Testbed Digitale Bewaring White Paper

XML en digitale bewaring

Testbed Digitale Bewaring is een initiatief van de Rijksarchiefdienst en het Ministerie van Binnenlandse Zaken en Koninkrijksrelaties. Het is een onderzoeksprogramma waar de praktische toepasbaarheid getoets wordt van verschillende manieren om (overheids-) informatie te bewaren en toegankelijk te houden voor de toekomst. Testbed Digitale Bewaring maakt deel uit van de stichting ICTU waar verschillende programma's zijn ondergebracht die alle ten doel hebben de digitale overheid op te bouwen.

ICTU
Nieuwe Duinweg 24-26
2587 AD Den Haag

Tel. 070 888 77 77
Fax: 070 888 78 88

E-mail testbed@ictu.nl
www.digitaleduurzaamheid.nl

Testbed Digitale Bewaring White Paper *XML en digitale bewaring.*

Den Haag, september 2002.

Inhoud

1 *Inleiding*[1]

Digitale archiefstukken zijn kwetsbaar. Al jarenlang wordt er gediscussieerd over de beste methoden om digitale archiefstukken voor de lange termijn te bewaren. Deze discussie zal ongetwijfeld nog jaren voortduren.[2] Er zijn verschillende theoretische oplossingen voorgesteld en er wordt momenteel wereldwijd onderzoek verricht naar manieren waarop digitale archiefstukken authentiek kunnen worden bewaard zonder dat de toegankelijkheid en de bruikbaarheid op de lange termijn in gevaar komen. In deze paper ligt de nadruk op XML, als bewaarstrategie. We plaatsen XML in de context van de huidige opvattingen en gewoonten met betrekking tot digitale bewaring, stellen vast welke thema's hierbij een rol spelen, en geven een overzicht van de huidige stand van de kennis over en het onderzoek naar XML.

1.1 Definitie van digitale bewaring

Het doel van digitale bewaring is te garanderen dat bestanden die elektronisch zijn aangemaakt met behulp van de huidige computersystemen en –applicaties, beschikbaar, bruikbaar en authentiek blijven gedurende de komende tien tot honderd jaar, wanneer de applicaties en systemen die werden gebruikt om het bestand te maken en te raadplegen hoogstwaarschijnlijk niet langer beschikbaar zullen zijn. Digitale bewaring omvat meer dan alleen het bewaren van de bits waaruit het bestand bestaat. We moeten deze ook kunnen *interpreteren*. Als dat niet mogelijk is, is de hoeveelheid bits niets anders dan een betekenisloze reeks nullen en enen. Tijdens het bewaarproces dient ook rekening te worden gehouden met zaken als de context, inhoud, structuur, vorm en gedrag van het bestand. Vorm en gedrag zijn aspecten die specifiek bij digitale bestanden horen. Misschien dat aan deze aspecten daarom de meeste aandacht moet worden geschonken voor het authentiek op de lange termijn bewaren van het digitale archiefstuk.

Er is een heel scala aan digitale formaten beschikbaar en om de zaken nog ingewikkelder te maken worden aan verschillende digitale objecten verschillende eisen gesteld ten aanzien van het bewaren. Deze kunnen afhankelijk zijn van de reden waarom het digitale archiefstuk wordt bewaard, de termijn dat het moet worden bewaard, de context en geschiedenis van het digitale archiefstuk, en het oorspronkelijke formaat.[3] Digitale bewaring betekent voor ieder digitaal object iets anders. Hoewel digitale bewaring vaak wordt beschouwd als het op zodanige wijze bewaren van het object dat het identiek is aan het oorspronkelijke formaat, is dit niet altijd noodzakelijk. Het is niet per definitie nodig om ieder aspect[4] van een digitaal archiefstuk te bewaren. Om deze reden wordt er onderzoek verricht om de essentiële aspecten van digitale

[1] De tekst van deze inleiding is vrijwel ongewijzigd overgenomen van het eerder gepubliceerde rapport 'Migratie: context en huidige status' / Testbed Digitale Bewaring, Den Haag, december 2001.
[2] Onder een lange termijn kan een periode van vijftig jaar of langer worden verstaan, zoals Bennett heeft aangegeven in *A Framework of Data Types and Formats, and Issues Affecting the Long Term Preservation Of Digital Material* (1997). Dit lijkt een redelijke tijdspanne om een bewaringsstrategie voor de lange termijn op te baseren. De technologische veranderingen over vijftig jaar zouden onze verwachtingen wel eens kunnen overtreffen en de toepasbaarheid van een goed ontwikkelde strategie kunnen beperken. Daarbij moet wel worden bedacht dat in de Archiefwet 1995 wordt uitgegaan van een periode van ten minste honderd jaar.
[3] In de Regeling geordende en toegankelijke staat archiefbescheiden (2002) (toelichting, artikel 8) wordt gesteld dat wat moet worden bewaard afhankelijk is van de vereisten ten aanzien van het werkproces waarvan het bestand deel uitmaakt.

archiefstukken te definiëren en de eisen die moeten worden gesteld ten aanzien van de authenticiteit. In alle gevallen moet het digitale archiefstuk echter zo worden bewaard dat het zijn integriteit behoudt, authentiek en bruikbaar is. Dit is een interessante uitdaging.

1.2 De noodzaak van digitale bewaring

Er is een verschil tussen papieren en digitale archiefstukken. Papieren archiefstukken kunnen worden waargenomen met behulp van de vijf zintuigen. Een digitaal bestand is daarentegen alleen waarneembaar met behulp van computerhardware en -software. De snelheid waarmee technologie veroudert maakt van digitale bewaring daarom een zaak die voor iedereen van belang is.

Digitale archiefstukken zijn afhankelijk van software. Ze zijn afhankelijk van de software die oorspronkelijk bedoeld was om ze te interpreteren of weer te geven. Als die software verouderd raakt, wat binnen enkele jaren kan gebeuren[5], ontstaat het probleem hoe het bestand nog kan worden gelezen zonder de oorspronkelijke software-applicatie. Het is niet waarschijnlijk dat verschillende versies van de applicatie het bestand op dezelfde manier zullen lezen, hetgeen kan leiden tot een verandering in het gebruikte bestand (de zichtbare of beschikbare weergave ervan) waardoor de integriteit van het digitale archiefstuk wordt aangetast. Sommige gegevens kunnen helemaal verloren gaan, op andere terreinen kunnen er gegevens bijkomen. Het is daarbij goed mogelijk dat een nieuwe versie niet met het origineel kan worden vergeleken, waardoor veranderingen onopgemerkt blijven. Iedere wijziging van een bestand kan gevolgen hebben voor de authenticiteit en de integriteit, wat weer gevolgen kan hebben voor de archivistische en wettelijke status. Dit kan, afhankelijk van het soort digitale archief en het gebruik ervan, tot problemen leiden, waarvan het zoekraken of verkeerd voorstellen van historische feiten niet het minste is.

Zelfs een eenvoudig computersysteem op kantoor maakt gebruik van een aantal verschillende software-applicaties. Van iedere soort applicatie bestaan verschillende versies van meerdere softwareproducenten. De snelheid waarmee nieuwe versies van software op de markt worden gebracht, met uitgebreide en nieuwe functies (die niet noodzakelijkerwijs op eerdere versies kunnen worden toegepast), draagt bij aan het probleem. Neem als voorbeeld het tekstverwerkingsprogramma MS Word® van Microsoft. Er zijn de afgelopen zes jaar vier nieuwe versies uitgekomen: Word 95, Word 97, Word 2000 en Word 2002. Daarnaast zijn er twee versies van Word gemaakt voor andere besturingssystemen dan Windows: Word 98 Special Edition voor Apple en IMac, en Word 2001 voor de Mac. Binnen deze versies bestaan er dan nog verschillende uitgaven. Deze uitgaven verschillen onderling weliswaar minder, maar kunnen in beginsel allemaal de integriteit of authenticiteit van een document aantasten.

Er zijn al legio voorbeelden van de snelheid waarmee digitale archiefstukken en gegevens ontoegankelijk kunnen worden. De specifieke gegevens van het eerste e-mailbericht in de jaren zestig, en door wie het werd verzonden, zijn niet meer beschikbaar. Sommige digitale archiefstukken uit het voormalige Oost-Duitsland zijn voor altijd verloren gegaan als gevolg van technologische veroudering. In een recent communiqué over het Joint Information Systems Committee (JISC) listserv werden

[5] Gail Hodge schrijft in *Best Practices for Digital Archiving* (1999) dat “verwacht kan worden dat ten minste om de twee tot drie jaar nieuwe versies zullen worden uitgebracht van databases, spreadsheets en tekstverwerkers, met tussentijds nog meer aanvullingen en kleine updates”. Het Public Record Office in Kew stelt in zijn *Guidelines on the Management, Appraisal and Preservation of Electronic Records* (1999) dat het ongebruikelijk zou zijn als migratie zich vaker dan eens per drie jaar voordoet.

artikelen genoemd over het verlies bij NASA van gegevens van de Viking-ruimtesondes die in het midden van de jaren '70 naar Mars werden gestuurd.[6]

Er zijn verschillende strategieën voor digitale bewaring. Hieronder wordt een korte analyse gegeven van zeven verschillende benaderingen.

1.3 Verschillende benaderingen

De belangrijkste bewaringsstrategieën zijn: *technologiebehoud, afdrukken op papier, emulatie, inkapseling, virtuele machinesoftware, XML, opslag in standaardformaten* en *migratie.* Voor al deze strategieën zijn de technische vereisten en kosten verschillend. Ook gelden verschillende eisen ten aanzien van het bewaren van de metagegevens.

1. *Technologiebehoud*. Een van de eerste toegepaste mogelijkheden was het bewaren van de technologie die nodig was om originele bestanden te openen, totdat de documenten zelf niet langer hoefden te worden bewaard. Dit is echter kostbaar en technologisch complex (hoewel sommige grote bedrijven in de praktijk nog altijd gebruikmaken van deze techniek). De ondersteuning van de software en de hardware verdwijnt uiteindelijk en de onderdelen om de hardware operationeel te houden worden steeds schaarser omdat producenten verouderde onderdelen niet langer op voorraad houden. Het aantal machines dat oude bestanden kan lezen neemt constant af om de eenvoudige reden dat computers niet oneindig blijven werken. Maar ook de kennis en vaardigheden om de hardware en software te gebruiken worden schaars en verdwijnen uiteindelijk.

2. *Afdrukken op papier*. Dit is een andere vroege benadering die ook nog steeds wordt toegepast. Maar het op papier afdrukken van alle bestanden is geen realistische bewaarmethode voor de meeste bestanden. Met het afdrukken op papier gaan functionele eigenschappen of andere gedragskenmerken verloren die de bestanden in hun digitale vorm wel hadden. Bepaalde informatie kan ook verloren gaan. 'Embedded' formules in een spreadsheet kunnen bijvoorbeeld niet worden afgedrukt. En databases zijn eenvoudigweg niet ontworpen om te worden afgedrukt. Een afgedrukte database-inhoud is altijd een selectieve view uit de database en bestaat nooit uit de oorspronkelijke gegevens zelf.

   Er zijn zowel rechterlijke uitspraken vóór als tegen het afdrukken op papier[7]. Volgens de National Library of Australia (NLA) kunnen 'platte gegevens' zoals tekst en bepaalde niet-bewegende afbeeldingen worden afgedrukt op papier zonder gegevensverlies maar mogelijkerwijs met enig verlies van functionaliteit.[8] Afdrukken op papier wordt vaak toegepast als tijdelijke oplossing voor bewaring terwijl naar een digitale oplossing wordt gezocht.

