

Wat betekent XML voor software engineers?

Danny Greefhorst, Reinier Balt

Inleiding

Het zal u niet ontgaan zijn: XML is een hype. Overal in de vakbladen en op conferenties is XML het terugkerende thema en lijkt het een wondermiddel waarmee alle software-problematiek wordt opgelost. Als software engineer krijg je te maken met XML. Maar wat kan je nu met XML en welke tools zijn er beschikbaar die het ontwikkelen met XML ondersteunen? Dat zijn de vragen waarop dit artikel een antwoord geeft.

We zullen dat doen door aan te geven wat XML precies is, en wat de belangrijkste onderdelen van XML zijn. Daarna wordt ingegaan op de belangrijkste toepassingen van XML, namelijk het beschrijven van documenten en het integreren van applicaties. Het grootste deel van het artikel geeft een overzicht van een aantal XML-tools en technieken die invloed hebben op het werk van een software engineer. Hierbij gaat het in eerste instantie om technieken die helpen bij het realiseren van applicaties zoals bijvoorbeeld XML-parsers. Een tweede categorie van technieken valt in de categorie "softwarelogistiek", waarbij de gehele administratie rond software-ontwikkeling van belang is.

Wat is XML

XML staat voor eXtensible Markup Language en is een taal waarmee gegevens tekstueel kunnen worden beschreven. Belangrijke eigenschappen van XML zijn de gestructureerde vorm en uitbreidbaarheid, wat betekent dat het soort documenten dat met XML kan worden beschreven in principe oneindig is. Merk op dat XML bedoeld is voor het beschrijven van gegevens en niet zozeer hun presentatie. Een XML-document dat dit artikel beschrijft zou er als volgt uit kunnen zien:

```
<?xml version="1.0">
<artikel>
  <auteur>Danny Greefhorst</auteur>
  <auteur>Reinier Balt</auteur>
  <titel>Wat betekent XML voor software engineers?</titel>
  <tijdschrift>Software Release Magazine</tijdschrift>
</artikel>
```

Zoals te zien is in het document kan met XML aangegeven worden wat de inhoud van een bepaald stuk van het document is door het te omgeven door zogenaamde “tags”. In een tag kunnen attributen worden gebruikt om extra eigenschappen te beschrijven. De toegestane tags en hun structuur kunnen worden vastgelegd in een “document type definition” (DTD). Door het opnemen van een DTD wordt het mogelijk om te controleren of een document aan deze structuur voldoet. De DTD van bovenstaand XML-document zou bijvoorbeeld kunnen zijn:

```
<!ELEMENT artikel      (auteur+, titel, tijdschrift)>
<!ELEMENT auteur      (#PCDATA)>
<!ELEMENT titel       (#PCDATA)>
<!ELEMENT tijdschrift (#PCDATA)>
```

In deze DTD zijn elementen gedefinieerd die overeenkomen met de tags uit het voorbeelddocument. Bij deze elementen is aangegeven waaruit ze bestaan; zo bestaat een artikel uit meerdere auteurs, een titel en een tijdschrift. De elementen auteur, titel en tijdschrift zijn gedefinieerd als #PCDATA, wat aangeeft dat ze uit een stuk tekst bestaan.

Standaard XML technologie

Het World Wide Web Consortium (W3C) [1] is verantwoordelijk voor de definitie van standaarden rond het world wide web en is in die rol ook verantwoordelijk voor XML. Nadat ze zijn goedgekeurd zijn deze standaarden terug te vinden in de vorm van "recommendations". Er is door het W3C een aantal aan XML-gerelateerde standaarden gedefinieerd waarvan er een aantal de status van recommendation heeft gekregen. We zullen deze standaarden hier kort toelichten aangezien ze veelvuldig zullen worden toegepast in combinatie met de XML-standaard zelf.

Namespaces – Deze standaard beschrijft een opdeling van namen in naamruimtes. Hierdoor kunnen documenten gebruik maken van documentdefinities in meerdere naamruimtes, en op een unieke manier refereren naar een specifiek element.

XSL – XSL staat voor eXtensible Stylesheet Language, en is net zoals zijn voorganger CSS (Cascading Style Sheets) bedoeld om de presentatie van een document te beschrijven. In tegenstelling tot CSS wordt XSL zelf ook in XML uitgedrukt. De standaard bestaat uit een taal voor het beschrijven van documenttransformaties (XSLT) en een vocabulaire voor het formatteren van documenten.

XML Schema – Omdat de uitdrukingskracht van DTD's vrij beperkt is wordt er gewerkt aan XML schema's. Deze schema's, welke worden uitgedrukt in standaard XML, maken het mogelijk om naast een beschrijving van de structuur van XML-documenten nog beter uitspraken te doen over de precieze inhoud (type) van elementen. Daarnaast bieden ze een mechanisme om elementen te specialiseren, waardoor documentdefinities eenvoudig kunnen worden hergebruikt.

