Multimedia Metadata Description Standards and Applications

Luk VLOEMANS

promotor :
Prof. dr. Wim LAMOTTE

Eindverhandeling voorgedragen tot het bekomen van de graad
Licentiaat in de informatica afstudeervariant databases

Woord vooraf

In de eerste plaats wil ik graag mijn promotor en begeleiders prof. dr. Wim Lamotte, dr. Kris Luyten en Peter Quax danken voor het regelmatig opvolgen van mijn thesis, voor de raadgeving en de feedback.

Een speciaal woord van dank gaat uit naar mijn ouders. Ze hebben mij moreel gesteund en de kans gegeven om mijn studies te volgen. Graag wil ik ook mijn broers Marc en Ben, en mijn beste vrienden Jo, Pieter en Roland bedanken voor hun aanmoedigingen en de leuke ontspanningsmomenten.

Een laatste dankwoord gaat uit naar Joshua Tauberer, voor het ter beschikking stellen en regelmatig bijsturen van bugs in zijn RDF-framework.

Inhoudsopgave

Lijst van figuren

List of Listings

Deel I

Inleidende hoofdstukken

Hoofdstuk 1

Introductie

Vooruitgang in de wetenschap heeft de computer omgetoverd van een kolossale mechanische machine tot een minuscule computerchip. Tegenwoordig vind je in haast elk gezin minstens één exemplaar. Veel mensen gebruiken de huidige generatie computers voor hun correspondentie, administratie of als multimediacentrum.

Met de opkomst van technologieën zoals digitale camera's, draagbare MP3-spelers en zakcomputers worden thuiscomputers overspoeld met digitale bestanden. Annotaties helpen bij het structureren, indexeren en doorzoeken van bestanden. Deze annotaties noemen we *metadata*. Metadata is informatie over informatie. Het bestaat uit gestructureerde manueel-toegevoegde (menselijke) annotaties (zoals auteur, beschrijving van het bestand) alsook annotaties die automatisch werden toegevoegd aan het bestand. (bijvoorbeeld creatiedatum, bestandsgrootte, e.a.)

Om metadata te structureren, en vooral om haar specificatie hard te maken beschrijft men metadata in standaarden. Deze thesis geeft de lezer een overzicht van huidige en toekomstige metadata standaarden.

Helaas bestaan er voor één enkel bestandstype (bv, een foto) dikwijls verschillende metadata standaarden. Juist zoals Kelvin en Celcius op hetzelfde ogenblik een maatstaf uitvonden ter beschrijving van temperatuur, bestaan er een heleboel organisaties die tegelijkertijd een standaard ontwikkelden ter beschrijving van een bepaald multimedia bestandstype. Vaak hebben deze standaarden overlappende elementen. Door een overaanbod aan standaard schema's worden softwareproducenten die metadata willen uitbuiten benadeeld. Ofwel verkiest men slechts een enkel aantal standaarden te ondersteunen in een applicatie , ofwel moet men extra energie investeren om zoveel mogelijk standaarden te ondersteunen.

Metadata standaarden die verschillende media beschrijven hebben ook vaak overlappende elementen. Deze informatie kan nuttig gebruikt worden door machines om verschillende type bestanden aan elkaar te linken op basis van auteur, classificatie of andere gemeenschappelijke elementen.

In hoofdstuk 3 bespreken we metadata standaarden die worden gebruikt ter annotatie van multimedia bestanden. In hoofdstuk 4 trachten we deze standaarden te vergelijken op basis van hun uitdrukkingskracht.

De resultaten van deze vergelijking worden in hoofdstuk 6 gebruikt in een applicatie die een gebruiker in staat stelt vragen te stellen op een geannoteerde set multimedia, zonder zich

zorgen te maken over de onderliggende metadata structuur.
In hoofdstuk 5 wordt het gebruik van de applicatie verduidelijkt aan de hand van een eenvoudige use case.

Hoofdstuk 2

Metadata Representatie en Ondervraging

Inhoudsopgave

In hoofdstuk 3 bespreken we metadata standaarden. Drie vormen van metadata representatie komen steeds terug bij deze standaarden. In dit hoofdstuk overlopen we kort de drie meest gebruikte methodes voor representatie van Metadata. Verder bespreken we SPARQL, een querytaal die werd gebruikt in de webclient applicatie (sectie 6.3) die in het kader van deze thesis werd ontwikkeld.

2.1 Metadata Representatie

2.1.1 EXtensible Markup Language

Voor gestructureerde opslag en uitwisseling van informatie kan men tegenwoordig beroep doen op de EXtensible Markup Language (XML). XML is een W3C recommendation. Het is een markup taal voor het beschrijven en structureren van informatie. XML-tags zijn niet voorgedefinieerd zoals in HTML. Vaak legt men de boomstructuur van een XML-specificatie vast door een Document Type Definition (DTD) of een XML Schema [XML].
Zoals in hoofdstuk 3 zal blijken wordt XML door een groot aantal multimedia metadata standaarden geadopteerd als definitie voor het structureren van metadata-elementen. De mogelijkheid om XML-documenten uit te breiden a.d.h.v. namespaces is voor vele metadata standaarden een belangrijk voordeel. Sommige schema's erven elementen over van reeds gevestigde metadata schema's. Dit hergebruik van gevestigde waarden bevordert compabiliteit tussen bedrijven of personen die informatie wensen uit te wisselen.

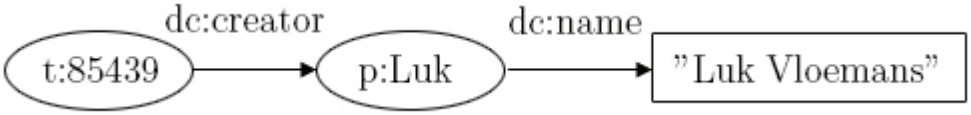
2.1.2 Resource Description Framework

Het Resource Description Framework [RDF] (RDF) van het W3C maakt de laatste jaren haar opkomst als werktuig voor metadata schema's. RDF werd ontwikkeld voor representatie van onderlinge relaties en eigenschappen van resources op het internet, maar kent ook zeker haar toepassing in multimedia metadata schema's, temeer omdat ook multimedia wordt uitgewisseld en ter beschikking gesteld op het internet.
RDF wordt vaak uitgedrukt in XML en voegt dankzij haar datamodel een semantische betekenis toe aan relaties tussen resources onderling of relaties tussen resources en hun eigenschappen. Een RDF-document bestaat ruwweg uit een opsomming van *statements*. Een statement is een triple waarmee een *subject* wordt gerelateerd aan een *object* d.m.v. een *predicate*. Elk subject en predicate wordt voorgesteld door een URI. Een object kan een literal (een waarde) voorstellen, of kan ook een URI bevatten. Beschouw als voorbeeld devolgende twee statements:

```
Namespace t = http://www.thesis.be/
Namespace dc = http://purl.org/dc/elements/1.1/
Namespace p = http://www.person.be/
```

Subject	Predicate	Object
t:85439	dc:creator	p:Luk
p:Luk	dc:name	"Luk Vloemans"

Figuur 2.1 geeft een voorbeeld van deze RDF-statements. Het eerste statement maakt duidelijk dat dit document (URI: http://www.thesis.be/85439) werd opgesteld door een object met URI *http://www.person.be/Luk*. Het object heeft als naam een literal-waarde "Luk Vloemans".



Figuur 2.1: RDF principe

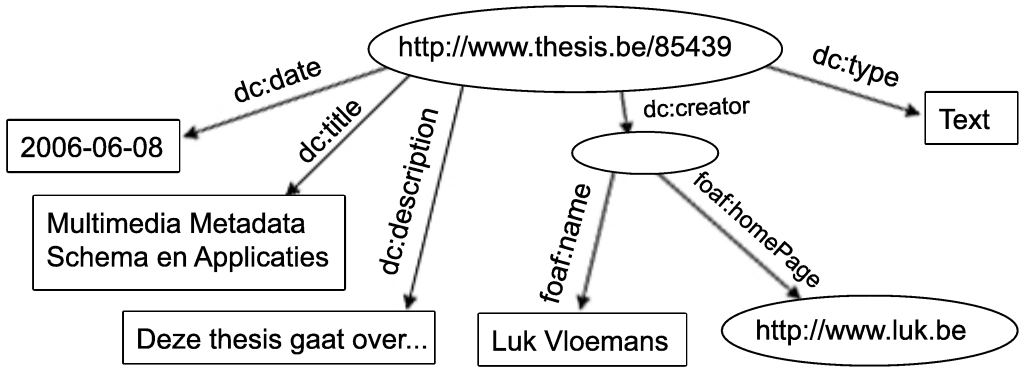
Figuur 2.1 demonstreert eveneens hoe RDF kan worden voorgesteld als een graaf. Elk statement in een RDF-bibliotheek introduceert een nieuwe boog in haar graaf. Zoals in hoofdstuk 3 zal blijken wordt ook RDF gebruikt in een aantal multimedia metadata schema's. In listing 2.1 geven we een meer uitgebreid voorbeeld van RDF, in conjuntie met Dublin Core (zie sectie 3.1.2.) en FOAF, een metadata schema ter beschrijving van personen. Figuur 2.2 geeft een visuele weergave van dit voorbeeld.

Listing 2.1: Dublin Core in RDF

```
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:dc="http://purl.org/dc/elements/1.1/"
 xmlns:foaf="http://xmlns.com/foaf/0.1/">

 <rdf:Description rdf:about="http://www.thesis.be/85439">
```

```
    <dc:creator>
     <rdf:Description foaf:name="Luk Vloemans">
      <foaf:homePage rdf:resource="http://www.luk.be"/>
     </rdf:Description>
    </dc:creator>
    <dc:title>
     Multimedia Metadata Schema en Applicaties
    </dc:title>
    <dc:description>
     Deze thesis gaat over...
    </dc:description>
    <dc:date>2006-06-08</dc:date>
    <dc:type>Text</type>
 </rdf:Description>
</rdf:RDF>
```



Figuur 2.2: RDF voorbeeld

2.1.3 Metadata Tagging

Een derde manier om metadata te bewaren is door haar op te slaan in de bytecode van een bestand. Deze methode beschouwen we binnen deze thesis als *tagging*. Ze is vergelijkbaar met een hashmap in een programmeertaal. Elk metadata-element heeft een unieke sleutel en waarde. De betekenis van de sleutels is meestal niet zelfbeschrijvend (in tegenstelling tot zelfbeschrijvende metadata-elementen in XML of RDF.) Metadata standaarden die hun informatie m.b.v. tagging opslaan voorzien vaak de mogelijkheid om hun elementen-set uit de breiden met nieuwe (domein-specifieke) sleutels. Echter, het gevaar dat nieuwe sleutels (die niet worden beschreven in de specificatie van een standaard) worden overschreven door general-purpose applicaties is bij tagging standaarden groter dan bij standaarden die zich baseren op XML of RDF.

2.2 Metadata Ondervraging

Sinds de conceptie van het RDF datamodel groeit het onderzoek naar een efficiënte standaard querytaal voor RDF. Mogelijk kan men RDF-data converteren naar het relationeel model en haar ondervragen met SQL [CDES05]. Voordelen hiervan zijn de bekende syntax van SQL en

reeds brede ondersteuning van SQL in huidige databasesystemen. Men kan ook een nieuwe querytaal definiëren die tracht de graafstructuur van RDF optimaal te benutten. Voorbeelden hiervan zijn RQL of Triple. Schematalen die beide de RDF graafstructuur benutten en tevens syntactisch gelijken op SQL zijn RDQL en SPARQL.
In deze thesis kozen we voor SPARQL als ondervragingstaal voor RDF. SPARQL [SPA06] staat voor de *SPARQL RDF Query Language*. Haar specificatie bevat een querytaal, een data access protocol (voor gebruik in webservices) en een beschrijving van de resultset van een SPARQL query. De Webclient applicatie (sectie 6.3) van deze thesis maakt gebruik van de querytaal en haar resultset. SPARQL werd geïntroduceerd door het W3C en DAWG (Data Access Working Group) en kan worden toegepast op databases met mappings naar RDF. De specificatie is heden een W3C candidate recommendation en wordt reeds gebruikt in verscheidene toepassingen van het semantisch web.

SPARQL ondervraagt een RDF-graaf d.m.v. een *subgraaf* die bestaat uit RDF-triples (subject, predicate, object). Deze SPARQL-subgraaf bevat typisch minstens één constante waarde (een literal of URI) en minstens één variabele. Variabelen worden onderscheiden van constanten door een vraagteken-prefix (bijvoorbeeld: *?alfa*.) De SPARQL-engine krijgt als input de RDF-graaf en de SPARQL-subgraaf en zal trachten de subgraaf te mappen op de RDF-graaf. Voor elke succesvolle mapping tussen de subgraaf en de RDF-graaf worden de geïnstancieerde variabelen teruggegeven.

Beschouw de SPARQL-query in Listing 2.2. De WHERE-clausule van de SPARQL-query bevat twee RDF-statements (gescheiden door een punt.) Deze RDF-statements vormen de subgraaf van de SPARQL-query. Figuur 2.3 geeft een visuele voorstelling van deze SPARQL-subgraaf. De ovalen met stippellijn stellen de variabelen voor. De SELECT-clausule is vergelijkbaar met de SELECT-clausule van SQL (projectie in een resultset.) Het SPARQL-voorbeeld in Listing 2.2, toegepast op de RDF-graaf in figuur 2.2 geeft als output "Luk Vloemans".

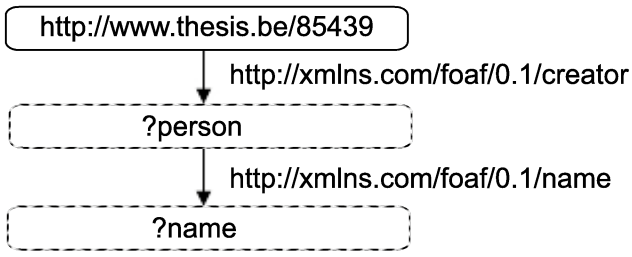
Listing 2.2: Eenvoudig SPARQL voorbeeld

```
SELECT ?name
WHERE
{
  <http://www.thesis.be/85439>
  <http://xmlns.com/foaf/0.1/creator>
  ?person .

  ?person
  <http://xmlns.com/foaf/0.1/name>
  ?name .
}
```

Een meer geavanceerd voorbeeld van een SPARQL-query wordt getoond in listing 2.3. Dit voorbeeld introduceert een aantal nieuwe clausules:

- De PREFIX-clausule introduceert een mechanisme voor namespaces.
- Het DISTINCT-sleutelwoord zorgt ervoor dat de resultset geen dubbels bevat.
- De FILTER-clausule wordt gebruikt om constraints toe te voegen aan een SPARQL-



Figuur 2.3: SPARQL-subgraaf voorbeeld

query (bijvoorbeeld: reguliere expressies, datum-vergelijkingen in combinatie met wiskundige- en booleaanse expressies.)

- De OPTIONAL-clausule wordt gebruikt om statements te definiëren, die niet noodzakelijk moeten worden bevredigd. Indien er variabelen in de OPTIONAL-clausule mappen op de RDF-graaf worden deze toegevoegd aan de resultset.
- De ORDER BY-clausule geeft aan in welke orde de resultset moet worden gerangschikt.
- De OFFSET waarde bepaalt hoeveel antwoorden van de resultset worden onthouden. De LIMIT waarde beperkt het aantal resultaten. Deze twee waarden zijn enkel nuttig in combinatie met ORDER BY. (De resultset van een SPARQL-query zonder ORDER BY clausule kan van volgorde wijzigen wanneer ze opnieuw wordt uitgevoerd.)

De beschouwde query ondervraagt een RDF-graaf van personen. Mensen met een e-mailadres (foaf:mbox) dat "@uhasselt.be" bevat worden teruggegeven in de resultset. De eerste tien resultaten worden onthouden. De resultset bevat maximaal vijf resultaten en wordt gerankschikt op e-mail. Wanneer er een persoon (voorgesteld als URI) een naam (dc:name) heeft wordt deze naam gemapped op de ?name-variabele. Naamloze personen kunnen echter ook voorkomen in de resultset (dankzij OPTIONAL). Wanneer de resultset dubbele resultaten bevat worden deze resultaten ontdubbeld.

Listing 2.3: SPARQL voorbeeld

```
PREFIX  dc:  <http://purl.org/dc/elements/1.1/>
PREFIX  foaf:  <http://xmlns.com/foaf/0.1/>
SELECT DISTINCT ?person ?name
WHERE {
  ?person foaf:mbox ?mailbox .
  OPTIONAL { ?person dc:name ?name }
  FILTER (regex(str(?mailbox), '@uhasselt.be') )
}
ORDER BY ?mailbox
LIMIT 5
OFFSET 10
```

Een SPARQL functionaliteit die niet voorkomt in bovenstaand voorbeeld is UNION. Dankzij UNION kunnen de resultaten van twee subgraven worden samengevoegd. Listing 2.4 geeft een voorbeeld van een SPARQL-query met een UNION. De query zal alle personen (URI) teruggeven wiens e-mailadres de substring "@uhasselt.be" bevat. Het predicaat *e-mailadres* mag voorkomen in namespace *foaf* of *vcard*.

Listing 2.4: SPARQL voorbeeld

```
PREFIX  vcard:  <http://www.w3.org/TR/vcard-rdf/>
PREFIX  foaf:  <http://xmlns.com/foaf/0.1/>
SELECT ?person
WHERE {
  { ?person foaf:mbox ?mailbox . }
  UNION
  { ?person vcard:EMAIL ?mailbox }
  FILTER (regex(str(?mailbox), '@uhasselt.be') )
}
```

In hoofdstuk 6 bespreken we hoe SPARQL werd gebruikt in de applicatie die in het kader van deze thesis werd ontwikkeld.

. De toegevoegde waarde van SPARQL t.o.v. reeds gevestigde querytalen zoals SQL of XQuery, is volgens [Mel05] haar manier om join operaties voor te stellen als een graaf. Waar men in SQL of XQuery voor elke boog een nieuwe JOIN-operatie of for-lus moet toevoegen kan men in SPARQL een join-operatie toevoegen door een extra statement toe te voegen aan de query subgraaf. Het is mogelijk SPARQL grotendeels om te zetten in traditionele SQL[1]. SPARQL en SQL beschikken over dezelfde grootorde uitdrukkingskracht. Zo is het in SPARQL onmogelijk de transitieve sluiting uit te drukken. Verder ondersteunt SPARQL in haar huidige specificatie een bescheiden set wiskundige-en vergelijkingsoperatoren. Zo is het bijvoorbeeld in de basis-specificatie van SPARQL onmogelijk een vierkantswortel, machtsverheffing of cosinus uit te drukken in een SPARQL filter. Voor zulke wiskunde berekeningen doet men beroep op *extention functions.* Deze zijn afhankelijk van het platform dat SPARQL ondersteunt.

[1] http://jena.sourceforge.net/sparql2sql/

Deel II

Onderzoek

Hoofdstuk 3

Multimedia Metadata Description Standards

Inhoudsopgave

In dit hoofdstuk bespreken we multimedia metadata standaarden. We richting extra aandacht op de uitdrukkingskracht van elke standaard (welke aspecten van de multimedia worden door de beschouwde standaard in het licht gezet). We bespreken tevens voor elke standaard in welke vorm haar beschrijvende set elementen wordt uitgedrukt (zoals besproken in sectie 2.1). In sectie 3.1 bespreken we generieke metadata standaarden (toepasbaar op alle media). In secties 3.2, 3.3 en 3.4 bespreken we respectievelijk standaarden die tekstuele, fotografische of audio multimedia annoteren. In hoofdstuk 4 gebruiken we de resultaten van dit hoofdstuk in een vergelijkende studie. Verbanden die in meerdere metadata standaarden terugkomen werden verwerkt in de applicatie die in het kader van deze thesis werd ontwikkeld (hoofdstuk 6).

Opmerking De beschouwde metadata standaarden voor video werden opgenomen in de generieke metadata standaarden.

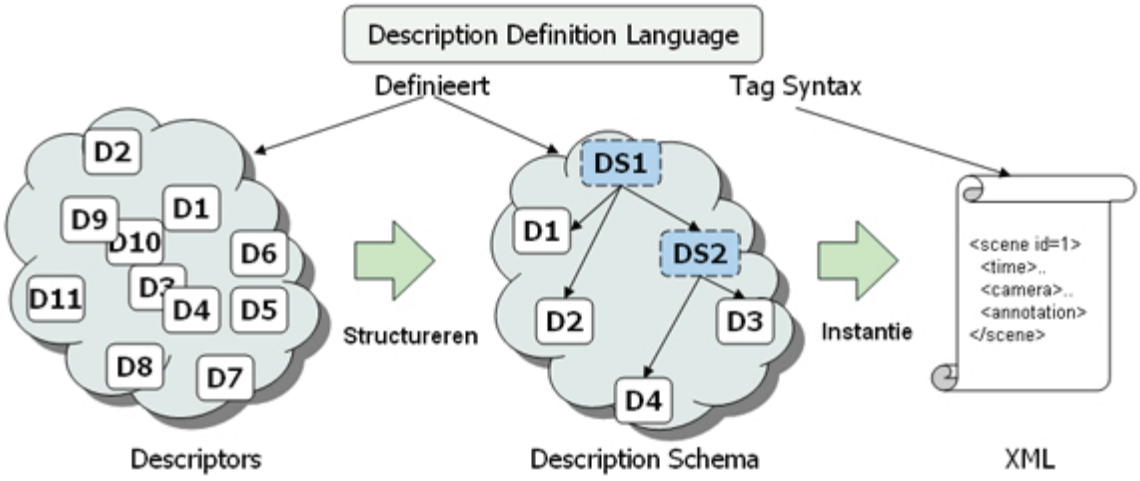
3.1 Generieke Metadata Standaarden

3.1.1 Multimedia Content Description Interface

Multimedia Content Description Interface [Mar04] (MPEG-7) is een ISO/IEC standaard door MPEG (Moving Picture Experts Group). Ze werd gelanceerd in 2001 en voorziet een set gestandariseerde tools ter beschrijving van multimedia.

MPEG-7 beoogt multimedia-zoekopdrachten zo simpel te maken als tekst-zoekopdrachten. Zo zou een eindgebruiker bijvoorbeeld a.d.h.v. MPEG-7 annotaties een audiobestand kunnen terugvinden door een fragment te neuriën (Query by humming). Hij zou d.m.v. een schets een set van videofragmenten of foto's kunnen opvragen die gelijken op die schets. Of hij zou tijdens het bekijken van een DVD enkel de videofragmenten kunnen selecteren waarin een bepaalde persoon meespeelt.

Om audiovisuele informatie te beschrijven beschikt MPEG-7 over een set van *descriptors* (D). Een descriptor beschrijft de syntax en semantiek van eigenschappen. Zo kan een descriptor low-level eigenschappen zoals kleur, motie of textuur beschrijven, alsook high-level eigenschappen zoals auteur of titel. De syntax van MPEG-7 descriptors wordt beschreven door de Description Definition Language (DDL), een uitbreiding[1] van XML-schema. De descriptors worden onderling gestructureerd door *description schema*(DS). MPEG-7 is uitbreidbaar door toevoeging van nieuwe descriptors en description schema. De onderlinge structuur van MPEG-7 descriptors, description schema en schema instanties wordt weergegeven in figuur 3.1.



Figuur 3.1: Abstractie van MPEG-7

De MPEG-7 standaard beschrijft enkel de syntax en semantiek van haar descriptions. Het

[1]De uitbreidingen aan XML-schema bestaan uit een aantal datatypes die noodzakelijk zijn voor annotatie van audiovisuele informatie. Bv: matrices, vectors en duratie in tijd.

annoteren van de audiovisuele informatie, alsook de metadata onderzoeken valt buiten het bereik van deze standaard. Dit laat men over aan ontwikkelingen binnen de industrie.

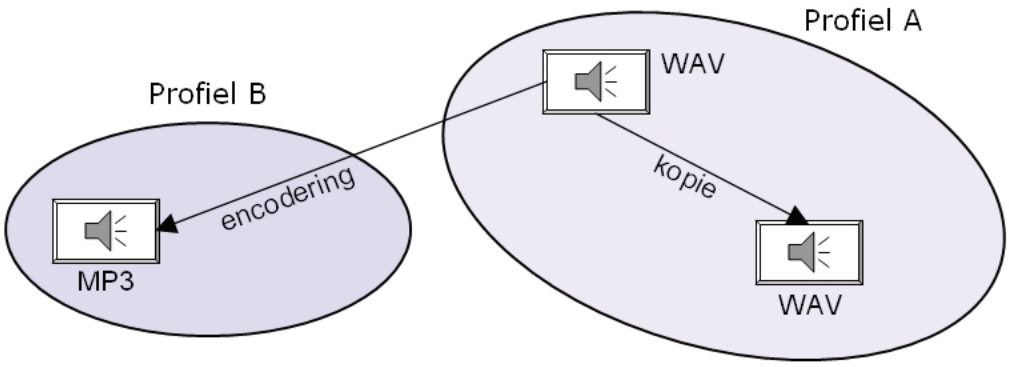
In de specificatie van de MPEG-7 standaard vinden we drie grote familie description schema. De eerste (Multimedia Description) wordt gebruikt ter annotatie van *inhoud* van audio en visuele informatie, alsook het management van deze informatie. De tweede set (Visual) biedt basis structuren en descriptors ter beschrijving van foto's of videofragmenten. De laatste set (Audio) biedt hetzelfde voor beschrijving van een audiosignaal.

De MPEG-7 metadata wordt opgeslagen binnen een datastream of in een apart bestand. Ze kan worden voorgesteld als tekst (XML) of als binair bestand. MPEG-7 gebruikt haar eigen binaire encodatie en-transmissiesysteem: BiM. Bij BiM-transmissie verwacht de ontvanger eerst het DDL-schema, en daarna de feitelijke data. Hierdoor kan men de XML sterk comprimeren. Met BiM is men niet verplicht de volledige XML-instantie te versturen, de ontvanger kan nodes selecteren om te downloaden. BiM heeft ook een groot aantal compressieschema voor de verschillende datatypes van MPEG-7.

Multimedia Description Schemes (MDS)

De MDS voorzien de beschrijvende elementen van MPEG-7. De MDS bestaan uit een groot aantal description schema die verschillende aspecten van audiovisuele informatie annoteren. In deze sectie bespreken we de vijf hoofdcategorieën: Content Management, Content Description, Navigation & Access, Content Organisation en User Interaction. Een applicatie is niet verplicht de MDS volledig in te vullen, voor verschillende doeleinden worden verschillende schema geselecteerd en gerelateerd. Naast de description schema introduceert de MDS ook een aantal descriptors (concept van tijd, duratie) en wiskundige structuren (vectors, matrices) die de description schema nodig hebben om audiovisuele informatie te annoteren.

Content Management biedt descriptors en schema's ter annotatie van de levenscyclus van een AV-fragment. Informatie die te maken heeft met de creatie of productie van een AV-fragment kan worden beschreven door de CreationInformation DS. Dit schema bevat descriptors voor de aanmaak (waar, wie, waarmee, wanneer,..) en classificatie (genre, stijl, taal) van een fragment, alsook de mogelijkheid om externe informatie te koppelen aan een fragment.
Gebruikersrechten van een AV-fragment worden beschreven door de UsageInformation DS. Met dit schema is het mogelijk te beschrijven wie het bestand mag gebruiken (Access Rights), hoeveel het kost of reeds opbracht, waar men het fragment kan verkrijgen en referenties naar eerder gebruik van het fragment.
Tenslotte introduceert het MediaInformation DS het concept van media-instanties en media-profielen. Een media-profiel beschrijft informatie over het formaat, de encodering en mogelijk een beoordeling van een AV-fragment. Een media-instantie komt overeen met een media-profiel. Een kopie van een fragment is een nieuwe instantie, maar behoudt hetzelfde profiel. Wanneer een fragment wordt gewijzigd (bijvoorbeeld naar een andere encodering) krijgt het eveneens een nieuw profiel. De beschrijving van een bepaald AV-fragment bevat mogelijk een aantal media-profielen met links naar de overeenkomende instanties. Figuur 3.2 demonstreert het onderscheid tussen profielen en instanties in MPEG-7. De drie rechthoeken stellen instanties voor. De twee cirkels stellen profielen voor.