3. *Emulatie.* De theorie achter emulatie is dat de enige manier om de authenticiteit en de integriteit van een digitaal archiefstuk op de lange termijn te waarborgen bestaat uit het blijvend verschaffen van toegang tot het bestand in de oorspronkelijke omgeving, dat wil zeggen het oorspronkelijke

[6] JISC listserv, vrijdag 3 augustus 2001, Een mooi praktijkvoorbeeld van digitale bewaring.
[7] De kwestie van NARA's GRS20 sleepte zich jaren voort. De rechter sprak zich aanvankelijk uit tegen GRS20 met als argument dat een e-maildocument niet gelijkgesteld kan worden aan een papieren document en 'dat in papieren afdrukken van een e-mail belangrijke delen van de elektronische versie kunnen ontbreken'. Deze uitspraak werd in hoger beroep echter ten gunste van de archiverende instantie herzien.
[8] National Library of Australia *Draft Research Agenda* 1998, p2.

besturingssysteem en de oorspronkelijke software-applicatie. Dit kan worden bereikt door niet alleen het bestand, maar ook een emulatiespecificatie te bewaren, die voldoende details van de oorspronkelijke omgeving bevat om die omgeving zonodig opnieuw op een toekomstige computer te kunnen creëren.

Sommigen zijn van mening dat emulatie te gecompliceerd is en te veel kansen op fouten biedt. Er is geen garantie dat we in staat zullen zijn de gehele computeromgeving van het bestand opnieuw te creëren, omdat we niet weten hoe toekomstige computers in elkaar zullen zitten. Op andere terreinen is echter niet zonder succes met emulatie geëxperimenteerd, en wellicht is het de enige manier om complexe databases of multimedia objecten te bewaren.

4. *Inkapseling.* Anders dan bij migratie wordt het bestand bij inkapseling wel in de oorspronkelijke vorm bewaard, maar kapselt deze in, samen met een serie instructies over de manier waarop het origineel moet worden geïnterpreteerd. Dit zou een gedetailleerde formele beschrijving moeten zijn van het bestandsformaat en de betekenis van de gegevens. De inkapselende laag zou bijvoorbeeld in XML kunnen worden geschreven. Als de oorspronkelijke software die wordt gebruikt voor het interpreteren van het gegevensbestand complex is, moet de beschrijving ook complex zijn en moet er op worden toegezien dat deze compleet is. Een vervolg op dit idee is het creëren van de beschrijving met een uitvoerbaar programma. De volgende paragraaf, 'Virtuele machinesoftware', is aan dit onderwerp gewijd.

5. *Virtuele machinesoftware.* Een variant op de emulatiebenadering is voorgesteld door Raymond Lorie van IBM.[9] In deze variant wordt het probleem van de interpretatie van gegevensbestanden in de toekomst opgelost door een programma te schrijven dat de interpretatie van de bestanden uitvoert in de machinetaal van een 'Universele Virtuele Computer' (UVC). Het programma moet worden geschreven op het moment dat het bestand wordt gearchiveerd en samen met het bestand worden bewaard. Het programma wordt uitgevoerd door wat Lorie een UVC Interpreter noemt, dat wil zeggen een virtuele machine. Om het bestand op een toekomstige computer te kunnen lezen is een UVC Interpreter nodig, die kan worden gemaakt met de specificaties van de UVC. Deze benadering is in principe dezelfde als die door het Java™ platform wordt gebruikt om de Java programma's van dit moment interoperabel te maken. Om dit efficiënt en praktisch te kunnen uitvoeren, zijn de belangrijkste kenmerken van de voorgestelde UVC-taal dat deze eenvoudig genoeg is om op een relatief directe manier de toekomstige virtuele machines te produceren, en algemeen genoeg om op grote schaal te worden gebruikt voor archiefdoeleinden, zodat de toekomstige virtuele machines er op een kosteneffectieve manier mee kunnen worden gemaakt. Met deze benadering kunnen de gegevens in ieder formaat worden opgeslagen en is de kennis die nodig is om de gegevens te decoderen ingekapseld in het UVC-programma.

   Deze benadering kan ook worden toegepast voor het archiveren van een programma. Dit lijkt meer op de volledige emulatiebenadering. De emulator kan op het moment van archiveren worden geschreven in de UVC-taal, zonder dat enige kennis is vereist van de toekomstige machine die het bestand moet openen.

[9] Raymond A. Lorie, *Long Term Preservation of Digital Information* (2000).

6. *Migratie* (waaronder *opslag in standaardformaten*). Zoals uit recente rapporten naar voren is gekomen, is dit de bekendste en meest toegepaste bewaarstrategie.[10]**.**Het is echter ook de meest bekritiseerde methode. Binnen de scope van Testbed wordt migratie gedefinieerd als het overzetten van bestanden van een hardwareconfiguratie of softwareapplicatie naar een andere configuratie of applicatie . Een eenvoudig voorbeeld hiervan is de migratie van een bestand van Word 6 naar Word 7, een complexer voorbeeld is de migratie van een bestand van Macintosh naar Windows. Een veelgehoord bezwaar tegen migratie is het gegeven dat de resultaten vaak onvoorspelbaar zijn, meestal door een gebrek aan documentatie of omdat er onvoldoende is getest. Als een nieuwe versie van software op de markt komt, voeren veel mensen simpelweg een update van hun documenten uit. Niet zelden leidt dit tot verlies van informatie, of dit nu om de inhoud, structuur, uiterlijk of de context van een bestand gaat. De nieuwe software leest het bestand niet altijd op dezelfde manier als de oorspronkelijke software met als gevolg dat inhoud en functionaliteit verloren kunnen gaan. Migratieresultaten zijn moeilijk te voorspellen, tenzij een substantieel deel van het werk vooraf wordt uitgevoerd wat betreft de specificaties van het bron- en doelformaat . Migratie kan van invloed zijn op de authenticiteit van een document. Ieder document dat wordt bewaard, moet worden bewaard als 'authentiek', omdat de betekenis en de geldigheid van het archiefbescheid anders niet kunnen worden gewaarborgd . Dit heeft zowel juridische als archivistische implicaties.Het Testbed heeft een andere white paper gewijd aan dit onderwerp: *Migratie: context en huidige stand van zaken* (december, 2001).

7. *XML.* Deze afkorting staat voor Extensible Markup Language, een taal om gegevens te verrijken met informatie over structuur en betekenis. Het is een open standaard die is gedefinieerd door het World Wide Web Consortium en is niet afhankelijk van een bepaalde soort platform. Conversie van bestanden naar XML kan worden beschouwd als een aparte migratietechniek. XML wordt echter ook beschouwd als het meest veelbelovende originele gegevensformaat voor archivering en interoperabiliteit, en verdient daarom te worden beschouwd als een op zichzelf staande aanpak. De rest van deze paper is gewijd aan XML.

[10] Het rapport van het onderzoeksproject InterPARES, *Preservation Strategies for Electronic Records, Round 1 (2000-2001) Where We Are Now: Obliquity and Squint?* (2001) bevat de resultaten van een onderzoek in 2000-2001 onder archiefinstellingen, waaruit bleek dat 4 van de 13 migratie noemden als de conserveringstechniek die werd gebruikt. Daarmee was het de meest gebruikte techniek. Zie ook Margaret Hedstrom, Digital Preservation: Problems and Prospects (2001); tevens Jeff Rothenberg en Tora Bikson, *Digital Preservation: Carrying Authentic, Understandable and Usable Records through Time* (1999).

2 *XML in de Regeling geordende en toegankelijke staat archiefbescheiden*

In Nederland schrijft de Archiefwet 1995 voor hoe overheidsorganen met hun archiefbescheiden moeten omgaan. Op 3 maart 2002 is de Regeling geordende en toegankelijke staat archiefbescheiden[11] van kracht geworden. Naast PDF en andere standaarden hebben XML en sommige van haar nevenstandaarden een prominente plaats in dit officiële document. De Regeling mag met recht visionair en stoutmoedig genoemd worden, omdat het enerzijds een jonge open standaard als XML en anderzijds een niet-open standaard als PDF voorschrijft. In dit hoofdstuk wordt gekeken naar wat er in de Regeling staat.

2.1 Begrippen

In de Regeling wordt allereerst een aantal begrippen gedefinieerd. Deze white paper tracht zoveel mogelijk aan te sluiten bij deze terminologie. Hieronder wordt een aantal begrippen uit art. 1 geciteerd:

In deze regeling wordt verstaan onder:
(c) bestand: een geheel van gegevens in een zelfde opslagformaat;
(d) besturingsprogrammatuur: de programmatuur die bestemd is ter besturing van een informatiesysteem;
(f) digitale archiefbescheiden: archiefbescheiden die uitsluitend met behulp van besturings- of toepassingsprogrammatuur geraadpleegd kunnen worden;
(k) opslagformaat: de code volgens welke gegevens op een gegevensdrager zijn opgeslagen;
(l) platform: geheel van apparatuur en besturingsprogrammatuur waarop de toepassingsprogrammatuur werkt;
(m) structuur: het logische verband tussen de elementen van een document of van een archief;
(n) toepassingsprogrammatuur: de programmatuur die bestemd is voor de ondersteuning van de uitvoering van een werkproces;
(o) vorm: de uiterlijke verschijning waarin de structuur en opmaak zichtbaar zijn;

Art. 2 bevat de volgende relevante tekst:

De zorgdrager zorgt ervoor dat van elk van de archiefbescheiden te allen tijde kan worden vastgesteld:
a. de inhoud, structuur en vorm bij het ontstaan, één en ander voor zover de inhoud, structuur en vorm kenbaar moesten zijn voor de uitvoering van het betreffende werkproces; en
b. op welk tijdstip en uit hoofde van welke taak of handeling het door het overheidsorgaan werd ontvangen of opgemaakt; en
c. de samenhang met de andere door het overheidsorgaan ontvangen en opgemaakte archiefbescheiden.

In bovenstaand artikel worden de cruciale begrippen *inhoud*, *structuur* en *vorm* genoemd. Naar de *context* wordt ook verwezen, al wordt deze term niet met name

[11] Regeling van de Staatssecretaris van Onderwijs, Cultuur en Wetenschappen, dr. F. van der Ploeg, 2001, houdende nadere regels omtrent de geordende en toegankelijke staat van te bewaren archiefbescheiden; hierna aangeduid als 'Regeling'; zie http://www.nationaalarchief.nl/images/3_2598.doc.

genoemd. Onvermeld blijft *gedrag* dat door het Testbed als de vijfde eigenschap van een digitaal object wordt beschouwd.

2.2 Pièce de résistance van de Regeling: dertien standaarden

Hieronder volgt de integrale tekst van het hart van de Regeling, art. 6, waarin een slordige dertien standaarden/talen/technieken worden voorgeschreven:

> Digitale archiefbescheiden dienen, uiterlijk op het tijdstip van overbrenging, als bedoeld in de artikelen 12 en 13 van de Archiefwet 1995, te worden opgeslagen volgens de volgende standaarden:
> a. voor character sets: ASCII (ISO/IEC 8859-1) of Unicode (ISO/IEC 10646-1);
> b. voor tekstbestanden: Portable document format (PDF) of SGML dan wel XML vergezeld van een stylesheet (XSL, CSS) dan wel TIFF of PDF met de metadata in een XML-wrapper;
> c. voor CAD/CAM bestanden; Portable document format (PDF) en STEP (Standard for the exchange of product data) als metadata standaard (ISO 10303);
> d. voor images/beelden (bitmapped): Portable document format (PDF) en indien gebruik gemaakt wordt van compressie: ITU T4 of ITU T6;
> e. voor databases: het oorspronkelijke opslagformaat of ASCII (flatfile, met veldscheidingstekens), vergezeld van documentatie bij voorkeur in XML-DTD over de structuur van de database (tenminste omvattende een compleet logisch datamodel met beschrijving van de entiteiten); queries dienen in de vraagtaal SQL (SQL2) te worden vastgelegd.