DOM – DOM staat voor Document Object Model, en biedt een standaard object-georiënteerde API (Application Programming Interface) voor het kunnen lezen en wijzigen van elementen van een XML-document.

XPath – XPath is een standaard waarmee selecties kunnen worden gemaakt uit XML-documenten. Met als invoer een zogenaamde padexpressie levert XPath een aantal nodes en attributen uit het objectmodel van het document die voldoen aan deze expressie. XPath vormt de basis voor standaarden als XPointer, XLink en XSL.

XPointer – Met XPointer is een standaard voor het adresseren van specifieke elementen binnen een XML-document. Dit is buitengewoon nuttig voor verwijzingen tussen documenten zoals gedefinieerd binnen XLink.

XLink – Xlink is een toepassing van XML waarmee het mogelijk wordt om verwijzingen tussen XML-documenten te definiëren. Deze verwijzingen zijn uitgebreider dan de verwijzingen zoals die bestaan in HTML. Zo is het mogelijk om de verwijzingen uit te breiden met extra informatie, kan er naar meerdere documenten verwezen worden, en is het mogelijk verwijzingen in een apart document te definiëren.

Naast deze W3C standaarden is er nog een aantal de facto XML standaarden die breed worden toegepast. Zo is er bijvoorbeeld een event-gebaseerde API voor het uitlezen van XML-documenten genaamd SAX (Simple API for XML).

Toepassingen van XML

De beschikbare XML standaarden kunnen op meerdere manieren worden toegepast. Zoals al eerder gezegd wordt er vaak onderscheid gemaakt tussen het gebruik van XML voor het beschrijven van documenten en voor het integreren van applicaties (via messaging technologie). De document toepassing heeft te maken met het beschrijven en presenteren van gegevens. Messaging is een toepassing waarbij XML gebruikt wordt voor het opmaken van berichten die elektronisch worden uitgewisseld.

XML is in eerste instantie ontwikkeld als opvolger van HTML. Een belangrijk voordeel van XML ten opzichte van HTML is de scheiding tussen inhoud en representatie. Een XML-document bevat alleen de inhoud, terwijl de representatie (bijvoorbeeld in HTML) wordt afgehandeld middels XSL transformaties. Hiermee wordt het mogelijk één XML-document op verschillende manieren te presenteren, bijvoorbeeld zowel in een web-browser als op een WAP-telefoon. XSL transformaties zouden zowel aan de kant van de web-server als in de web-browser uitgevoerd kunnen worden. Het Cocoon [2] project bij Apache is een goed voorbeeld van een servergebaseerd XML-publicatieraamwerk. Goede ondersteuning voor XSL in web-browsers ontbreekt op dit moment nog.

Een andere belangrijke toepassing van XML is applicatie-integratie, waarbij berichten die worden uitgewisseld worden beschreven in XML. De belangrijkste reden waarom XML hier wordt toegepast is dat de flexibiliteit van XML ervoor zorgt dat wijzigingen in gegevens die een applicatie levert minder snel leiden tot aanpassingen in andere applicaties. Ook kan XML een belangrijke rol spelen bij de transformatie van berichtformaten door toepassing van XSL. Het transport van XML berichten wordt afgehandeld met bestaande technologie zoals object request brokers of message oriented middleware. Ook ontstaan er nieuwe transport-standaarden zoals bijvoorbeeld SOAP (Simple Object Access Protocol) [3], die een standaard voor XML-procedureaanroepen over HTTP definieert.

Bij applicatie-integratie is belangrijk dat applicaties het eens zijn over de definities en de betekenis van uitgewisselde documenten. Het W3C heeft hiertoe het Resource Description Framework (RDF) gedefinieerd, waarmee semantische informatie kan worden uitgedrukt. Daarnaast is er een groot aantal initiatieven om te komen tot standaard documentdefinities (vocabulaires) voor businessdocumenten, zoals XML portal [4], Microsoft BizTalk [5] en het door de Verenigde Naties ondersteunde ebXML [6].

Ontwikkelhulpmiddelen

Er zijn genoeg toepassingen van XML, maar welke hulpmiddelen heeft een software engineer om deze te realiseren? Software die met XML om moet kunnen gaan heeft een aantal ingrediënten nodig. In de eerste plaats moet een XML-document ingelezen kunnen worden. Hiertoe is een XML-parser benodigd. Deze parsers zijn er in twee soorten: validerende en niet-validerende parsers. Een validerende parser controleert hierbij niet alleen de standaard XML-syntaxregels, maar ook of het XML-document valide is volgens de DTD. Na het parsen van het document kan met DOM, SAX of XPath informatie worden opgezocht in een XML-document. Verder is het mogelijk om het document te wijzigen en het gewijzigde document weer weg te schrijven.