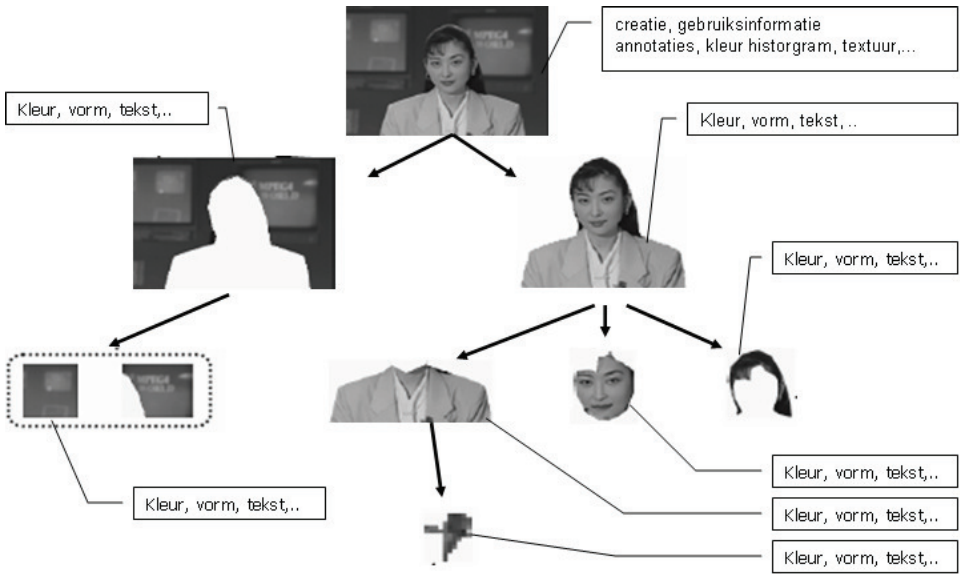


Figuur 3.2: Profielen en instanties in MPEG-7 MDS

Content Description voorziet descriptors ter beschrijving van de structuur of semantiek van audiovisuele informatie. De semantische schema's kunnen worden gerelateerd aan de structurele.
Structureel is het mogelijk elk object uit een scène of een fragment van een groter geheel voor te stellen als een segment (Segment DS). Een segment is vergelijkbaar met een abstracte klasse uit object-georiënteerde programmeertalen. Haar afgeleide klassen bestaan uit segmenten ter beschrijving van audio, video of (mogelijk 3D) afbeeldingen. Een segment bevat mogelijk temporele alsook spatial informatie. Zo zal een audio-segment enkel temporele samples in een muziekstuk beschrijven, terwijl een audio-visueel segment het geluid en de beweging van een regio beschrijft.
Structurele segmenten kunnen worden ondergebracht in een boom of graafstructuur. Een video-segment kan bijvoorbeeld worden verdeeld in een aantal sub-video-segmenten die op hun beurt verder opsplitsen in bewegende regio's en tenslotte stilstaande afbeeldingen. (boomstructuur)
Elk segment kan beschrijvingen bevatten uit de *content management* (zie hierboven), tekstuele annotatie of annotatie eigen aan het type. Figuur 3.3 is een voorbeeld van hiërargische decompositie van een stilstaande afbeelding. Het hoofdsegment bevat informatie over de foto zelf, de kinderen bevatten informatie over de vorm, kleur en tekstuele annotatie.

De graafstructuur beschrijft onderlinge relaties tussen segmenten in AV-materiaal. De keuze tussen een graaf-of boomdecompositie voor een AV-fragment is afhankelijk van haar toepassing.
Voor sommige toepassingen volstaat geen structurele decompositie. Daarom voorziet MPEG-7 ook semantische schema's (Semantic DS). De schema's worden juist zoals bij de structurele schema afgeleid van een basisklasse, in dit geval het SemanticBase DS. De afgeleide schema's beschrijven objecten (bv: meubel,..), objectagenten (bv: een persoon), concepten (bv: harmonie), gebeurtenissen (bv: pianospelen) en tenslotte de semantische status van een object (bv: weegt 30 kilogram). De semantische schema worden opnieuw verbonden in een graaf-of boomstructuur. De bogen van deze structuren duiden ditmaal op onderlinge relaties.

Naast concrete semantische beschrijvingen voorziet de Semantic DS ook abstracties. Een standaard abstractie is een abstractie van een object of event (zie hierboven). Een event



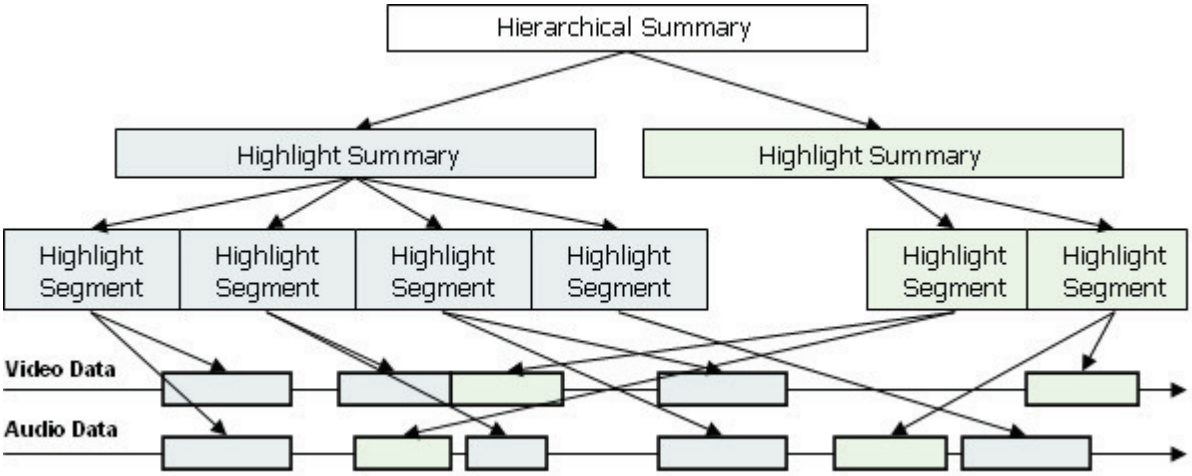
Figuur 3.3: Hiërargische decompositie d.m.v. segmenten in MPEG-7 MDS [Mar04]-10

beschrijft een specifieke entitiiteit in ruimte of tijd. (bijvoorbeeld: Tom speelt piano). Een abstract event beschrijft een klasse van events met dezelfde karakteristieken. (Bijvoorbeeld: persoon speelt piano.) Dit is handig voor zoekalgoritmen.
Een tweede type abstractie is de media abstractie. Deze abstractie werkt op een meer algemeen niveau en beschrijft eenzelfde type audio-visuele fragmenten. Bijvoorbeeld, een nieuwsbericht.

Navigation and Access bevat descriptors die AV-informatie indexeren. Summaries (Summarization DS) worden voornamelijk gebruikt om een gebruiker snel zicht te geven op de aangeboden informatie. Summaries zijn sequentiëel (een slideshow) of hiërargisch (een boomstructuur). Een film kan bijvoorbeeld hiërargisch worden onderverdeeld in hoofdstukken, maar ook in fragmenten die handelen over hetzelfde onderwerp. (bijvoorbeeld actiescènes, belangrijke gebeurtenissen, e.a.) Figuur 3.4 geeft een voorbeeld van twee hiërargische summaries van hetzelfde AV-fragment.

De View DS maakt het mogelijk een fragment op te delen. Het voorziet een aantal schema's ter decompositie van een AV-fragment in tijd, ruimte en frequentie. Deze schema's kunnen worden gestructureerd in een boom zodat applicaties een selectie kunnen maken uit de verschillende mogelijkheden.
De Variation DS beschrijft tenslotte variaties van eenzelfde mediafragment. Zo kan de gebruiker van een bepaald AV-fragment een variatie selecteren die overeenkomt met zijn bandbreedte, decompressiemethodes of taalinstellingen.

De *Collections* categorie voorziet schema's om events, objects, segmenten of fragmenten te verzamelen. Met het CollectionStructure DS is het mogelijk een verzameling te annoteren met eenzelfde attribuut. Zo hebben een aantal foto's mogelijk hetzelfde kleurhistogram. Verder kunnen verzamelingen onderling worden gerelateerd. Figuur 3.5 bevat een voorbeeld van drie collecties foto's met hun onderlinge relaties. (aanval, doelpunt, gejuich).



Figuur 3.4: Hiërargische summary in MPEG-7 MDS

De *User Interaction* DS beschrijft tenslotte gebruikersvoorkeuren, geschiedenis en instellingen. Het stelt schema's ter beschikking die een systeemagent kan gebruiken om conclusies te trekken uit voorgaande acties van gebruikers.

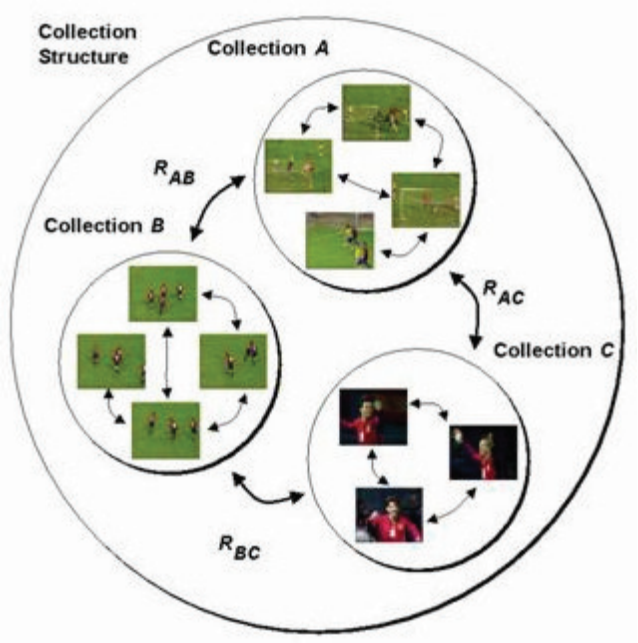
Video Description Schema

De low-level visuele descriptors [Sik01] die MPEG-7 voorziet zijn kleur, textuur, vorm, motie, localisatie en gezichtsherkenning. Ze kunnen worden toegepast op objecten, segmenten of events uit de MDS. Naast deze descriptors introduceert MPEG-7 Visual ook een aantal basisstructuren, nl. Grids, tijd met (ir)reguliere intervals, 3D-descriptors, een 2D assenstelsel descriptor en een descriptor voor interpolatie over tijd.

MPEG-7 Visual bevat een rijke set descriptors voor beschrijving van kleur. De twee basisdescriptors zijn ColorSpace (RGB, Ycbcr,..) en de Quantisation van een color space. De overige kleurdescriptors zijn ontworpen om zoekopdrachten in visuele fragmenten te ondersteunen. De DominantColor descriptor beschrijft de meest voorkomende kleuren in een fragment en hun onderlinge relatie. Dit is bijvoorbeeld handig voor het terugvinden van een vlag. De ScalableColor descriptor voorziet een histogram ter beschrijving van een stilstaand fragment. Op Figuur 3.6 zijn de twee linkse foto's sterker gerelateerd dankzij hun kleurhistogram.

De colorLayout descriptor beschrijft een 8x8 pixels thumbnail van een bepaald fragment, ter ondersteuning van bijvoorbeeld sketch-queries.
De colorStructure description beschrijft een visueel fragment met een histogram, juist zoals de ScalableColor descriptor, maar deze descriptor gebruikt ook spatiale informatie in haar historgram. (d.m.v. een 8x8 pixel sliding-window). Hierdoor merkt het spatial verschillen op die de ScalableColor descriptor niet zou ontdekken.
Tenslotte berekent de GoF/GoPColor descriptor een kleurhistogram voor bewegende beelden. Het gebruikt hiervoor een gemiddelde berekening van de ScalableColor descriptor over een aantal fragmenten.

De *texture* descriptors leggen verbanden tussen patronen in visuele fragmenten. Zo is het



Figuur 3.5: Collecties in MPEG-7 MDS [Mar04]-19

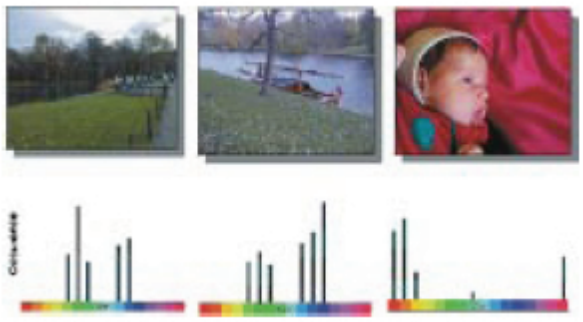
mogelijk op basis van een patroon, een weiland in een luchtfoto te herkennen. We beschouwen twee types texture descriptors: homogene en non-homogene.
De homogenous-texture descriptors worden toegepast op patronen en gebruiken 62 bytes om de vorm en deviatie van een patroon te beschrijven.
De EdgeHistogram descriptor wordt gebruikt om fragmenten met non-homogene vormen te beschrijven. Hij verdeelt een fragment in 16 gelijke blokken en past op elke blok een parametrisatie van vijf edges toe. (verticaal, twee diagonaal, horizontaal en één non-lineaire.) De Edgehistogram kan (samen met kleur) bijdragen aan image-to-image matching.

Vorm descriptors van MPEG-7 beschrijven (op een lossy manier) 2D-objecten op basis van de regio die ze innemen (Region Shape D) of op basis van de omlijning van het object (Contour Shape D). 3D objecten kunnen worden voorgesteld door een verzameling van 2D views vanuit verschillende hoeken, of op basis van een histogram dat de vorm van het object compact beschrijft.

De *motie* descriptors beschrijven camerabeweging (mogelijk automatisch ingevuld door de camera), beweging van een object (door interpolatie functies op een set van keypoints), parametrische motie (affine translaties tussen regio's in een bewegende scène) en bewegingsintensiteit in een scène (voorgesteld als integerwaarde tussen 0 en 5).

Met de *localisatie* descriptors kan men een regio binnen een fragment afbakenen met een polygoon, alsook deze regio volgen in tijd (Spatio Temporal Locator.)

Tenslotte voorziet de *face recognition* descriptor de mogelijkheid om fragmenten te doorzoeken naar gezichten die overeenkomen met een gegeven gelaat.



Figuur 3.6: Het kleurhistorgram ondersteunt Image-to-Image matching in MPEG-7 Visual [Sik01]-2

Audio Description Schema

MPEG-7 heeft twee categorieën descriptors voor annotatie van audio. Low en high-level descriptors. De eerste categorie biedt werktuigen die de tweede categorie gebruikt ter ondersteuning van audio-zoekopdrachten. (Bijvoorbeeld query-by-humming, of stemherkenning.) De Low-level descriptors van MPEG-7 deelt men op in zes klassen, die elk een audiosample op een andere manier voorstellen.

De *Basic*-klasse bevat een descriptor die de vorm van de wave-curve (minimum en maximum) voorstelt (AudioWaveform D) en een descriptor die het vermogen van een signaal beschrijft. (AudioPower D.)
De *Basic-spectral*-klasse biedt een globale descriptor (AudioSpectrumEnvelope) die (in vector) een logaritmisch-frequentie spectrum van een audio-signaal voorstelt. Binnen deze klasse leiden vier descriptors af van de AudioSpectrumEnvelope: De AudioSpectrumCentroid Descriptor stelt het gewogen gemiddelde van een spectrum voor. Deze descriptor wordt gebruikt om te bepalen of het audiosignaal werd gedomineerd door een hoge of lage frequentie. De AudioSpectrumSpread beschrijft de afwijking van het spectrum in verhouding tot haar gewogen gemiddelde. (Dit helpt in het onderscheiden tussen ruis of tonen.) De AudioSpectrumFlatness descriptor wordt tenslotte gebruikt om het audiosignaal af te wegen tegen een vlak spectrum (bv: ruis), een grote afwijking heeft kans op aanwezigheid van tonale componenten.
De *signal parameters*-klasse bevat de AudioFundamentalFrequency descriptor die de basisfrequentie (1ste harmonische) van een signaal beschrijft, terwijl de AudioHarmonicity descriptor meerdere harmonischen bevat, voor vergelijking met andere signalen.
De *timbral temporal*-klasse wordt gebruikt om temporele attributen (buiten pitch of vermogen) van een audiosignaal te beschrijven. Meer precies, de LogAttackTime descriptor beschrijft de tijd die een audiosignaal nodig heeft om van stilte tot maximale amplitude te stijgen. (Maakt onderscheid tussen lange en korte noten mogelijk.) De TemporalCentroid descriptor beschrijft waar in de tijd een signaal het sterkts was. Dit is handig om audiosignalen te onderscheiden die dezelfde LogAttackTime bezitten, maar toch verschillen. (bijvoorbeeld een stervende piano-toets t.o.v. een blijvende orkelklank.)
De *Timbral Spectral*-klasse bevat descriptors voor spectral analyse in lineaire-frequency verhouding. Het bevat een desciptor die het gewogen gemiddelde van het spectrum voorstelt (Spectral Centroid D) en een descriptor die het gewogen gemiddelde van de hoogste pieken

in het spectrum weergeeft (HarmonicSpectralCentroid D). Verder bevat deze klasse drie descriptors die verdere afwijkingen t.o.v. het spectrum voorstellen.
De *Spectral Basis*-klasse bevat tenslotte twee descriptors die m.b.v. wiskunde projecties interessante eigenschappen van een volledig spectrum kunnen filteren. Deze eigenschappen worden vooral gebruikt ter classificatie van audio-fragmenten.
Een laatste low-level descriptor is de Silence descriptor. Ze wordt gebruikt om stiltes te beschrijven.

De High-level audio descriptors van MPEG-7 worden onderverdeeld in vijf tool-sets: Een schema om audio samen te vatten in een "vingerafdruk" (Audio Signature DS), een schema (Music Instrument Timbre DS) voor onderscheid tussen muziekinstrumenten (harmonische instrumenten t.o.v. slaginstrumenten). Een schema voor beschrijving van melodie en ritme van een audiofragment (Melody Description), een schema voor geluidsherkenning (Sound Recognition). Tenslotte een schema voor beschrijving van gesproken tekst. (Spoken Content).

3.1.2 Dublin Core

Dublin Core [Dub05] (DC) is de meest bekende set metadata elementen. Ze dankt haar naam aan een workshop die werd gehouden in Dublin, Ohio in 1995. Het onderhoud en de uitbreiding van de Dublin Core wordt verzorgd door de Dublin Core Metadata Initiative (DCMI). Het doel van de Dublin Core was het definiëren van een basisset elementen die zou worden gebruikt ter annotatie van web resources. Vandaag bestaat deze set uit 15 elementen: Title, Creator, Subject, Description, Publisher, Contributor, Date, Type, Format, Identifier, Source, Language, Relation, Coverage en Rights. DC dankt haar succes aan haar simpliciteit. Ze kan worden toegepast in elke annotatietaal op elk digitaal object. De volgorde van de objecten is van geen belang, en elementen kunnen zonder restricties worden herhaald. De inhoud van de annotatievelden is eveneens open ter interpretatie. Zo kan het veld coverage bijvoorbeeld voor één type bestand spatial data bevatten, terwijl het voor een ander bestand temporele data of beide kan bevatten.

Critici vonden de basis set Dublin Core elementen (ook simple of unqualified DC genoemd) niet nauwkeurig genoeg om een bepaald digitaal bestand te annoteren. Om die reden ontstond een tweede set Dublin Core elementen: Qualified DC. Deze laatste set is een uitbreiding van Simple DC. De huidige Qualified DC specificatie introduceert zeven nieuwe basiselementen (Instructional Method, Audience, Provenance, AccrualMethod, AccrualPeriodicity, AccrualPolicy, Rightsholder), alsook drieëndertig specialisaties van bestaande elementen. Zo zijn er bijvoorbeeld voor het Simple DC element *date* acht specialisaties (created, modified,..).
Qualified DC is ook meer specifiek over de formattering van sommige elementen. Zo specifieert Qualified DC bijvoorbeeld de formattering van een datumstring, geografisch punt, tijdsinterval, taalcode, e.a.
Voor een aantal velden van Qualified DC raadt men gebruik van een aantal welbekende classificatie systemen of thesauri aan. Het is belangrijk te vermelden dat Qualified DC geen restrictie oplegt voor deze velden. De voorgestelde formatteringen en classificaties zijn slechts suggesties.
Listing 3.1 geeft een voorbeeld van Dublin Core in de head-tag van HTML. In HTML worden de hoofdelementen geïdentificeerd met een DC-prefix. De gespecialiseerde elementen starten met prefix DCTERMS.

Listing 3.1: Dublin Core voorbeeld in HTML

```
<head>
  <title>Luk Vloemans, Thesis</title>
  <link rel="schema.DC" href=
    "http://purl.org/dc/elements/1.1/"/>
  <link rel="schema.DCTERMS" href=
    "http://purl.org/dc/terms/"/>
  <meta name="DC.title" lang="nl" content=
    "Thesis Website"/>
  <meta name="DC.creator" content=
    "Luk Vloemans"/>
  <meta name="DCTERMS.issued" scheme=
    "DCTERMS.W3CDTF" content="2006-05-01"/>
  <meta name="DC.identifier" scheme=
    "DCTERMS.URI" content="http://www.lukv.be"/>
  <meta name="DCTERMS.abstract" content=
    "Dit document bevat een changeLog voor
     de thesis van Luk Vloemans." />
  <meta name="DC.format" scheme=
    "DCTERMS.IMT" content="text/html"/>
  <meta name="DC.type" scheme=
    "DCTERMS.DCMIType" content="Text"/>
</head>
```

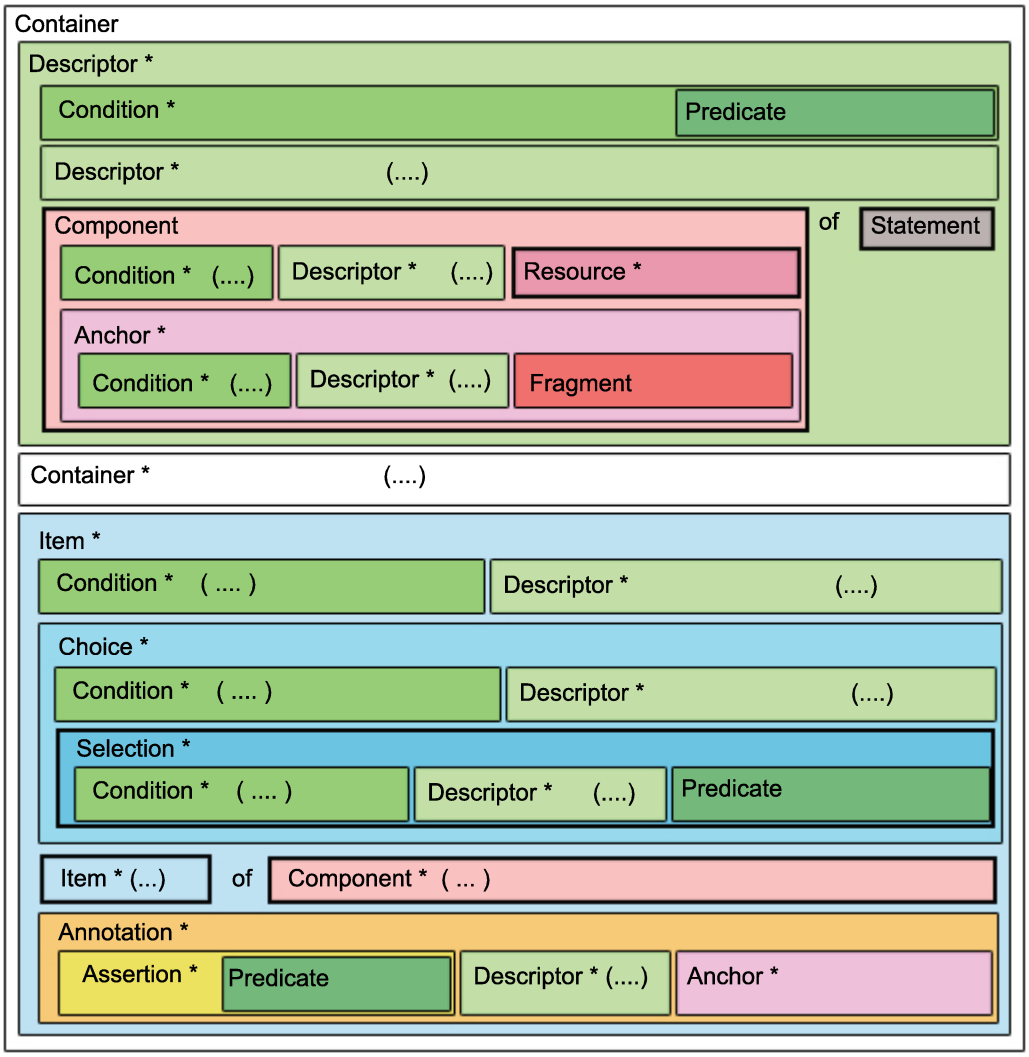
3.1.3 MPEG-21 Digital Item Declaration Language

MPEG-21 [BH02] is een multimedia framework dat verzending en consumptie van digitale bestanden over een groot aantal netwerken en apparaten standariseert. MPEG-21's nadruk ligt op eenvoud (de gebruiker mag zich niet bewust zijn van achterliggende architectuur), beheer van rechten (productiehuizen hekelen de huidige illigale vrije markt op het internet) en netwerkmogelijkheden (een korte daling in netwerkcapaciteit mag geen invloed hebben op een video die een gebruiker bekijkt, het beeld mag hoogstens dalen in kwaliteit.)

Centraal in de MPEG-21 architectuur staat de representatie van een *digitaal item*. Deel twee van de MPEG-21 specificatie [mpe05] beschrijft een abstract model alsook een schemataal (Digital Item Declaration Language) om digitale items voor te stellen. Een digitaal item bevat typisch één of meerdere multimediabestanden (muziek, video, ondertitels, muziekteksten..), en metadata gerelateerd aan deze bestanden. In het kader van deze thesis concentreren we ons op de technieken die MPEG-21 gebruikt voor het annoteren van een digitaal item (DI).

Het abstract model van MPEG-21 DID beschrijft termen en concepten om digitale items voor te stellen. Binnen het model is een digitaal item het element dat wordt gebruikt (beschreven, uitgewisseld, verzameld.) Het model is de basiseenheid waarop verdere MPEG-21 functies worden gebouwd.
Figuur 3.7 geeft een overzicht van het MPEG-21 DID Model. Een element met een ster (*) kan geen, eenmaal of meerdermaal voorkomen binnen het element van haar ouder. (Bijvoorbeeld, een container kan meermaals voorkomen in een andere container). De verschillende elementen in het model worden in kleurcodes onderscheiden. Een element dat meermaals voorkomt, wordt op de figuur afgekort met (....). Elementen die verplicht zijn binnen hun parent werden gemarkeerd met een vette rand.

De Digital Item Declaration Language (DIDL) is gebaseerd op het abstract model (neemt de structuur volledig over) en wordt uitgedrukt in XML Schema. In de DIDL is het mogelijk



Figuur 3.7: Het MPEG-21 DID Model

meerdere metadata schema's over te nemen (d.m.v. namespaces in XML schema). De DIDL onderscheidt veertien verschillende elementen. De root van een DIDL instantie bevat typisch één container of item-element.

- Het **container** element is een structuur die containers of items groepeert. Het descriptor element binnen de container biedt informatie over de verzameling elementen van de container.

- Een **item** element is de representatie van een digitaal bestand (bijvoorbeeld een video). Het kan sub-items, (verschillende representaties van hetzelfde bestand) of components (containers voor bytecode) bevatten. Een item kan choices bevatten waardoor een systeem/persoon keuzes kan maken tussen verschillende sub-items/components. (bijvoorbeeld: bij video, een keuze tussen MPEG of Quicktime). Verder bevat een item

descriptors (informatie over het item), condities (beperking van de context van het item) en annotaties.

- Het **component** element is een wrapper voor resources. Het bevat descriptors over de resource en anchors die gedeeltes binnen de resource beschrijven. Verder is het ook mogelijk condities toe te voegen aan components. Het **resource** element bevat de bytecode (of referentie naar de bytecode) van een bestand. De attribuutset van een resource bevat mogelijk het MIME-type, gebruikte compressie (contentEncoding) of encodering (encoding, default base64)

- Het **statement** element bevat enkel tekst (of een referentie naar tekst). De inhoud is mogelijk gestructureerd in XML. Het bevat dezelfde attribuutlijst als een resource element. Een statement is afhankelijk van haar context. Bijvoorbeeld: In de context van een choice zal haar inhoud gebruikt worden als interface naar de gebruiker. In de context van een resource bevat het statement de metadata van de resource. (bijvoorbeeld: MPEG-7 descriptors, of Dublin Core elementen).

- Een **descriptor** element associeert informatie met het element waardoor het wordt geëncapsuleerd. Een descriptor gebruikt components of statements. Descriptors bevatten vaak metadata. (bijvoorbeeld: thumbnails in components of auteursinformatie in een statement)

- Een **fragment** element wordt gebruikt om een stuk media te identificeren binnen een groter geheel. Een **anchor** element bevat bindings tussen descriptors en fragments van een resource. Bijvoorbeeld: een tekst-uitleg van een stuk van een foto of de naam van een woorvoerder tijdens een audio-conversatie. In het eerste voorbeeld is de descriptor een tekst en het fragment een set coördinaten binnen een foto. In het tweede voorbeeld is de descriptor een dc:name element en het fragment een offset in seconden.

- Een **choice** element wordt gebruikt om een user/systeem een keuze te laten maken over de representatie van een bepaald element. Om dit te bekomen gebruikt het choice element **selection** elementen. Een selection element definiëert een booleaanse waarde, waarop later kan worden getest door **condition** elementen. Een conditie element bevat een attribuut *require* dat een conjuntie van variabelen bevat die waar moeten zijn, en een attribuut *except* waarvan geen enkel element mag waar zijn. (disjunctie bekomt men door meerdere conditie elementen op te sommen.) Listing 3.2 geeft een voorbeeld van dit mechanisme. In dit voorbeeld kan een gebruiker tijdens het beluisteren van een audiofragment extra videomateriaal bekijken en selecteren in welk formaat hij de video wil bekijken.