Interessant is dat voor tekstbestanden enerzijds 'character-based' standaarden als SGML en XML (gebaseerd op respectievelijk ASCII en Unicode) worden voorgeschreven en anderzijds de binaire formaten TIFF en PDF. In de Regeling wordt niet nader toegelicht waarom voor tekstdocumenten o.a. TIFF (Tagged Image File Format) is gekozen en voor afbeeldingen juist geen TIFF (of bijvoorbeeld JPEG of GIF), maar PDF. Deze paper zal verder niet ingaan op images, wel is er in hoofdstuk 3 ('Extended Family van XML') een verwijzing naar Scalable Vector Graphics, een op XML gebaseerde standaard om afbeeldingen vast te leggen. Lid e roept de vraag op: waarom gebruik maken van een DTD als de inhoud van de database zelf niet in XML hoeft worden opgeslagen (zie voor de behandeling van databases de casus van Arbeidsvoorziening in hoofdstuk 4)? De Regeling zegt niets over de vraag of van een transactioneel systeem als een database niet de stand van zaken op meerdere tijdstippen moet worden bewaard.

2.3 Toelichting op de toelichting

In de toelichting op artikel 6 wordt XML nog verder besproken:

> Een andere optie voor tekstbestanden is XML (Extensible Markup Language) in combinatie met een stylesheet. Afkomstig uit de uitgeverswereld[12] is XML nog geen echte, maar een de facto standaard. XML is een subset van de standaard SGML (Standardised Generic/alised Markup Language[13]) en is verwant aan de webtaal HTML. Met behulp van XML wordt de structuur van (een bepaald type) documenten in een zogenaamd Document Type Description[14] (DTD) opgeslagen. Voor het vastleggen van de vorm van documenten kan een style-sheet gebruikt worden. Cascading Style Sheets (CSS), Extensible Stylesheet Language (XSL), dan wel XSL

[12] SGML wordt veel gebruikt in de uitgeverswereld, XML is juist afkomstig uit de internetwereld.
[13] Bedoeld wordt *Standard Generalised* Markup Language. Soms wordt inderdaad de term 'generic' in plaats van 'generalised' gebruikt.
[14] Meer standaard is: Document Type *Definition.*

Transformations (XSLT) kan dan gebruikt worden. De inhoud van een document tenslotte wordt in ASCII-formaat[15] met XML “tags” opgeslagen.

Wat betreft stylesheets is een nadere toelichting op zijn plaats. Een stylesheet bevat namelijk slechts instructies voor programmatuur om op basis van een XML-bestand vorm zoals bijvoorbeeld een PDF- of HTML-bestand te genereren. Door de stylesheets te bewaren heeft men slechts een recept op basis waarvan vorm geproduceerd kan worden; de garantie dat deze er onder alle omstandigheden hetzelfde uit komt te zien heeft men niet (zie ook het volgende hoofdstuk).

[15] XML ondersteunt juist Unicode, de opvolger van ASCII (zie het volgend hoofdstuk).

3 XML en haar familie van standaarden

Dit hoofdstuk beschrijft XML en de aan deze taal verwante standaarden zoals SGML en XSL die in de Regeling worden genoemd.

3.1 Hors-d'oeuvre: vorm, opmaak, structuur en inhoud[16]

Laten wij als voorproefje de centrale gedachte van XML illustreren aan de hand van dit document, dat u leest van een computerscherm of op papier. In beide gevallen kunt u aan de *opmaak* van de gepubliceerde vorm zien dat deze zin deel uitmaakt van de sectie met titel "Hors-d'oeuvre: vorm, opmaak, structuur en inhoud." Het verschil blijkt onder andere uit een onderscheid in stijl: de titel is vet en cursief (in beide gevallen van het lettertype 10-punts Ariel). De globale *structuur* van de white paper blijkt onder andere uit de *opmaak* van de hoofdstuktitels, die een hoogte van 18 punten hebben. Dit soort *opmaak*informatie staat verborgen in het digitale *bestand*, wiens *opslagformaat* alleen met behulp van de oorspronkelijke *toepassingsprogrammatuur*, in dit geval MS Word® (de op dit moment de *de facto* standaardtekstverwerker), kan worden gelezen.[17] In de papieren *vorm* is deze expliciete informatie verloren gegaan.

Voor deze white paper stond de *opmaak* van tevoren vast in de vorm van een sjabloon. Maar stel dat de uiteindelijke *vorm* van het publicatiemedium (bijvoorbeeld het World Wide Web, waar men titels verschillende kleuren kan geven) niet bekend was geweest. In dat geval zou het wenselijk zijn om zo min mogelijk *opmaak* aan te brengen, zodat wanneer de *inhoud* en het doel vaststaan het document in zijn uiteindelijk vorm gegoten kan worden. Daarnaast zou men zich (nog) niet willen binden aan specifieke *toepassingsprogrammatuur* met haar eigen *opslagformaat*, maar onafhankelijk hiervan willen blijven door gebruik te maken van een open standaard waarin alle informatie expliciet toegankelijk is. Juist omdat XML aan dit soort wensen tegemoet komt is deze beschrijvingstaal in korte tijd zo populair geworden. Laten wij eens kijken hoe een deel deze paper (sterk vereenvoudigd) in deze taal, c.q. *opslagformaat* eruit zou kunnen zien:

```
<Whitepaper>
  <Papertitel>XML en digitale bewaring</Papertitel>
  <Hoofdstuk>
        <Hoofdstuktitel>XML en haar familie van standaarden</Hoofdstuktitel>
        <Lead>Dit hoofdstuk beschrijft (...)</Lead>
        <Sectie>
            <Sectietitel>Hors-d'oeuvre: vorm, opmaak, structuur en inhoud</Sectietitel>
            <Alinea>Laten wij als voorproefje de centrale gedachte van XML illustreren aan de
            hand van dit document, dat u leest van een computerscherm of op papier. In beide
            gevallen kunt u aan de opmaak van de gepubliceerde
            <RegelingBegrip>vorm</RegelingBegrip> zien (...)</Alinea>
        </Sectie>
  </Hoofdstuk>
</Whitepaper>
```

[16] In deze sectie worden de sleutelbegrippen uit de Regeling cursief weergegeven.
[17] Om een indruk te krijgen hoe ontoegankelijk een dergelijk formaat is, opene men de file van dit tekstdocument met behulp een eenvoudige editor als Notepad. Eventueel kan men een *migratie* uitvoeren naar het meer onafhankelijke RTF (Rich Text Format)-*opslagformaat* of een *migratie* naar de versie van deze tekstverwerker voor het *platform* Apple met zijn eigen *apparatuur* en *besturingsprogrammatuur*.

Het eerste wat opvalt is dat XML-*bestand* bestaat uit herkenbare lettertekens die men met een eenvoudige editor kan lezen. Meer geavanceerde programma's[18] herkennen de structuur van een dergelijke file en tonen—om de menselijke lezer te helpen—de verschillende bouwstenen van de tekst in verschillende kleuren.[19] In dit geval wordt de tekst van de white paper zelf met zwart weergegeven. Een belangrijk verschil is dat bijvoorbeeld de titel van deze sectie (voorafgegaan door de *tag* <Sectietitel>) in precies hetzelfde lettertype staat als de rest van de XML-tekst (in dit geval 9-punts Ariel). Een XML-file bevat namelijk geen *opmaak* en zoals met andere *flatfiles* (d.w.z platte tekstbestanden) kan men in principe alleen gebruik maken van één lettertype. Om het de lezer gemakkelijker te maken is het wel gebruikelijk om door middel van tabs de inbedding van een XML-bestand aanschouwelijk te maken. Het 'root'-element Whitepaper dat door middel van de open-*tag* <Whitepaper> en de sluit-*tag* </Whitepaper> het geheel omvat, heeft op een niveau lager als 'kinderen' de elementen Papertitel en Hoofdstuk. Hoofdstuk op zijn beurt bevat weer de element Hoofdstuktitel, Lead en Sectie en zo takt de XML-boom zich verder voort.

Uiteraard valt er nog veel meer te zeggen over dit kleine stukje XML. Laten we echter het voor dit moment laten bij een opmerking over de *structuur* van de white paper die in het XML-formaat expliciet door middel van de *tags* wordt benoemd. Deze *tags* bevatten steeds de namen van de elementen, die beschrijven wat de inhoud van element is in plaats van hoe het eruit moet zien. Dit laatste komt duidelijk naar voren bij het element <RegelingBegrip>vorm</RegelingBegrip> dat deel uitmaakt van het element Alinea. Door middel van de tags wordt aangegeven dat 'vorm' een begrip uit de Regeling is. Op een later moment kan besloten worden hoe dit in de uiteindelijke *vorm(en)* eruit moet zien door in een stylesheet de regel op te nemen dat een dergelijk element in cursief of in groen moet worden *opgemaakt*. Op een zelfde manier zou men de term 'de facto' die in de eerste alinea ook in cursief staat kunnen taggen als bijvoorbeeld UitheemseUitdrukking of Latijn en op een later moment beslissen hoe dit type element getoond moet worden. Het moge duidelijk zijn dat door de inhoudelijke benoeming van tekstelementen in XML dit soort documenten (afhankelijk van de precisie van de tagging) zeer gericht doorzocht kunnen worden.

3.2 Grootmoeder ASCII: van bit naar letterteken

De standaard ASCII (American Standard Code for Information Interchange) voor lettertekens is een lichtend voorbeeld van een technologische standaard die wereldwijd wordt gebruikt voor de uitwisseling van tekstuele informatie in digitale vorm. Een contrast hiermee vormt het platform-specifieke EBCDIC van IBM. Men zou het vaststellen van de ASCII-oerstandaard kunnen vergelijken met de uitvinding van het alfabet. In het laatste geval is er een afspraak over wat voor tekens bepaalde klanken representeren en in het eerste geval wordt met behulp van afgesproken combinaties van bits naar lettertekens verwezen.

Omdat de oorspronkelijk ASCII-verzameling (ISO/IEC 646) alleen 128 lettertekens (op basis van 7 bits) definieerde, had men vooral in Europa problemen om letters met accenten of andere alfabetten als het Griekse hiermee vast te leggen. Vandaar dat er een uitgebreidere ASCII kwam op basis van 8 bits (dus met het dubbele aantal mogelijkheden): ISO 8859; de Regeling gaat uit van de character set voor het Latijnse alfabet: ISO/IEC 8859/1. Ook deze jas bleek echter niet ruim genoeg te zijn en daarom is het initiatief ontstaan om alle lettertekens in de wereld vast te leggen: Unicode (ISO/IEC 10646, zie www.unicode.org). De Regeling schrijft naast ASCII Unicode voor

[18] Bijvoorbeeld Microsoft Internet Explorer 6.0.
[19] Omdat de kans dat de lezer een papieren versie zonder kleuren voor zich heeft liggen, gaan we ervan uit dat we het zonder deze leeshulp moeten stellen.

en deze standaard vormt voor XML de basis voor haar gegevensvastlegging. Men verwacht dat uiteindelijk alle besturings- en toepassingsprogrammatuur gebruik zal maken van Unicode. Omdat in de meeste gevallen slechts gebruik wordt gemaakt van een deelverzameling van het enorme Unicode, zijn er in de praktijk verschillende aanpakken om de performance op pijl te houden. Een uitgebreide behandeling van deze materie valt buiten de scope van deze paper. Voor XML is van belang om te weten dat aan het begin van een XML-file wordt aangegeven welke character set wordt gebruikt. In het geval van het stukje XML in de vorige sectie luidt het begin van het bestand als volgt:

```
<?xml version="1.0" encoding="UTF-8"?>
<Whitepaper>
        <Papertitel>XML en digitale bewaring</Papertitel>
(…)
```

De 'Processing Instruction' voor verwerkende programmatuur op de eerste regel geeft aan dat het XML van versie 1.0 betreft die de lettertekens van UTF-8 gebruikt.[20] UTF staat voor UCS Transformation Format en UCS staat weer voor Universal Character Set. Voor de technisch minder geïnteresseerde lezer is het voldoende om te onthouden dat SGML in principe gebruik maakt van ASCII en XML van Unicode.

