut für das OrderRequestHeader-Element. Da die Lieferanten bei Verwendung dieses Attributs nur dann zum Ausführen des Bestellauftrags aufgefordert werden, wenn alle Artikel verfügbar sind, lassen sich auf Wunsch Teillieferungen vermeiden.

Weitere Informationen:

„OrderRequestHeader“ auf Seite 70

```
<OrderRequest>
    <OrderRequestHeader
            orderID="DO1234"
            orderDate="2000-03-12T13:30:23+8.00"
            type="new"
            requisitionID="R4321"
            shipComplete="yes">
        <Total>
            <Money currency="USD">12.34</Money>
        </Total>
        <ShipTo>
        . . .
        </ShipTo>
    </OrderRequestHeader>
    <ItemOut>
    . . .
    </ItemOut>
</OrderRequest>
```

Neues ShortName-Element

ShortName ist ein neues, optionales Element, das innerhalb von Description-Elementen für Artikel verwendet wird.

```
<Description xml:lang="en-US">
    <ShortName>Big Computer</ShortName>
    This wonder contains three really big disks, four CD-ROM drives, two Zip drives, an
    Ethernet card, much more memory than you could ever use, four CPUs on two
    motherboards.  We'll throw in two monitors, a keyboard and the cheapest mouse we
    can find lying around.
</Description>
```

Weitere Informationen:

„ItemDetail“ auf Seite 86

ShortName steht für einen kurzen Artikelnamen (30 Zeichen empfohlen, maximal 50 Zeichen), der in die für den Benutzer angezeigten Produktlisten hineinpasst. Da es bisher nur ein Description-Element gab, wurden längere Beschreibungen oft an unbekannten Stellen abgeschnitten.

Beschaffungsanwendungen und andere cXML-Clients sollten in Feldern mit Längenbeschränkungen an Stelle des abgeschnittenen Description-Werts den Wert des Attributs ShortName anzeigen. Wenn das Attribut ShortName nicht definiert ist, kann weiterhin der abgeschnittene Description-Text ausgegeben werden.

Es wird nicht empfohlen, in ShortName und Description denselben Namen anzugeben. Stattdessen sollte mit ShortName der Name des Produkts und mit Description eine Beschreibung der Produktdetails angegeben werden.

Das CIF 3.0-Katalogformat wurde ebenfalls erweitert, um das Attribut ShortName zu unterstützen. Der entsprechende CIF-Feldname lautet Short Name.

Neue Statustransaktion für Bestellaufträge

In cXML 1.1 ist eine neue Transaktion implementiert, mit der der Status von Bestellungen an E-Commerce-Netzwerkhubs übertragen werden kann.

Diese Transaktion stützt sich auf das neue OrderReference-Element, das den Status mit dem zuletzt vom Käufer empfangenen OrderRequest-Element verknüpft.

Neues OrderReference-Element

Im neuen OrderReference-Element ist das Statusupdate mit einem bestimmten OrderRequest-Dokument verknüpft. Das Element wiederholt obligatorische Attribute der OrderRequestHeader- und cXML-Elemente aus dem OrderRequest-Dokument und fügt optionale Kennungen hinzu, die durch den Lieferanten generiert wurden.

Weitere Informationen:

„DocumentReference“ auf Seite 87

```
<OrderReference
    payloadID="0c300508b7863dcclb_14999"
    timestamp="2000-01-08T14:36:05-07:00"
    orderID="DO4321"
    orderDate="2000-01-08T13:56:23-07:00"
    supplierOrderID="27-33-00-08-01"
</OrderReference>
```

Dieses Element wird von der neuen StatusUpdateRequest-Transaktion genutzt.

Neue StatusUpdateRequest-Transaktion

Auftragsverarbeitungspartner (z. B. Fax- oder EDI-Anbieter) senden die neue StatusUpdateRequest-Transaktion an die entsprechenden E-Commerce-Netzwerkhubs, um den Status des Bestellauftrags zu bestimmen. Da der Auftragsstatus auf dem Netzwerkhub angegeben wird, ist er für Käufer und Lieferanten gleichermaßen sichtbar. Zusätzlich können Lieferanten mit dieser Transaktion Käufern ermöglichen, den Status der Dokumentenverarbeitung innerhalb des Lieferantenunternehmens zu sehen.

Weitere Informationen:

„StatusUpdateRequest“ auf Seite 88

> **Hinweis:** Mit dieser Transaktion werden interessierte Partner über Änderungen am Liefer- und Verarbeitungsstatus von *Bestellauftrags-dokumenten* informiert. Eine Aussage zum Lieferstatus der Artikel selbst ist damit nicht möglich.

Eine Änderung ist besonders wichtig: Wenn ein Vermittlerhub ein OrderRequest-Dokument erfolgreich weiterleitet, kann er den ursprünglichen Absender oder einen vorhergehenden Hub über die erfolgreiche Übertragung informieren. Die Übergänge zwischen verschiedenen Warteschlangen und Verarbeitungsschritten auf Lieferanten- oder Hubseite sind unter Umständen auch für den Käufer von Interesse.

Neues Followup-Element

Das Followup-Element innerhalb des OrderRequestHeader-Elements definiert, an welchen URL die StatusUpdateRequest-Dokumente in Zukunft gesendet werden sollen. Diese Adresse wird als Eingangsadresse für alle Dokumente verwendet, die später auf das aktuelle OrderRequest-Dokument verweisen.

Weitere Informationen:

„Followup“ auf Seite 74

Index

T

U

V

X

Z

www.cxml.org