Listing 3.2: Condities en Selecties in MPEG21 DIDL

```
<DIDL xmlns="urn:mpeg:mpeg21:2002:02-DIDL-NS">
 <Item>
  <Choice minSelections="1" maxSelections="1">
   <Descriptor><Statement mimeType="text/plain">
     Wilt u ook video, terwijl u audio beluistert?
   </Statement></Descriptor>
   <Selection select_id="YES_VIDEO"><Descriptor>
     <Statement mimeType="..">Ja</Statement>
   </Descriptor></Selection>
```

```
   <Selection select_id="NO_VIDEO"><Descriptor>
     <Statement mimeType="..">Nee</Statement>
   </Descriptor></Selection>
  </Choice>
  <Choice minSelections="1" maxSelections="1">
   <Descriptor><Statement mimeType="text/plain">
       Welk formaat verkiest u?
   </Statement></Descriptor>
   <Selection select_id="VIDEO_MPEG"><Descriptor>
    <Statement mimeType="..">MPEG-1/2</Statement>
   </Descriptor></Selection>
   <Selection select_id="VIDEO_QT"><Descriptor>
    <Statement mimeType="..">Quicktime</Statement>
   </Descriptor></Selection>
  </Choice>
  <Component>
   <Resource ref="audio.mp3" mimeType=".."/>
  </Component>
  <Component>
   <Condition require="YES_VIDEO VIDEO_MPEG"/>
   <Resource ref="video.mpg" mimeType=".."/>
  </Component>
  <Component>
   <Condition require="YES_VIDEO VIDEO_QT"/>
   <Resource ref="video.mov" mimeType=".."/>
  </Component>
 </Item>
</DIDL>
```

- **Annotations** bevatten tenslotte gebruikersopmerkingen (in descriptors/anchors). Gebruikers kunnen annotaties toevoegen zonder het bestand of gerelateerde metadata te wijzigen. Annotaties kunnen ook *assertions* bevatten. Assertions worden gebruikt om variabelen van conditions in te vullen. In bovenstaand voorbeeld had een assertion bijvoorbeeld ervoor kunnen zorgen dat een applicatie automatisch een quicktime video verkiest.

Alle elementen van MPEG-21 DIDL (behalve conditions en assertions) mogen metadata attributen van andere schema's bevatten.

3.1.4 Extensible Metadata Platform

Adobe Systems lanceerde in 2001 het Extensible Metadata Platform [Ado05] (XMP). Het is een metadata schema voor annotatie van elk type bestand. XMP metadata is geëncodeerd in XML-geformateerde tekst en maakt gebruik van een subset van RDF. XMP informatie wordt binnen een bestand opgeslagen in een *XMP packet* dat eenvoudig herkenbaar is dankzij een speciale unicode marker. Een XMP Parser kan dankzij deze marker het packet eenvoudig terugvinden, en op basis van de marker de gebruikte encodering (UTF-8, UTF-16, UTF-32) en byteorde (big of little endian) bepalen.

Listing 3.3 toont de layout van het XMP Packet. De xpacket-tag bevat de speciale marker en de daaropvolgende unieke string. De RDF Descriptions binnen de rdf:RDF-tag worden best gegroepeerd op basis van namespace. Het about attribuut van deze tags is steeds leeg mits

het XMP packet steeds informatie bevat over het bestand waarin ze werd geëncapsuleerd. Op het einde van een XMP Packet raadt men aan een aantal (2-4KB) whitespaces toe te voegen, zodat men een bestand niet steeds moet herschrijven wanneer het XMP-packet uitbreidt.

Listing 3.3: Layout van het XMP Packet

```
<?xpacket begin=" " id="W5M0MpCehiHzreSzNTczkc9d"?>
  <x:xmpmeta xmlns:x="adobe:ns:meta/">
    <rdf:RDF xmlns:rdf= ...>
        <rdf:Description about="" xmlns:dc=".."   />
        <rdf:Description about="" xmlns:exif=".." />
        <rdf:Description about="" xmlns:tiff=".." />
        ...
    </rdf:RDF>
  </x:xmpmeta>
  <!-- Whitespaces als vulling -->
<?xpacket end="w"?>
```

Voor gesloten bestandstypes of bestandsformaten die geen ruimte voorzien voor encapsulatie van extra informatie voorziet men een generieke manier om XMP metadata extern op te slaan. In dit geval wordt aangeraden de metadata te voorzien van een sleutel die het gerelateerde bestand identificeert. (Bijvoorbeeld m.b.v. een MD5-sleutel van de bytecode.) XMP maakt gebruik van XML namespaces. Hierdoor introduceert XMP een krachtig voordeel: uitbreiding naar andere schema's. De specificatie bevat reeds uitbreidingen naar externe schema's zoals EXIF, Photoshop, Camera Raw, PDF en Dublin Core. Verder introduceert XMP zes sets metadata elementen voor verscheidene doeleinden.

Het *XMP Basic Schema* (xmlns:xmp) bevat negen elementen die het bronbestand verder beschrijven dan Dublin Core. Het *XMP Rights Management Schema* (xmlns:xmpRights) beschrijft welke organisatie de rechten van het bronbestand beheert en op welke manier het bestand mag worden gebruikt. Het *XMP Media Management Schema* (xmlns:xmpMM) is ontworpen als hulp voor digitale bibliotheken (DAM[2]-systemen). Het *XMP Job Ticket Schema* (xmlns:xmpBJ) beschrijft in welke workflows het bestand momenteel wordt gebruikt. Het *XMP Paged-Text Schema* (xmlns:xmpTpg) bevat een aantal elementen ter annotatie van geschreven tekst. Tenslotte bevat het *XMP Dynamic Media Schema* (xmlns:DM) een set elementen ter beschrijving van audio en visuele informatie. Tabel 3.1 geeft een overzicht van de verschillende namespaces.

Adobe Systems tracht de industrie te overtuigen van de kracht van het XMP platform. Om dit te bekomen heeft het bedrijf XMP opgenomen in haar programma suite. Verder heeft ze de specificatie vrijgegeven onder open licentie en voorziet men een SDK ter ondersteuning van software ontwikkelaars. Adobe heeft stappen ondernomen bestaande metadata schema's uit te drukken in XMP. De Samenwerking met IPTC (sectie 3.1.5) is hier een voorbeeld van: IPTC4XMP.

3.1.5 IIM, IPTC Header en IPTC4XMP

De IPTC[3] is een consortium van grote nieuwsagentschappen. In 1991 ontwierp deze groep een standaard voor uitwisseling van alle mogelijke type bestanden, voornamelijk nieuwsitems, fo-

[2]Digital Asset Managent
[3]International Press Telecommunications Council

Bestandsinformatie	xmp:BaseURL, xmp:Label, xmpidq:Scheme, xmp:Identifier, xmpDM:fileDataRate
Auteursinformatie	xmpDM:artist, xmpDM:composer, xmpDM:engineer
Legaal	xmpRights:Certificate, xmpRights:Marked, xmpRights:Owner, xmpRights:UsageTerms, xmpRights:WebStatement, xmpDM:copyright
Geografische oorsprong	xmpDM:scene, xmpDM:shotLocation
ProductieProces	xmp:Advisory, xmp:CreatorTool, xmpBJ:JobRef, xmpDM:projectRef, xmpDM:tapeName, xmpDM:altTapeName, xmpDM:contributedMedia
Digital Asset Management	[xmpMM] DerivedFrom, DocumentID, History, ManagedFrom, Manager, ManageTo, ManageUI, ManagerVariant, RenditionClass, RenditionParams, VersionID, Versions
DateTime	xmp:CreateDate, xmp:MetadataDate, xmp:ModifyDate, xmpDM:startTimecode, xmpDM:altTimecode, xmpDM:duration, xmpDM:shotDate, xmpDM:videoModDate, xmpDM:audioModDate, xmpDM:metadataModDate, xmpDM:releaseDate
Technische Text-Representatie	xmpTPg:MaxPageSize, xmpTPg:Npages, xmpTPg:Fonts, xmpTPg:Colorants, xmpTPg:PlateNames
Technische Audio-Representatie	[xmpDM:] audioSampleRate, audioSampleType, audioChannelType, audioCompressor, speakerPlacement, absPeakAudioFilePath, relativePeakAudioFilePath, tempo, introTime, outCue, relativeTimestamp, numberOfBeats, key, stretchMode, timeScaleParams, resampleParams, beatSpliceParams, timeSignature
Technische Video-Representatie	[xmpDM:] videoFrameRate, videoFrameSize, videoPixelAspectRatio, videoPixelDepth, videoColorSpace, videoAlphaMode, videoAlphaPremultipleColor, videoAlphaUnityIsTransparent, videoCompressor, videoFieldOrder, pullDown, loop, scaleType
Geëncapsuleerd Bestand	xmp:Thumbnails, xmpDM:shotName, xmpDM:logComment, xmpDM:markers
Beschrijvend	xmp:Nickname, xmp:Rating
Audio-Beschrijving	xmpDM:album, xmpDM:trackNumber, xmpDM:genre, xmpDM:instrument
Overerving	Dublin Core, EXIF, IPTC4XMP

Tabel 3.1: Elementen in XMP

to's of audiobestanden. De standaard heet het Information Interchange Model [IPT99] (IIM).

Het model beschrijft hoe een bestand wordt geëncapsuleerd in een digitale enveloppe die haar metadata bevat. Deze metadata is eigen aan haar cultuur: de perswereld. Ze bevat een uitgebreide pers-taxonomie, waarin men een object kan annoteren met een type (bv: Interview, Forecast) alsook een gestructureerde inhoudsbeschrijving (bv: Disasters & Accidents: Nuclear Accident). Deze informatie, samen met keywords en een korte inhoud stellen een journalist in staat een digitaal object sneller te doorzoeken en te beoordelen.

Verder beschrijft de specificatie elementen voor beschrijving van tijd, auteur, locatie en productieproces. Afhankelijk van het bestandstype van het object bestaan er ook elementen voor beschrijving van technische metadata van foto, audio en-tekstbestanden.

Adobe Systems adopteerde een subset van het IIM voor de annotatie van JPEG, PSD en TIFF bestanden. De 19 elementen werden geëncapsuleerd binnen de bestandsformaten in *Image Resource Blocks* zonder de bronbestanden te verminken. Deze subset kende een succes in de digitale fotografie en werd door een groot aantal softwarepakketten geaccepteerd als defacto-standaard voor digitale foto's. Ze kreeg als titel de *IPTC Header*. Tabel 3.2 bevat een overzicht van de elementen van IIM. De elementen die werden geadopteerd door Adobe staan gemarkeerd met een (*).

In 2001 introduceerde Adobe een nieuw metadata Framework: XMP (sectie 3.1.4). Adobe en de IPTC creëerden in 2004 een uitbreiding van deze standaard om de IPTC Headers te migreren van de Image Resource Blocks naar de nieuwe standaard. Deze uitbreiding heet IPTC4XMP [IPT05]. Naast de IPTC header elementen werden acht extra elementen toegevoegd uit de originele IIM specificatie waaronder de ondersteuning voor de IIM perstaxonomieën en informatie om de auteur van een object te contacteren via e-mail, post of telefoon. Een overzicht van de elementen in IPTC4XMP staan beschreven in Tabel 3.3.

3.1.6 Windows Media MetaFiles

Microsoft heeft naast haar eigen bestandsformaten voor audio en video (wmv, wma, asf) ook een metadata formaat ter beschrijving van haar bestandsformaten. Het formaat noemt Windows Media Metafiles [Mic06] en wordt uitgedrukt in XML. Een metafile wordt gebruikt om audiovisuele fragmenten te annoteren en te groeperen in een playlist. De metafiles zijn ontworpen voor de Windows Media Player, maar worden reeds (gedeeltelijk) ondersteund in open source software.

Een metafile bevat een lijst van links naar de audiovisuele data. Per audiovisueel fragment voorziet men elementen om titel, auteurs, copyright en-beschrijvingsinformatie te annoteren. Andere metadata kan worden opgeslagen in Param-tags. Naast beschrijvende metadata is het ook mogelijk elk fragment te beperken (Bijvoorbeeld: speel enkel de eerste vier minuten van fragment A) of de gebruikersmogelijkheden te beperken (Bijvoorbeeld: persoon mag niet pauseren of terugspoelen)

Metafiles zijn vooral ontworpen voor commerciële doeleinden. Zo is het mogelijk bij elk bestand een bescheiden banner (194x32 pixels) te definiëren die de mediaspeler moet tonen

Interne Tags	Model Version, Destination, Envelope Number, Envelope Priority, Record Version, Action Advised, Reference Service, Reference Date, Reference Number, Subfile, Picture Number
Bestands informatie	File Format, File Format Version, Product I.D., UNO, ARM Identifier, ARM Version, Object Name(*), Filesize
Auteursinformatie	Service Identifier, By-Line(*), By-Line Title(*), Credit(*), Source (*), Contact, Writer/Editor(*)
Legaal	Copyright (*)
Geografische oorsprong	City(*), Sub-location, Province/State(*), CountryCode, CountryName(*)
ProductieProces	Edit Status, Editorial Update, Urgency(*), Originating Program, Program Version, Original Transmission Reference(*)
DateTime	Date/Time Sent, Date/Time Released, Date/Time Expiration, Date/Time Created (*), Date/Time Digital Creation
Technische Text-Representatie	Coded Character Set
Technische Foto-Representatie	Image Type, Image Orientation, Pixels/Line, Numer of Lines, Pixel Size in Scanning Direction, Pixel Size Perpendicular To Scanning Direction, Supplement Type, Colour Representation, Interchange Colour Space, Colour Sequence, ICC Input Colour Profile, Colour Calibration Matrix, Lookup Table, Number of Index Entries, Colour Palette, Number of Bits/Sample, Sampling Structure, Scanning Direction, Image Rotation, Data Compression Method, Quantisation Method, End Points, Excursion Tolerance, Bits/Component, Maximum Density Range, Gamma Compensated Value.
Technische Audio-Representatie	Audio Type, Audio Sample Rate, Audio Sampling Resolution, Audio Duration
Thumbnail	ObjectData Preview File Format, ObjectData Preview File Format Version, ObjetData Preview Data
Inhoudelijk Beschrijvend	Object Type Reference, Object Attribute Reference, Subject Reference, Category(*), Supplemental Category(*), Fixture Identifier, Keywords(*), Content Location Code, Content Location Name, Special Instructions(*), Object Cycle, Headline(*), Caption(*), Rasterized Caption, Language Identifier, Audio Outcue

Tabel 3.2: Elementen in IIM

Auteur	Creator, CreatorContactInfo, CreatorJobTitle, DescriptionWriter, Provider, Source
Legaal	CopyrightNotice, RightUsageTerms
Geografische oorsprong	City, Country, CountryCode, Location, Province/State
ProductieProces	JobID
DateTime	DateCreated
Inhoudelijk Beschrijvend	Description, HeadLine, Instructions, IntellectualGenre, Keywords, Scene, SubjectCode, Title

Tabel 3.3: Elementen in IPTC4XMP

tijdens playback. Verder kan een mediabestand d.m.v. een commando de mediaspeler de opdracht geven een *event* uit te voeren. Events zijn een opsomming van mediafragmenten en worden gedefiniëerd in het metafile. Het event geeft ook aan welke actie de mediaspeler moet nemen na afloop van het event. Een typisch voorbeeld van een event is een reclamespot. Bijvoorbeeld: Een gebruiker streamt een tenniswedstrijd vanop een webserver. De webserver stuurt om de 10 minuten een boodschap naar de mediaspeler om een bepaalde reclameboodschap (een event in het mediafile) af te spelen. Na de reclameboodschap kan de gebruiker de wedstrijd verder bekijken.
Wanneer een portaalsite een profiel bezit van de gebruiker die multimedia aanvraagt wordt het reclame-aspect nog interessanter. Op basis van profielgegevens kan een scriptingtaal een metafile genereren dat events bevat die zijn afgesteld op het profiel van een gebruiker.
Sinds versie negen van de Windows Media Player is het eveneens mogelijk om in metafiles (per fragment) aan te geven dat de multimedia moet worden afgespeeld binnen een bepaalde webpagina. De Media Player opent dan de webpagina in haar hoofdvenster (Playing Now), en speelt de visuele informatie af in de webpagina (d.m.v. de HTML Object-tag.) Dit heeft krachtige commerciële doeleinden.

Listing 3.4 toont een synthese-voorbeeld van Windows Media Metafiles. Het bestand speelt eerst een promotieclip voor UHasselt. Tijdens playback toont de Media Player een banner van UHasselt. Hierna speelt een promotieclip voor stad Hasselt. Beide clips kunnen een event (Reclame1) aanroepen. De gebruiker heeft niet de mogelijkheid om de reclame door te spoelen. (ClientSkip=No)

Listing 3.4: WMM voorbeeld

```
<ASX version = "3.0">
 <TITLE>WMM Demo</TITLE>
 <ABSTRACT>Dit is een WMM-demo</ABSTRACT>
 <COPYRIGHT>Luk Vloemans 2006</COPYRIGHT>
 <MOREINFO HREF="http://www.msdn.com" />

 <!-- Event: Reclame in apart ASX bestand -->
 <EVENT NAME="Reclame1" WHENDONE="RESUME">
   <ENTRYREF HREF="http://www.eg.com/reclame.asx"
       CLIENTSKIP="NO" />
 </EVENT>

 <!-- Promoclipje voor UHasselt -->
 <ENTRY>
  <MOREINFO HREF="http://www.uhassellt.be" />
  <ABSTRACT>Bezoek de UHasselt</ABSTRACT>
  <TITLE>Promoclip voor uHasselt</TITLE>
  <AUTHOR>UHasselt</AUTHOR>
  <PARAM name='created' value='2006-05-06'>
  <COPYRIGHT>(c)2006 UHasselt</COPYRIGHT>

  <REF HREF = "http://../uhasselt.wmv" />

  <BANNER HREF="http://www.uhasselt.be/img/1.jpg">
   <ABSTRACT>Bezoek UHasselt</ABSTRACT>
   <MOREINFO HREF = "http://www.uhasselt.be" />
  </BANNER>
 </ENTRY>
```

```
 <!-- Promoclipje van Stad Hasselt -->
 <ENTRY>
  <TITLE>Studentenstad Hasselt</TITLE>
  <AUTHOR>Stad Hasselt</AUTHOR>
  <COPYRIGHT>(c)2006 Hasselt</COPYRIGHT>
  <ABSTRACT>Stad in groei: Hasselt</ABSTRACT>
  <REF HREF = "http://../hasselt.wmv" />
 </ENTRY>
</ASX>
```

3.2 Metadata voor tekst

3.2.1 Text Encoding Initiative

Het Text Encoding Initiative [SMB02] (TEI) is een standaard die guidelines voorziet voor annotatie en (vooral) markup van literatuur. De TEI wordt onderhouden door een consortium dat voornamelijk bestaat uit universiteiten, colleges en onderzoeksbibliotheken. Het initiatief ontstond uit een conferentie in New York (1987). De eerste versie van de TEI werd in 1994 gelanceerd. De huidige TEI-versie is P4.
De TEI is groot, en omvat een DTD met meer dan 400 elementen voor annotatie van elk type text. (bv: proza, een toneelstuk, een wetenschappelijk document, ..) Gebruikers van de TEI zullen typisch een subset-DTD van de TEI selecteren die voor hun van toepassing is. De specificatie is enkel een implementatie-suggestie, gebruikers mogen vrij de DTD uitbreiden naar hun eigen noden.

Een TEI-document (instantie) wordt uitgedrukt in SGML of (sinds P4) in XML. Listing 3.5 toont de globale structuur van een TEI-document.

Listing 3.5: Skeletstructuur van een TEI instantie.

```
<TEI.2>
  <teiHeader> [ informatie over de tekst ] </teiHeader>
  <text>
    <front> [ voorwoord, erkenning, ... ] </front>
    <body> [ tekst in markup ] </body>
    <back> [ appendices, colofon, ... ] </back>
  </text>
</TEI.2>
```

De teiHeader-tag omvat de eigenlijke *metadata* van een TEI-bestand. Listing 3.6 geeft een voorbeeld van een mogelijke teiHeader voor deze thesis. Binnen de teiHeader kunnen vier tags voorkomen met elk hun aparte functie:

- **fileDesc** is een verplichte tag voor elk TEI-bestand. Het omvat informatie over personen die te maken hadden bij de creatie van het bestand (auteur, sponsers, vertalers, e.a.), de titel van de tekst, versie informatie, bestandsgrootte, publicatie-informatie (instituut/drukkerij, datum,..) en bronvermeldingen. Hiernaast bevat het eveneens een manier om het document te identificiëren in een groter geheel (bijvoorbeeld: deel van een conferentie-publicatie.)

- **encodingDesc** geeft meer informatie achter beslissingen die werden genomen tijdens de markup van de tekst. (Bijvoorbeeld: hoe werden quotaties behandeld, segmentering in zinnen of paragrafen, welke correcties werden doorgevoerd,..). Verder is het mogelijk in deze tag een taxonomie te definiëren waarnaar wordt verwezen vanuit de tekst. Tenslotte kan men in deze tag aangeven hoeveel keer een bepaalde tag voorkomt binnen de tekst en welke implicaties een tag heeft op de layout van een bestand. (bijvoorbeeld: p-tag komt 322 maal voor, en heeft een indentatie van 30 pixels.)

- **profileDesc** profileert een tekst op basis van haar creatie (bv: juni 2006, België), taal (bv: Nederlands) en classificatie (t.o.v. een thesaurus of m.b.v keywords). Het dient vooral ter ondersteuning van zoekopdrachten.

- **revisionDesc** bevat tenslotte een log van alle veranderingen die werden doorgevoerd. Elke wijziging wordt geannoteerd met een datum, een verantwoording en een link naar de persoon of organisatie die de wijziging doorvoerde.

Listing 3.6: Voorbeeld van een teiHeader

```
<teiHeader>
 <fileDesc>
  <titleStmt>
   <title>Multimedia Metadata Descr. Standards</title>
   <author>Luk Vloemans</author>
   <respStmt>
    <resp>Promotor</resp>
    <name>prof. dr. Wim Lamotte</name>
   </respStmt>
   ...
  </titleStmt>
  <extent>44532 bytes</extent>
  <publicationStmt>
   <distributor>tUL</distributor>
    <address>
     <addrLine>Agoralaan gebouw D,
        3590 Diepenbeek</addrLine>
    </address>
   <date value="2006">2006</date>
  </publicationStmt>
  <sourceDesc>
   <bibl>
    Adobe Systems Inc. XMP Specification,
    1.0 edition, june 2005. (http:/www.adobe.com)
   </bibl>
    ...
  </sourceDesc>
 </fileDesc>
 <encodingDesc>
  <projectDesc>
   <p>Eindverhandeling voorgedragen tot het
      behalen van de graad van Licentiaat
      in de Informatica afstudeervariant Databases
   </p>
  </projectDesc>
  <editorialDecl>
   <correction>
```

```
    <p>
       Tekst werd aangepast na elke
       revisie van thesis-coach.
    </p>
   </correction>
  </editorialDecl>
 </encodingDesc>
 <profileDesc>
  <creation>
    <date value="2006-06">Juni 2006</date>
    <name type="place">Hasselt, Belgium</name>
  </creation>
  <langUsage>
   <language wsd="NL" usage="100">Nederlands</language>
  </langUsage>
  <textClass>
   <keywords>
    <list>
    <item>metadata</item>
    <item>schema</item>
    <item>standaard</item>
    </list>
   </keywords>
  </textClass>
 </profileDesc>
 <revisionDesc>
  <list>
    <item>
     <date value="2006-04-27">27 april 2007</date>
     Revisie door <name>Peter Quax</name>
    </item>
    ....
  </list>
 </revisionDesc>
</teiHeader>
```

Het tekstgedeelte van de TEI bevat steeds een front, een body en een backgedeelte.
De front(matter) bevat typisch een titelblad en een aantal pagina's ter inleiding van de hoofdmaterie. Het titelblad kan men beschouwen als de kaft van een boek of een publicatie. Het bevat typisch velden zoals een titel, datum, auteur, e.a. Verder bevat de front mogelijk groepen tekst (gestructureerd door div-tags) die dienen ter inleiding van de tekst. (Bijvoorbeeld: het voorwoord, abstract, erkenning, opdraging,..)
De back(matter) bevat enkel groepen tekst (in div-tags) die bijdragen aan de inhoud van de body-tag. (Bijvoorbeeld: appendices, nota's, een bibliografie, colofon, etc.)

De body-tag bevat de inhoud van het document. De tekst wordt geformatteerd door markup-tags. Typisch wordt een document gesegmenteerd in hoofdstukken. Wanneer slechts één niveau onderverdeling noodzakelijk is voldoet het gebruik van de div-tag. Voor hiërargische onderverdeling gebruikt men div1 (hoogste niveau) t.e.m div7-tags (laagste niveau). Elk segment kan men voorzien van een header en een footer en verder onderverdelen in paragrafen, zinnen en zelfs zinstukken. De structuur van de body-tag is nauw verwant met de materie die ze beschrijft. TEI voorziet verschillende technieken voor markup van proza, vers, drama of gesproken tekst. Elk van deze type documenten heeft een verschillende tekst-layout. Zo

kan men in een stuk drama elke zin van een dialoog annoteren met een spreker, of event-tags toevoegen voor wanneer er iets gebeurt op de scène.
De markup-mogelijkheden van de TEI zijn vergelijkbaar met de output van een tekstverwerker. De TEI voorziet paginanummering, notatie-mogelijkheden, lijst en-tabelstructuren, formattering (cursief, bold,..), referenties,.. Verder is het mogelijk stukken tekst te identificeren. (Bijvoorbeeld: een quotatie, een datum, een afkorting,..)

TEI-Lite is een subset van de TEI, die origineel bedoeld was ter demonstratie van de kracht van de TEI. Vandaag wordt de TEI Lite echter aanvaard als defacto-standaard en overgenomen door de meeste implementaties van de TEI.

3.2.2 Metadata Encoding And Transmission Standard

De Metadata Encoding And Transmission Standard [Fed05] (METS) is een XML schema voor het beschrijven van digitale bibliotheek objecten. Meestal bestaan zulke objecten uit een aantal subobjecten die door de METS worden gegroepeerd. METS wordt vooral gebruikt voor het management van digitale objecten of tijdens communicatie tussen digitale bibliotheken. (of tussen bibliotheken en gebruikers). METS kan naast annotatie van tekstuele bestanden (bv: een boek) ook audio of visuele informatie annoteren. Een interessante eigenschap van METS is dat het metadata opsplitst in verschillende categorieën: Beschrijvende, structurele en technische metadata. METS introduceert zelf geen nieuwe metadata-elementen, maar steunt volledig op reeds bestaande schema's. (bijvoorbeeld, Dublin Core, VRA, TEI, NISO,..).

Een METS document bestaat uit zeven secties met elk hun aparte functie:

- De **METS Header** <metsHdr> bevat informatie over het METS packet. Het beschrijft wanneer het packet werd aangemaakt en gewijzigd, in welke status het packet zich bevindt (bv: afgewerkt) en welke agents bijdragen aan het packet. Een agent is een persoon of een organisatie en heeft steeds een rol t.o.v. het digitaal bestand (bv: creator, editor, archivist..)

- Na de header volgt een sectie met **beschrijvende metadata** <dmdSec>. METS voorziet de mogelijkheid om beschrijvende metadata los te koppelen van het METS-document (URI link naar een ander schema). Het is ook mogelijk om metadata (bijvoorbeeld Dublin Core) binnen het METS document op te slaan, in XML of binaire (base64) vorm.

- Voor **administratieve metadata** bevat METS vier verschillende onderverdelingen: Technische metadata (bestandscreatie, formaat, etc..), Rechten (copyright, licentie-informatie,..), Herkomst (informatie over de analoge bron van het digitaal bestand) en workflow-informatie (bron/doel-informatie, alsook transformaties toegepast op het bestand). Voor deze vier soorten metadata gebruikt METS hetzelfde systeem als voor beschrijvende metadata, metadata wordt opgehaald d.m.v. een URI, of metadata wordt binnen het METS document opgeslagen. De administratieve tags bevatten steeds een referentie-ID zodat andere METS-elementen kunnen refereren naar de metadata.

- De **File Section** <fileSec> wordt gebruikt om de bronbestanden die het METS-bestand beschrijft op te sommen. Deze bestanden kunnen worden gestructureerd in groepen. Groepen kunnen zelf opnieuw worden omvat door grotere groepen. Als voorbeeld nemen we een digitale versie van een ingescanned boek. De hoofdgroep omvat het boek

zelf. Elke subgroep van het boek bevat een pagina van het boek. Tenslotte bestaat een pagina-groep uit verschillende bestanden (ingescande tekst en foto's).
Elk bestand bevat een MIME-type, byteSize en creatiedatum. Het kan worden opgeslagen binnen het METS document (handig voor transmissie) of worden gerefereerd (d.m.v. een URI). Elk bestand kan tevens een pointer bevatten naar een element binnen de administratieve/beschrijvende metadata. (bijvoorbeeld een TEI-header, een DublinCore-beschrijving.)
Bestanden die dezelfde inhoud bevatten (maar bijvoorbeeld in andere formattering) worden gelinkt d.m.v. een GROUPID attribuut.

- De **Structural Map** wordt gebruikt om overzichten te creëeren van files. Een overzicht bevat pointers naar de file-section of mogelijk pointers naar andere METS documenten (voor zeer uitgebreide overzichten). Een filepointer kan een stuk van een bestand aanduiden (bijvoorbeeld een stuk tekst binnen een XML document). Elk overzicht wordt omvat door een element. Op die manier is het mogelijk (juist zoals MPEG-7) verschillende views te creëeren van dezelfde informatie. Elk element kan mogelijk worden gerelateerd met metadata uit de beschrijvende of administratieve metadata.

- De **Structural Links** sectie wordt gebruikt om links te definiëren tussen verschillende div's binnen de Structural Map, dit is vooral handig wanneer METS wordt gebruikt als container voor een website.