Wellicht ten overvloede zij opgemerkt dat lettertypes (fonts) zoals Ariel en Times wat anders zijn dan de abstracte lettertekens zoals deze in ASCII en Unicode worden vastgelegd (en waarmee computers met elkaar communiceren). Lettertypes geven de lettertekens hun uiteindelijk vorm op scherm en papier, zodat de menselijke lezer ze met zijn oog kan waarnemen.[21]

3.3 De moeder van XML: SGML

Als ASCII de moeder van SGML is, dan is GML (Generalised Markup Language) de vader. GML bracht de idee van generieke gegevensvastlegging, waarin de inhoud ontdaan van vorm wordt gecodeerd, in de praktijk. Deze voorloper is ontwikkeld in 1969 door een aantal pioniers van IBM, die ook bijgedragen hebben aan de definitie van SGML (Standard Generalised Markup Language), dat in 1986 een onafhankelijke standaard werd (ISO 8879:1986). SGML is bijvoorbeeld met veel succes bij Boeing ingezet voor het beschrijven van vliegtuigonderdelen en produceren van handboeken. Ook veel uitgevers gebruiken met veel genoegen deze taal tot op de huidige dag.

Omdat SGML echter onder andere vrij ingewikkeld is en er relatief weinig programmatuur voor is geschreven, heeft zij nooit een hoge vlucht genomen.[22] Dit geldt niet voor twee verwante talen: XML en HTML, die men—omdat zij toepassingen zijn van SGML—haar dochters zou kunnen noemen.[23]

Hieronder worden twee aspecten van SGML behandeld, markup en DTD, die zij aan HTML en XML heeft doorgegeven.

3.3.1 Markup: verrijking

'Markup' wordt dikwijls in het Nederlands vertaald als 'opmaak'. Deze laatste term suggereert echter dat een taal als SGML de opmaak vastlegt, wat juist niet het geval is

[20] Dit is ook de default encoding, dus als er niets staat is het UTF-8.
[21] En eventueel klank geven door ze uit te spreken.
[22] Terecht noemt de Regeling deze taal; als echter een organisatie nu SGML of XML wil invoeren, ligt de keuze van de laatste meer voor de hand.
[23] XML kent als subset van SGML bijvoorbeeld niet inclusie en exclusie. In tegenstelling tot SGML en HTML moeten in XML 'haakjes sluiten', d.w.z. een open-*tag* moet een corresponderende sluit-*tag* hebben.

(voor opmaak van SGML is juist de taal DSSSL[24] in het leven geroepen). Daarom is het beter om deze term onvertaald te laten of—voor de taalpuristen—de term ‘verrijking’ te gebruiken. De verrijking bestaat uit de tags die steeds het begin en het einde van een element aangeven en in principe de inhoud beschrijven. Men zou de verrijking als ingebedde metadata kunnen beschouwen die informatie geeft over de data zelf, d.w.z. de inhoud. Dit is een goed moment om een ander onderdeel van SGML (en HTML en XML) te introduceren. We hergebruiken daarvoor het stukje XML uit de eerste sectie dat ook correct SGML is:

```
<Whitepaper>
    <Papertitel>XML en digitale bewaring</Papertitel>
    <Hoofdstuk id="H3">
        <Hoofdstuktitel>XML en haar familie van standaarden</Hoofdstuktitel>
        <Lead>Dit hoofdstuk beschrijft (...)</Lead>
        <Sectie vertrouwelijkheid ="hoog">
            <Sectietitel>Hors-d’oeuvre: vorm, opmaak, structuur en inhoud</Sectietitel>
            <Alinea>Laten wij als voorproefje de centrale gedachte van XML illustreren aan de
hand van dit document, dat u leest van een computerscherm of op papier. In beide gevallen kunt
u aan de opmaak van de gepubliceerde <RegelingBegrip>vorm</RegelingBegrip> zien (...)
</Alinea>
        </Sectie>
    </Hoofdstuk>
</Whitepaper>
```

Aan twee elementen zijn nu elk een ‘attribute’ toegevoegd.[25] Het element Hoofdstuk heeft de identificatie ‘H3’ gekregen en Sectie een attribuut ‘vertrouwelijkheid’ met waarde ‘hoog’. In het eerste geval kan men dit attribuut gebruiken om bijvoorbeeld verwijzingen elders in het document naar dit element te maken. Een attribuut geeft aanvullende informatie over een element, men zou het daarom ‘metametadata’ kunnen noemen.[26]

3.3.2 DTD: structuur voor een type document

Via een zogenoemde Document Type Definition (DTD) kan men de structuur die door middel van de tags in een XML-document ligt besloten expliciet en afzonderlijk vastleggen. Bovendien kunnen regels gegeven worden voor bijvoorbeeld het aantal malen dat een element mag voorkomen. Hieronder volgt de DTD voor het stuk SGML/XML in de vorige sectie:

```
<!ELEMENT Whitepaper (Papertitel, Hoofdstuk+)>
<!ELEMENT Papertitel (#PCDATA)>
<!ELEMENT Hoofdstuk (Hoofdstuktitel, Lead?, Sectie*)>
<!ELEMENT Hoofdstuktitel (#PCDATA)>
<!ELEMENT Lead (#PCDATA)>
<!ELEMENT Sectie (Sectietitel, Alinea+)>
<!ELEMENT Sectietitel (#PCDATA)>
<!ELEMENT Alinea (#PCDATA | RegelingBegrip?)*>
<!ELEMENT RegelingBegrip ANY>
<!ATTLIST Hoofdstuk
    id ID #IMPLIED
>
```

[24] DSSSL (Document Style and Semantics Specification Language) wordt niet in de Regeling genoemd. Met komst van XML, CSS en XSL is zij ook meer op de achtergrond geraakt.
[25] Voor de kleurloze lezer zijn deze toevoegingen hierboven vet gemaakt. De editor beeldt een attribuut standaard in rood af
[26] Overigens vindt een minimalistisch ingestelde purist als Bert Bos van de W3C dat XML nog mooier was geweest als attributes buiten de standaard waren gehouden. Door middel van een sub-element kan de informatie van een attribute namelijk ook worden vastgelegd.

```
<!ATTLIST Sectie
    vertrouwelijkheid (geen | laag | hoog) #IMPLIED
>
```

Met behulp van een geformaliseerd notatiesysteem wordt aangegeven waaruit elk element bestaat. Dus het element Whitepaper (eerste regel) bijvoorbeeld omvat (niet meer en niet minder dan één) Papertitel en één of meerdere Hoofdstukken. Op de volgende regel wordt aangegeven dat Papertitel een 'blaadje van de boom' is: het bestaat zelf niet uit andere elementen, maar mag alleen tekst bevatten. Onderaan worden met behulp van ATTLIST de twee attributen gedefinieerd.

In een DTD kan de structuur van een bepaald type documenten worden vastgelegd. Zij kan daarmee dienen als losstaande documentatie van de structuur, als recept voor nieuwe documenten van dit type en als grammatica aan de hand waarvan men programmatuur kan laten controleren of een document 'valide' is (zie 3.6).

3.4 De zuster van XML: HTML

Door middel van een DTD zijn de verschillende versies van HTML (HyperText Markup Language; de eerste versie dateert van 1990) in SGML vastgelegd. Versie 4 van HTML[27] is echter tegenwoordig ook gedefinieerd als toepassing van XML: XHTML. Daarmee is HTML naast een zuster ook een dochter van XML geworden (hierdoor kan programmatuur die XML kan bewerken dit ook met XHTML).

Vaak wordt de belangrijkste vernieuwende eigenschap van HTML vergeten: de mogelijkheid om 'hyperlinks' aan te brengen. Dit concept zou men de vader van HTML kunnen noemen en vormt één van de ingrediënten van het succes van het World Wide Web (WWW). Door verwijzingen naar elementen van hetzelfde document of andere documenten op het Internet op te nemen ontstaat een dynamisch geheel, dat een andere manier van publiceren en lezen met zich mee heeft gebracht.[28]

HTML zou men wel een opmaaktaal kunnen noemen omdat haar (door een DTD vastgestelde tags) verwijzen naar opmaakbegrippen als 'bold' (de tag 'B'). Aan de andere kant zijn HTML-tags als H1, H2 en H3 voor verschillende niveau's van koppen redelijk abstract.

HTML is mooi en populair, maar heeft ook flinke beperkingen die ertoe hebben geleid dat XML werd verwekt: een uiterlijk niet zo interessante, maar wel veel intelligentere zuster. De belangrijkste beperkingen van HTML zijn:

- Zij is niet uitbreidbaar (extensible): de tags liggen vast in de HTML-standaard.
- Inhoudelijke tagging is maar zeer beperkt mogelijk: inhoud en opmaak zijn onlosmakelijk verbonden.
- De daadwerkelijke visualisatie is afhankelijk van de instellingen van de browser. Men is er dus nooit zeker van dat een lezer precies de vorm voor zijn ogen krijgt die de auteur van de HTML bedoeld heeft.

3.5 De standaard XML

Zoals we gezien hebben bouwt XML voort op het solide fundament van SGML, maakt gebruik van het universele Unicode en heeft geleerd van de ervaring met HTML. Uit het feit dat versie 1.0 van de XML-specificatie uit 1998[29] nog geen opvolgers heeft gehad

[27] Zie http://www.w3.org/TR/html4; versie 4.01 dateert van 24 december 1999.
[28] En een grote uitdaging voor degene deze nieuwe vorm van publiceren moet archiveren.
[29] Vastgesteld door de Word Wide Web Consortium op 10 februari 1998 (zie http://www.w3.org/TR/2000/REC-xml-20001006).

blijkt dat deze standaard een schot in de roos was. Gezien het prestige van de World Wide Web Consortium is het vreemd dat de Regeling in de toelichting op art. 6 XML een “de facto standaard” noemt. In deze sectie wordt verdere uitleg over XML. Ook beschrijft gaat deze sectie in op de programmatuur die nodig is om XML te verwerken.

3.5.1 Voortgezette XML: namespaces, empty elements, etc.

Het voorbeeld van de eerste sectie is hieronder toegespitst op XML en uitgebreid met metadata in de traditionele zin: bij wijze van voorbeeld de gegevens van de opdrachtgever en auteur (met de aanvullingen vet afgebeeld).

```
<?xml version="1.0" encoding="UTF-8"?>
<Whitepaper xmlns="xml.ictu.nl" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.ictu.nl/xml/schemas WhitepaperICTU.xsd">
    <Meta>
        <Opdrachtgever>
            <Naam>Jacqueline Slats</Naam>
            <Functie>Programmamanager Digitale Duurzaamheid</Functie>
            <Email>Jacqueline.Slats@ictu.nl</Email>
        </Opdrachtgever>
        <Auteur>
            <Naam>Hette Bakker</Naam>
            <Functie>Senior Consultant CGE&amp;amp;Y</Functie>
            <Email>Hette.Bakker@cgey.nl</Email>
        </Auteur>
    </Meta>
    <Papertitel>XML en digitale bewaring</Papertitel>
    <Hoofdstuk id="H1"></Hoofdstuk>
    <Hoofdstuk id="H2"/>
    <Hoofdstuk id="H3">
        <Hoofdstuktitel>XML en haar familie van standaarden</Hoofdstuktitel>
        <Lead>Dit hoofdstuk beschrijft (...)</Lead>
        <Sectie vertrouwelijkheid="hoog">
            <Sectietitel>Hors d’oeuvre: vorm, opmaak, structuur en inhoud</Sectietitel>
            <Alinea>Laten wij als voorproefje de centrale gedachte van XML illustreren aan de hand van dit document, dat u leest van een computerscherm of op papier. In beide gevallen kunt u aan de opmaak van de gepubliceerde <RegelingBegrip>vorm</RegelingBegrip> zien (...)
</Alinea>
        </Sectie>
    </Hoofdstuk>
</Whitepaper>
```

In het bovenstaande XML-bestand zijn nog twee andere Hoofdstuk-elementen toegevoegd (met de id’s ‘H1’ en ‘H2’). Aan de hand hiervan kunnen we zien dat een element ook leeg mag zijn. Een verkorte notatie van een ‘empty element’ komen we tegen bij het tweede element: in dit geval volgt de slash uit de sluit-tag na de naam van het element en mag bij hoge uitzondering[30] met één tag worden volstaan.