XML-parsers zijn ruimschoots beschikbaar voor verschillende programmeertalen en in de vorm van bibliotheken en componenten. Microsoft, IBM en Sun zijn hierin voorlopers. IBM en Sun nemen daarnaast samen met anderen deel in de Open Source XML initiatieven van Apache [7]. Daar zijn veel implementaties van bovengenoemde technologieën vrij beschikbaar. Zolang applicaties gebruik maken van parsers die zich houden aan de standaard API's (DOM, SAX) is het zelfs mogelijk om eenvoudig een parser te vervangen door een andere.

Voor het maken van XML-documenten en de bijbehorende documentdefinities is een XML-editor handig. De DTD zou door de editor gebruikt kunnen worden om de gebruiker te helpen bij het maken van een correct en valide XML-document, bijvoorbeeld door aan te geven wat valide invoer is of door controle-acties uit te voeren. Daarnaast zal zo'n editor goede zoekmogelijkheden moeten bieden, inclusief ondersteuning voor XPath. Tenslotte zou een editor in XSL transformaties uit kunnen voeren op een XML-document en het resultaat tonen en bewaren. Er zijn verschillende editors beschikbaar met de bovengenoemde functionaliteit, zoals bijvoorbeeld XMLSpy [8].

Een ander aspect is integratie van XML in ontwikkelomgevingen. Zo zouden XML editors geïntegreerd kunnen worden in ontwikkelomgevingen. Daarnaast zou een object-georiënteerde ontwikkelomgeving mogelijkheden kunnen bieden om klassedefinities te genereren uit XML documentdefinities, en om de toestand van objecten in te lezen en weg te schrijven in XML documenten. Het voordeel hiervan is dat dit transparant voor de ontwikkelaar plaats kan vinden en dat de gegenereerde code sneller is dan een standaard XML-parser. Een voorbeeld van een dergelijk initiatief is Sun's Adelard, welke momenteel nog in ontwikkeling is. Een voorbeeld van een ontwikkelomgeving met XML-integratie is Forté for Java van Sun.

Kijkend naar de ontwikkelingen op het gebied van component based development, is er duidelijk een trend om alle middleware-gerelateerde code (transacties, security, persistentie) te ontkoppelen van de bedrijfsregels. Een ontwikkelaar maakt hierbij alleen nog bedrijfscomponenten en alle middleware gerelateerde code wordt in een container geplaatst. Bekende voorbeelden hiervan zijn Enterprise JavaBeans en COM+. Zo'n container zou ook XML kunnen ondersteunen, zonder dat de ontwikkelaar hiervan op de hoogte hoeft te zijn. Een aanroep naar een component zou dan transparant via een XML-gebaseerde transportlaag kunnen verlopen. Er wordt op dit moment op door verschillende middlewareleveranciers zoals TIBCO, NEON, en IBM gewerkt aan een dergelijke transportlaag.

Softwarelogistieke ondersteuning

Softwarelogistiek [9] heeft betrekking op de inrichting en ondersteuning van de definitie, bouw, assemblage, verspreiding, ondersteuning en evolutie van softwareproducten. Terwijl software vanuit het oogpunt van ontwikkeling wordt beschouwd als het eindresultaat, is het vanuit softwarelogistiek gezien een 'object' van manipulatie, administratie, opslag en transport. Software-ontwikkeling is daarbij niet meer dan één van de stappen.

XML kan bij softwarelogistiek een belangrijke rol spelen door het bieden van een standaard voor het beschrijven van logistieke gegevens. Een dergelijke standaard kan van belang zijn voor uitwisseling tussen softwarelogistieke-tools zoals case-tools, configuratiebeheersystemen en repositories. Daarnaast biedt XML een standaard voor het configureren van applicaties. We zullen in deze paragraaf een aantal voorbeelden van relevante initiatieven schetsen.

XMI [10] staat voor XML Metadata Interchange, en is een voorstel van de Object Management Group voor een standaard voor uitwisseling van modellen en hun bijbehorende meta-modellen. De standaard beschrijft hoe metamodellen, die zijn gedefinieerd in de CORBA-specifieke Meta Object Facility (MOF), kunnen worden vertaald naar XML-documentdefinities. Daarnaast geeft XMI aan hoe modellen die voldoen aan deze metamodellen kunnen worden uitgewisseld middels XML-documenten. Als voorbeeld heeft de OMG de metamodellen van UML en MOF vertaald naar XML-documentdefinities. Hierdoor wordt het mogelijk dat UML-modellen kunnen worden uitgewisseld tussen bijvoorbeeld CASE tools, object-georiënteerde ontwikkelomgevingen en repositories. Zo kan bijvoorbeeld een model gedefinieerd in Rational Rose zonder problemen vertaald worden naar Java klasse definities.