- Behaviour Links bevatten behaviours. Behaviours worden gebruikt om METS-bestanden te koppelen aan uitvoerbare code. (bijvoorbeeld een webservice) Een behaviour bestaat uit een *definitie*, waarin wordt vermeld welke functies kunnen worden toegepast (bijvoorbeeld een WSDL, of ander METS object) en een *mechanism*. Het mechanism is een referentie naar de uitvoerbare code (bijvoorbeeld een URL). Elk behaviour kan worden gelinkt aan administratieve metadata.

3.2.3 Publishing Requirements for Industry Standard Metadata

Publishing Requirements for Industry Standard Metadata. (PRISM) is een open metadata standaard die wordt onderhouden door IDEAlliance.
De PRISM-werkgroep ontstond in 1999 uit de samenwerking tussen uitgevers, rechthouders en handelaars van geschreven pers. Een XML standaard ter annotatie van uitgewisselde artikels zou geautomatiseerde verwerking en management van deze artikels bevorderen.
In 2001 ontstond uit dit orgaan de PRISM-specificatie [IDE05c] voor elektronische publicaties.

PRISM wordt uitgedrukt in W3C RDF. Haar specificatie introduceert vijf namespace schema:

- **Dublin Core Namespace**: De PRISM Dublin Core namespace (xmlns:dc) is een subset van Qualified Dublin Core. PRISM legt hogere restricties op aan de inhoud van de Dublin Core elementen. (Dublin Core legt geen restricties op, enkel voorkeuren.) In PRISM mag bijvoorbeeld het dc:format-element enkel een MIME-type bevatten en mag er in het dc:coverage element enkel temporele informatie worden opgeslagen.

- **PRISM Namespace**: (xmlns:prism) Het PRISM schema introduceert eenenvijftig elementen. Hiervan werden tien elementen overgenomen door een nieuwere versie van

Dublin Core en worden dus in de laatste versie van PRISM als overbodig beschouwd. De overblijvende eenenveertig elementen bestaan uit:

  - verdere uitbreidingen van Dublin Core elementen (dc:date(6), dc:relation(6), dc:rights(4) en dc:subject(6))
  - elementen voor identificatie van de tekst binnen een elektronische publicatie (12)
  - verantwoordelijke uitgever of organisatie (2)
  - andere: category, teaser, byteCount, wordCount, complianceProfile (5)

- **Rights Namespace**: Het rechtenschema (xmlns:prl) bestaat enkel uit drie basiselementen. De set is met opzet klein. Indien deze niet volstaat raadt PRISM aan een schema over te erven binnen de industrie. De voorgestelde elementen bevatten condities voor plaats, industrie en gebruik.

- **Inline Markup Namespace**: De (xmlns:pim) Inline Markup namespace werd toegevoegd om woorden of zinnen binnen een artikel te annoteren. De mogelijkheden zijn: gebeurtenis, industriesector, locatie, object-titel, organisatie, persoon of quotatie. De inline markup kan handig worden gebruikt voor zoekopdrachten. (bv: 'vind alle documenten waar Bill Gates in wordt vernoemd'.) Listing 3.7 toont een toepassing van de pim-elementen. (voor eenvoud zijn de attributen van de pim elementen weggelaten.)

Listing 3.7: Toepassing van inline markup elementen met PRISM

```
<p><pim:organization>Microsoft</pim:organization>
billionaire <pim:person>Bill Gates</pim:person>
has been hit with a cream pie in <pim:location>
Brussels</pim:location>. Gates responded mildly
saying <pim:quote>'the pie, just wasn't that
good.'</pim:quote></p>
```

- **Pam Namespace**: de PAM namespace werd toegevoegd ter ondersteuning van het PAM-aggregator DTD. (zie verder)

Voor de opslag van PRISM metadata raadt men gebruik van het XMP framework aan. Men kan PRISM ook opslaan als apart XML document.
PRISM introduceert een aantal taxonimiën ter invulling van sommigen PRISM-ementen. Deze taxonomieën zijn: **Resource Type Vocabulary** voor het dc:type element. (artikel, boek, illustratie, magazine,..) **Resource Category Vocabulary** voor het dc:category element. (abstract, interview, nieuwsbericht,..) **Rights en Usage Vocabulary** voor het prl:usage element. (gebruik, geen gebruik, eenmalig, ..) **PAM class Vocabulary** ter invulling van het class-attribuut in XHTML van PAM (zie verder.)

Indien men een nieuwe taxonomie nodig heeft voorziet PRISM een namespace (pcv) om taxonomieën te definiëren. Dit doet men d.m.v. een lijst pcv:descriptors op te sommen binnen een XML-document. Dit document kan dan worden uitgewisseld tussen producenten en aggregators als ondubbelzinnige declaratie van de inhoud van een bepaald element. De bovenstaande taxonomieën werden op deze manier gedefinieerd. Listing 3.8 geeft een voorbeeld van een pcv:descriptor.

Listing 3.8: PCV Descriptor in PRISM

```
<pcv:Descriptor rdf:about="Astrophysics">
  <pcv:label>Astrophysics</pcv:label>
  <pcv:broaderTerm rdf:resource="#Physics" />
  <pcv:narrowerTerm rdf:resource="#Cosmology" />
  <pcv:synonym>celestial mechanics</pcv:synonym>
  <pcv:definition>
    Includes cosmology; space plasmas;
    and interstellar and interplanetary
    gases and dust.
  </pcv:definition>
  <pcv:code>84</pcv:code>
</pcv:Descriptor>
```

De PRISM-specificatie is voldoende voor de uitwisseling van metadata tussen producenten en aggregators. Het werd echter snel duidelijk dat er ook nood was aan een specificatie om PRISM-metadata te versturen, samen met haar gerelateerde tekst. Hiervoor werd de PRISM Aggregator DTD (PAM) [IDE05b] geïntroduceerd. De DTD gebruikt een subset van W3C XHTML DTD. De elementen en taxonomieën die PAM gebruikt werden toegevoegd aan de PRISM-specificatie (zie hierboven.)

Een PAM-bericht start steeds met een pam:message. Een pam:message bevat minstens één pam:article (en een aantal namespace declaraties). Elk article bevat(naast namespace entaaldeclaraties) een header en mogelijk een body statement. De pam:header bevat de PRISM-metadata. Een speciaal element van de pam:header is het pam:status element. Dit element beschrijft wat de ontvanger van het PAM-artikel moet doen met het artikel (Toevoegen aan de database, verwijderen, updaten of corrigeren). Bij verwijdering of correctie is een body-element niet noodzakelijk. Bij een correctie bevat de header tevens een prism:hasCorrection element dat dient ter verduidelijking van een vorige versie.
Het pam:body element bevat de inhoud van het artikel in XHTML formaat. In de pam:body bestaan twee mogelijke soorten markups. De elementen uit de PRISM pim-namespace worden gebruikt om elementen aan te duiden binnen de tekst (zie hierboven). De PAM class Vocabulary (zie hierboven) wordt gebruikt om belangrijke segmenten binnen de tekst aan te duiden die typisch een andere layout krijgen wanneer ze worden vertoond in magazines. (bijvoorbeeld: byline, deck (=teaser-intro), sidebar,..).

3.3 Metadata voor afbeeldingen

3.3.1 Technical Metadata for Digital Still Images

Door gebrek aan een algemene standaard voor technische metadata en metadata omtrent productie van digitale foto's besloten de National Information Standards Organization (NISO) en de Association for Information and Image Management (AIIM) in 1999 een specificatie te ontwikkelen die deze doelen volbracht.
De beoogde metadata is belangrijk voor het achterhalen van de gedetailleerde herkomst van een gegeven digitale foto, en om ervoor te zorgen dat de foto accuraat wordt getoond aan de eindgebruiker.
De standaard kreeg de werktitel *Technical Metadata for Digital Still Images* [NIS02] maar wordt afgekort met Z39.87.

De elementen van de Z39.87 specificatie beschrijven basisinformatie over het digitaal bestand (grootte, checksum, orientatie) alsook haar formaat (MIME-type, compressie, segmentering, e.a.). De beschrijvingen omtrent oorsprong zijn gedetailleerd. Zo is het mogelijk om voor een gegeven digitaal beeld te achterhalen welk medium (bedrijf, computer, besturingssysteem, scanninghardware, scanningsoftware, e.a.) verantwoordelijk was voor haar creatie. Verder voorziet de specificatie een recursief element om vroegere transformaties te raadplegen. Elk niveau van de recursie beschrijft de toegepaste transformaties op een eerdere versie, alsook de gehele set elementen die die vorige versie beschreven.

De Z39.87 specificatie beschrijft niet hoe de metadata moet worden opgeslagen, noch in welk formaat. Ze is vrij beschikbaar als XML schema.[4] Haar werktitel luidt *Technical Metadata for Digital Still Images*. Het eerste ontwerp van deze specificatie was klaar in juni 2002. De specificatie wordt momenteel herbekeken en mogelijk voorgelegd aan ANSI [5] voor goedkeuring als Amerikaanse nationale standaard.

3.3.2 Exchangeable Image File Format

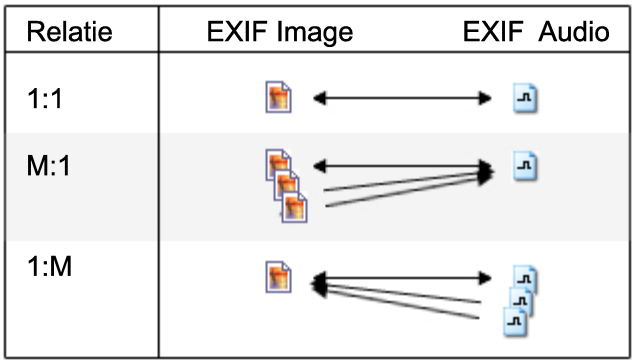
Het Exchangeable Image File Format [Jap02] (EXIF) werd ontwikkeld door de JEITA (Japan Electronics and Information Technology Association) in een poging de informatie die wordt uitgewisseld door digitale camera's en software te vereenvoudigen en te standaardiseren.
De standaard is meer dan een metadata standaard; ze is tevens een uitbreiding op de JPEG en TIFF bestandsformaat specificaties. Ze beschrijft welke compressie mag worden toegepast en op welke locatie en formattering haar metadata binnen een bestand moet worden opgeslagen. Dankzij deze restricties wordt EXIF opgenomen in firmware van een groot aantal apparaten. (Printers, Photoprinters, digitale camera's, e.a.).
Een aantal elementen van deze standaard werden overgeërfd van de TIFF 6.0 bestandsformaat specificatie. Elk EXIF-element wordt voorzien van een unieke waarde (tag) waarmee het element kan worden gerelateerd aan haar betekenis. De volledige set elementen kan worden onderverdeeld in twee hoofdcategorieën: Elementen die technische informatie over de foto beschrijven en elementen die technische informatie over de digitale camera en haar instellingen tijdens opname beschrijven. Voorbeelden van deze eerste categorie zijn breedte, compressie of oriëntatie. De tweede categorie bestaat uit elementen als belichting, digitale zoom, of flash. Sinds versie 2.0 biedt de EXIF standaard ook een set elementen voor beschrijving van geografische informatie. Camera's uitgerust met een GPS kunnen o.a. positie, hoogte, bewegingsvector en aantal verbonden satellieten wegschrijven in EXIF.
Tenslotte biedt EXIF een bescheiden ruimte voor persoonlijke notities, een link met een audiobestand en opslagruimte voor een thumbnail met gerelateerde metadata.
Omdat de EXIF standaard vandaag op grote schaal wordt toegepast is het belangrijk dat toekomstige versies compatible zijn met voorgangers.

EXIF staat bekend om haar tagging systeem voor foto's, maar bevat tevens een bescheiden set elementen ter annotatie van audio. EXIF Audio is gebaseerd op RIFF WAVE, een audio-containerformaat van Microsoft en IBM. RIFF bevat typisch een lijst chunks: Twee headerchunks, een datachunk (bytecode van audio-bestand), en een mogelijke LIST-chunk. De LIST-chunk (type INFO) bevat de metadata van het audiobestand. EXIF Audio erft zes metadata-elementen over van de LIST-chunk van RIFF: een titel (INAM), genre (IGNR),

[4] `http://www.loc.gov/standards/mix/`
[5] American National Standards Institute

datum (ICRD), commentaar (ICMT), verantwoordelijke software (IART) en een copyright-vermelding (ICOP).
Verder voegt EXIF een vijfde chunk toe aan de RIFF-container. Deze container heeft dezelfde structuur als de LIST-chunk, maar heeft het type *exif*. Software die de exif-chunk niet herkent zal deze chunk gewoon overslaan. De exif-chunk bevat zeven mogelijke elementen: versie (ever), relaties met foto-reeks (erel), tijd (etim), camera (ecor), camera-model (emdl), gebruikersannotaties (eucm) en een laatste element dat vrij mag worden ingevuld door implementaties (emnt).

De relatie tussen EXIF audio-en fotobestanden is eenvoudig: Wanneer een audiobestand werd opgenomen tijdens een fotosessie bevat elke foto een pointer naar het audiobestand. Het audiobestand zelf wijst enkel naar het eerste foto-bestand. Dezelfde soort relatie bestaat tussen audiobestanden die worden gerelateerd aan één foto. Toch kan op die manier (door relaties te vergelijken) eenvoudig de onderlinge structuur worden achterhaald. (one-to-one, one-to-many of many-to-one.) Zie figuur 3.8



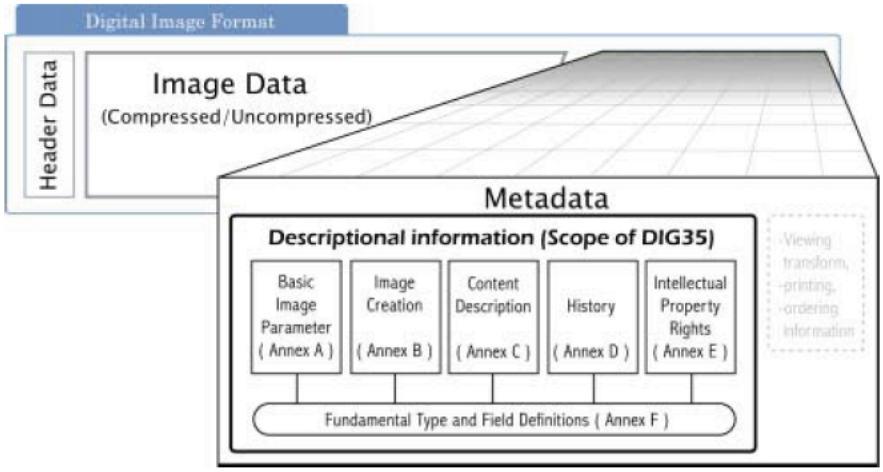
Figuur 3.8: Relaties tussen EXIF-bestanden

3.3.3 DIG35 Metadata Specification for Digital Image Metadata

DIG35 Metadata Specification for Digital Image Metadata [Int01] (DIG35) is een metadata standaard van de DIG (Digital Imaging Group). De standaard werd ontwikkeld door een werkgroep die bestond uit giganten binnen de digitale fotografie industrie. (bv: Adobe, Agfa, Canon, Kodak, Fuji, HP, Microsoft, e.a.)
DIG35 hekelt het feit dat software vaak metadata verliest tijdens conversie van bestandsformaten. De oorzaak hiervan is volgens de DIG35 specificatie het gebrek aan een standaard metadata-formaat ter annotatie van digitale fotografie. DIG35 is ontworpen voor bestandsformaten die encapsulatie van metadata toelaten. (bv: JPEG, TIFF, SPIFF). Door standarisatie zou de kost om de metadata te laten overleven, tijdens conversie naar een ander formaat, beperkt zijn.
DIG35's syntax wordt bepaald door XML schema (alsook een DTD) en onderscheidt vijf grote metadata *blocks*. Elk block beschrijft een uniek aspect van een foto. Implementaties zijn niet verplicht alle blocks in te vullen en mogen vrij elk block uitbreiden met eigen elementen. Figuur 3.9 geeft een idee van de architectuur van DIG35. In de komende paragraaf bespreken we de vijf blocks XML schema.

Interne Tags	ExifVersion, FlashPixVersion
Bestandsinformatie	ImageUniqueID
Auteur	Artist
Legaal	Copyright
DateTime	DateTime, DateTimeOriginal, DateTimeDigitized, SubSecTime, SubSecTimeOriginal, SubSecTimeDigitized
Inhoudelijk Beschrijvend	Description, HeadLine, Instructions, IntellectualGenre, Keywords, Scene, SubjectCode, Title
Technische Foto-Representatie	ImageWidth, ImageLength, Bits/Sample, Compression, PhotomtricInterpretation, Orientation, Samples/Pixel. PlanarConfiguration, YCbCrSubSampling, YCbCrPositioning, XResolution, Yresolution, ResolutionUnit, StripOffsets, RowsPerStrip, StripByteCounts, TransferFunction, WhitePoint, PrimaryChromaticities, YCbCrCoefficients, ReferenceBlackWhite, Colorspace, PixelXDimension, PixelYDimension, ComponentsConfiguration, CompressedBits/Pixel
Thumbnail	JPEGInterchangeFormat, JPEGInterchangeFormatLength
Beschrijvend	ImageDescription, MakerNote, UserComment
Recording Camera	Make, Model, Software, ExposureTime, Fnumber, ExposureProgram, SpectralSensivity, ISOSpeedRatings, OECF, ShutterSpeedValue, ApertureValue, BrightnessValue, ExposureBiasValue, MaxApertureValue, SubjectDistance, MeteringMode, LightSource, SubjectArea, FlashEnergy, SpatialFrequencyResponse, FocalPlaneXResolution, FocalPlaneYResolution, FocalPlaneResolutionUnit, SubjectLocation, ExposureIndex, FileSource, SceneType, CFAPattern, CustomRendered, ExposureMode, WhiteBalance, DigitalZoomRatio, SceneCaptureType, GainControl, Contrast, Saturation, Sharpness, DeviceSettingDescription, SubjectDistanceRange
Gerelateerde bestanden	RelatedSoundFile
GPS	GPSVersionID, GPSLatitudeRef, GPSLatitude, GPSLongitudeRef, GPSLongitude, GPSAltitudeRef, GPSAltitude, GPSTimeStamp, GPSSatellites, GPSStatus, GPSMeasureMode, GPSDOP, GPSSpeedRef, GPSSpeed, GPSTrackRef, GPSTrack, GPSImgDirectionRef, GPSImgDirection, GPSMapDatum, GPSDestLatitudeRef, GPSDestLatitude, GPSDestLongitudeRef, GPSDestLongitude, GPSDestBearingRef, GPSDestBearing, GPSDestDistanceRef, GPSDestDistance, GPSProcessingMethod, GPSAreaInformation, GPSDateStamp, GPSDifferential

Tabel 3.4: Elementen in EXIF



Figuur 3.9: DIG35 architectuur [Int01]-p8

- **Basic Image Parameter** is het equivalent van een *header* die men in de meeste fotografie-bestanden aantreft. Het bevat naast een unieke identifier (UUID) algemene informatie over het bestandsformaat, basis-afmetingen, gebruikte compressie, voorkeur-afmetingen en toegepast kleurpatroon. De informatie van dit schema bevat typisch redundante informatie die reeds voorkomt in de header van de geannoteerde foto. (Tenzij de foto geen header heeft, zoals bijvoorbeeld een bitmap). In het geval dat beide voorkomen heeft de header van het bestandsformaat voorkeur op het Basic Image Parameter block. Dit block wordt enkel gebruikt ter ondersteuning van indexering of zoekopdrachten.

- **Image Creation** bevat informatie over het tijdstip en het gebruikte toestel om de digitale foto te creëeren. DIG35 maakt een onderscheid tussen een scanner en een camera. Voor beide toestellen bevat het Image Creation block een exhaustieve opsomming van instellingen van het apparaat. Voor een camera kan men bijvoorbeeld lichtinval, zoom-instellingen of resolutie opslaan. Voor een scanner kan men naast instellingen ook informatie opslaan over het bronfragment dat gescanned werd (papieren foto of negatief).

- **Content Description Metadata** wordt typisch gebruikt voor classificatie en descriptie van foto's. Het is mogelijk de foto te annoteren met een titel, timestamp, commentaar, GPS-positie of zelf-gedefiniëerde eigenschap. De eigenschappen kunnen worden gelinkt aan een thesaurus. Verder kan men in deze block ook objecten binnen een foto identificeren. Zo is het mogelijk een persoon, organisatie, voorwerp of gebeurtenis te localiseren m.b.v. een rechthoek of een polygoon. Een gebeurtenis kan relaties tussen objecten weergeven, alsook worden opgesplitst in sub-gebeurtenissen. Tenslotte voorziet dit block een attribuut om audiobestanden te linken met de foto.

- **History Metadata** block bevat een aantal elementen die de geschiedenis van een foto weergeven. Het *Processing Summary* element bevat een aantal booleaanse waarden die in een oogopslag moeten duidelijk maken welke delen van de foto (over haar volledige levenscyclus) reeds werden aangepast. (bijvoorbeeld: cropped, colors-adjusted, meta-

data changed,..)
De *Image Processing Hints* bevatten dezelfde informatie als het bovenstaande, maar ze zijn meer volledig en eisen dat de opgesomde operaties voorkomen in de juiste volgorde. De *Previous History Metadata* bevat tenslotte de volledige DIG35 metadata van een vorige versie van de foto. Als de foto werd opgemaakt door compositie van verschillende foto's is het mogelijk de metadata van elke bron-foto te encapsuleren binnen de History tag. (recursief)

- **Intellectual Property Rights** (IPR). Deze block bevat alle elementen die weerslag hebben op de rechten verbonden met een bepaalde foto. De block bevat een *names*-element waarin de verschillende personen (fotograaf, producent, rechthouder, model, ..) die verantwoordelijk zijn voor de conceptie van de foto in worden gedefiniëerd. Het *description*-element bevat het copyright van de foto. Het *dates*-element bevat een lijst van datums waarop de foto een wijziging onderging, samen met een kleine beschrijving van die wijziging. (bv: foto genomen, foto ontwikkeld, foto ingescanned, foto aangepast,..)
  Het *Exploitation* element bevat een aantal velden die de gebruiker toegangs-of gebruiksrecht beschrijven. Het bevat tevens een container voor IPR-management systemen. Het *Contact Point*-element bevat contactgegevens van de rechthouders. Het *IPR-history*-element is tenslotte een container voor IPR-elementen die van toepassing waren op vorige versies van de foto.

Het zesde block metadata (**Fundamental Metadata Types and Fields**) dat zichtbaar is op figuur 3.9 beschrijft datatypes en structuren die worden gebruikt doorheen de schema's van de DIG35 specificatie. Het bevat elementen ter beschrijving van cijfers, taal, locaties, tijd, e-mail, website, personen,..etc.

De DIG35-specificatie beschrijft hoe de DIG35 metadata kan worden geïnserteerd in bestaande bestandsformaten zoals TIFF, JPEG of FlashPix. Het beschrijft alle mogelijke opties, zoals bijvoorbeeld het embedden van DIG35 metadata binnen de EXIF metadata van JPEG en TIFF, of toevoeging van nieuwe pointers binnen een bestandsstructuur naar de DIG35 metadata.
DIG35 wordt gebruikt als metadata standaard voor het JPEG2000 bestandsformaat.

3.3.4 Digital Image Submission Criteria Metadata

De Digital Image Submission Criteria Metadata [IDE05a] (DISC) is een specificatie die wordt gebruikt tijdens uitwisseling van digitale foto's. Haar scope is de communicatie tussen fotografen of agentschappen en magazines. De DISC-specificatie werd gepubliceerd in 2004 en is een initiatief van IDEAlliance. DISC legt geen restricties op in welke taal haar set elementen mag worden uitgedrukt, maar raadt gebruik van XML of RDF aan. De locatie van de metadata wordt eveneens niet vastgelegd in de specificatie, hoewel men gebruik van XMP aanraadt. (Men stelt een uitbreiding van de Adobe XMP Suite ter beschikking op hun website.)
De werkgroep die DISC specifieerde hield rekening met bestaande intiatieven zoals IPTC (sectie 3.1.5), XMP (sectie 3.1.4), Dublin Core (sectie 3.1.2) en PRISM (sectie 3.2.3). De DISC specificatie bevat mappings van haar eigen set elementen naar elk van deze standaarden.
De set elementen die DISC omvat is klein (zestien elementen) en eenvoudig. Juist zoals bij de Dublin Core zijn de elementen van de DISC zelfbeschrijvend en legt men geen restricties op voor de formattering van hun inhoud. (met uitzondering op het 'Date Created' veld.)

De elementen van DISC zijn: City, Copyright Notice, Country, Creator, Creator Contact Info, Date Created, Description, Headline, Instructions, Job ID, Keywords, Location, Original File Name, Provider, Province State en Source. Listing 3.9 geeft een voorbeeld van DISC metadata in XML.

Listing 3.9: DISC voorbeeld in XML

```
<!-- foto beschrijft EDM gebouw op
     universitaire campus te Diepenbeek -->
<example>
  <creator>Luk Vloemans</creator>
  <creatorContactInfo>
    luk.vloemans@student.uhasselt.be
  </creatorContactInfo>
  <orginalFileName>DSC00001.jpg</originalFileName>
  <provider>tUL</provider>
  <copyrightNotice>(C) tUL 2006</copyrightNotice>
  <source>tUL</source>
  <country>Belgium</country>
  <provinceState>Limburg</provinceState>
  <city>Diepenbeek</city>
  <location>Universiteitspark</location>
  <headline>EDM</headline>
  <description>EDM bij daglicht.</description>
  <dateCreated>03/06/2006</dateCreated>
  <instructions>Uploaden naar website</instructions>
  <keywords>EDM, Universiteitspark</keywords>
  <jobId>001</jobId>
</example>
```

3.4 Metadata voor audio

3.4.1 Identify an MP3

Toen de Moving Picture Experts Group het audioformaat MP3 introduceerde bevatte de standaard weinig ruimte voor metadata. Er waren enkel een aantal vlag-bits voorzien voor zeer rudimentaire informatie (Voorbeelden: Dit bestand heeft copyright rechten, dit bestand is een originele versie, e.a.). In 1996 bracht een software programma (Studio3) hier verandering in. Het programma verminkte de MP3 bestanden door 128 bytes informatie toe te voegen aan het bestand. Omdat programma's die deze informatie niet kenden er slecht op reageerden werd besloten deze informatie toe te voegen aan het einde van het MP3 bestand. De informatie kreeg de titel *Identify an MP3* (ID3-tag) en werd dankzij haar succes de eerste defacto-standaard voor het annoteren van MP3 bestanden.

De eerste versie van ID3(v1) [ID303] bevat een bescheiden set metadata elementen. Zo is het mogelijk een audiofragment te annoteren met een titel, artiestnaam, album, jaar van uitvoering, muziekgenre en extra commentaar. ID3v1 werd door het commentaarveld te beperken, uitgebreid naar ID3v1.1 met een extra element, nl. Nummer op het album.

Ondanks haar populariteit is ID3v1.1 voorbijgestreefd. De redenen hiervoor zijn divers: De zeven metadata velden zijn onvoldoende en te klein om een audiofragment behoorlijk te beschrijven. Daarbij is ID3v1.1 niet interessant voor streaming-doeleinden, mits de ID3v1.1 tag zich achteraan het bestand bevindt. Tenslotte is ID3v1.1 niet uitbreidbaar.

De *ID3v2-Tag* werd de opvolger van ID3v1.1. De nieuwe standaard is compatibel met haar voorganger, zodat software die enkel ID3v1.1 ondersteunt de ID3v2 header overslaat. De ID3v2-Tag bevindt zich voor de bytecode van het audiobestand (en mogelijk ook gedeeltelijk achteraan.) Ze is eenvoudig uitbreidbaar en beschrijft 82 standaard metadata elementen (zie tabel 3.5). Elk metadata element wordt voorgesteld door een *frame* binnen de tag. Elk frame is voorzien van een header die aanduidt hoe groot het frame is en een string van vier letters die specifieert welk type informatie het frame bevat.

De set metadata elementen voor het annoteren van auteursinformatie is uitgebreid. Zo kan men niet enkel artiest of groep, maar ook dirigent, tekstschrijver, componist, e.a. vermelden. Binnen deze categorie is het frame TMCL opmerkelijk, het voorziet ruimte voor annotatie van elk lid van het orkest en welk instrument deze persoon bespeelt. Naast een set elementen voor beschrijving van legale termen zoals copyright en gebruikstermen bevat de standaard ook elementen voor commerciële doeleinden. Zo kan iemand achterhalen hoeveel het audiofragment kost en waar hij het bestand kan aankopen.

Voor subjectieve annotaties heeft ID3v2 dezelfde set genres overgeërfd als ID3v1, maar ze zijn ongelimiteerd uitbreidbaar. Nieuw is het frame voor beschrijving van de gemoedstoestand die een bepaald fragment oproept (TMOO) of een frame voor populariteitsscore voor een bepaald fragment. (POPM)

ID3v2 is niet louter beschrijvend. Sommige elementen van de specificatie hebben effect op de manier waarop het fragment wordt afgespeeld. Zo is het mogelijk een stilte (TDLY) in te lassen voor het afspelen van een fragment. De gebruiker kan ook het tempo(SYTC), volume(RVA2), echo(RVRB) of equalisatie(EQU2) van een fragment instellen. Tijdens het beluisteren van muziek is het interessant de songtekst van het fragment te volgen. Met het USLT frame wordt een songtekst gesynchroniseerd met de muziek. Dit is vooral interessant voor karaoke muziek. Het ETCO frame is op dezelfde manier gesynchroniseerd maar beschrijft gebeurtenissen in het fragment. Bijvoorbeeld: "Start van de intro", "Start van het refrein".