De XML-file bevat nu ook een drietal attributen bij het element Whitepaper. De afkorting ‘xmlns’ in de eerste twee verwijst naar het concept ‘namespace’. Het zou te ver voeren om in het kader van deze white paper dieper in te gaan op dit onderwerp, we volstaan met de mededeling dat namespaces het mogelijk maken om op verschillende plaatsen van dezelfde elementnaam gebruik te maken. Meervoudige definitie van bijvoorbeeld het element Naam hoeft dan niet tot spraakverwarring te leiden. Uit de verwijzing naar het HTTP-protocol in de laatste twee attributen blijkt dat XML gebruik maakt van het

[30] SGML en HTML zijn in dit opzicht minder streng.

WWW om de unieke definities van elementen van een XML-bestand te publiceren en te vinden.

Tot slot voor de lezer die het naadje van de kous wil weten: in het element Functie van het element Auteur van het Element Meta staat '**CGE**&**Y**' (dat in bijv. HTML-vorm als 'CGE&Y' getoond moet worden). Met behulp van de 'entity' & wordt het letterteken & 'geëscapet': een verwerkende processor weet hierdoor dat zij & letterlijk moet nemen en niet beschouwen als het speciale teken dat het begin van een entity aangeeft. De letter & is namelijk net als de haakjes < en > (entities < em >) een teken dat de computer gebruikt om XML te lezen en te ontleden.

3.5.2 XML is leesbaar voor mens en machine

Dit laatste brengt ons bij de programmatuur die XML verwerkt. Idealiter hoeft een gebruiker namelijk nooit de confrontatie met de hoekige haakjes aan te gaan, maar blijft XML 'onder water.' Een stuk programmatuur dat XML kan lezen wordt een XML-parser genoemd. Alleen als een XML-bestand aan de XML-specificatie voldoet (o.a. doordat alle 'haakjes sluiten') kan de parser hiermee uit de voeten. Dergelijke zogenoemde 'well-formed' XML voldoet aan de basisgrammatica van XML.

In het kader van bijvoorbeeld het COVAX-project is XML-software in kaart gebracht.[31] De lijst van programmatuur groeit echter met de dag, bovendien worden veel bestaande programma's in rap tempo 'XML-enabled' gemaakt.

3.6 De structuur beschreven: dochter XML-Schema

XML heeft van haar moeder SGML het DTD-mechanisme overgeërfd. Echter, omdat een DTD bijvoorbeeld zeer beperkt datatypes kan definiëren en zelf geen XML is, wordt zij verdrongen door een andere standaard: W3C-schema.[32] Officieel heet deze standaard XML Schema Definition Language (XSDL), in de praktijk wordt de naam W3C-schema of XML-schema gebruikt. Omdat deze standaard relatief ingewikkeld is, gaan er stemmen op om tot een eenvoudiger taal te komen om de structuur van een type XML-document vast te leggen. Alleen de toekomst zal leren of W3C-schema breed geaccepteerd zal worden; in deze white paper wordt alleen deze schema-standaard behandeld. Op dit moment wordt deze taal steeds breder ondersteund, maar juist XML-gebruikers van het eerste uur blijven gebruik maken van DTD's.

3.6.1 Het (vereenvoudigde) schema van deze paper

Om te laten zien hoe een XML-schema er daadwerkelijk uitziet is, wordt hieronder het schema gegeven voor het type document dat in 3.5.1 is afgebeeld. Uit de eerste regel blijkt dat het schema zelf weer XML is, maar daardoor is de notatiewijze in vergelijking met een DTD veel woordrijker en ingewikkelder geworden.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="xml.ictu.nl" xmlns="xml.ictu.nl"
xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified">
    <xs:element name="Whitepaper">
        <xs:complexType>
            <xs:sequence>
                <xs:element name="Meta">
                    <xs:complexType>
                        <xs:sequence>
```

[31] Zie http://www.covax.org/public_docum/p_documets.htm.
[32] Op 2 mei 2001 aangenomen als officiële W3C Recommendation zie http://www.w3.org/TR/xmlschema-1/ en http://www.w3.org/TR/xmlschema-2/.

```
                    <xs:element name="Opdrachtgever" type="persoonType"/>
                    <xs:element name="Auteur" type="persoonType"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="Papertitel" type="xs:string"/>
        <xs:element name="Hoofdstuk" maxOccurs="unbounded">
            <xs:complexType>
                <xs:sequence minOccurs="0">
                    <xs:element name="Hoofdstuktitel" type="xs:string"/>
                    <xs:element name="Lead" type="xs:string" minOccurs="0"/>
                    <xs:element name="Sectie" maxOccurs="unbounded">
                        <xs:complexType>
                            <xs:sequence>
                                <xs:element name="Sectietitel" type="xs:string"/>
                                <xs:element name="Alinea" maxOccurs="unbounded">
                                    <xs:complexType mixed="true">
                                        <xs:choice maxOccurs="unbounded">
                                            <xs:element name="RegelingBegrip"
                                                minOccurs="0"/>
                                        </xs:choice>
                                    </xs:complexType>
                                </xs:element>
                            </xs:sequence>
                            <xs:attribute name="vertrouwelijkheid" use="optional">
                                <xs:simpleType>
                                    <xs:restriction base="xs:string">
                                        <xs:enumeration value="geen"/>
                                        <xs:enumeration value="laag"/>
                                        <xs:enumeration value="hoog"/>
                                    </xs:restriction>
                                </xs:simpleType>
                            </xs:attribute>
                        </xs:complexType>
                    </xs:element>
                </xs:sequence>
                <xs:attribute name="id" type="xs:ID"/>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
</xs:complexType>
</xs:element>
<xs:complexType name="persoonType">
    <xs:sequence>
        <xs:element name="Naam"/>
        <xs:element name="Functie"/>
        <xs:element name="Email"/>
    </xs:sequence>
</xs:complexType>
</xs:schema>
```

3.6.2 Kleindochters van XML: XML-vocabularies

XML zelf is slechts een zogenoemde metataal die de syntaxis levert waarmee men zelf een taal kan vastleggen. Met behulp van XML-schema's kan men deze definiëren. Het resultaat wordt wel een (XML-)vocabulary genoemd: als het ware een kleindochter van XML. Men zou kunnen zeggen dat XML-schema als moeder de syntaxis levert en dat een concreet toepassingsgebied als vader zorgt voor de semantiek. In het geval van de het hierboven gepresenteerde schema zou dit deel uit kunnen maken van een vocabulary voor publicaties van ICTU ("IctuML") of een andere organisatie. Op dit

moment zijn er zeer veel initiatieven om vocabularies af te spreken; slechts een klein deel hiervan zal doorbreken als algemeen geaccepteerde standaard. Een gebruiker c.q. organisatie moet de afweging maken of het gunstiger is om van een bestaande vocabulary gebruik te maken of zelf met XML-schema aan de slag te gaan. De ervaring leert dat juist de inhoudelijke discussie over hoe een type document is opgebouwd (en opgebouwd zou moeten zijn!) veel tijd in beslag kan nemen.

3.6.3 De structuur controleren: validators

Een XML-document kan aan het begin verklaren dat het aan een bepaald schema (of DTD) voldoet (het is overigens niet verplicht een schema te hebben).[33] Als een document zegt een 'instance'[34] te zijn van een abstract type document dat vastgelegd is in een schema, dan kan men controleren of dit inderdaad zo is met behulp van een validator. Dit stuk programmatuur (ingebouwd in bijvoorbeeld een XML-editor) neemt het (well-formed) XML-document en houdt het als het ware naast het schema. Het resultaat van dit validatieproces is het antwoord wel of niet valide; veel validators geven in het laatste geval hulp bij het opsporen van grammaticale fouten.

Valideren kan men op verschillende momenten, bijvoorbeeld aan de bron, dus alvorens een XML-document te versturen of bij ontvangst van een XML-instance. Dit is een zeer probaat middel om een sjabloon af te dwingen[35] en de kwaliteit van gegevens te waarborgen.

3.7 Naar de vorm: dochter XSL en aangetrouwde CSS

Een XML-document is door zijn gestructureerde en consistente opbouw uitermate leesbaar voor een computer. Voor de doorsnee menselijke gebruiker is het echter eerder een halffabrikaat dat nog een meer toegankelijke vorm moet krijgen (zonder de puntige haakjes). Met behulp van het stylesheet-mechanisme kan men een dergelijke vorm genereren; bijvoorbeeld een HTML-, PDF- of PostScript-bestand dat met behulp van de bijbehorende toepassingsprogrammatuur op het scherm, op papier of zelfs als klank kan worden waargenomen. In deze sectie komen de stylesheets die in de Regeling genoemd worden aan de orde en de bijbehorende programmatuur.

3.7.1 Cascading Style Sheets

Zoals boven uiteengezet is HTML een enorm succes geworden, maar heeft deze taal grote beperkingen. Marktpartijen begonnen al snel eigen ongestandaardiseerde uitbreidingen op HTML te formuleren om aanvullende opmaakaanwijzingen te kunnen geven. Zo voegde de browser Netscape het element CENTER toe, dat echter niet geïnterpreteerd kan worden door een spraak-converter.[36] Om als het ware de standaard HTML te redden stelde de W3C in 1996 de standaard Cascading Style Sheets (CSS) vast als instrument om opmaak te faciliteren. CSS, dat weer een aparte taal vormt (dus geen SGML/XML is), kent op dit moment twee generaties: CSS1 en CSS2. [37] Om een indruk te geven van hoe CSS eruit ziet en wat voor informatie het bevat volgt hieronder zonder verdere toelichting een fragment.

```
BODY {
    font-family: Arial, Helvetica, sans-serif;
```

[33] Het document in 3.5.1 verwijst bijvoorbeeld naar het schema WhitepaperICTU.xsd.
[34] Een mogelijke vertaling van deze Engelse term is 'voorkomen'. Vergelijk een object als instance van een klasse in object-georiënteerd denken of meer filosofisch: een concrete stoel als voorkomen van de idee stoel.
[35] Een gewone tekstverwerker staat de gebruiker toe om af te wijken van een sjabloon.
[36] Op deze laatste vorm die een tekst kan krijgen gaat de Regeling niet in. Wel worden als fysieke media geluidsdragers genoemd.
[37] CSS kunnen niet alleen in combinatie met HTML, maar ook met XML en XSL worden gebruikt: zie http://www.w3.org/TR/NOTE-XSL-and-CSS.