De toepassing van XML is veel breder dan het uitwisselen van UML-modellen. Zo wordt het met XML mogelijk om allerlei gegevens die in het logistieke proces worden uitgewisseld te standaardiseren. Deze standaard kan vervolgens worden gebruikt voor communicatie tussen applicaties, maar kan tevens worden gezien als standaard interface op een gemeenschappelijke repository. Door deze gegevens gezamenlijk in een enterprise repository op te slaan wordt het mogelijk om de relaties tussen gegevens te documenteren, en bijvoorbeeld te gebruiken voor impactanalyse.

Naast het bieden van een uitwisselingsstandaard kan XML ook helpen bij het standaardiseren van configuratiebestanden. Een bekend voorbeeld is het gebruik van XML voor Enterprise JavaBeans (EJB) deployment descriptors [11]. Een deployment descriptor is een XML-document dat omgevings specifieke eigenschappen van het EJB component beschrijft. Zo is er bijvoorbeeld in gedefinieerd welke gebruikers toegang hebben tot het component en op welke wijze gegevens van het component worden opgeslagen in een database. Door het definiëren van een op XML-gebaseerde deployment descriptor wordt het mogelijk om componenten met verschillende applicaties die voldoen aan de EJB standaard te configureren, te implementeren en uit te voeren.

Een ander voorbeeld van het gebruik van XML als configuratie-standaard is Ant [12]; een initiatief van het Apache Jakarta project. In Ant worden XML-documenten gebruikt om het bouwvoorschrift van een Java applicatie in te definiëren. Een dergelijk bestand is hierbij te zien als een Java-specifieke variant van een "makefile". Platformspecifieke constructies zijn echter niet toegestaan in Ant; uitbreidingen worden gedefinieerd in standaard Java. Dit heeft als belangrijk voordeel dat met Ant Java-applicaties ook op elk platform kunnen worden gebouwd.

Een voorbeeld van het gebruik van XML voor het beschrijven van software voor software distributie is het Open Software Description Format (OSD) [13]. OSD geeft een vocabulaire voor het beschrijven van componenten, versies, onderliggende structuur en relaties tussen componenten. OSD, dat deel uitmaakt van het Microsoft Zero Administration initiatief, is gerelateerd aan Microsofts Channel Definition Format (CDF) [14]. CDF wordt gebruikt voor het beschrijven van "push" webpagina's, en is beschikbaar in Microsoft Internet Explorer.

Conclusies

XML is een interessante standaard die prima geschikt is voor het publiceren van documenten en het integreren van applicaties, maar ook als ondersteuning bij softwarelogistiek. XML is het experimentele stadium voorbij; naast de standaard is er een veelvoud van tools beschikbaar voor de software engineer om nu toepassingen te kunnen realiseren die gebruik maken van XML.

XML is echter niet het wondermiddel voor al uw software-ontwikkelproblematiek. Zo is XML slechts een syntax voor het beschrijven van documenten, en moeten er duidelijke afspraken over de betekenis van deze documenten worden gemaakt en vastgelegd. Daarnaast biedt XML geen hulpmiddelen (diensten) voor het omgaan met documenten, zoals bijvoorbeeld het transporteren. Daar zou bijvoorbeeld de SOAP-infrastructuur voor kunnen worden gebruikt. Tenslotte zijn de XML-standaarden nog steeds in beweging en maken tools vaak gebruik van verouderde of onvolwassen standaarden, waardoor aanpassing van de software in de toekomst nodig zal zijn.

Danny Greefhorst is onderzoeker bij het Software Engineering Research Center in Utrecht.

Reinier Balt is consultant bij Verdonck, Klooster & Associates in Zoetermeer.

Referenties

- [1] <http://www.w3c.org>
- [2] <http://xml.apache.org/cocoon>
- [3] Sander Duivestijn, Arno Harteveld, "SOAP = (CBD)2 + XML", Software Release Magazine, Array Publications, Mei 2000.
- [4] <http://www.xml.org>
- [5] <http://www.biztalk.org>
- [6] <http://www.ebxml.org>
- [7] <http://xml.apache.org>
- [8] <http://www.xmlspy.com>
- [9] Gert Florijn, Danny Greefhorst, Hugo Boer, "Softwarelogistiek", Software Release Magazine, Array Publications, Juni 2000.
- [10] <http://www.omg.org/xml/>
- [11] <http://java.sun.com/j2ee/>
- [12] <http://jakarta.apache.org/ant/>
- [13] Arthur van Hoff, Hadi Partovi, Tom Thai, "The Open Software Description Format (OSD)", Microsoft Corp. and Marimba, Inc., Augustus 1997.
<http://www.w3.org/TR/NOTE-OSD.html>
- [14] C. Ellerman, "Channel Definition Format", Microsoft Corp., Maart 1997.
<http://www.w3.org/TR/NOTE-CDFsubmit.html>