3.4.2 APE Tag

De APE tag [Kle] werd ontworpen voor annotatie van het lossless *Monkey's Audio* formaat. De specificatie wordt heden ook toegepast op andere audioformaten zoals WavPack, OptimFROG en MP3. De eerste versie, APEv1 bestaat uit een aantal *Tag items* en een *Tag footer*. De footer bevat een marker waarmee software de tag kan terugvinden alsook de Tag-grootte en het aantal Tag items die ze bevat. Een Tag item bevat een ASCII sleutel, de grootte en het type van het item alsook de waarde van de sleutel. De tweede versie van APE (APEv2) biedt ondersteuning voor UTF-8 tekst en bevat naast een footer ook een tag-header. De header werd toegevoegd om de APE Tag binnen een audiobestand eenvoudiger op te sporen.

De APE specificatie bevat dertig beschrijvende basis elementen zoals artiest, songnummer of Album. Table 3.6 geeft hier een volledig overzicht van.

De APE specificatie veroorzaakt soms verwarring voor software omdat haar marker (APETAGEX) de ID3v1 marker bevat (TAG). Hiermee kan software de APE tag verkeerdelijk interpreteren als ID3v1.

Interne Tags	FrameEncryption(ENCR), FrameGroupID(GRID), SignatureFrame(SIGN), NextTag(SEEK)
Bestands informatie	Unique file identifier (UFID), ISRC (TSRC), MIME (TFLT), OriginalFilename (TDFN)
Auteursinformatie	Lead artist/group(TPE1), Band/Orchestra(TPE2), Conductor(TPE3), ModifiedBy(TPE4), Original Artist/Group(TOPE), LyricsBy(TEXT), Original LyricsBy(TOLY), Composer(TCOM), InstrumentByList (TMCL), Others(TIPL), EncodedBy(TENC), ArtistURL(WOAR)
Legaal	Copyright(TCOP), ProducerNotice(TPRO), Publisher(TPUB), FileOwner(TOWN), InternetRadioStation(TRSN), RadioStationOwner(TRSO), CopyrightURL(WCOP), PublisherURL(WPUB), TermsOfUse(USER), Ownership(OWNE)
Commercieel	CommercialURL(WCOM), Payment (WPAY), CommercialFrame(COMR)
DateTime	EncodingDT (TDEN), OriginalReleaseDT (TDOR), RecordingDT(TDRC), ReleaseDT(TDRL), TaggingDT(TDTG)
Technische Audio-Representatie	BPM (TBPM), Length(TLEN), Key(TKEY), PlaylistDelay(TDLY), MPEGLocationLookupTable(MLLT), SyncTempo(SYTC) RelVolAdj(RVA2), Equalisation(EQU2), Echo(RVRB), StreamingBuff(RBUF), StreamingPos(POSS), SoftwarePrivateTag(PRIV), Encryptie(AENC)
Geëncapsuleerd	Picture(APIC), OtherFile(GEOB)
Beschrijvend	Comments(COMM), Content group(TIT1), Songname(TIT2), Subtitle(TIT3), Genre(TCON), Mood(TMOO), SyncEvents(ETCO), UnSyncLyrics(USLT), SyncLyrics(SYLT), Popularity(POPM)
Herkomst	Album(TALB), Original Album(TOAL) Part of TALB (TPOS), Item within TPOS (TRCK), Set Subtitle (TSST), MediaType (TMED), SourceURL(WOAS), OfficialURL (WOAF), InternetRadioURL(WORS), CDID(MCDI), DeviceSettings(TSSE)
Gerelateerd	ExternalInfo(LINK)
Taal	Language(TLAN)
Playlist	AlbumSortOrder(TSOA), PerformerSortOrder(TSOP), TitleSortOrder(TSOT), PlayCounter(PCNT)

Tabel 3.5: Elementen in ID3v2

Bestands informatie	Catalog, ISRC
Auteursinformatie	Artist, Publisher, Conductor, Composer, LC, Bibliography
Legaal	Copyright, Publicationright
Geografische oorsprong	Record Location
DateTime	Year, Record Date
Beschrijvend	Comment, Abstract
Audio-Beschrijving	Title, Genre, Index, Introplay
Herkomst Medium	Album, Debut Album, Track, EAN/UPC, ISBN, Media
Gerelateerde bestanden	File
Taal	Language
Andere	Dummy

Tabel 3.6: Elementen in APE Tag

3.4.3 ITunes Music Library

iTunes is een applicatie die werd gelanceerd door Apple voor het afspelen en organiseren van audio-bestanden. Het programma wordt gebruikt als standaard interface tussen de populaire draagbare audio-speler iPod, en de iTunes Music Store (iTMS). iTunes maakt gebruik van metadata die latent wordt opgeslagen binnen muziekbestanden (zoals bijvoorbeeld de ID3 tag), maar bevat hiernaast ook een eigen metadata bibliotheek, de iTunes Music Library [Bor04].
De bibliotheek is een XML bestand dat zich bevindt op de harde schijf van een gebruiker. Een iPod bevat typisch een subset van deze metadata. (enkel de informatie die behoort tot de bestanden die zich op de iPod bevinden.) De iTunes Music Library bevat voor elk muziekfragment achtentwintig metadata elementen. De elementen zijn grotendeels zelfbeschrijvend en kunnen worden gemapped op andere tagging formaten zoals ID3. Listing 3.10 geeft een voorbeeld van een iTunes annotatie. Na een opsomming van de audiofragmenten bevat de bibliotheek eveneens een opsomming van playlists die refereren naar de verschillende muziekfragmenten. De structuur van de iTunes Music Library wordt bepaald door een DTD[6].

Listing 3.10: voorbeeld iTunes Music Library

```
<key>001</key>
<dict>
  <key>Track ID</key><integer>001</integer>
  <key>Name</key><string>Sweet Georgia Brown</string>
  <key>Artist</key>
     <string>Count Basie & His Orchestra</string>
  <key>Composer</key>
     <string>Bernie/Pinkard/Casey</string>
  <key>Album</key> <string>Prime Time</string>
  <key>Genre</key><string>Jazz</string>
  <key>Kind</key>
     <string>Protected AAC audio file</string>
  <key>Size</key><integer>3771502</integer>
  <key>Total Time</key><integer>219173</integer>
  <key>Disc Number</key><integer>1</integer>
  <key>Disc Count</key> <integer>1</integer>
```

[6] http://www.apple.com/DTDs/PropertyList-1.0.dtd

```
  <key>Track Number</key><integer>3</integer>
  <key>Track Count</key><integer>8</integer>
  <key>Year</key><integer>1977</integer>
  <key>Date Modified</key>
     <date>2004-06-16T18:10:55Z</date>
  <key>Date Added</key>
     <date>2004-06-16T18:08:31Z</date>
  <key>Bit Rate</key> <integer>128</integer>
  <key>Sample Rate</key><integer>44100</integer>
  <key>Play Count</key><integer>3</integer>
  <key>Play Date</key><integer>-1119376103</integer>
  <key>Play Date UTC</key>
     <date>2004-08-17T16:39:53Z</date>
  <key>Rating</key><integer>100</integer>
  <key>Artwork Count</key><integer>1</integer>
  <key>File Type</key><integer>1295274016</integer>
  <key>File Creator</key> <integer>1752133483</integer>
  <key>Location</key><string>file://c:/...</string>
  <key>File Folder Count</key><integer>4</integer>
  <key>Library Folder Count</key><integer>1</integer>
</dict>
```

3.5 Besluit

Door een overaanbod aan metadata standaarden die vaak zelfs overlappende elementen bevatten is het niet altijd eenvoudig om metadata uit te buiten. Software ontwikkelaars die in hun producten gebruik willen maken van metadata kunnen maar een beperkt aantal technologieën ondersteunen. De keuzes die ze maken zijn dus cruciaal.
Het zou kortzichtig zijn om voor een media-specifieke standaard te opteren indien er een generieke oplossing voorhanden is die even expressief is. Om volledig voorbereid te zijn op de toekomst kan men daarbij best kiezen voor een uitbreidbare standaard. Er kunnen immers nieuwe initiatieven opduiken, waarop men vandaag de dag niet kan inspelen.

Van de standaarden die we in dit hoofdstuk hebben besproken voldoen MPEG-7 en XMP aan deze criteria. IIM is generiek, maar helaas niet uitbreidbaar. De keuze tussen MPEG-7 en XMP wordt voornamelijk bepaald door het doel van de software. Hoewel XMP en MPEG-7 beiden ondersteuning bieden voor *media-specifieke annotaties*, gaat MPEG-7 dieper in op de structuur van het medium (bv:. annotaties ter ondersteuning van query by humming). MPEG-7 is dan ook aangewezen voor software die hiervan gebruik wenst te maken.

Indien *algemene annotaties* echter volstaan lijkt XMP ons de beste keuze. XMP is eenvoudiger te gebruiken wegens de toegankelijkere specificatie en kan op een transparante manier worden geëncapsuleerd in arbitraire mediabestanden. Een bijkomend voordeel van XMP is het feit dat het *in haar specificatie* reeds een aantal schema's overerft van bestaande succesvolle initiatieven (bv. Dublin Core en Exif). Hierdoor vergt bestaande software die reeds gebruik maakte van deze schema's slechts weinig aanpassingen. [GvOH05] beschrijft een lijst van vereisten voor praktische multimedia annotaties, waaronder: een eenvoudige specificatie, uitbreidbaarheid en hergebruik van bestaande vocabularies; eigenschappen die we ook terugvinden in XMP. In deze publicatie bekritiseert men verder de grootte en complexiteit van MPEG-7. Er wordt zelfs gesuggereerd dat het bestaan van MPEG-7 organisaties afschrikt

om te investeren in nieuwe initiatieven voor multimedia annotatie[7]. Tenslotte is het gebruik van RDF in XMP een pluspunt gezien de groeiende interesse voor het semantisch web.

Dit hoofdstuk vormt de basis voor hoofstuk 4 waarin we de verschillende metadata standaarden vergelijken.

[7]Nieuwe intiatieven worden vaak afgewezen omdat ze het werk van MPEG-7 dupliceren, maar desondanks staat men ook twijfelachtig tegenover adoptie van MPEG-7 wegens diens complexiteit.

Hoofdstuk 4

Vergelijking

Inhoudsopgave

In hoofdstuk 3 hebben we een aantal metadata standaarden beschouwd die worden gebruikt ter annotatie van multimedia. In deze sectie zullen we de verschillende schema's trachten te vergelijken op basis van hun uitdrukkingskracht. Hoewel elke standaard werd ontwikkeld voor een specifiek doel zijn er tussen standaarden onderling vaak overlappende elementen. Een abstractie van deze overlappende elementen kan nuttig worden gebruikt wanneer we een lijst van bestanden willen ondervragen die worden geannoteerd door verschillende standaarden. De webclient-applicatie die werd ontwikkeld in het kader van deze thesis zal de resultaten van deze sectie gebruiken.

Niet alle standaarden die werden besproken in hoofdstuk 3 komen in aanmerking voor een onderlinge vergelijking. Standaarden die worden gebruikt als container voor multimedia bestanden (zoals METS, MPEG-21 of WMM) beschouwen we niet, mits ze zelf geen nieuwe metadata elementen introduceren. Verder zullen we MPEG-7 in deze vergelijking niet opnemen. We proberen in dit hoofdstuk immers zoveel mogelijk gemeenschappelijke elementen te identificeren. MPEG-7 is echter de enige beschouwde standaard die veel dieper ingaat op de structuur van audio-visuele informatie. Deze annotaties komen niet voor in de andere standaarden, en bieden dus geen meerwaarde voor de vergelijking. De enige basis waarop we MPEG-7 wel kunnen vergelijken zijn algemene annotaties. Maar zoals we in sectie 3.5 aanhaalden is het gebruik van MPEG-7 voor dit doeleinde te complex. Het wegvallen van MPEG-7 heeft echter als gevolg dat we geen vergelijking kunnen maken tussen video-annotatie standaarden, vermits we nu enkel nog beschikken over XMP.

De standaarden die we onderling vergelijken staan opgesomd in tabel 4.1 samen met de

Standaard	**prefix**	**Namespace**
DISC	disc	http://www.ns.be/disc/2.0/
TEI-header	tei	http://www.ns.be/tei/p4/
IPOD	ipod	http://www.ns.be/ipod/
PRISM	prism	http://prismstandard.org/namespaces/basic/1.2/
	pam	http://prismstandard.org/namespaces/pam/0.0/
	pim	http://prismstandard.org/namespaces/pim/1.2/
QDC	dc	http://purl.org/dc/elements/1.1/
APE	ape	http://www.ns.be/apev2/
XMP	xmp	http://ns.adobe.com/xap/1.0/
	xmpRights	http://ns.adobe.com/xap/1.0/rights/
	xmpMM	http://ns.adobe.com/xap/1.0/mm/
	xmpBJ	http://ns.adobe.com/xap/1.0/bj/
	xmpTPg	http://ns.adobe.com/xap/1.0/t/pg/
	xmpDM	http://ns.adobe.com/xmp/1.0/DynamicMedia/
ID3v2	id3	http://www.ns.be/id3v2/
IPTC4XMP	iptc4xmp	http://iptc.org/std/Iptc4xmpCore/1.0/xmlns/
EXIF	exif	http://ns.adobe.com/exif/1.0/
Z39.87	mix	http://www.loc.gov/mix/
DIG35	dig35	http://www.digitalimaging.org/dig35/1.1/xml/

Tabel 4.1: Gebruikte standaarden voor vergelijking.

gerelateerde namespaces. De element-sets die niet beschikken over een default namespace kregen een namespace toegewezen onder het domein *http* : *//www.ns.be/*. Sommige standaarden introduceren meerdere namepaces zoals XMP of PRISM.
In sectie 4.1 vergelijken we de metadata standaarden door de sets elementen die ze introduceren rechtstreeks te mappen op elkaar. Dit kunnen we ook zien als een één-op-één mapping. Deze mapping is recht toe, recht aan. Bijvoorbeeld, het element "artiest" in ID3v2 heeft een rechtstreekse mapping op het element "artist" van XMP.
We voeren deze studie uit om in de applicatie van de thesis een set multimedia met heterogene annotaties te ondervragen. De verbanden die we aantreffen binnen dit hoofdstuk spelen hierbij een cruciale rol. In sectie 4.2 gaan we op zoek naar interessante relaties tussen verschillende elementen van metadata standaarden. We trachten een aantal voorbeelden uit te werken waarbij verscheidene metadata elementen toedragen aan een groter geheel. Bijvoorbeeld: we zouden kunnen besluiten dat de kwaliteit van een foto in relatie staat met haar resolutie, compressie, kleurenpatroon, etc.

4.1 Directe mapping

In de komende secties bespreken we rechtstreekse mapping (of één-op-één mapping) tussen metadata standaarden a.d.h.v. vergelijkingstabellen. Elke vergelijkingstabel stelt een aparte categorie voor (bijvoorbeeld bestandsannotaties, auteursannotaties, ...). De rijen van de vergelijkingstabellen stellen standaard-onafhankelijke elementen voor binnen een categorie. We definiëren deze vanaf nu als zijnde dimensies. De kolommen van de vergelijkingstabellen bevatten de standaarden van tabel 4.1. Elke cel in een vergelijkingstabel baseert zich op de aan-of afwezigheid van die bepaalde dimensie binnen de beschouwde standaard. Een gekleurde cirkel binnen een cel toont aan dat men de dimensie kan uitdrukken in die standaard.

Een kleurloze cirkel binnen een cel betekent dat de overeenkomstige standaard geen nieuw attribuut introduceert ter beschrijving van dat element, maar dat de standaard wel elementen overerft van een andere standaard (bijvoorbeeld Dublin Core) die dat element wel uitdrukt.
Sommige metadata standaarden zijn uitbreidbaar (bijvoorbeeld: door toevoeging van namespaces in XML.) We beperken ons echter tijdens deze bespreking tot de elementen die elke standaard zelf introduceert, of overerft zoals beschreven in hun specificatie. Bijlage A bevat een detailweergave van de tabellen die in deze sectie worden geïntroduceerd.

Opmerking De metadata elementen die wij beschouwen als deel van de TEI standaard werden overgenomen van de TEI-header, zoals beschreven in sectie 3.2.1.

Vergelijkingstabel bestandselementen

De bestandsannotaties omvatten elementen die worden gerelateerd met een bestand als alleenstaand geheel. Figuur 4.1 geeft een overzicht van de vergelijking op basis van bestandsannotaties.
MIMEtype kan men mappen op de specificaties van MIME door de Internet Engineering Task Force (IETF). De bestandsgrootte wordt meestal uitgedrukt in een grootorde van het aantal bytes. Bestandsversie en bestandsformaat bepalen welke versie van het MIME-type wordt gebruikt (bv: JFIF 1.02). Het veld byteorde (Z39.87) wordt uitgedrukt in big-of little endian. De checksum (Z39.87) wordt tenslotte gebruikt om zich ervan te vergewissen dat een geannoteerd bestand niet werd verminkt tijdens transmissie (Z39.87 legt geen restricties op over de inhoud van de checksum.)
Sommige standaarden leggen restricties op over de inhoud van identifiers: APE en ID3v2 baseren zich op ISRC, een internationale standaard voor digitale vingerafdrukken voor audiobestanden of EAN/UPC, een bekende barcode identifier.

	DC	XMP	IPTC4XMP	TEI	PRISM	DISC	EXIF	Z39_87	DIG35	IPOD	APE	ID3v2
Bestandsnaam		●				●						●
ID	●	●			○		●	●	●	●	●	●
Titel	●	○		●	○				●	●		
Bestandsgrootte	●	○			●	●		●		●		
Bestandsversie									●			
Bestandsformaat									●			
MIME	●	○			○			●	●	●		●
ByteOrder								●				
Checksum								●				

Figuur 4.1: Vergelijking van bestand annotaties

Vergelijkingstabel auteurselementen

De auteursannotaties omvatten elementen die te maken hebben met de persoon die het bestand oorspronkelijk maakte of personen die op zijn minst een contributie leverden. Zoals uit figuur 4.2 blijkt, heeft elke standaard een element ter beschrijving van de hoofdauteur van een multimedia bestand. Voor audio-standaarden zal dit eerder de band/zanger/spreker bevatten, terwijl deze elementen bij annotatie van afbeeldingen de verantwoordelijke fotograaf of grafische vormgever bevatten. Zoals reeds beschreven in sectie 3.4.1 bevat ID3v2 een heleboel tags om contributies te annoteren (ondergebracht in de tabel in rij "andere").
Een laatste auteursannotatie in figuur 4.2 bevat elementen die personen of organisaties beschrijven die verantwoordelijk waren voor de verspreiding van de multimedia.

	DC	XMP	IPTC4XMP	TEI	PRISM	DISC	EXIF	Z39_87	DIG35	IPOD	APE	ID3v2
Creator	●	●	●	●	●	●	●	●	●	●	●	●
ContactInfo Creator		○	●			●						
Creator Title		○	●									
Contributie (alg)	●	○			○							
Componist		●								●	●	●
Engineer		●										
Sponsor/Funder				●								
Dirigent											●	●
Andere												●
Verspreiding	●	○		●	●						●	●

Figuur 4.2: Vergelijking van auteur annotaties

Vergelijkingstabel datum elementen

In deze sectie vergelijken we de verschillende mogelijkheden die de beschouwde standaarden introduceren ter beschrijving van datums. Figuur 4.3 geeft een overzicht van de beschouwde datum-elementen. Een datum wordt door de meeste standaarden voorgesteld als (een subset van) ISO-8601 [ISO04]. Dit is een internationale standaard voor representatie van datum en tijd. Het wordt gerepresenteerd in een string: YYYY-MM-DDThh:mm:ss.sTZD. Standaarden die afwijken van deze representatie zijn DISC (DD/MM/YYYY) en Exif (YYYY:MM:DD HH:MM:SS).
De "andere"-rij bevat voor de (qualified) Dublin Core elementen de datums wanneer een bestand werd vrijgegeven, aanvaard, verstuurd en geannoteerd met copyright. XMP en PRISM erven deze elementen over. PRISM bevat tenslotte datum-elementen die zouden verschijnen op de cover van een magazine.

	DC	XMP	IPTC4XMP	TEI	PRISM	DISC	EXIF	Z39_87	DIG35	IPOD	APE	ID3v2
Datum (alg)	●	○			○							
Creatiedatum	●	●	●	●	●	●	●	●	●	●	●	●
Aangepast	●	●		●	●					●		
Release Datum		●									●	●
Gedigitaliseerd		○					●					●
Gepubliceerd	●	○			●							
Metadata toegevoegd		●										●
Gebruik/Verval Datum	●				●							
Andere	●	○			●							

Figuur 4.3: Vergelijking van datum annotaties

Vergelijkingstabel legale elementen

De meeste beschouwde metadata standaarden introduceren de mogelijkheid om legale aspecten van multimedia op te nemen. Figuur 4.4 geeft een vergelijkingstabel voor deze categorie. Het copyright-statement, de rechthouder en voorwaarden worden meestal voorgesteld door een string informatie, zonder vooropgestelde formattering. Qualified DC, XMP, DIG35 en ID3v2 voorzien ook elementen om legale informatie te relateren met online resources. DIG35 voorziet tenslotte containers voor gebruik door een IPR (Intellectual Property Rights) Management systeem.

	DC	XMP	IPTC4XMP	TEI	PRISM	DISC	EXIF	Z39_87	DIG35	IPOD	APE	ID3v2
Copyright	●	●	●	●	●	●	●		●		●	●
Rechthouder	●	●	●		●				●		●	
Voorwaarden	●	●	●		●				●			●
Link	●	●			○				●			●
Andere									●			

Figuur 4.4: Vergelijking van legale annotaties

Vergelijkingstabel geografische elementen

In tabel 4.5 staat beschreven welke standaarden elementen introduceren voor het beschrijven van een locatie, een scène of GPS-informatie. De standaarden die geen geografische metadata introduceerden werden niet opgenomen in de tabel. Exif bevat opmerkelijk veel elementen voor GPS-annotatie. Enkele uitzonderlijke voorbeelden zijn doelbestemming, snelheid en draaihoek van een camera t.o.v. de doelvector.

	DC	XMP	IPTC4XMP	PRISM	DISC	EXIF	DIG35	APE
Locatie	•	○	•	•	•		•	•
Country/Prov/City		○	•		•		•	
CountryCode		○	•					
ZipCode							•	
Scene		•	•				•	
GPS		○				•	•	

Figuur 4.5: Vergelijking van geografische annotaties

Vergelijkingstabel foto elementen

We beschouwen in deze vergelijking enkel de standaarden die nieuwe elementen introduceren voor afbeeldingen (EXIF, DIG35 en Z39.87). XMP erft de elementen van EXIF over (XMP werd om die reden niet opgenomen in deze tabel). In deze categorie zou een gedetailleerde vergelijking tussen de onderlinge elementen van elke standaard ons te ver zou afleiden van het doel van de thesis. Daarom groeperen we de elementen op een hoger niveau. Figuur 4.6 geeft hiervan een overzicht. De geschiedenis-elementen bevatten informatie over een vorige versie van de afbeelding, en de operaties die erop werden uitgevoerd. De voorkeursresolutie bepaalt op welke schaal de foto best kan worden gerepresentateerd.

	EXIF	Z39_87	DIG35
Resolutie	•	•	•
Kleur/Pixel informatie	•	•	•
Compressie		•	•
Camera	•	•	•
Camera-instellingen	•	•	•
Scanner		•	•
Scanner-instellingen		•	•
Relatie met audio	•		•
Voorkeursresolutie		•	•
Geschiedenis		•	•

Figuur 4.6: Vergelijking van foto annotaties

Vergelijkingstabel audio elementen

Figuur 4.7 toont aan welke standaarden elementen toedragen ter beschrijving van audio. We overlopen kort de dimensies die meer uitleg vereisen.

Tempo beschrijft het aantal beats per minute. Het herkomstmedium beschrijft de bron vanwaar het audiobestand afkomstig is (CD album, Radio,...). Elementen voor het audiosig-

naal bevatten bijvoorbeeld Sample rate of bitrate. Playlist informatie bevat o.a. informatie over hoe men het bestand moet ordenen in een alfabetische lijst, het aantal maal dat men het bestand reeds beluisterde of een subjectieve maatstaf voor populariteit van het bestand. Events duidt op een lijst tijdscodes die wordt gebruikt om interessante fragmenten binnen een audiobestand te identificeren (bijvoorbeeld refrein).
Playback informatie heeft effect op hoe het fragment mag worden afgespeeld. Bv.: volume, equalisatie-of echoaanpassingen, hoeveel seconden stilte men moet respecteren in het begin van een fragment, of indien het bestand moet worden herhaald in een oneindige lus (Bv.: een hartslag).
Tenslotte vermelden we een aantal interessante elementen die in figuur 4.7 werden voorgesteld als "andere" elementen: APE bevat een preview-element dat het meest interessante stuk in een audiofragment aanduidt. XMP bevat een element voor de voorkeursinstellingen van de audiospeakers. ID3v2 bevat tenslotte elementen voor encryptie, kostprijs, aankoop, songtekst en encapsulatie van digitale bestanden (waaronder thumbnails).

	ID3v2	APEv2	IPOD	XMP
Tempo	•		•	•
Lengte	•		•	•
Toonaard	•			•
Taal	•	•		•
Genre	•	•	•	•
HerkomstMedium	•	•	•	•
Audiosignaal				•
Playlist informatie	•		•	•
Events	•			•
Playback	•			•
Andere	•	•		•

Figuur 4.7: Vergelijking van audio annotaties

Besluit directe mappings

De vergelijkingstabellen in deze sectie kunnen we gebruiken ter ondersteuning van generieke zoekopdrachten in één dimensie. Zo kunnen we bijvoorbeeld weten welke standaarden ondersteuning bieden voor GPS-coördinaten. Een verzameling heterogene multimedia bestanden met verschillende typen annotaties kunnen we op basis van deze dimensies onderling relateren. Wanneer we trachten de verschillende dimensies te combineren kunnen we eveneens meer interessante zoekopdrachten ondersteunen. In sectie 4.2 gaan we hier verder op in.

4.2 Samengestelde mapping

De directe vergelijkingen uit de vorige sectie geven een duidelijk beeld van de dimensies die we kunnen uitdrukken met de beschouwde metadata standaarden, evenals de dimensies die

overlappen tussen meerdere standaarden. Het is echter interessanter om ook tussen verschillende dimensies verbanden te leggen. De dimensies die met elkaar verband houden dragen toe tot een groter geheel dat we in deze sectie definiëren als een *samengestelde mapping*. Via deze samengestelde mappings zou een gebruiker meer geavanceerde zoekopdrachten kunnen stellen. We illustreren dit idee a.d.h.v. een aantal uitgewerkte voorbeelden. We starten met de samengestelde mapping beeldkwaliteit.

4.2.1 Beeldkwaliteit

Een gebruiker zou graag een verzameling foto's ondervragen op basis van *beeldkwaliteit*. De lijst van bestanden is echter groot, en bevat (onvolledige) annotaties verspreid over verschillende metadata standaarden. De gebruiker heeft weinig kennis van de factoren die toedragen tot een goede beeldkwaliteit (compressie, resolutie- of kleurinstellingen, ...). Hij zou enkel de foto's van hoge (of misschien zelfs lage) kwaliteit willen opvragen.

We kunnen dit begrip beeldkwaliteit trachten te mappen op verschillende dimensies die voorkomen in de beschouwde metadata standaarden. Door aan elk relevant element een geschaleerde waarde en een gewicht toe te kennen kunnen we het begrip beeldkwaliteit mappen op een gewogen gemiddelde. Zulk gewogen gemiddelde introduceert twee krachtige voordelen.
Enerzijds kunnen we met één zoekopdracht *verschillende metadata schema's* ondervragen. Wanneer een element niet kan worden uitgedrukt in een bepaalde standaard kunnen andere elementen die wél aanwezig zijn binnen die standaard toedragen aan de evaluatie van het gemiddelde.
Anderzijds kan dit gewogen gemiddelde flexibel om met *onvolledige annotaties*. Zelfs wanneer een bestandsannotatie (bijvoorbeeld Exif) slechts informatie bevat over één element (bijvoorbeeld toegepaste compressie), kan deze informatie alsnog doorslag geven om het bestand op te nemen in de resultaatset van de zoekopdracht.