```
    margin-left : 0px;
    margin-top : 0px;
    margin-bottom : 0px;
    margin-right : 0px;
}
.result{
    background-color : #FFFFFF;
}
.resultheader{
    background-color : #BCBCBC;
}
.inactivelink{
    color : Gray;
}
.big_black
{
   COLOR: black;
   FONT-FAMILY: Arial, Helvetica, sans-serif;
   FONT-SIZE: large
}
```

3.7.2 XSL-FO en XSLT

CSS en DSSSL (zie 3.3.1) zou men beide als vader van van Extensible Stylesheet Language[38] (XSL) kunnen beschouwen. XSL maakt gebruik van de ervaring opgedaan met deze oudere standaarden en gebruikt als syntaxis XML. Dit laatste heeft tot gevolg dat deze taal net als XML-schema niet erg compact is. Men zou kunnen zeggen dat XSL uit twee delen bestaat: XSL-FO en XSLT. XSL-FO (Formatting Objects) is zeer rijk, men kan ermee bijvoorbeeld een stylesheet formuleren dat de instructies bevat om uit een bijpassend XML-document een PDF-file te genereren.[39]

XSL is veel krachtiger dan de naam stylesheet doet vermoeden. Met behulp van XSL kan men een XML-document omzetten naar o.a. een ander XML-document of een HTML-file. Filteren is ook mogelijk; zo zou men in een XSL-stylesheet kunnen vastleggen dat bij transformatie naar een HTML- of XML-document voor extern gebruik bepaalde elementen weggelaten moeten worden (bijvoorbeeld aan de hand van het attribuut ‘vertrouwelijkheid’ in 3.3.1). In dit geval kan men voor intern en extern gebruik (èn archivering) gebruikmaken van één XML-brondocument dat met één of meerder stylesheets wordt getransformeerd. Het deel van XSL dat voor dergelijke omzettingen zorgt, is tegenwoordig uitgekristalliseerd tot een aparte standaard: XSL Transformations (XSLT).

Hieronder volgt ter illustratie het begin van een XSL-stylesheet die deel uitmaakt van een demo ontwikkeld voor Arbeidsvoorziening (zie het volgende hoofdstuk). Vet afgebeeld zijn de HTML-tags die aangeven dat het te behandelen XML-document naar HTML moet worden omgezet. Men ziet ook een verwijzing naar een CSS-stylesheet (cipers.css).

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
   <xsl:template match="/">
      <html>
         <head>
            <link rel="stylesheet" href="/cipers.css"/>
```

[38] Vastgesteld op 15 oktober 2001 door de W3C; zie http://www.w3.org/TR/xsl.
[39] Voor het genereren van PDF zie bijvoorbeeld ook http://www.tallcomponents.com/.

```
        </head>
        <body>
            <table class="main_text" width="80%" align="center" cellspacing="0">
                    <tr>
                    <td>
                        <font class="big_blue">Resultaat</font>
                    </td>
                </tr>
            </table>
            <xsl:apply-templates />
        </body>
      </html>
    </xsl:template>
    <xsl:template match="@expiry-date">
      <xsl:variable name="date">
          <xsl:value-of select="."/>
      </xsl:variable>
      <!-- Day -->
      <xsl:value-of select="substring($date, 7, 2)" />
      <xsl:text>-</xsl:text>
      <!-- Month -->
      <xsl:value-of select="substring($date, 5, 2)" />
      <xsl:text>-</xsl:text>
      <!-- Year -->
      <xsl:value-of select="substring($date, 1, 4)" />
          </xsl:template>
```

3.7.3 Stylesheet-processors

Om XML daadwerkelijk te transformeren aan de hand van de instructies in een stylesheet, is een stylesheet-processor nodig. Deze programmatuur maakt steeds vaker deel uit van browsers en andere software. Overigens zijn XSL en bijpassende programma's niet de enige mogelijkheid om XML-documenten te transformeren. Ervaren programmeurs kunnen met bijvoorbeeld de programmeertaal Perl sneller tot resultaten komen.

3.8 Extended family van XML

Na de behandeling van de XML zelf en haar belangrijkste nevenstandaarden worden hieronder nog kort een aantal XML-gerelateerde standaarden geïntroduceerd.[40]

- XPath is een standaard om informatie binnen een XML-document te localiseren. XML-schema en XSLT maken gebruik van XPath.
- De bevragingstaal Xquery zou men kunnen beschouwen als de XML-tegenhanger van SQL. Deze nieuwe standaard bouwt voort op XPath.
- SVG (Scalable Vector Graphics) is een op XML gebaseerde standaard die erop is gericht om afbeeldingen als abstracte geometrische vormen op te slaan in plaats van als een verzameling puntjes (een bitmap).[41]
- Het Resource Description Framework[42] biedt een complexe manier om XML-documenten zo op te zetten dat ze gemakkelijker kunnen worden geïnterpreteerd als metadata. Een initiatief dat hiermee raakvlakken heeft zijn de Topic Maps (ISO/IEC 13250). Op dit moment worden er pogingen gedaan om beide standaarden te combineren.

[40] Bezoek voor meer en actuele informatie de website van de W3C: www.w3.org.
[41] SVG richt zich net als XML zelf meer op de intelligente inhoud, terwijl een bitmap net als PDF en HTML de bevroren vorm bewaard.
[42] Zie http://www.w3.org/RDF/.

3.9 Bij wijze van samenvatting

Ter recapitulatie volgt hieronder een globale indeling van de behandelde standaarden in generaties en naar gebruik:[43]

Generatie	Letters	Inhoud	Structuur	Omzetting	Vorm	Verwijzing
↓	ASCII	SGML	DTD	(DSSSL)		(HyTime)
↓	Unicode	↓	↓	↓	HTML,CSS1	HTML
↓	↓	XML	↓	XSL	HTML,CSS2	↓
↓	↓	↓	(Schema)	↓	(XHTML)	(Xpath)

Om de herkomst en de samenhang tussen XML en haar nevenstandaarden XSL en (XML-)Schema nader aan te geven volgen hieronder een aantal optellingen:

XML = Unicode + SGML + XML-grammatica
Schema-taal = XML + Schema-grammatica
XSL = XML + XSL-grammatica (+ CSS + DSSSL)

Bovenstaande vergelijkingen hebben betrekking op de drie genoemde talen. Hieronder volgen vergelijkbare optellingen voor de concrete 'uitdrukkingen' die met deze talen kunnen worden gedaan:

XML-document = inhoud (data) + metadata + structuur
Schema = mogelijke structuur + datatypen
XSL-stylesheet = transformatie- c.q. opmaak-regels

Om tot slot van deze inleiding inzicht te geven wat programmatuur met deze standaarden kan doen geven we de volgende combinaties:

XML-document + parser → well-formed?: ja of nee
Well-formed XML-document + parser → bewerken XML-document
Well-formed XML-document + schema + validator → valide?: ja of nee
Well-formed (+ valide) XML-document + XSL-stylesheet + XSL-processor → getransformeerd XML-document of document in opgemaakte vorm (bijv. HTML of PDF)

Tot slot volgen hieronder regels hoe met behulp van (gegenereerd) HTML of PDF een publiceerbare vorm aan een mens getoond kan worden.

HTML-document (+ CSS-stylesheet) + browser (viewer) → vorm (op het scherm)
HTML-document (+ CSS-stylesheet) + browser + printer → vorm (op papier)
HTML-document + spraakconverter → geluid
PDF-document + viewer → vorm (op het scherm)
PDF-document + viewer + printer → vorm (op papier)

[43] Tussen haakjes staan de standaarden die waarschijnlijk te oud of te nieuw waren om in de Regeling te worden opgenomen.

4 *XML en digitale bewaring in de praktijk*

In hoofdstuk 2 zagen we dat de Regeling een groot aantal standaarden voor digitale bewaring voorschrijft. Hoofdstuk 3 trachtte de herkomst en de functie van de XML-gerelateerde standaarden nader in kaart te brengen. In dit hoofdstuk gaan we eerst in op de vraag wat de 'unique selling point' van XML is voor digitale bewaring. Hierna volgt een vergelijking met PDF en beantwoording van een aantal veelgehoorde vragen. Vervolgens keren wij terug naar de drie strategieën uit hoofdstuk 1 waarin XML een rol speelt: inkapseling, migratie en natuurlijk XML zelf. Van elk van deze aanpakken wordt een praktijkgeval besproken. Algemene aspecten zoals metadata, beveiliging en opslag komen aan bod bij de meest betrokken strategie.Tot slot van dit hoofdstuk en deze white paper volgt een opsomming van de waardevolle eigenschappen van XML met een ontnuchterende opmerking.

4.1 XML en digitale bewaring

XML heeft zich in de digitale wereld inmiddels een solide positie verworven en haar veroveringstocht zet zich voort. Zij wordt steeds meer de lingua franca voor digitale gegevensuitwisseling, een gemeenschappelijk taal vergelijkbaar met het Engels nu en het Latijn in de Middeleeuwen. Niet alleen wegens deze wijde verspreiding is XML uiterst belangrijk voor digitale bewaring, maar ook omdat zij de achilleshiel van digitale documenten beschermt: de afhankelijkheid van uitstervende besturings- en toepassingsprogrammatuur. Dit doet zij door platform- en programma-onafhankelijk te zijn. Een belangrijke rol hierbij speelt de scheiding tussen inhoud, structuur en vorm. Omdat juist de digitale *vorm* voor veel afhankelijkheid zorgt (van bijvoorbeeld het bedrijf Adobe in het geval van PDF), wordt de kans veel groter dat de in XML geabstraheerde *inhoud* een lange reis door de tijd kan maken. Een redelijk intelligent wezen c.q. computer zal over een paar honderd jaar een digitaal object geschreven in XML kunnen ontcijferen. Zelfs als XML eventueel net als Latijn[44] een dode taal wordt, zou onze verre nazaat met de XML-specificatie[45] in de hand er uit moeten komen.

Hoofdstuk 3 liet zien dat sommige van de verwanten van XML al weer op hun retour zijn (bijvoorbeeld de DTD) en andere zich nog in de praktijk moeten bewijzen (bijvoorbeeld XML-schema en XSL). Algemene acceptatie kan een standaardisatie-orgaan noch een ministerie namelijk afdwingen. Men vergete trouwens niet dat XML zelf pas vier jaar oud is en veel van haar kinderen nog in de couveuse liggen of er net uitkomen. Men kan dus moeilijk verwachten dat zij en haar familie op stel en sprong het probleem van digitale duurzaamheid oplossen. Op dit punt kunnen wij ook weinig hulp van de softwareleveranciers verwachten. Deze zullen blijven proberen monopolieposities te verwerven door particuliere standaarden in te voeren;[46] bovendien zijn XML-toepassingen als digitale marktplaatsen en webdiensten commercieel vele malen aantrekkelijker dan digitale duurzaamheid. Het is dus zaak om net als de Regeling geen

[44] Het Vaticaan tracht overigens het Latijn in leven te houden door voor nieuwe begrippen nieuwe Latijnse woorden vast te stellen. Latijn en Grieks zijn net als XML zeer 'extensible': men kan gemakkelijk nieuwe woorden als kardiogram en automobiel samenstellen die ook nog in de gehele wereld gebruikt worden.
[45] Plus eventueel de Unicode-tabel, een schema van het type document met documentatie en misschien een Nederlands en Engels woordenboek (voor de betekenis van de tag-namen).
[46] Of in het geval van hardware-leveranciers erop aansturen dat er veel geëmuleerd wordt, hiervoor is namelijk veel geheugenruimte en dus hardware nodig. Veel versies en een korte levensduur van soft- en hardware en de daaruit voortvloeiende conversies en migraties hebben ook een positief effect op de omzet van de producenten.

afwachtende houding aan te nemen, maar voortvarend aan de slag te gaan. Op het gebied van digitale preservering is XML namelijk nog weinig beproefd; er zijn voornamelijk theoretische beschouwingen aan deze taal gewijd.