De formule voor het gewogen gemiddelde van de beeldkwaliteit ziet er uit als volgt:

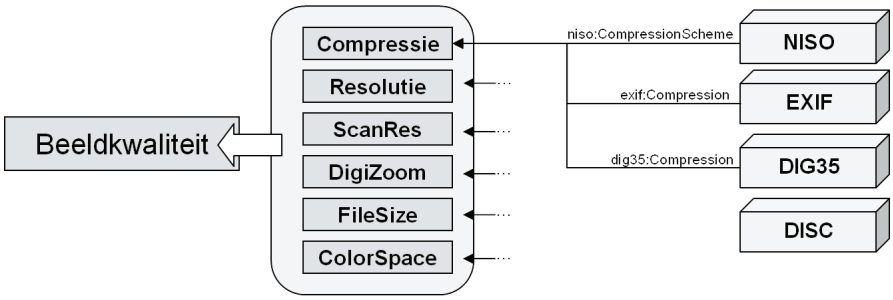
$$\overline{beeldkwaliteit} = \frac{\sum_{i=1}^{N} Gewicht_i * Eval_i(Elem_i)}{\sum_{i=1}^{N} \begin{cases} Gewicht_i, & \text{als } Elem_i \text{ bestaat} \\ 0, & \text{anders} \end{cases}} \tag{4.1}$$

Opmerkingen In vergelijking 4.1 berekent de Eval-functie de geschaleerde waarde voor een bepaalde dimensie. In het verdere verloop van deze sectie gebruiken we voor elke Eval-functie een schaal tussen 0 en 10. Tevens zorgt de noemer ervoor dat elementen die niet aanwezig zijn het eindresultaat niet beïnvloeden. De vergelijking houdt enkel rekening met de elementen die ter beschikking zijn.

Het schaleren en wegen van de verschillende elementen kan men subjectief interpreteren. Een *perfecte* schalering en gewichtstoekenning van deze elementen valt buiten het bestek van de thesis. We willen in de thesis de voordelen van het relateren van verschillende metadata-elementen aantonen. Daarenboven is het nuttig op te merken dat bepaalde evaluatiefuncties

afhankelijk zijn van evoluties binnen de industrie (de meeste professionele camera's beschikken tegenwoordig over een resolutie van meer dan 8.0 megapixels, terwijl deze resolutie over een aantal jaren enkel nog zal toebehoren aan de instapmodellen).

Figuur 4.8 verduidelijkt welke dimensies toedragen aan het begrip beeldkwaliteit. Elke dimensie mapt op zijn beurt op elementen van verschillende metadata standaarden. Wanneer we de verschillende metadata standaarden voor afbeeldingen overlopen vinden we voor beeldkwaliteit een aantal belangrijke factoren. We kiezen voor elke factor een voorbeeld-evaluatiefunctie, en gewichtstoekenning.



Figuur 4.8: Samengestelde mapping voor beeldkwaliteit

- Compressie: compressie-technieken hebben een grote invloed op de beeldkwaliteit. Wanneer het metadata schema aangeeft dat er geen compressie of lossless compressie (RLE[1], LZW[2],...) werd toegepast, kiezen we de maximale waarde op onze schaal. Bij lossy compressie is de waarde afhankelijk van het toegepaste compressieschema en mogelijk compressieniveau.

$$Eval_{compr}(s,n) = \begin{cases} 10, & lossless \\ 8, & \text{mogelijk lossy, n onbekend} \\ evalLossy(s,n), & \text{lossy, n bekend} \end{cases}$$

  De parameter s staat voor het compressieschema, terwijl de parameter n het compressieniveau bevat. In bovenstaande vergelijking wordt voor formaten die mogelijk lossy zijn (zoals JPEG of JPEG2000) waarde 8 toegekend wanneer hun compressie-niveau onbekend is. In dat geval kunnen we best veronderstellen dat de compressie weinig invloed had op de beeldkwaliteit, om geen resultaten uit te sluiten[3]. Wanneer we weten dat een lossy compressie werd toegepast met een bepaalde compressiefactor bepaalt de evalLossy-functie de uiteindelijke waarde. Hoe groter de toegepaste compressiefactor, hoe kleiner de waarde van de evalLossy-functie. Een perfecte uitwerking van deze functie valt buiten het bestek van de thesis. Voor het uitgewerkt voorbeeld op het einde van deze sectie werd de volgende evalLossy-functie gebruikt voor JPEG-compressie:

$$evalLossy(JPEG, niveau) = \left\lceil \frac{100 - niveau}{12,5} \right\rceil$$

[1]Run-length encoding
[2]Lempel Ziv Welch encoding, bekend van het GIF bestandsformaat
[3]JPEG2000 ondersteunt immers ook lossless compressie.

Bij een JPEG-compressie van 10%, 25% en 83% evalueert bovenstaande functie respectievelijk waarden 8, 6 en 2. Merk op dat de uitkomst van deze functie nooit hoger kan zijn dan de waarde 8. Ons inziens is dit een goede keuze aangezien we beelden met lossless compressie hoger inschatten dan beelden met lossy compressie.
Het gewicht voor de gebruikte compressie moet uiteraard zwaar doorwegen in vergelijking 4.1.

- Resolutie: foto's met een hoge resolutie kunnen meer detail bevatten en zouden dus moeten resulteren in een grotere waarde voor de beeldkwaliteit dan identieke foto's met een lagere resolutie. Grootte wordt uitgedrukt in megapixels (het product van pixelbreedte en -hoogte, gedeeld door een miljoen). Voor de schalering van de resolutie gebruiken we een logaritmische functie. Hierdoor zal de schaal naarmate de resolutie stijgt geleidelijk stijgen in subjectieve waarde. Foto's met 5 megapixels of hoger krijgen de hoogst mogelijke waarde. Proefondervindelijk bekomen we de volgende evaluatiefunctie:

$$Eval_{res}(Mpx) = \begin{cases} 10, & \textit{indien } \text{Mpx} > 5.0 \\ \lceil log_{1.5}(Mpx * 10) \rceil & \textit{indien } \text{Mpx} \leq 5.0 \end{cases}$$

Voorbeelden van de uitkomst van bovenstaande evaluatiefunctie op een aantal bestandsformaten:

Breedte	Hoogte	Mpx	Schalering	Subjectief
640	480	0.3072	3	slecht
800	600	0.4800	4	onvoldoende
1024	786	0.7864	6	voldoende
1600	1200	1.92	8	goed
3072	2304	7.0	10	uitstekend

Het gewicht van de resolutie zou ook zwaar moeten doorwegen in de vergelijkingsfunctie.

- Colorspace: Dit veld beschrijft welk kleurenmodel wordt gebruikt om de foto weer te geven. De modellen die in de beschouwde standaarden kunnen worden uitgedrukt zijn o.a. RGB, CMYK, YCbCr of CieLab. Het heeft weinig nut een bepaald kleurenmodel te verkiezen boven een ander, aangezien het gebruikte kleurenmodel geen zichtbaar effect heeft op de beeldkwaliteit. Als een gebruiker echter afbeeldingen verkiest met een bepaald kleurmodel (bijvoorbeeld CMYK voor publishing) kan hij deze nog steeds bekomen d.m.v. een specifieke zoekopdracht op de dimensie kleurmodel.
We kunnen echter wel een kleurenmodel dat enkel grijswaarden uitdrukt benadelen in de evaluatie.

$$Eval_{ColS}(Elem_i) = \begin{cases} 4, & \text{indien } Elem_i \text{ enkel grijswaarden bevat} \\ 10, & \text{indien } Elem_i \text{ anders} \end{cases}$$

Het gewicht voor de colorspace mag niet hoog doorwegen. Enkel wanneer er geen andere informatie ter beschikking is zouden we kunnen discrimineren op basis van grijswaarden. Op deze manier benadelen we bijvoorbeeld niet onnodig artistieke grijswaarden-foto's.

- Bestandsgrootte: kleine bestanden bevatten meestal minder informatie dan grotere bestanden. Men kan dus algemeen stellen dat een kleine bestandsgrootte duidt op een

slechte beeldkwaliteit[4]. De grootte van een bestand kan vooral een belangrijke factor zijn wanneer men over geen enkele andere annotatie beschikt. Hierdoor mag de waarde van deze schalering dus niet sterk doorwegen in de evaluatiefunctie. Een mogelijke evaluatiefunctie voor bestandsgrootte is:

$$Eval_{size}(KBytes) = \begin{cases} 10, & \text{indien KBytes} > 1000 \\ \left\lceil \frac{KBytes}{100} \right\rceil, & anders \end{cases}$$

Er zijn een aantal dimensies die een sterke correlatie hebben met de resolutie: nl. digitale zoom en scanresolutie. Men zou deze dimensies niet in rekening kunnen brengen voor de evalutie van de beeldkwaliteit. Ze zijn echter wel nuttig indien men foto's met een digitale zoom of met een lage scanresolutie extra wil benadelen t.o.v. andere foto's met dezelfde resolutie.

- Digitale zoom: De digitale-zoom functie op een camera heeft een negatief effect op de beeldkwaliteit van de uiteindelijke foto. Digitale zoom zal cropping toepassen waardoor de resolutie afneemt. Hierdoor verliezen we beeldkwaliteit. Hoe groter de digitale zoom, hoe kleiner de geschaleerde waarde. Een mogelijke evaluatiefunctie voor digitale zoom is:

$$Eval_{zoom}(factor) = \begin{cases} 10, & indien factor = 1.0 \\ \left\lceil log_{10}^{-1}(factor + cte) \right\rceil, & indien factor > 1.0 \end{cases}$$

  Waarbij *cte* gelijk is aan 0.2, zodat de evalutie van een zoom-factor 1.1 gelijk is aan 9. Als we deze evaluatiefunctie toepassen op waarden met digitale zoom 1.2, 1.9 en 3.0 verkrijgen we respectievelijk een waarde van 7, 4 en 2.

- ScanningResolution: Het aantal PPI (Pixels Per Inch) waarmee een foto werd ingescand heeft tevens een sterke invloed op de uiteindelijke resolutie. Hoe hoger het aantal PPI, hoe hoger de resolutie. We veronderstellen de volgende evaluatiefunctie voor scanresolutie:

$$Eval_{scanRes}(ppi) = \begin{cases} 10, & \text{indien ppi} \geq 600 \\ 8, & \text{indien } 300 \leq ppi < 600 \\ 5, & \text{indien } 200 \leq ppi < 300 \\ 3, & \text{indien } 100 \leq ppi < 200 \\ 2, & \text{indien ppi} < 100 \end{cases}$$

Zoals beschreven in de inleiding van deze sectie kunnen we de evaluatiefuncties die we introduceerden in bovenstaande opsomming gebruiken in vergelijking 4.1. Tabel 4.2 toont aan welke dimensies kunnen worden uitgedrukt in de foto-standaarden. We tonen het gebruik van de samengestelde functie *beeldkwaliteit* aan in een voorbeeld:

[4]Dit is niet volledig correct: de bestandsgrootte is ook afhankelijk van het type informatie dat de foto bevat (bv.: JPEG-compressie is effectiever bij een klein aantal verschillende frequenties, hetgeen bij zulke foto's dus resulteert in een kleinere bestandsgrootte, hoewel de algemene beeldkwaliteit identiek is).

	DISC	NISO Z39.87	DIG35	EXIF
Compressie	¬	√	√	¬
CompressieNiveau	¬	√	¬	¬
Kleurmodel	¬	√	√	√
Digitale Zoom	¬	¬	¬	√
Bestandsgrootte	√	√	¬	¬
Resolutie	¬	√	√	√

Tabel 4.2: Mogelijke dimensies in beschouwde standaarden.

Tabel 4.2.1 toont de configuratie van 11 digitale foto's. De laatste kolom van tabel 4.2.1 bevat het resultaat van het gewogen gewicht. Een lege cel in de tabel betekent dat de overeenkomstige dimensie niet kan worden uitgedrukt in die metadata standaard[5]. Het symbool [−] duidt op een ontbrekende waarde. Om het gewogen gewicht te bekomen kozen we voor de evaluatiefuncties de volgende gewichten:

- Compressie: 0.3
- Resolutie: 0.1
- Kleur: 0.01
- Digitale Zoom: 0.07
- Bestandsgrootte: 0.05
- Scanresolutie: 0.3

Opmerking In dit voorbeeld werden de optionele dimensies (digitale zoom en scanresolutie) ook in rekening gebracht.

		Compr	LvL	Color	DZ	Size	Scan	Width	Height	
1	DISC					[-]				0
2	NISO	JPEG	90	RGB		100kb	[-]	640	480	2
3	NISO	JPEG	90	RGB		954kb	[-]	3072	2304	4
4	DIG35	[-]		Gray			200ppi	[-]	[-]	5
5	EXIF			sRGB	2.0			1024	786	5
6	EXIF			sRGB	[-]			1024	786	6
7	NISO	JPEG	[-]	YcbCr		342kb	[-]	800	600	7
8	DISC					734kb				8
9	EXIF			sRGB	[-]			2048	1536	9
10	EXIF			sRGB	[-]			3072	2304	10
11	NISO	LZW	[-]	sRGB		2348kb	600ppi	3072	2304	10

We overlopen kort de elf voorbeelden uit tabel 4.2.1. De foto's staan gerankschikt op stijgende beeldkwaliteit.:

[5]Een aantal dimensies kan men steeds afleiden uit de attributen van een bestand (bv.: bestandsgrootte, resolutie). We willen hier echter illustreren wat de mogelijke waarde van de evaluatiefunctie is bij het ontbreken van die attributen. We baseren ons immers enkel op de informatie die aanwezig is binnen de metadata.

- Foto 1 is een foto zonder metadata.
- Foto's 2 en 3 zijn voorbeelden van foto's van verschillende grootte met een hoge compressiefactor (90%).
- Foto 4 is een ingescande foto met een grijs kleurmodel.
- Foto's 5 en 6 zijn identiek, maar foto 5 werd gemaakt met digitale zoom (2x).
- Foto 7 is een 800x600 kleurenfoto met mogelijk lossy JPEG compressie.
- Foto 8 is een foto waarvan we enkel de bestandsgrootte kennen.
- Foto's 9 en 10 zijn kleurenfoto's met zeer hoge resoluties.
- Foto 11 is een (lossless compressie) kleurenfoto met zeer hoge resolutie (7mpx), ingescand op 600ppi.

4.2.2 Portretfoto

Het principe van een samengestelde mapping werd reeds uitgelegd in sectie 4.2.1 a.d.h.v. de samengestelde mapping beeldkwaliteit. In deze sectie bespreken we de samengestelde mapping *portretfoto*. We schetsen het voordeel van deze samengestelde functie m.b.v. een korte use case:

Use Case Een studente geneeskunde wil graag op haar nieuwe profielpagina een foto plaatsen. In haar persoonlijke folder met afbeeldingen vindt ze een groot aantal foto's van verschillende aangelegenheden. De studente heeft geen kennis van digitale fotobewerking en gaat dus op zoek naar een portretfoto.

Op dezelfde manier als in sectie 4.2.1 bespreken we de verschillende evaluatiefuncties die toedragen aan de berekening van de samengestelde mapping. De vergelijkingsfunctie voor de mapping portretfoto is analoog aan vergelijking 4.1.

- Portrait Mode: een digitale camera beschikt meestal over modi voor verschillende fotografische doeleinden (landschap, actie, portret,...). We discrimineren irrelevante modi t.o.v. de portret-mode. Voor overige modi (bv.: manuele instelling, normale modus, ...) kunnen we deze dimensie best niet opnemen in de vergelijkingsfunctie. De modedimensie is duidend en kan dus best zwaar doorwegen in de vergelijkingsfunctie. Een mogelijke evaluatiefunctie voor portret mode is:

$$Eval_{pMode}(m) = \begin{cases} 10, & \text{indien } m = portret \\ 0, & \text{indien } m = landscape, actie, \ldots \\ N/A & anders \end{cases}$$

Opmerking De uitkomst N/A in bovenstaande evaluatiefunctie duidt op de engelse term *not applicable*. Wanneer een evaluatiefunctie deze waarde teruggeeft is het belangrijk dat het gerelateerde gewicht in de vergelijkingsfunctie niet wordt opgenomen.

- Resolutie: Men kan stellen dat de breedte van een ideale portretfoto 75% zo groot is als haar hoogte (verhouding 3:4). Wanneer men een portretfoto maakt met een camera zal men deze meestal in verticale positie houden. Wanneer een professionele fotograaf een foto bijsnijdt zal hij in de context van een portretfoto ook trachten te werken naar de verhouding van een pasfoto. Als de breedte of hoogte onderling sterk afwijken (naar een balk-vorm) moet deze functie een zeer lage waarde teruggeven. Het gewicht van deze evaluatiefunctie mag sterk doorwegen in de vergelijkingsfunctie. Een mogelijke evaluatiefunctie voor de resolutie is in deze context:

$$Eval_{res}(i = \frac{breedte}{hoogte}) = \begin{cases} 10 * (1 - |i - 0.75|), & \text{indien } 0.35 \leq i \leq 1.15 \\ 0, & \text{indien } anders \end{cases}$$

  Deze functie geeft voor resoluties 480x640, 200x400 en 100x400 respectievelijk waarden 10, 7.5 en 0 teruggeven.

- Pitch: DIG35 bevat een waarde die uitdrukt in welke *pitch* een foto werd getrokken. De pitch wordt uitgedrukt in een waarde tussen -90 (camera naar de grond gericht) of +90 (camera naar de hemel gericht). Een goede portretfoto (bv. een pasfoto) wordt genomen met een pitch tussen -10 en 10 graden. Wanneer deze informatie ter beschikking is moeten we haar zwaar laten doorwegen in de vergelijkingsfunctie. Een mogelijke evaluatiefunctie voor pitch is:

$$Eval_{pitch}(i = |p|) = \begin{cases} 10, & \text{indien } 0 \leq i \leq 10 \\ 0, & \text{indien } 45 \leq i \leq 90 \\ \left\lceil log_{10}^{-1}(i * 10) \right\rceil, & anders \end{cases}$$

- Roll: DIG35 bevat eveneens een maatstaf voor de rotatie van de camera tijdens opname (op een schaal van -180 tot 180 graden, waarbij 0 graden de camera horizontaal werd opgesteld.) Een perfecte portretfoto wordt dus opgenomen bij een hoek van -90 of 90 graden (o.w.v. dezelfde redenering die werd toegepast bij de evaluatiefunctie resolutie). Een mogelijke evaluatiefunctie voor de dimensie roll:

$$Eval_{roll}(i = |r|) = \begin{cases} 10, & \text{indien } 70 \leq r \leq 110 \\ 0, & \text{indien } anders \end{cases}$$

  Het gewicht van deze dimensie mag niet zo zwaar doorwegen als het gewicht van de pitch. Een foto in 4:3 verhouding (met een pitch van 0 graden) kan later immers later nog bijgesneden worden tot de juiste verhouding van een portretfoto.

- Subject Distance: De afstand tussen camera en object wordt uitgedrukt in meters. Wanneer deze informatie voorhanden is, kunnen we stellen dat een portretfoto wordt genomen tussen 0.5 en maximaal 4 meter. Afwijkende waarden worden gediscrimineerd:

$$Eval_{afstand}(m) = \begin{cases} 10, & \text{indien } 0.5 \leq m \leq 4 \\ 0, & \text{anders} \end{cases}$$

  De afstand tussen camera en model is een belangrijke maatstaf en kan best zwaar doorwegen in de vergelijkingsfunctie.

- Beschrijvende Inhoud: DIG35 bevat beschrijvende informatie over de inhoud die wordt afgebeeld binnen een scène. Wanneer een persoon wordt afgebeeld kunnen we niet met zekerheid besluiten dat de foto een portretfoto is. Echter, wanneer meerdere personen, een gebeurtenis, een object of een organisatie worden afgebeeld kunnen we uitsluiten dat de foto een portretfoto is. Wanneer deze informatie ter beschikking is kan haar gewicht best zwaar doorwegen in de vergelijkingsfunctie. De volgende evaluatiefunctie krijgt als input vier collecties (p,e,t,o) respectievelijk (people, events, things, organisations):

$$Eval_{inhoud}(p,e,t,o) = \begin{cases} 0, & \text{indien e,t,o} \neq leeg \\ 10, & \text{anders, indien aantal personen in p=1} \\ 0, & \text{anders} \end{cases}$$

  Opmerking Een collectie wordt in een RDF-graaf voorgesteld door een opsomming van de kinderen (rdf:bag, rdf:seq of rdf:alt). In een SPARQL-subgraaf is het niet mogelijk het aantal elementen van een collectie te ondervragen (aggregatie). We kunnen in een SPARQL-subgraaf dus niet het onderscheid maken tussen een collectie met één of meerdere elementen. We komen hier in sectie 6.3.4 op terug.

- Exposure Time: Het aantal seconden dat een foto werd belicht wordt in deze dimensie uitgedrukt. Deze waarde zou voor een portretfoto niet hoger mogen doorwegen dan een aantal honderste van een seconde. Een hoge exposure-time vereist dat de persoon tijdens het nemen van de foto niet beweegt. Wanneer dit wel gebeurt zal dat leiden tot wazige artefacten. Een perfecte inschatting van deze valt buiten het bestek van de thesis. We nemen als bovengrens voor deze dimensie 0.05 seconden.

$$Eval_{exposureT}(s) = \begin{cases} 10, & \text{indien s} \leq 0.05 \\ 0, & \text{anders} \end{cases}$$

  Het gewicht van de exposure dimensie mag zwaar doorwegen in de vergelijkingsfunctie maar heeft een sterke correlatie met de dimensie portrait mode. Daarom is het gebruik van deze dimensie slechts aangewezen wanneer men extra wil discrimineren op exposure.

Bijlage C.1 bevat een uitgewerkt voorbeeld voor de beeldkwaliteit.

Opmerking De Bluetooth sensor die werd gebruikt in de PDA applicatie van de thesis is in de context van deze samengestelde mapping een interessante extra dimensie. Wanneer we weten dat er mensen in de buurt waren op het moment dat de foto genomen werd is er een hogere kans op een portretfoto van een van deze personen.

4.2.3 Muziek Voorkeur

Als derde uitgewerkt voorbeeld beschouwen we de samengestelde mapping *muziekvoorkeur*. Met deze samengestelde mapping tonen we het nut van de combinatie van conventionele dimensies (zoals in bovenstaande voorbeelden) en dimensies die zich baseren op gebruikersinstellingen aan. Gebruikersinstellingen interpreteren we als een *hashmap* uit een conventionele programmeertaal. Het bevat voor een aantal sleutelwaarden, een subjectieve getalwaarde die werd ingesteld door de gebruiker. Evaluatiefuncties die afhankelijk zijn van een hashmap

definiëren we in deze sectie als *hashfuncties*.
Een hashfunctie gebruikt haar hashmap om een waarde toe te kennen binnen de vergelijkingsfunctie. Wanneer deze waarde niet wordt teruggevonden (of wanneer de hashfunctie niet beschikt over een hashmap) wordt dit type functie niet opgenomen in het gewogen gewicht.
Een hashmap kan het unieke sleutelwoord *else* bevatten waardoor een mismatch steeds resulteert in de waarde die overeenkomt met het sleutelwoord *else*.
De evaluatie van een hashfunctie (HF) is als volgt:

$$HF(hashmap, waarde) = \begin{cases} hashmap(waarde), & \text{indien waarde} \in hasmap \\ hashmap(else), & \text{indien else} \in hasmap \\ N/A, & \text{anders} \end{cases}$$

Een aantal dimensies die mappen op muziek voorkeur maken gebruik van deze hashfunctie. We zullen telkens kort een mogelijke hashmap aanhalen. Omdat een hashfunctie persoonlijke voorkeur uitdrukt moet ze natuurlijk zwaar doorwegen in de vergelijkingsfunctie.

- FavLang: een gebruiker met een uitgesproken voorkeur voor Franse en Nederlandstalige muziek, maar een afkeur van Engelse muziek zou devolgende hashmap kunnen gebruiken:

  NL ⇒ 10
  FR ⇒ 10
  EN ⇒ 0

  De sleutelwaarden in bovenstaande mapping zijn gebaseerd op ISO-639-1 [ISO02], een bekende standaard voor representatie van taal.

- FavYear: met deze hashfunctie kan een gebruiker favoriete tijdsperiodes aanduiden, evenals periodes van weinig interesse discrimineren. Een mogelijke hashfunctie voor een gebruiker die een grote voorkeur heeft voor de zomer van 1969, maar een hekel heeft aan hedendaagse muziek is als volgt:

  { 1969-07-01 → 1969-08-31 } ⇒ 10
  { 2000-00-00 → 3000-00-00 } ⇒ 0
  else ⇒ 7

  Merk op dat FavYear gebruikt maakt van een hashmap die een datum mapt binnen een interval. De hashmap van FavYear baseert zich op ISO-8601 [ISO04].

- FavGenre: de hashmap van deze hashfunctie bevat de favoriete muziekgenres van de gebruiker. Een mogelijk voorbeeld:

  Rock ⇒ 10
  Pop ⇒ 8
  Soul ⇒ 2
  else ⇒ 6

  De hashmap van FavGenre baseert zich op de genres gedefinieerd in [ID303].

- FavMood: ID3v2 bevat een element dat de *mood* van een bepaald muziekfragment beschrijft. Een mogelijke hashmap voor Favmood is als volgt:

  Happy ⇒ 10
  Relaxing ⇒ 7
  Romantic ⇒ 0
  Sad ⇒ 0

  ID3 specifieert geen taxonomie voor de inhoud deze categorie. We definiëren in de uitwerking een kleine taxonomie ter illustratie.

- TimesPlayed (TP): Het aantal maal dat een muziekfragment werd afgespeeld door de gebruiker kan duiden op een voorkeur voor dat fragment. Een mogelijke evaluatiefunctie voor TimesPlayed is:

$$Eval_{TP}(n,c,d) = \begin{cases} N/A, & \text{indien dateCheck(c,d) = false} \\ \lceil \frac{n}{10} \rceil, & \text{indien n} \leq 100 \wedge \text{ dateCheck(c,d) = true} \\ 10, & \text{anders} \end{cases}$$

  Waarbij n een waarde is die uitdrukt hoeveel maal het fragment reeds werd afgespeeld. c is de creatiedatum van het betand. d is tenslotte een constante die uitdrukt hoeveel dagen nodig zijn vooraleer een bestand oud genoeg is om het op te nemen in de beoordeling. De functie dateCheck vergelijkt d met de huidige datum. Indien het aantal dagen groter is, of gelijk is aan d geeft deze functie *true* terug.
  De dateCheck functie zorgt ervoor dat nieuwe bestanden die werden toegevoegd aan het systeem niet onterrecht worden gediscrimineerd t.o.v. oude bestanden. We veronderstellen een maximale waarde voor een bestand dat honderd maal werd afgespeeld. Een lineaire vergelijking (trapvorm) bepaalt de evaluatie van waarden lager dan honderd. Het gewicht van deze evaluatiefunctie mag zwaar doorwegen in de vergelijkingsfunctie.

  Opmerking Een mogelijke variant voor de TimesPlayed functie kan het aantal maal dat een bestand werd afgespeeld, afwegen t.o.v. de aantal dagen dat het bestand bestaat. Deze functie is echter sterk afhankelijk van gebruiker tot gebruiker (een intensieve luisteraar zal veel hoger scoren). Om deze reden is bovenstaande functie stabieler.

- Rating: sommige metadata standaarden voorzien een element dat een subjectieve beoordelingswaarde toekent aan een muziekfragment. Deze waarde is in deze samengestelde functie belangrijk en mag zwaar doorwegen in de vergelijkingsfunctie. De parameter r in de onderstaande evaluatiefunctie voor rating heeft een waarde tussen 0 en 100.

$$Eval_{rating}(r) = \frac{r}{10}$$

Bijlage C.2 bevat een uitgewerkt voorbeeld voor deze samengestelde mapping.

4.2.4 Andere

In dit hoofdstuk hebben we een aantal samengestelde mappings uitgewerkt. In deze sectie overlopen we een aantal extra mogelijke samengestelde functies. We vermelden steeds welke factoren toedragen tot de vergelijkingsfunctie.

Audiokwaliteit

Op dezelfde manier als de samengestelde mapping beeldkwaliteit zou het interessant zijn een set audiobestanden te ondervragen op *audiokwaliteit*. In deze samengestelde functie zijn de samplerate (aantal samples per seconde) en het aantal audiokanalen (mono, stereo,...) belangrijke factoren. Verder zijn de toegepaste compressie (lossless of lossy) en de herkomst van het fragment (radio, cd, dvd,...) belangrijke indicatoren. Een factor die eveneens meespeelt in deze vergelijking is de bitrate (aantal verwerkte bits per seconde). Hierbij merken we op dat bitrate enkel kan worden toegepast wanneer we beschikken over een verzameling homogene bestandstypen. Wanneer we beschikken over een heterogene verzameling (bv.: WAV, MP3 en Ogg Vorbis bestanden) is de bitrate (in de context van deze samengestelde functie) waardeloos.

Audiogenre

Wanneer we beschikken over de dimensie genre kunnen we de samengestelde mapping *audiogenre* bepalen a.d.h.v. deze factor. Wanneer we niet beschikken over deze informatie kunnen we trachten het audiogenre van een fragment te bepalen d.m.v. een aantal andere factoren die aanwezig zijn binnen de metadata van het audiobestand. Veel van deze factoren kunnen subjectief worden geïnterpreteerd. Daarom is deze samengestelde mapping afhankelijk van een groot aantal hasfuncties:
Men kan stellen dat oude muziek (bv.: +40 jaar) mapt op het audiogenre *oldies*. Een muziekfragment waarvan we de naam van een dirigent kennen, kan mappen op het audiogenre *classical*, *symphony* of *choir*. De songtekst (lyrics) van een bestand kan eveneens toedragen tot de evaluatie van het genre. Bv.: Een songtekst met veel vloekwoorden zal bijvoorbeeld toedragen tot het genre *rap*, een songtekst met een zinsdeel dat uitermate veel wordt herhaald kan duiden op het genre *house*.
Kennis over de gebruikte instrumenten kan ook bijdragen tot het genre, zeker wanneer we weten hoeveel instrumenten werden gebruikt. (Bv.: 20 violen kunnen we mogelijk mappen op *classical*, terwijl één trompet kan mappen op het genre *jazz*.
Een laatste factor is aantal BPM (beats per minute). Bv.: Wanneer we weten dat het aantal BPM hoger is dan negentig kunnen we besluiten dat we te maken hebben met *house* of dergelijke variant.