4.2 XML versus PDF?

Vaak worden XML en PDF opgevoerd als de twee concurrenten waaruit men moet kiezen om een document duurzaam te bewaren. In deze richtingenstrijd delven de andere twee standaarden die door de Regeling worden genoemd, TIFF en SGML, het onderspit. Omdat PDF en XML zo complementair zijn ligt het echter meer voor de hand om te besluiten XML *en* PDF te gebruiken ter bewaring van een document dan te kiezen tussen XML en PDF. Beide standaarden inzetten is ook een vorm van risicospreiding: kan men het ene formaat over honderd jaar niet meer lezen, dan heeft men altijd het andere nog. Idealiter ontstaat er een open standaard die de rol van PDF over kan nemen, zodat men voor het veiligstellen van het digitale erfgoed niet afhankelijk is van één enkel bedrijf.

In hoofdstuk 3 heeft de lezer kunnen zien dat men PDF kan genereren uit XML met behulp van XSL-FO. Een PDF-bestand naar XML (of een ander formaat) converteren daarentegen is een herculeswerk. Het is nu eenmaal gemakkelijker een mammoet tot leven te wekken op basis van gepreserveerd DNA (lees XML) dan op basis van een foto (lees PDF).

4.3 Vragen en tegenwerpingen

Haalt XML de honderd jaar? Dat is moeilijk te voorspellen. Als men kijkt naar het tempo waarin deze taal wordt geaccepteerd en geïntegreerd, zou men verwachten dat onze computers over tientallen jaren nog met de rechthoekige XML-haakjes zullen werken. Een ontwikkeling als met de vergelijkbare onafhankelijke standaard ASCII, die nu tot Unicode evolueert, zou eventueel kunnen plaatshebben. En zelfs als XML weer in onbruik raakt zijn de XML-bestanden redelijk gemakkelijk te converteren naar een nieuw formaat. In XML zijn de gegevens namelijk gestructureerd vastgelegd en van metadata voorzien: ideale omstandigheden voor een geautomatiseerde migratie.

XML legt alles open; hoe zit het met het waarborgen van de authenticiteit en integriteit? Dat is inderdaad een zorgpunt, maar dit geldt niet alleen voor XML. Per slot van rekening kan ook aan het binaire formaat van PDF gerommeld worden en dit geldt ook voor papieren documenten.

Is XML niet te ingewikkeld? Dat valt best mee; het ‘onderwater-scherm’ van WP 5.1 (waarin de verborgen opmaaktekens zichtbaar waren) was ingewikkelder. Als men eenmaal het concept achter SGML en XML begrijpt, is de rest technische invulling. Bovendien is het de bedoeling om de gewone gebruiker niet in aanraking te laten komen met de puntige XML-haakjes.

Is er niet het gevaar dat leveranciers toch hun proprietary substandaarden gaan ontwikkelen? Ja, hiervoor bestaat een gevaar, want op die manier kan men een monopolie-positie verwerven. Microsoft leek deze weg in te slaan door met een eigen schema-standaard, XDR-schema, te komen, toen ze het definiëren van de officiële schema-standaard door de W3C te traag vond gaan. Uiteindelijk conformeerde de softwarereus uit Redmond zich wel aan de schema-specificatie van de W3C. SQL, dat in de Regeling wordt opgelegd als taal voor database-queries, is een voorbeeld van een taal waaraan producenten elk hun eigen specifieke features toevoegen; de vraag die daarom opkomt is: welk dialect van SQL bedoelt de Regeling? XML daarentegen is een strikt gecodificeerde taal; aanvullingen op haar worden als aparte standaarden

voorgesteld aan de W3C en, na een streng selectieproces, eventueel geaccepteerd als officiële Recommendation.

Heeft de invoering van XML niet veel voeten in aarde? Er is zeker sprake van een leercurve die niet moet worden onderschat. Omdat XML op zoveel gebieden wordt ingezet, mag men echter verwachten dat over enkele jaren de kennis en expertise op dit gebied gemeengoed is geworden. Op dit fundament kan men dan specifieke XML-expertise ten behoeve van digitale archivering opbouwen.

4.4 Strategie 4: Inkapseling

Deze benadering richt zich op behoud van het oorspronkelijke formaat. XML wordt vaak genoemd als taal waarin metadata en instructies over het te bewaren object kunnen worden vastgelegd. In de deze sectie passeren eerst een aantal termen die in deze context worden gebruikt de revue. Na een korte bespreking van metadata volgt een beschrijving van het VERS-project.

4.4.1 Wrappers, containers, inkapseling en kapstok

De Regeling noemt een 'XML-wrapper ' als middel om metadata aan PDF en TIFF-bestanden toe te voegen. Hoewel men zich er iets bij kan voorstellen, heeft deze term (nog) niet een vaste betekenis. De San Diego Supercomputer Center beschouwt bijvoorbeeld een wrapper als een stuk software dat door een 'mediator' wordt gebruikt.[47] Het Roquade project daarentegen gebruikt juist de term 'container' voor de 'verpakking' van digitale archiefbescheiden.[48] Een stap verder dan inkapseling is om XML ook te gebruiken als 'kapstok', waaraan (delen van) documenten in bijv. TIFF- of PDF-formaat worden gehangen. In dit geval vormt XML de ruggegraat van het digitale archiefbescheid.

4.4.2 Metadata

In het vorige hoofdstuk hebben we gezien dat metadata, d.w.z. informatie over informatie, integraal onderdeel is van XML (in de vorm van tags). Ook voor het vastleggen van metadata in de nauwere, archivistische betekenis van het woord biedt XML uitstekende faciliteiten. Vandaar dat men XML op dit punt tegenkomt bij de andere strategieën, bij emulatie zou XML bijv. de taal kunnen zijn waarin technische metadata wordt verwoord. Adobe, eigenaar van de PDF-standaard, heeft onlangs het eXtensible Metadata Platform gelanceerd,[49] dat ook gebruik maakt van XML voor het vastleggen van metadata.

Als men een vaste verzameling van metadata heeft afgesproken (en dat is vaak veel moeizamer dan de technische implementatie!), kan deze worden vastgelegd in de vorm van een XML-schema dat weer gebruikt wordt door schema's voor specifieke documenten. Deze standaardisering is belangrijk, want anders weet een digitaal archief niet wat het aan metadata kan verwachten.

---

[47] "A wrapper is a piece of software that acts as a translator between the native format of an information source and a commonly agreed protocol (XML for us). The end-user or application interacts with a piece of software called mediator that collects information from multiple wrappers ", pagina 4 van *Methodologies for the Long-Term Preservation of and Access to Software-Dependent Electronic Records*, http://www.sdsc.edu/NHPRC/Pubs/nhprcfeb2k.doc.

[48] "It was decided to work out the idea of XML containers. So the Archival Information Packages (AIP), to be stored in the electronic archive, will be wrapped in XML." *An electronic Archive for academic communities* (Dekker, R. et al, Nov 2001). Het begrip AIP is afkomstig uit het Open Archive Information System (OAIS)-model.

[49] Zie http://partners.adobe.com/asn/developer/xmp/download/docs/MetadataFramework.pdf.

4.4.3 Casus: VERS

In Australië is met de Victorian Electronic Records Strategy een pioniersproject met succes afgesloten. In het eindrapport (uit 1999)[50] wordt met gepaste trots gemeld dat het mogelijk is om electronische archiefbescheiden gedurende een lange termijn te bewaren. Hiertoe wordt een standaard formaat voorgesteld met de volgende kenmerken:

- De documenten, context en authenticatie moeten worden ingekapseld in één object en niet verspreid worden opgeslagen.
- De gegevensstructuur moet het aanbrengen van lagen van metadata (het 'ui-model') mogelijk maken.
- XML moet gebruikt worden voor het coderen van de ingekapselde archiefbescheiden.
- Elk elektronisch archiefbescheid moet een digitale handtekening krijgen.

In de demonstrator die als product van het project werd opgeleverd, werden de documenten zelf omgezet naar PDF. Omdat PDF een binair formaat is en XML juist op tekst gebaseerd, moesten de PDF-bestanden vòòr de inkapseling in XML omgezet worden naar tekstbestanden.[51]

4.5 Strategie 6: Migratie (naar XML)

Gestructureerde gegevens zoals die zich in een database of een spreadsheet bevinden lenen zich erg goed voor migratie naar XML. In principe zou men de inhoud van database-tabellen één-op-één kunnen vertalen naar elementen in XML. Hiermee zou echter ook veel non-informatie (samenhanged met de technische implementatie van de oude database) mee verhuizen naar de te archiveren bestanden. Migratie naar XML zal minder haken en ogen hebben als in de toekomst bij het ontwerp van een systeem rekening gehouden wordt met een uiteindelijke verhuizing naar XML.

In deze sectie wordt een recent uitgevoerd project op het gebied van migratie van databases gepresenteerd. Hierna komen twee onderwerpen, integriteit en opslag, aan de orde, die uiteraard ook voor de andere strategieën van belang zijn.

4.5.1 Casus : Databases van Arbeidsvoorziening

Voor Arbeidsvoorziening heeft Cap Gemini Ernst & Young (CGE&Y) een proof of concept (POC) uitgevoerd die heeft aangetoond dat het technisch haalbaar is om de inhoud van databases om te zetten naar XML. Arbeidsvoorziening, die op het moment van schrijven geliquideerd wordt, moet namelijk haar digitale bestanden op orde hebben voordat ze deze kan overdragen aan haar rechtsopvolger(s). Het betreft in eerste instantie terabytes aan gegevens in databases behorende bij meer dan tien systemen. De noodzaak voor gegevensbehoud is extra acuut wegens de miljoenenclaims op het gebied van het Europees Sociaal Fonds (ESF).

De Regeling bevat, zoals in Hoofdstuk 2 al werd vermeld, het voorschrift dat databases in het oorspronkelijk formaat of als flatfile moeten worden bewaard, terwijl de structuur bij voorkeur in een DTD moet worden vastgelegd. In het geval van Arbeidsvoorziening heeft de Rijksarchivaris het advies gegeven om de inhoud van databases om te zetten naar XML. Binnen een maand (maart van 2002) kon worden aangetoond dat:

- De inhoud van een database kan op transparante wijze worden uitgelezen en in ruwe XML omgezet.

[50] Zie http://www.prov.vic.gov.au/vers/final.htm.
[51] In dit geval gebruikte men de bekende standaard Base64, die ook door emailsystemen wordt toegepast.

- Deze XML kan vervolgens stapsgewijs worden omgezet naar well-formed XML die is verrijkt met metadata over in de eerste plaats de vernietigingstermijn;[52] op basis van deze metadata kunnen gegevens wier 'houdbaarheidsdatum' is verstreken worden vernietigd.
- Een rudimentaire bevragingstool kan de XML doorzoeken en de resultaten presenteren met behulp van XSL(T).

De omzetting naar XML was een ingrijpende migratie, die niet zozeer technisch als wel theoretische vragen opleverde, bijvoorbeeld:

- Wat is het domein van de te transformeren data: hoe kunnen we beslissen welke database-tabellen moeten worden meegenomen (veel tabellen zijn namelijk applicatie-specifiek en niet inhoudelijk)?
- Hoever mag men denormaliseren: welke tabellen mogen worden samengevoegd zonder verlies van toegankelijkheid en authenticiteit?
- Hoe vertalen wij de abstracte handelingen genoemd in het Basis Selectie Document naar houdbaarheids-metadata per tabel/element?