Landschapsfoto

Voor een landschapsfoto kunnen we ongeveer op dezelfde wijze als voor een portretfoto (zie sectie 4.2.2) conclusies trekken uit de beschikbare metadata. Een landschapsfoto zal voor Exposure Time en Pitch dezelfde evaluatiefuncties bevatten. Het gewicht voor pitch mag voor een landschapsfoto nog sterker doorwegen. Men zal immmers nooit een landschapsfoto nemen, met de camera naar de grond of hemel gericht. Een grote Subject Distance zullen we positief laten doorwegen. Voor verhouding tussen hoogte en breedte van de foto zullen we in dit geval streven naar een 4:3 verhouding. Wanneer de camera werd ingesteld op de Portrait Mode *landscape* weegt dit positief door in de vergelijkingsfunctie. Tenslotte zullen we voor objecten op de foto (een persoon, gebeurtenis, organisatie of voorwerp) een negatieve score toekennen.

Andere

De bovenstaande samengestelde mappings tonen aan dat verbanden tussen directe mappings kunnen worden gebruikt voor interessante zoekopdrachten. We sommen in de thesis geen exhaustieve lijst van mogelijkheden op. Ongetwijfeld bestaan er binnen het bereik van de de beschouwde standaarden nog een aantal voorbeelden.

4.2.5 Hoger Niveau Samengestelde Mapping

Wanneer twee of meerdere samengestelde mappings een overeenkomstig verband uitdrukken zouden we deze mappings kunnen gebruiken in een nieuwe samengestelde mapping. Op die manier introduceren we een derde niveau mapping waarmee de gebruiker zoekopdrachten kan uitvoeren. Als voorbeeld beschouwen we de samengestelde mappings beeldkwaliteit en audiokwaliteit. We zouden hiermee een samengestelde dimensie *kwaliteit* kunnen definiëren die gebruik maakt van beide mappings.

4.3 Besluit

In sectie 4.1 introduceerden we een aantal directe mappings tussen de beschouwde metadata standaarden van hoofdstuk 3. De generieke logische verbanden (auteur, tijd, ...) zorgen ervoor dat we over verschillende multimedia, met heterogene annotatieschema's zoekopdrachten kunnen stellen. In sectie 4.2 werd duidelijk dat we m.b.v. verbanden tussen directe mappings op een hoger niveau zoekopdrachten kunnen combineren in *samengestelde mappings*. Deze informatie over de metadata biedt krachtige mogelijkheden. Aangezien metadata informatie over data is, kunnen we informatie over metadata beschouwen als meta-metadata. De Object Management Group (OMG[6]) definieert meta-metadata ook als *M2*[7] oftewel metadata van het 2^{de} niveau [XMI98]. We kunnen met de samengestelde mappings, schema's van verschillende grootte-orde uitdrukkingskracht ondervragen. Zoals we hebben aangetoond is deze methode ook toepasbaar op annotaties met ontbrekende informatie.

Zoals reeds vermeld hebben de waarden die we toekennen aan de gewichten in de samengestelde mappings een belangrijke invloed op het resultaat. Elk gewicht is daarbij afhankelijk van de wensen van een gebruiker. Bv.: het gewicht voor TimesPlayed in de samengestelde mapping *muziek voorkeur* is waarde nul voor een gebruiker die deze dimensie onbelangrijk vindt. Daarom is er nood aan hoge instelbaarheid (customization) in een applicatie die samengestelde mappings ondersteund.

Als uitbreiding zouden we de samengestelde mappings nog onderling kunnen relateren om op deze manier tot metadata van het 3^{de} niveau te komen, zoals beschreven in 4.2.5.

In hoofdstuk 6 bespreken we op welke manier we deze functionaliteit ondersteunen.

[6] `http://www.omg.org/`

[7] Hierbij staat *M0* voor data (metadata van het 0^{de} niveau) en *M1* voor metadata (metadata van het 1^{ste} niveau). In feite wordt deze notatie gebruikt voor metamodellen, maar we kunnen ze ook toepassen op metadata in het algemeen.

Deel III

Ontwikkeling

Hoofdstuk 5

Use Case

Inhoudsopgave

5.1 Use Case

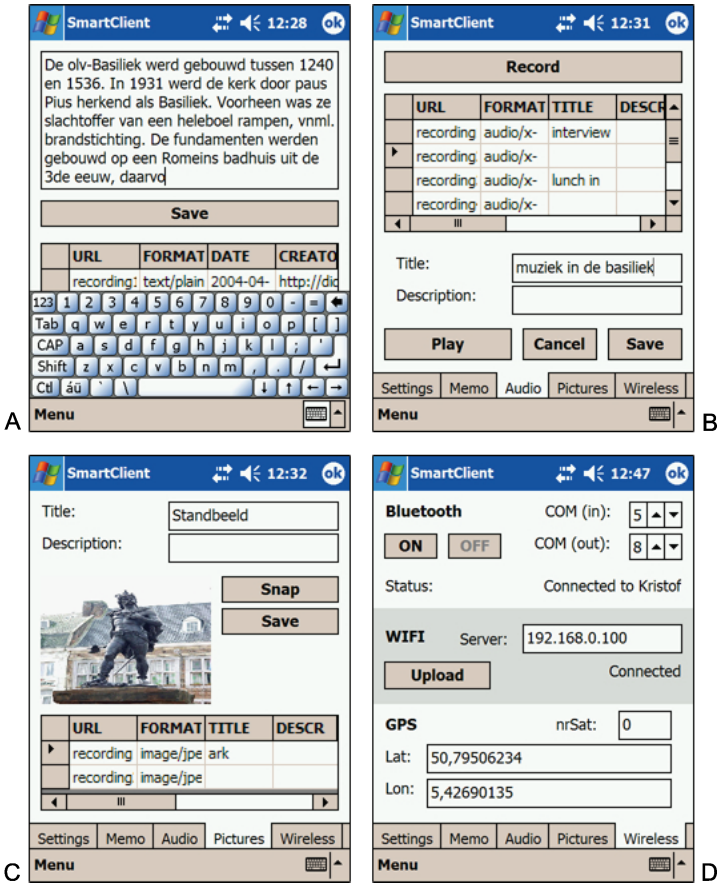
In het kader van deze thesis werd als praktische toepassing een applicatie ontwikkeld met drie componenten: een PDA-client component, een servercomponent en een webclient component. In dit hoofdstuk bekijken we een voorbeeldscenario waarin het gebruik en de voordelen van bovenstaande applicaties duidelijk wordt. In hoofdstuk 6 bespreken we technische details en implementatie.

5.2 Uitweiding

Christine in Tongeren

Christine gaat met haar klas op excursie naar Tongeren, de oudste stad van België. Met een lijst van vragen over de oude Romeinen mag Christine met haar vrienden de stad en haar cultuur verkennen. Ze neemt foto's (figuur 5.1-C) met haar PDA van de belangrijkste artefacten, monumenten en schouwspelen. Ze bezoekt o.a. het Begijnenhof, het Wapenmuseum, het Gallo-Romeins museum, het standbeeld van Ambiorix, de Basiliek en andere bezienswaardigheden. Af en toe neemt ze een kort interview (figuur 5.1-B) af van een voorbijganger over de betekenis van een bepaald object, of neemt ze nota's (figuur 5.1-A) als ze iets interessants verneemt van een gids of brochure. Wanneer Christine een vriend(in) ontmoet synchroniseert ze haar PDA met de PDA van haar vriend(in) via Bluetooth. (figuur 5.1-D)
Thuisgekomen synchroniseert Christine haar PDA met de multimedia-server en kan ze rustig haar bestanden bekijken.

Een jaar later krijgt Christine de opdracht een opstel te schrijven over de koning van de Gallische stam, Ambiorix. Christine herinnert zich haar excursie naar Tongeren en denkt dat ze mogelijk notities of foto's van het ambiorix-standbeeld kan gebruiken in haar opstel.



Figuur 5.1: PDA component in Tongeren

Christine gebruikt de webclient om in te loggen op de multimedia-server. Ze weet dat de bestanden werden gemaakt in Tongeren, dus stelt ze een filter in, die de multimedia die werden geregistreerd in een straal van 5 kilometer in de omstreek van Tongeren teruggeeft. (figuur 5.2)

Helaas is haar eerste poging niet productief genoeg, de zoekopdracht heeft 321 resultaten. Christine is nogmaals in Tongeren geweest, twee jaar geleden met haar ouders op de zondagsmarkt en een aantal maanden geleden naar een theatervoorstelling. De geografische filter was dus niet nauwkeurig genoeg.

In een tweede poging besluit Christine een tijdsgebonden filter toe te voegen zodat de fragmenten van 2004 en 2006 niet meer worden beschouwd. De excursie naar Tongeren vond zeker ergens plaats tussen april en juni 2005. (figuur 5.3) De toegepaste filter geeft inderdaad dit maal enkel nog resultaten van de excursie naar Tongeren. De resultaatset is echter nog steeds te groot, ze heeft tijdens de excursie erg veel bestanden geregistreerd. Christine heeft niet de tijd om een honderdtal fragmenten te doorzoeken.

Gelukkig herinnert Christine zich dat ze dicht bij een vriendin stond toen ze bij het Ambiorixbeeld stond. Deze informatie kan ze gebruiken als extra filter om het aantal resultaten

Figuur 5.2: Eerste filter: geografische omtrek

nog verder af te slanken. Inderdaad, na de derde filter vindt Christine enkel nog een klein aantal multimediafragmenten waarvoor het niet moeilijk zal zijn de Ambiorix gerelateerde bestanden terug te vinden. (figuur 5.4)

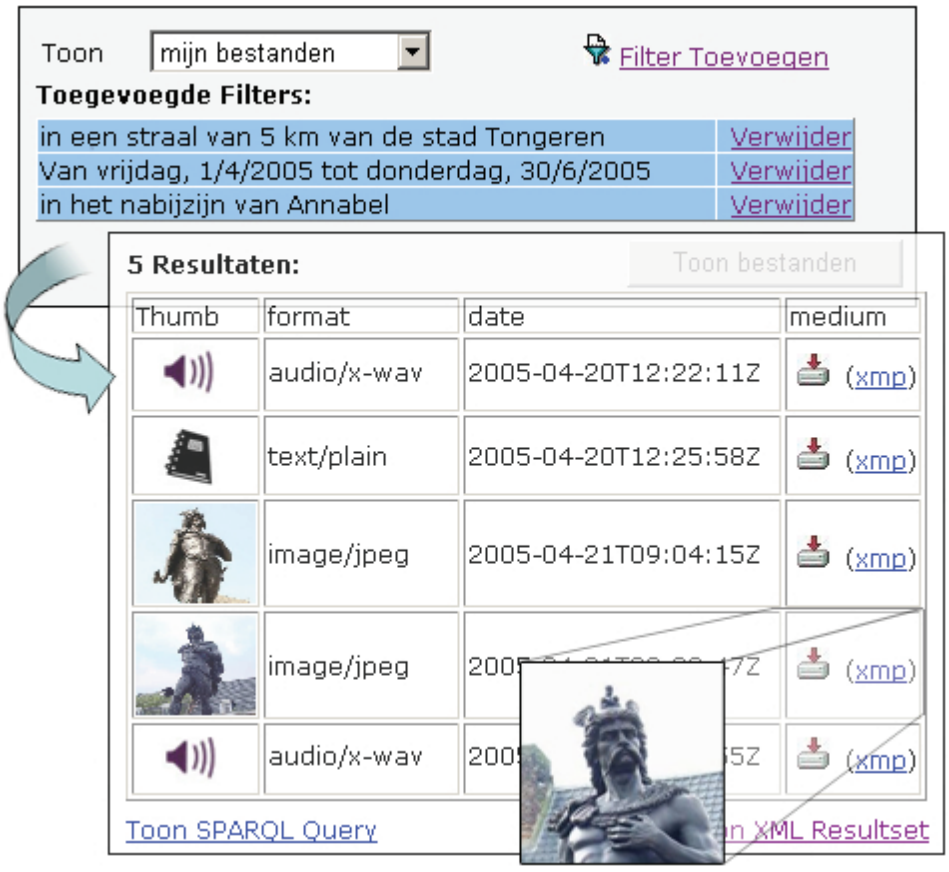
Christine in Antwerpen

Christine is vrijwilliger bij VZW Lentekind, een centrum voor kinderzorg en gezinsondersteuning in Antwerpen. Elke zomer gaat Christine, samen met een tiental andere vrijwilligers helpen in de speelpleinwerking van deze vereniging.
Dit jaar zijn er helaas te weinig volwassen begeleiders om de speelpleinwerking te behouden. Christine wil graag helpen door promotie te voeren voor de werking. Ze besluit een poster te maken die ze kan verspreiden in een aantal hogescholen en universiteiten.

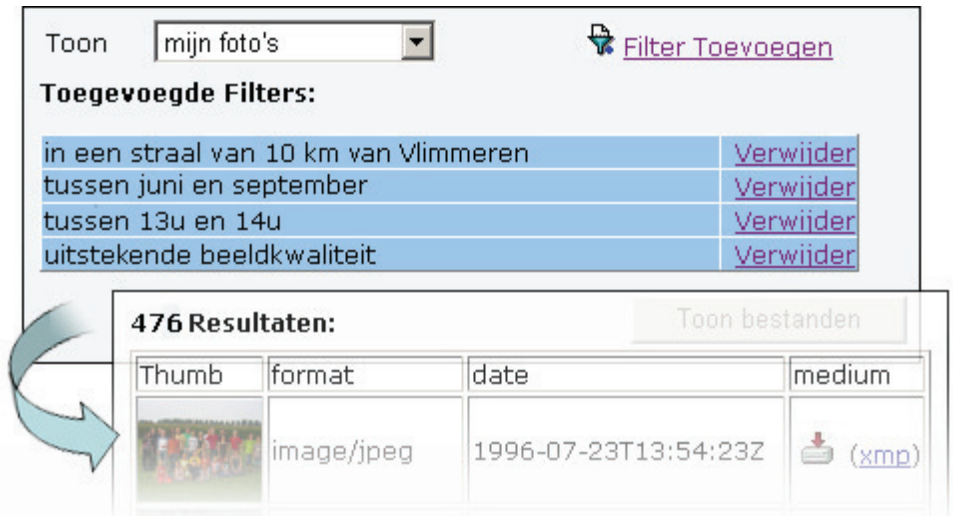
Christine logt in op de webclient en kiest een filter die resultaten teruggeeft van elke zomerperiode (tussen juni en september), binnen een straal van 10 kilometer van Vlimmeren (het dorpje van Lentekind). Christine vindt een heleboel foto's met vrienden en kinderen. De kinderen van het opvangcentrum mogen o.w.v. legale redenen niet verschijnen op een foto. Daarom besluit Christine een extra tijdsfilter toe te voegen, die enkel foto's teruggeeft tussen 13u en 14u 's middags, wanneer de leiding pauze heeft.

De resultaatset bevat ditmaal enkel nog foto's van haar vrienden, maar verschillen qua resolutie en kwaliteit. Dit komt omdat Christine vaak haar digitale camera moest instellen op een lagere kwaliteit o.w.v. gebrek aan opslagruimte. Christine stelt daarom de filter beeldkwaliteit in op *uitstekend*. De foto's van de resultaatset bevatten nu inderdaad enkel foto's van hoge kwaliteit. Christine selecteert een leuke groepsfoto en kan aan de slag met haar programma voor digitale fotobewerking.

Figuur 5.3: Tweede filter: tijdsgebonden



Figuur 5.4: Resultaten van derde filter



Figuur 5.5: Filter met beeldkwaliteit: uitstekend

Hoofdstuk 6

Implementatie

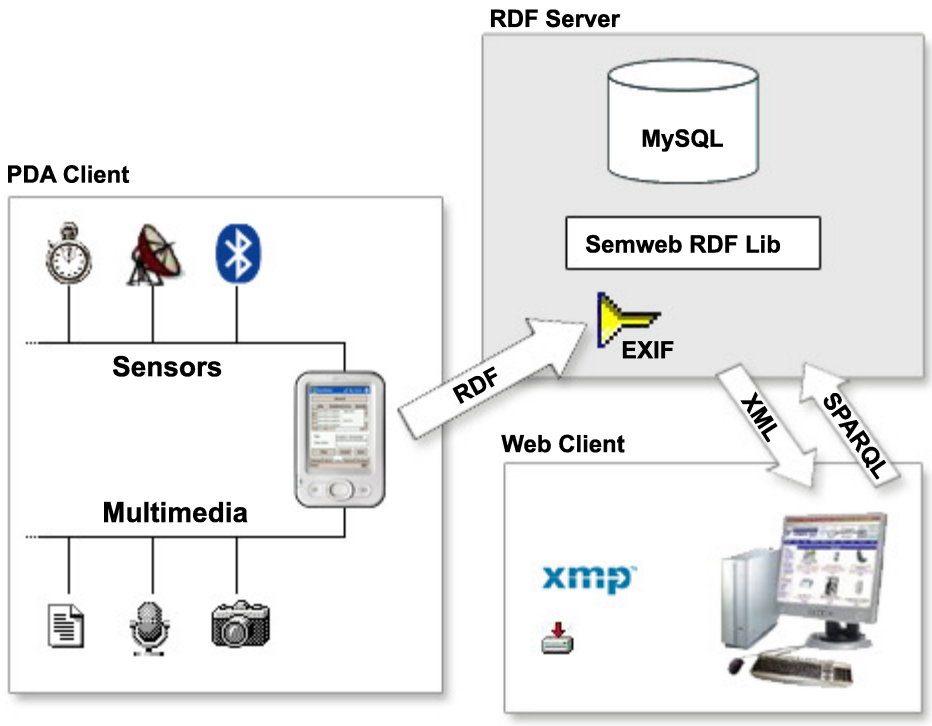
Inhoudsopgave

In dit hoofdstuk bespreken we de applicatie die werd ontwikkeld in het kader van de thesis. Figuur 6.1 geeft een overzicht van de componenten van deze applicatie en de gebruikte technologieën. In sectie 6.1 bespreken we de mobiele component die een gebruiker in staat stelt geannoteerde multimedia op te slaan op een PDA en vervolgens deze informatie door te sturen naar de server component. In sectie 6.2 beschrijven we de server die wordt gebruikt voor permanente opslag van RDF metadata. In sectie 6.3 bespreken we tenslotte de webcomponent die de gebruiker in staat stelt de metadata te onderbevragen a.d.h.v. SPARQL.

Opmerkingen

De metadata schema's die werden gebruikt in hoofdstuk 4 worden in deze applicatie opgeslagen op de server. Deze schema's zijn beschreven in RDF (bv. XMP), XML Schema (bv. DIG35) of in een set elementen (bv. ID3v2). Om alle schema's op een uniforme manier te kunnen benaderen veronderstellen we dat alle schema's beschreven zijn in RDF. Dit is ons inziens geen beperking aangezien steeds meer metadata beschreven wordt in RDF. Voor deze applicatie gebruikten we een recht toe, recht aan vertaling van het XML Schema in RDF. Een volledige vertaling van XML Schema naar RDF is mogelijk maar omslachtig [HL01] en valt buiten het bestek van deze thesis. We gebruiken tabel 4.1 als basis voor de namespaces van de applicatie.

Sommige dimensies eisen een vooropgestelde formattering. We nemen als voorbeeld een



Figuur 6.1: Overzicht van applicatie

dimensie die zich baseert op een datum. Zoals aangegeven in sectie 4.1 wijken DISC en EXIF af van de standaard formattering, nl. ISO-8601 [ISO04]. We veronderstellen dat de beschouwde standaarden overeenkomen in de formattering voor dimensies die semantisch dezelfde waarde uitdrukken. In een eenvoudige preprocessing stap zouden deze waarden kunnen worden geconverteerd naar een uniforme syntax.

We veronderstellen dat de RDF-server naast de PDA component tevens beschikt over andere leveranciers voor geannoteerde multimedia. De PDA component dient ter illustratie van een mogelijke leverancier.

6.1 Mobiele Leverancier voor Metadata

De PDA component die voor deze thesis werd ontworpen is geschreven in C# m.b.v. het Microsoft Compact .NET Framework. Vooraleer men de PDA component kan gebruiken moet men eerst via de Web component een profiel aanmaken op de server.

De PDA component dient ter illustratie van een multimedia metadata leverancier. De metadata wordt aangeleverd door een aantal sensoren (GPS, Bluetooth, Tijd,..) die de PDA ter beschikking heeft. Figuur 5.1 toont de user interface van de PDA component.

De PDA component beschikt over een abstracte klasse Sensor, waarvan de verschillende

sensoren afleiden. Bij initialisatie van de applicatie wordt verduidelijkt welke sensoren mogen worden gebruikt op basis van een XML document (*schema.xml*). Listing 6.1 geeft een voorbeeld van dit schema. Het schema bevat voor elke sensor een element dat later wordt gebruikt om de sensor te identificeren in RDF. Wanneer een nieuwe sensor ter beschikking is (bijvoorbeeld een temperatuursensor) moet men enkel een extra klasse afleiden van de Sensor klasse, en haar element toevoegen aan het XML document.

Wanneer een multimedia fragment wordt opgenomen worden de gewenste sensoren uit het XML-document vergeleken met de beschikbare sensors. Bij elke match wordt (indien de sensor actief is) de waarde van de sensor bewaard in een RDF bibliotheek.

Wanneer de PDA component toegang heeft tot een draadloos netwerk kan men de bewaarde gegevens uploaden naar de Server component. Elk multimedia bestand wordt dan samen met haar metadata (een RDF-bestand) via SOAP doorgestuurd naar de webserver.

Listing 6.1: XML initialisatie schema

```
<?xml version="1.0"?>
<Schema
  xmlns:dc="http://purl.org/dc/elements/1.1/"
  xmlns:dig35="http://www.digitalimaging.org/dig35/1.1/xml/"
  xmlns:exif="http://ns.adobe.com/exif/1.0/"
>
 <Sensors>
  <Sensor key='DATE'      elem='dc:date'/>
  <Sensor key='CREATOR'   elem='dc:creator'/>
  <Sensor key='PROXIMITY' elem='dig35:personName'>
  <Sensor key='LAT'       elem='exif:GPSLatitude'/>
  <Sensor key='LON'       elem='exif:GPSLongitude'/>
 </Sensors>
 <Multimedia type='Pictures'>
  <Annotation key='FORMAT'  elem='dc:format'/>
  <Annotation key='TITLE'   elem='dc:title'/>
  <Annotation key='DESCR'   elem='dc:comment'/>
 </Multimedia>
 <Multimedia type='AudioFragments'>
  <Annotation key='FORMAT'  elem='dc:format'/>
  <Annotation key='TITLE'   elem='dc:title'/>
  <Annotation key='DESCR'   elem='dc:comment'/>
 </Multimedia>
 <Multimedia type='Memos'>
  <Annotation key='FORMAT'  elem='dc:format'/>
 </Multimedia>
</Schema>
```

6.2 RDF Server voor Metadata Opslag

De server component van deze thesis biedt permanente opslag voor RDF-gestructureerde metadata. De server voorziet een aantal functies die via SOAP door de metadata leveranciers en de webclient kunnen worden aangeroepen. Tabel 6.1 geeft een overzicht van de functies die de SOAP service van de webserver voorziet, met overeenkomstige commentaar. In sectie 6.2.1 bespreken we SemWeb, een library voor het .NET platform die permanente opslag biedt en

addFile(byte[] file, str RDF)	Het binair bestand wordt toegevoegd aan de MySQL database. De RDF wordt geparsed en toegevoegd aan de library
executeSparQL(str query)	de SPARQL select query wordt uitgevoerd en de XML-resultset wordt teruggegeven.
getFile(int id, bool isImage)	geeft byte array terug van het gevraagde bestand. Op basis van isImage tracht de server een XMP packet toe te voegen aan het bestand.
getXMP(int id)	geeft XMP packet terug van het gerelateerde bestand (id)
login(str login, str pwd)	indien login en paswoord overeenkomen, geeft deze functie een unieke persoonID (string).
regMap(str id, dataset map)	deze functie registreert een hashMap binnen de server. Hashmaps werden besproken in sectie 4.2.3.

Tabel 6.1: Functies die de server aanbiedt via SOAP.

ondervraging van RDF-gestructureerde informatie ondersteunt. In sectie 6.2.2 bespreken we kort hoe een digitale foto die werd doorgestuurd vanuit de PDA-client wordt onderzocht voor EXIF metadata. Tenslotte bespreken we in sectie 6.2.3 hoe de webserver gebruik maakt van het Adobe XMP framework.

6.2.1 SemWeb

In de thesis kozen we de querytaal SPARQL voor de Webclient. Hierdoor was er nood aan een library voor permanente opslag van RDF en ondervraging op basis van SPARQL. Twee libraries werden in dit kader beschouwd: Dave Beckett's Redland RDF Application Framework, en Joshua Tauberer's Semantic Web/RDF Libary (SemWeb). Beide platformen boden voldoende ondersteuning voor de Server component. SemWeb werd specifiek ontwikkeld voor het .NET platform. Dit gaf uiteindelijk de doorslag, mits de rest van de applicatie ook werd ontwikkeld voor .NET.

SemWeb gebruikt MySQL als database voor permanente opslag van RDF. Voor ondersteuning van SPARQL gebruikt SemWeb de SPARQL engine voor Sesame[1] via IKVM[2].

In de eerste fasen van ontwikkeling gaf SemWeb goede resultaten voor SPARQL queries. In meer uitgebreide queries werden een aantal gebreken of bugs aangetroffen in de library. Zo was het oorspronkelijk niet mogelijk OPTIONAL en UNION statements te gebruiken in een query. Verder was er oorspronkelijk geen ondersteuning voor extension functions. Door deze problemen te melden aan de ontwikkelaar, die daaropvolgend de nodige bugfixes en updates

[1] `http://www.openrdf.org/`
[2] `http://www.ikvm.net/`

doorvoerde, werd het SemWeb platform expressief genoeg om de SPARQL-queries, beschreven in sectie 6.3, te ondersteunen.

De SPARQL-queries die we toepassen op de RDF-metadata worden gegenereerd door de webclient (sectie 6.3). De samengestelde mappings van sectie 6.3.2 doen beroep op extension functions. Deze functies (geschreven in C#) worden geregistreerd in de SemWeb-bibliotheek tijdens de initialisatie van de server. De extension functions worden berekend in main memory en resulteren in een booleaanse waarde of getalwaarde waarop men in een FILTER-statement van SPARQL kan testen.

We beschouwen als voorbeeld de extension function beeldkwaliteit (zie sectie 4.2.1). Deze functie verwacht (in juiste volgorde) als input 8 dimensies (Compression, CompressionLevel, Width, Height, ScanningResolution, DigitalZoom, FileSize en Colorspace) en 6 gewichten. De functie weet dat het eerste gewicht overeenkomt met de twee eerste dimensies (compressie en compressieniveau) en dat het 2de gewicht overeenkomt met de breedte en de hoogte. De functie berekent de vergelijkingsfunctie en vertaalt het resultaat naar een integerwaarde tussen 0 en 10.

In sectie 4.2.3 bespraken we een samengestelde mapping die steunt op *hashmaps*. De hashmaps worden gebruikt binnen de extension functions van de samengestelde mapping. Ze moeten dus aanwezig zijn binnen de serveromgeving. Een gebruiker kan via SOAP zijn hashmaps registreren (functie addMap in tabel 6.1).
Bij de evaluatie van een dimensie (binnen een samengestelde mapping) die afhankelijk is van een hashmap zal de webserver in haar lijst van hashmaps zoeken naar de overeenkomstige ID. Wanneer deze tabel niet wordt aangetroffen zal de dimensie niet doorwegen in de vergelijkingsfunctie van de samengestelde mapping.

6.2.2 EXIF

JPEG-bestanden die worden ontvangen vanuit een leverancier worden doorzocht op EXIF metadata. De klasse verantwoordelijk voor het scannen en parsen naar EXIF-metadata werd geschreven door Asim Goheer[3]. Deze klasse werd aangepast zodat de EXIF informatie wordt teruggegeven in RDF-formaat.

6.2.3 XMP

In hoofdstuk 3 bespraken we het Extensible Metadata Platform (XMP) van Adobe Systems. Hierin werd duidelijk dat encapsulatie van gestandaardiseerde metadata belangrijk is voor de overleving van deze metadata. Daarom werd voor deze thesis XMP gebruikt om metadata te encapsuleren in JPEG bestanden. Metadata voor bestanden die geen encapsulatie toelaten wordt geschreven naar een apart bestand, zoals beschreven in de XMP specificatie.

Adobe voorziet een C++ SDK voor het creeëren van XMP packets. De RDF Server is geschreven in C#, dus was het nodig om een DLL die vanuit C# managed code kon worden aangeroepen te schrijven. De XMP DLL voorziet vijf functies waarmee XMP packets kunnen

[3] `http://www.codeproject.com/csharp/exifextractor.asp`

worden aangemaakt. (Listing 6.2)

Init() initialiseert een XMP handle die een leeg packet instantieert. Deze functie geeft een pointer terug naar de XMP handle. *RegisterNameSpace* registreert een namespace in de XMP Handle. Objecten met een onbekende namespace veroorzaken excepties binnen de XMP SDK.
SetXMPProperty voegt een RDF statement toe aan de XMP library. *GetPacket* geeft een string terug die het XMP Packet bevat. *AddPacket* zal tenslotte trachten het XMP packet in een JPEG bestand te voegen.