De Regeling schrijft geen standaard voor ter bewaring van de functionaliteit van een database.[53] Dat hiervoor geen algemeen aanvaard recept bestaat is begrijpelijk, want applicatie-migratie is vele malen ingewikkelder dan data-migratie. Toch is het vanuit archivistisch oogpunt belangrijk dat toekomstige generaties inzicht hebben in wat met deze terabytes aan gegevens werd gedaan, hoe ze door de gebruiker werden gezien en op welke wijze ze konden worden bevraagd. Wat betreft het laatste punt schrijft de Regeling (in art. 6) SQL voor als taal voor queries. Over deze queries worden geen nadere aanwijzingen gegeven. Omdat tijdens de POC de databasegegevens werden omgezet naar XML, werd voor bevraging niet van SQL maar van XSL(T) gebruik gemaakt. De bouw van een volwaardig bevragingstool, de keuze van opslag voor de XML-bestanden en de opzet van het benodigde digitale archief vielen buiten de scope van de POC en zullen op een later tijdstip aan bod moeten komen.

4.5.2 Integriteit

Een aspect dat bij migratie een grote rol speelt is de zekerheid dat de integriteit van de gemigreerde gegevens intact blijft. Dit is zeker het geval bij transformatie naar XML waarbij irrelevante data worden weggelaten en metadata toegevoegd: de resulterende XML-file ziet er heel anders uit dan de download uit de database. Gelukkig kan men in dit opzicht gebruik maken van de ruime ervaring op migratie-gebied. Door middel van transparante procedures zal de Rijksarchiefinspectie of eventueel een rechter het vertrouwen moeten hebben dat bij de totstandkoming van de XML-bestanden en hun bewaring niet met de gegevens is geknoeid. In dit opzicht verschilt het digitale bestand niet wezenlijk van het papieren archiefbescheid (waaraan ook iets toegevoegd kan worden). Het samenwerkingsverband tussen mens en machine dat de Regeling in art. 1a noemt bij de omschrijving van het begrip archiefbeheerssysteem is ook hier van toepassing.[54]

[52] Op grond van het Basis Selectie Document dat per handeling (art. 1h van de Regeling: een complex van activiteiten ter vervulling van een taak of op grond van een bevoegdheid) de bewaartermijn voorschrijft. Een Basis Selectie Document wordt gebruikt als instrument voor de selectie van archiefbescheiden.
[53] Toelichting op art. 6: "Voor databases is een keuze voor een specifieke standaard, die voldoet aan de eisen die authenticiteit stelt met betrekking tot de verschijningsvorm van schermen en rapporten, (nog) niet mogelijk gebleken."
[54] "Een geheel van mensen, methoden, procedures, gegevensverzamelingen, opslag-, verwerkings- en communicatieapparatuur en andere middelen, bestemd tot het beheer van archiefbescheiden."

4.5.3 Opslag

Het onderwerp archiefbeheerssysteem brengt ons op de vraag hoe de producten van de migratie, de XML-bestanden, kunnen worden opgeslagen. Hierop volgt onmiddellijk de wedervraag: wat wilt u met deze bestanden doen? Als men slechts één keer in de vijftig jaar een vraag hierover verwacht, dan zou men kunnen overwegen om ze slechts op een duurzame fysiek drager vast te leggen en dit medium aan het Nationaal Archief over te dragen. Mits de XML-files voorzien zijn van goede documentatie (zoals de Regeling voorschrijft) kan men ze te rechter tijd uit de mottenballen halen.[55] Wil men snel toegang hebben of zelfs via een interface de bevrager ondersteunen (en hiermee de toegangsrechten regelen), dan moet men aan een nieuwe database denken. Op dit moment zijn er dan twee opties: een ('klassieke') relationele database of een native XML database. Een verdere bespreking van deze opties valt buiten de scope van deze paper.

4.6 Strategie 7: XML (vanaf het begin)

In de vorige strategie werd XML achteraf ingezet. Deze aanpak wordt een stuk gecompliceerder als er sprake is van ongestructureerde documenten waarvan men achteraf de structuur wil expliciteren in de vorm van XML-tags. Men kan dit niet altijd vol-automatisch doen, maar zal altijd met handwerk en menselijke controle geconfronteerd worden. Als men bijvoorbeeld in deze paper het element RegelingBegrip (zie 2.1) wil onderscheiden, dan is het gunstiger om direct bij invoer de tags aan te brengen. Het is niet verwonderlijk dat er initiatieven zijn om voor kantooromgevingssoftware XML als onderliggend formaat te gebruiken. Het open-source pakket OpenOffice (zie www.openoffice.org) is hiervan een voorbeeld. Ook gezien het feit dat Microsoft binnen Office XP ook gebruik maakt van XML lijkt de tendens te zijn dat de bijbehorende documenten XML vanaf het begin als opslagformaat krijgen.

4.6.1 Wil het authentieke document nu opstaan?

Als het originele formaat van een document al XML is, voldoet men hiermee ook aan de voorschriften van de Regeling. Eventueel kan men nog een stylesheet toevoegen, waarin opmaakinstructies zijn vastgelegd. Vaak zal echter het bestand dat met één of meerdere stylesheets wordt gegenereerd het bestand zijn dat men verzendt, c.q. publiceert. Dit kan, zoals we in hoofdstuk 3 zagen, bijvoorbeeld een HTML-, PDF-, of PostScript-bestand zijn. Juist omdat deze de vorm biedt die uiteindelijk door de lezer wordt gezien, zou een archivaris juist deze bestanden met vermelding van verzend- c.q. publicatie-moment etc. willen bewaren. Dit brengt het aantal van de typen bestanden dat samenhangt met archivering in XML op vier:

1. XML-documenten (instances)
2. Schema's die de structuur van een type XML-document vastleggen
3. Stylesheets die opmaak- c.q. transformatieregels bevatten
4. 'Bevroren' vormen met eventueel timestamps

De Regeling geeft aan dat wat betreft tekstbestanden de types 1 en 3 moeten worden bewaard.

4.6.2 Casus: Uitgaande e-mail van het Testbed

Om het gebruik van XML voor e-mail in de praktijk te brengen is een demonstrator ontwikkeld die bestaat uit twee applicaties: een webservice en een extensie voor het Outlook-e-mailprogramma van Microsoft. Outlook is op een zodanige wijze aangepast

[55] Het probleem van bewaring van de oorspronkelijke functionaliteit en de bevraging, die bij de gepresenteerde casus aan de orde zijn gekomen, laten we hier even buiten beschouwing.

dat de mail die door de gebruiker wordt geschreven ‘onder water’ in XML formaat wordt opgebouwd. Daarnaast wordt de gebruiker geconfronteerd met een invulscherm waarin hij metadata in moet voeren, welke in de XML van het emailbericht wordt verwerkt. Uiteindelijk kan de auteur een preview in HTML van het e-mailbericht bekijken en versturen. De validatie van de XML, de transformatie naar HTML en het centrale opslaan van de e-mail wordt uitgevoerd door een webservice. Een aandachtpunt voor deze pilot is de realisatie van de XML transformatie en bewaring zonder dat het gebruiksgemak van het mailen word beperkt. Twee belangrijke winstpunten voor de organisatie zijn dat (a) officiële e-mail gevalideerd wordt vòòr verzending en een vaste opmaak heeft en (b) deze emailberichten centraal opgeslagen worden in XML met de benodigde metadata.

4.7 Tot slot: de voordelen van XML en een ontnuchtering

Ter afsluiting worden hieronder de belangrijkste eigenschappen van XML nog eens opgesomd:

- Platform- en programma-onafhankelijk
- Een open standaard, breed geaccepteerd en toegepast
- Het concept van scheiding van inhoud, structuur en vorm in de praktijk
- Uitbreidbaar en controleerbaar (net als een natuurlijke taal)
- Leesbaar voor mens en machine
- Gratis

Hoewel XML veel perspectieven biedt op het gebied van digitale duurzaamheid, is het tot slot belangrijk om te benadrukken dat XML niet het wondermiddel is dat men voor elke digitale verduurzaming moet voorschrijven. XML, haar nevenstandaarden en hun gebruik vormen een complexe materie; veel pionierswerk zal nog moeten worden verricht.

5 *Bibliografie*

Bourret, Ronald	XML and Databases (2002) http://www.rpbourret.com/xml/XMLAndDatabases.htm
COVAX	State of the Art (2000) http://www.covax.org/public_docum/p_documets.htm
Dekker, R et al	An electronic archive for academic communities (Nov 2001) http://www.library.tudelft.nl/e-archive/Documenten/Resultaten/e.archive.acad.comm2.doc
Dürr, E & Lourens, W	Emulation and Conversion: Design and Implementation of an Electronic Archive (Nov 2001) http://www.library.tudelft.nl/e-archive/Documenten/Resultaten/report3.pdf
Dürr, E; Lourens, W; & Meer, K.v.d.	Emulation and Conversion: Organisational and Architectural Overview of an Electronic Archive (2001) http://www.library.tudelft.nl/e-archive/Documenten/Resultaten/reportone13.pdf
Gilheany, Steve	XML for Records Managers (2002) http://www.archivebuilders.com/whitepapers/22033p.pdf
Hedstrom, Margaret	Digital Preservation: Problems and Prospects (2001) http://www.si.umich.edu/CAMILEON/camileon%20Presentations/margaretpresentation.pdf
Hodge, Gail	Best practices for Digital Archiving: An Information Life Cycle Approach (2000) http://www.dlib.org/dlib/january00/01hodge.html
InterPARES	Preservation Strategies for Electronic Records, Round 1 – Where are We now? Obliquity and Squint http://www.interpares.org
Klyne, G	An XML Format for Email Messages (2002) http://www.ietf.org/internet-drafts/draft-klyne-message-rfc822-xml-03.txt
Lorie, Raymond A	A Project on Preservation of Digital Data (2001) http://www.rlg.org/preserv/diginews/diginews5-3.html - feature2
National Library of Australia	A Draft Research Agenda for the Preservation of Physical Format Digital Publications (1998) http://www.nla.gov.au/policy/rsagenda.html
Ploeg, Dr. F. van der	Regeling geordende en toegankelijke staat archiefbescheiden (2002) http://www.nationaalarchief.nl/images/3_2597.doc
Public Record Office (Kew, UK)	Guidelines for the Management, Appraisal and Preservation of Electronic Records (1999) http://www.pro.gov.uk/recordsmanagement/eros/guidelines/default.htm

Public Record Office Victoria (Australië)	Victorian Electronic Records Strategy Final Report http://www.prov.vic.gov.au/vers/published/final.htm
Rothenberg, Jeff & Bikson, Tora	Digital Preservation: Carrying Authentic, Understandable and Usable Records Through Time (1999) http://www.digitaleduurzaamheid.nl/bibliotheek/docs/final-report_4.pdf
SDSC/NHPRC	Methodologies for the Long-Term Preservation of and Access to Software-Dependent Electronic Records http://www.sdsc.edu/NHPRC/Pubs/nhprcfeb2k.doc.
Testbed Digitale Bewaring	Migratie: Context en Huidige Stand van Zaken (Den Haag, december 2001) http://www.digitaleduurzaamheid.nl/bibliotheek/docs/white_paper_migratie.pdf

6 *Websites*

http://www.digitaleduurzaamheid.nl Website Testbed Digitale Bewaring

http://www.covax.org/ Contemporary Culture Virtual Archives in XML

http://www.nationaalarchief.nl/images/3_2598.doc Website Nationaal Archief; Regeling Geordende en toegankelijke staat archiefbescheiden

http://www.jiscmail.ac.uk/ *JISC* Listserv archives

http://www.w3.org/TR/ W3C Technical Reports and Publications

http://www.interpares.org/ InterPARES Web site

http://www.rlg.org/preserv/diginews/ RLG Diginews website

http://www.prov.vic.gov.au/vers/published/final.htm VERS Final Report site

http://www.pro.gov.uk/recordsmanagement/eros/ PRO (UK) Records Management

http://www.sdsc.edu/NHPRC/Pubs/nhprcfeb2k.doc. San Diego Supercomputer Centre

http://www.antwerpen.be/david/ Project DAVID website

http://www.si.umich.edu/CAMILEON/ Project CAMILEON website