Appendix B bevat de output van de XMP SDK packet sniffer, toegepast op de output van de webclient.

Listing 6.2: XMP DLL voorziene functies

```
int init();
void addPacket(int cls, string source, string dest);
setXMPProperty(int cls,string ns, string elem,
  string val);
string getPacket(int cls);
void registerNameSpace(int cls, string ns,
  string prefix);
```

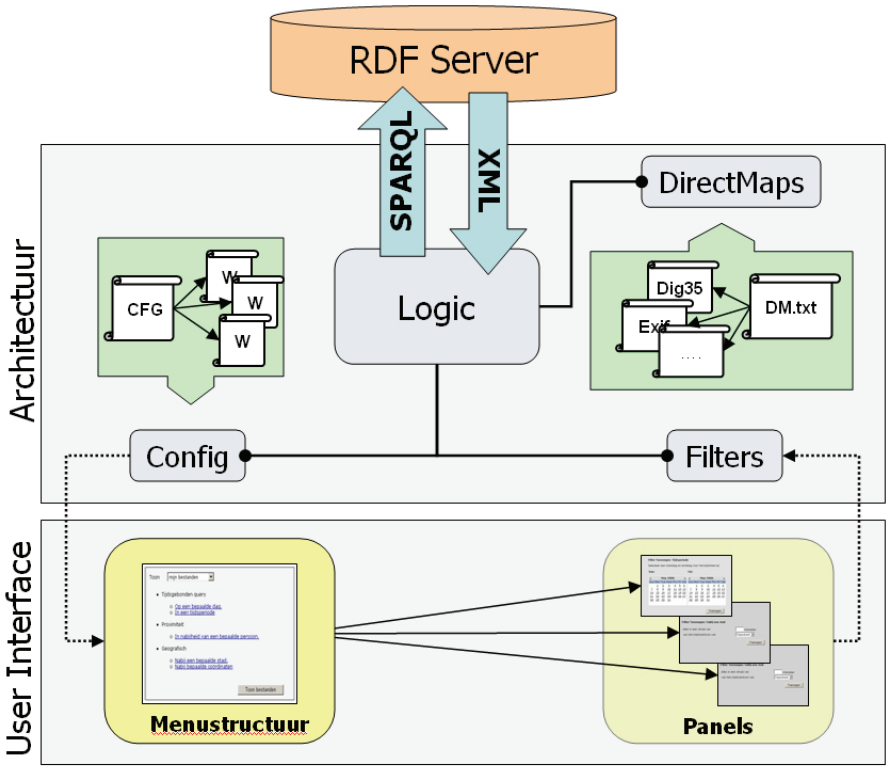
6.3 Webclient voor Metadata Ondervraging

Een gebruiker kan m.b.v. de webclient component zijn verzameling multimedia ondervragen. De webclient component werd geschreven in ASP.NET. In de use case van hoofdstuk 5 werd het gebruik van de webclient gedemonstreerd. In sectie 6.3.1 bespreken we eerst de architectuur van de webclient. Daarna bespreken we in sectie 6.3.2 hoe de webclient deze architectuur gebruikt voor het genereren van SPARQL-queries. Deze queries ondersteunen de directe en samengestelde mappings uit hoofdstuk 4.

Opmerking In deze sectie zullen we bespreken hoe de webclient de graaf in het WHERE-statement van de uiteindelijke SPARQL-query genereert. Om verwarring te voorkomen definiëren we de verschillende typen grafen die aan bod komen in deze sectie. We spreken over een $\underline{graaf}$ wanneer we het hebben over het pad tussen een resource en value in de RDF-structuur van een metadata-standaard (bv. bij Dublin Core kunnen we met één RDF-triple de auteur van een bestand achterhalen). Wanneer we verschillende $\underline{grafen}$ gebruiken om een filter samen te stellen spreken we over een $\overline{graaf}$. Tenslotte, wanneer we spreken over de graaf in de SPARQL-query (die mogelijk uit meerdere filters bestaat) spreken we over de *SPARQL-graaf*.

6.3.1 Architectuur van de webclient

Figuur 6.2 geeft een overzicht van de architectuur van de webclient. De groene diagramma duiden op stappen die worden uitgevoerd bij het opstarten van de component (preprocessing). De *Logic* klasse is het hart van de webclient. Ze stuurt de structuur van de user interface, zet de filters van een gebruiker om in SPARQL, communiceert met de RDF-server via SOAP en zet tenslotte de XML-resultaten van deze queries om in HTML. De Logic klasse maakt gebruik van drie hulpklassen voor deze doeleinden te bekomen:



Figuur 6.2: Webclient architectuur

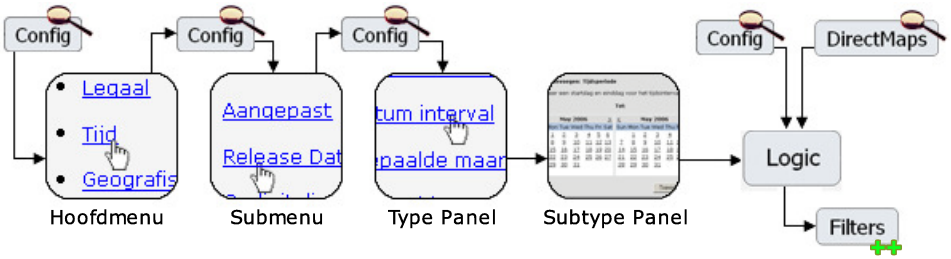
- De Config klasse bevat de verzameling van mogelijke zoekopdrachten waarover de webclient beschikt. Wanneer de applicatie wordt opgestart zal deze klasse een configuratiebestand inlezen waarin elke zoekopdracht structureel wordt voorgesteld als een rij uit de volgende tabel:

1:?	Category	SubCat	Key(s)	Filter	Proze	URL
1	Persoon	Auteur	author	text	Deze..	[-]
1	Persoon	Contributie	contributor	text	Deze..	[-]
1	Persoon	Componist	composer	text	Deze..	[-]
..	..	..	..	..	..	[-]
n	SamenG	Beeldkwaliteit	compr; res;..	bld	Deze..	bld.txt
..	..	..	..	..	..	[-]

De eerste kolom van dit bestand bevat het type van de query (samengesteld of direct). De 2de en 3de kolommen bevatten termen die worden gebruikt in de user interface. De 4de kolom bevat de dimensie die overeenkomt met de beschouwde query. In het geval van een samengestelde mapping zal deze kolom meerdere dimensies bevatten. De 5de kolom bevat welke filter van toepassing is op deze zoekopdracht. De user interface zal op basis van dit veld beslissen welk panel het moet weergeven aan de gebruiker (we komen hier later op terug). De voorlaatste kolom bevat uitleg over de filter, ter verduidelijking naar de gebruiker. De laatste kolom wordt tenslotte enkel gebruikt voor samengestelde

functies. Het bevat de URL van een bestand waarin de gewichten van de samengestelde functie worden gedefiniëerd.

De Config klasse bouwt de user interface dynamisch op a.d.h.v. bovenstaand configuratiebestand. Figuur 6.3 toont welke stappen moeten worden doorlopen voor het selecteren van een filter. De Config klasse bepaalt de inhoud van het hoofdmenu en elk overeenkomstig submenu. Wanneer een gebruiker klikt op een submenu, zal de user interface op basis van de overeenkomstige filter in bovenstaande tabel een gepast panel tonen aan de gebruiker. In dit panel kan de gebruiker kiezen welk mogelijk subtype filter hij wil toepassen.
In het geval van een samengestelde mapping doet de Logic klasse beroep op de Config klasse voor de gewichten van de vergelijkingsfunctie. We komen hierop terug in sectie 6.3.2



Figuur 6.3: Toevoeging van een filter in de webclient

- De DirectMaps klasse wordt eveneens gevuld vanuit een configuratiebestand bij opstart van de applicatie. Het configuratiebestand (DM.txt uit figuur 6.2) bevat een lijst URL's naar schema-bestanden. Elk schema-bestand bevat enerzijds een lijst namespaces (meestal slechts één) en een lijst directe mappings tussen een sleutelwaarde (de dimensie) en een element binnen die standaard. Onderstaande tabel geeft een voorbeeld van het schema-bestand voor Dublin Core:

Type	Key	Waarde
Namespace	dc	http://purl.org/dc/elements/1.1/
Keyvalue	author	?mm dc:creator ?value
Keyvalue	date	?mm dc:date ?value
Keyvalue	format	?mm dc:format ?value
..	..	..

De eerste kolom in bovenstaand voorbeeld bevat het type van de rij (Namespace of Keyvalue). Een *namespace* bevat een prefix en een URL van dat schema. Een *keyvalue* bevat een sleutelwaarde en een *graaf* (uitgedrukt in RDF-triples, gescheiden door een punt). De *graaf* wordt gebruikt om de locatie van de overeenkomstige sleutelwaarde binnen de RDF-structuur van een bepaald type metadata (bv. Dublin Core) te achterhalen. Bijvoorbeeld voor de de sleutel *author* moeten we in een Dublin Core namespace één niveau diep (via dc:creator) zoeken naar een waarde.
De SPARQL variabelen *?mm* en *?value* duiden respectievelijk op de identifier van het multimedia bestand en de waarde binnen de metadata. Elke *graaf* moet deze waarden

bevatten.

Voor waarden die zich dieper bevinden binnen de structuur van een RDF-schema, moet de *graaf* knopen die zich bevinden tussen ?mm en ?value voorstellen door een genummerde variabele $?n_i$. Bijvoorbeeld:

?mm exif:flash $?n_1$. $?n_1$ exif:fired ?value

Wanneer binnen een standaard een bepaald veld door meerdere dimensies kan worden uitgedrukt, gebruiken we het UNION-statement van SPARQL om deze *graaf* samen te stellen. Bijvoorbeeld, wanneer we zoeken op de creatiedatum van een bepaald bestand zullen we voor Dublin Core devolgende *graaf* gebruiken:

{?mm dc:date ?value UNION {?mm dc:created ?value}}

Wanneer we nood hebben aan een *graaf* die meerdere variabelen teruggeeft (bv. bij geografische coördinaten of bij de resolutie van een foto) zullen we twee genummerde ?value parameters gebruiken:

?mm exif:ImageWidth ?value1 . ?mm dc:ImageLength ?value2

Tenslotte, wanneer een dimensie zich bevindt binnen een RDF collectie (rdf:bag, rdf:seq, rdf:alt) zullen we voor elke collectie het type van de collectie, en de variabele ?col toevoegen aan de *graaf*:

?mm dc:contributor ?n1 . ?n1 rdf:type rdf:bag . ?n1 ?col ?value

In sectie 6.3.2 leggen we uit hoe de webclient de DirectMaps klasse gebruikt voor het samenstellen van een filter.

- Wanneer een gebruiker een filter aanmaakt wordt deze toegevoegd aan het singleton object Filters. Deze klasse bevat voor elke filter een ID, een proza beschrijving, het type (direct of samengesteld) en de *graaf* die werd berekend door de Logic klasse. Deze klasse is in feite slechts een container die door de Logic klasse zal worden gebruikt om de SPARQL-graaf te genereren.

Bovenstaande architectuur zorgt voor een scheiding tussen de mogelijke dimensies (en daarbijhorende metadata schema's) enerzijds en de logica van de webclient component anderzijds. Hierdoor is het eenvoudig om nieuwe dimensies of nieuwe metadata schema's (via de Config respectievelijk de DirectMaps klasse) toe te voegen aan de applicatie. Deze werkwijze waarbij de architectuur bij de introductie van nieuwe schema's of dimensies niet moet worden aangepast, is vergelijkbaar met de generieke metadata tools beschreven in [REG99], [Wau98] en [VDO99].

6.3.2 Compositie van SPARQL queries

In deze sectie bespreken we op welke manier de Webclient SPARQL queries genereert m.b.v. de architectuur die we beschreven in sectie 6.3.1.

Voor elke filter die een gebruiker toevoegt zal de uiteindelijke SPARQL-query een $\overline{graaf}$ bevatten binnen haar WHERE-statement. Elke $\overline{graaf}$ binnen dit statement mapt op de unieke variabele *?mm* binnen het SELECT-statement. We illustreren a.d.h.v. een voorbeeld op welke manier een $\overline{graaf}$ wordt samengesteld:

Voorbeeld voor directe mapping

Scenario Een gebruiker kiest in het hoofdmenu een filter voor personen. In het submenu selecteert hij dat hij wil zoeken op auteur. De user interface toont het panel voor de mogelijke stringvergelijkingen. De gebruiker kiest voor een identieke string-match. In het laatste panel vult hij in dat hij alle multimedia van de auteur "Roald Dahl"wil opvragen.

De functie van het panel genereert een filter (zie sectie 6.3.3) voor deze instelling. De filter voor identieke stringvergelijking ziet er uit als volgt: *FILTER(?var="Roald Dahl")*

De Logic klasse krijgt deze filter als input, samen met de sleutelwaarde (author), het type (directe mapping), de kardinaliteit (singulier[4]) en een proza beschrijving ("Auteur met naam Roald Dahl") van de zoekopdracht.

Met de sleutelwaarde zoekt de Logic klasse op in DirectMaps welke standaarden deze sleutel uitdrukken. Dit resulteert in een verzameling $\underline{grafen}$. De Logic klasse gebruikt de $\underline{grafen}$ en de filtergegevens om de uiteindelijke $\overline{graaf}$ te construeren die de intentie van de gebruiker uitdrukt. Hierbij worden de volgende stappen uitgevoerd:

- De variabele van de filter moet overeenkomen met de *?value*-node van elke $\underline{graaf}$. Daarom wordt een nieuwe variabele gemaakt die daarenboven uniek is binnen de gehele SPARQL-graaf (bv. ?a13). De ?var-variabele binnen de filter en de ?value-variabelen binnen elke $\underline{graaf}$ worden hierdoor vervangen[5]:

FILTER(?var="Roald Dahl")	⇒	FILTER(?a13="Roald Dahl")
?mm ape:artist ?value	⇒	?mm ape:artist ?a13
?mm exif:artist ?value	⇒	?mm exif:artist ?a13
...	⇒	...

- Vervolgens moeten we ervoor zorgen dat variabelen in $\underline{grafen}$ die langer zijn dan één RDF-triple niet toevallig mappen op variabelen in de volledige SPARQL-graaf. Daarom vervangen we elke tussenknoop in elke $\underline{graaf}$ door een unieke nieuwe variabele. Bijvoorbeeld:

?mm vb:Creator ?n1
?n1 vb:name ?a13
⇓
?mm vb:Creator ?a53
?a53 vb:name ?a13

[4]Voor sommige dimensies hebben we nood aan filters met twee parameters. Bv.: geografische queries hebben nood aan een binaire filter die inwerkt op latitude en longitude.

[5]Voor binaire filters zullen we volledig analoog de ?value1 en ?value2 variabelen omzetten in twee unieke variabelen.

- Als voorlaatste stap moeten we ervoor zorgen dat zoekopdrachten op RDF-collecties correct worden afgehandeld. We beschouwen hiervoor de collectie (rdf:seq) dc:creator. In de RDF ziet een instantie van deze collectie er uit als volgt:

Subject	Predicate	Object
URI	dc:creator	$node_{879}$
$node_{879}$	rdf:type	rdf:seq
$node_{879}$	rdf:_1	"waarde1"
$node_{879}$	rdf:_2	"waarde2"
$node_{879}$	rdf:_3	"waarde3"
...	...	...

URI is hierbij het beschreven onderwerp en $node_{879}$ is een willekeurige node. Zoals we hadden gedefiniëerd in sectie 6.3.1 ziet de $\underline{graaf}$ van deze collectie op dit moment uit als volgt (?a54 is een willekeurige tussenknoop):

?mm dc:creator ?a54 .
?a54 rdf:type rdf:seq .
?a54 ?col ?a13

Opnieuw vervangen we voor elke $\underline{graaf}$ deze variabele ?col door een unieke variabele binnen de SPARQL-graaf. Elke nieuwe variabele slaan we op in een tijdelijke stack:

?mm dc:creator ?a54 . ?a54 rdf:type rdf:seq . ?a54 ?col ?a13
$\Downarrow$
?mm dc:creator ?a54 . ?a54 rdf:type rdf:seq . ?a54 ?a864 ?a13

Stack.push(?a864)

De stack zullen we gebruiken om te eisen dat elke ?col variabele syntactisch mapt op een predicaat van de vorm rdf:_n. Dit doen we door een extension function *checkCols()* toe te voegen aan het FILTER-statement. De checkCols functie krijgt als parameters de ?col variabelen van de stack:

FILTER(?var="Roald Dahl") .
$\Downarrow$
FILTER(checkCols(?a864,?a934,?...) && (?var="Roald Dahl")) .

- Tenslotte moeten we de filter en de grafen samenstellen in een $\overline{graaf}$ die de zoekopdracht van de gebruiker uitdrukt. Om dit te bekomen voegen we de $\overline{grafen}$ samen d.m.v. elke $\underline{graaf}$ te combineren in een verzameling UNION-statements. We sluiten deze verzameling af met het FILTER-statement.

Listing 6.3: SPARQL voorstelling van directe mapping: auteur.

```
{
  {?mm ape:artist ?a13}
    UNION {
```

```
      {?mm exif:artist ?a13}
      UNION {
        {?mm id3:TPE1 ?a13}
        UNION {
          {
            ?mm dc:creator ?a54 .
            ?a54 rdf:type rdf:bag .
            ?a54 ?a864 ?a13
          }
          UNION { ... }
        }
      }
    }
  FILTER( checkCols(?a864,?...) && (?var=``Roald Dahl"))
}
```

Het UNION statement in listing 6.3 zorgt ervoor dat de filter resultaten zoekt in alle beschouwde metadata schema's. Wanneer één van deze annotaties de waarde "Roald Dahl"bevat, zal (althans voor deze filter) het overeenkomstig bestand (?mm) worden geselecteerd.

Voorbeeld voor samengestelde mapping

Scenario Als voorbeeld voor een samengestelde mapping gebruiken we beeldkwaliteit uit sectie 4.2.1. Een gebruiker kiest voor de beeldkwaliteit een waarde tussen uitstekend (10) of barslecht (1). We kiezen als voorbeeld alle waarden die hoger zijn dan 5.

Analoog als bij de directe mapping genereert de functie van het panel een filter, sleutelwaarde, type, kardinaliteit (n.v.t.) en beschrijving van de zoekopdracht. Het type is ditmaal een samengestelde functie, en de filter wordt ditmaal voorgesteld d.m.v. een string: FILTER($functie$ > 5). De samengestelde functie wordt op basis van deze informatie bepaald als volgt:

- De Logic klasse zoekt op basis van de sleutelwaarde de dimensies en gewichten van de samengestelde functie op. De $functie$ parameter wordt d.m.v. deze informatie aangepast als volgt:

  FILTER(Beeldkwaliteit(?compr, ?comprLvl ?width, ?height, ?scanRes, ?digiZoom, ?fileSize, ?Colorspace, 0.3, 0.1, 0.3, 0.07, 0.05, 0.01) > 5)

  De functie Beeldkwaliteit is een *extension function* van SPARQL. Ze bevindt zich op de server en verwacht precies acht dimensies en zes gewichten (het 1ste gewicht heeft effect op de compressie en het compressieniveau, terwijl het 2de gewicht effect heeft op de resolutie, wat wordt berekent door dimensies ?width en ?height). De evaluatie van de functie gebeurt op de webserver.

- In de webclient is het belangrijk dat we in bovenstaande extension function de dimensies kunnen mappen op de juiste elementen binnen de aanwezige metadata. Hierbij mogen we niet eisen dat elke dimensie aanwezig binnen de metadata. De $\overline{graaf}$ voor beeldkwaliteit wordt berekent als volgt:

Voor elke dimensie bepalen we de directe mapping (zie bovenstaande sectie waar het voorbeeld *auteur* werd uitgewerkt). De directe mapping bevat hier geen FILTER-statement, enkel een UNION van $\underline{grafen}$[6]. We zorgen ervoor dat de ?value variabelen in de $\underline{grafen}$ ditmaal worden vertaald naar een variabele binnen de extension function. We plaatsen het sleutelwoord OPTIONAL voor deze $\overline{graaf}$, en bekomen devolgende optionele directe mapping (bv. voor compressie):

```
OPTIONAL{
   {
      ?mm dig35:compression ?compr
      UNION { ?mm mix:compressionscheme ?compr }
   }
}
```

- We verzamelen de directe mappings en voegen op het einde van de verzameling de filter toe. Zo bekomen we de $\overline{graaf}$ die de samengestelde mapping beeldkwaliteit uitdrukt:

```
{
OPTIONAL{ directe mapping compressie }
OPTIONAL{ directe mapping width }
OPTIONAL{ directe mapping height }
...
FILTER ( Beeldkwaliteit ( ?compr, ?width, ..., 0.05. 0.01) > 5)
}
```

Samenstelling SPARQL-query

Zoals beschreven in sectie 6.3.1 wordt elke $\overline{graaf}$ bewaard in de Filters klasse. Wanneer een gebruiker na toevoeging van extra filters een zoekopdracht lanceert zal de Logic klasse de $\overline{grafen}$ van de Filters klasse samenvoegen tot de uiteindelijke SPARQL-graaf. Dit gebeurt als volgt:

- De DirectMaps klasse voorziet de Logic klasse van de namespaces (en prefices) die worden gebruikt in de query. Deze lijst wordt opgesomd d.m.v. een lijst PREFIX-statements:

```
PREFIX dc:  <http://dublincore.org/documents/dcmi-terms/>
PREFIX exif: <http://www.w3.org/2003/12/exif/ns/>
PREFIX id3:  <http://www.ns.be/id3v2/>
...
```

- Daarna wordt het uniek SELECT-statement toegevoegd:

```
SELECT ?mm
```

[6] Wanneer er in de UNION collecties worden gebruikt bevat de $\underline{graaf}$ wel een FILTER-statement.

- Tenslotte gebruikt de Logic klasse de lijst $\overline{grafen}$ uit de Filters klasse. Het volstaat deze $\overline{grafen}$ op te sommen in een lijst, gescheiden door een punt. In SPARQL betekent dit dat enkel resultaten die mappen op elke $\overline{graaf}$ worden geselecteert in de resultset. De lijst $\overline{grafen}$ wordt omsloten door het WHERE-statement:

```
WHERE {
{ subgraaf1 } .
{ subgraaf2 } .
{ ... }
}
```

Deze query wordt via SOAP verstuurt naar de RDF-server. Wanneer de metadata van een multimediabestand voldoet aan de eisen van elke $\overline{graaf}$ binnen de SPARQL-graaf zal haar identifier worden opgenomen in de XML resultset. Deze XML bevat voor elke variabele in het SELECT-statement (in ons geval slechts ?mm), een waarde-mapping (Listing 6.4).

Listing 6.4: SPARQL XML resultset

```
<sparql xmlns="http://www.w3.org/2005/sparql-results#">
  <head>
    <variable name="mm" />
  </head>
  <results ordered="false" distinct="false">
    <result>
      <binding name="mm">
        <uri>http://didactiek.edm..../identifier#46</uri>
      </binding>
    </result>
    <result>
      <binding name="medium">
        <uri>http://didactiek.edm..../identifier#287</uri>
      </binding>
    </result>
    .....
  </results>
</sparql>
```

De webclient zet deze resultaten om in HTML voor representatie naar de gebruiker.

Opmerking In hoofdstuk 5 werd naast de identifier ook steeds de creatiedatum en het bestandsformaat van het bestand opgevraagd.

6.3.3 SPARQL Filters

Voor het genereren van elk FILTER-statement gebruiken we een grafisch *Panel*. Wanneer de gebruiker een filter wil toevoegen zullen we het overeenkomstig panel zichtbaar maken in de user interface. Voorbeelden van zulke panels worden afgebeeld bovenaan figuren 5.2 en 5.3.

Wanneer een gebruiker klikt op "Toevoegen"gebruiken we de event handler van deze knop om het FILTER-statement samen te stellen. Voor de meeste filters maken we gebruik van de tools die SPARQL aan boord heeft:

- De datatypes van XML Schema: string, double, float, decimal, boolean en dateTime.

- Conversiefuncties tussen deze datatypes.
- Wiskundige en logische operatoren.

We overlopen kort de verschillende mogelijke filters:

- Getalwaarde: Een getalwaarde kunnen we casten naar typen double, float of decimal. We gebruiken de wiskundige operatoren voor vergelijking. Bijvoorbeeld:

  ```
  FILTER( xsd:double(?value) > 13.2)
  ```

- String: Een stringvergelijking is recht toe, recht aan in SPARQL. We kunnen eveneens reguliere expressies gebruiken:

  ```
  FILTER( ?value = "waarde")
  FILTER(regex(str(?value),"^aarde"))
  ```

  De reguliere expressies die worden ondersteund in SPARQL zijn een subset van reguliere expressies van XPath.

- Tijd: voor het vergelijken van tijd, of het bepalen van een tijdsinterval typecasten we de ISO-8601 datum naar het type xsd:dateTime, en gebruiken wiskundige operatoren voor vergelijking. Bv.:

  ```
  FILTER(
    xsd:dateTime("2005-12-31") < xsd:dateTime(?value) &&
    xsd:dateTime(?value) < xsd:dateTime("2006-12-01")
  )
  ```

  Voor het zoeken naar een bepaalde maand (bv. alle resultaten tussen september en november, onafhankelijk van het jaar) kunnen we zulke filter niet gebruiken. In dit geval gebruiken we een extension function die de maand filtert uit de datum, en vervolgens omzet naar een getalwaarde. Bijvoorbeeld:

  ```
  FILTER(
    09 <= parseDate(?value,"mm") &&
    parseDate(?value,"mm") <= 11)
  )
  ```

- GPS: We willen de straal berekenen rondom een bepaalde locatie. Hiervoor bereken we een bounding box rondom deze locatie.[7] Met de hoeken van deze bounding box, kunnen we eenvoudig in SPARQL resultaten generen als volgt:

[7] Merk op dat een rechthoek meer resultaten teruggeeft dan een cirkel-berekening. We hebben echter gekozen voor deze oplossing omdat SemWeb op dat moment nog geen ondersteuning bood voor extension functions. Zonder extension functie zijn we gebonden aan de operatoren van SPARQL. De extension functions werden na uitwerking van dit algoritme toegevoegd aan SemWeb.

```
FILTER (
latOndergrens < ?value1 &&
?value1 < latBovengrens &&
?value2 < lonRechtergrens &&
lonLinkergrens < ?value2
)
```

De berekening van de grenzen, op basis van een straal aantal kilometers wordt besproken in appendix D.

6.3.4 Beperkingen van SPARQL

SPARQL bevat twee restricties die in de applicatie zouden toedragen tot de evaluatie van directe of samengestelde mappings. In de eerste plaats biedt SPARQL geen ondersteuning voor de transitieve sluiting. Wanneer we dus een RDF-graaf a.d.h.v. SPARQL willen ondervragen moeten we de structuur van deze graaf kennen. M.b.v. de transitieve sluiting zouden we een relatie tussen een identifier en een property kunnen definiëren zonder de volledige relatie tussen bron en doel te kennen. Bijvoorbeeld, deze $\underline{graaf}$:

$$\text{?mm exif:flash } ?n_1 \text{ . } ?n_1 \text{ exif:fired ?value}$$

Zouden we m.b.v. de transitieve sluiting kunnen uitdrukken als volgt:

$$\text{?mm transClose(exif:fired) ?value}$$

Dit gebrek beperkt ons echter niet in expressiviteit, we kunnen met de volledige relaties nog steeds het gewenste resultaat bekomen.

Verder biedt SPARQL geen ondersteuning voor aggregatie op collecties. Zoals we in sectie 4.2.2 reeds aanhaalden zou het nuttig zijn het aantal elementen binnen een collectie te gebruiken in een evaluatiefunctie. Bijvoorbeeld:

$$\begin{gathered}\text{?mm dc:creator ?node .} \\ \text{?node rdf:type rdf:seq .} \\ \text{FILTER(COUNT(?node) = 1)}\end{gathered}$$

Bovenstaande functie zouden we kunnen gebruiken om enkel bestanden te zoeken die werden aangemaakt door één enkele auteur. Helaas is dit niet mogelijk in SPARQL.

6.4 Besluit

Er werd een generieke metadata query tool ontwikkeld die zoekopdrachten stelt over een verzameling multimedia met heterogene annotaties. Deze zoekopdrachten werden gebaseerd op de directe en samengestelde mappings van hoofdstuk 4. De applicatie is generiek in de zin dat ze haar user interface dynamisch opbouwt gebaseerd op een aantal configuratiebestanden die de query mogelijkheden en de ondersteunde metadata schema's beschrijven. Een zoekopdracht kan bestaan uit een conjuntie van filters en wordt uitgevoerd m.b.v. de RDF-querytaal SPARQL.

De applicatie bestaat uit een server component die zorgt voor metadata opslag, een web-client om zoekopdrachten uit te voeren en een PDA component als één van de mogelijkheden om metadata in het systeem in te voeren. Verder kan de server component als proof-of-concept van encapsulatie, XMP packets aan JPEG-bestanden toevoegen.

In [VDO99] wordt een generieke metadata query tool beschreven met een gelijkaardige architectuur. Een metadata schema komt hier echter overeen met een relationeel database-model. Als querytaal wordt er dan ook gebruik gemaakt van SQL. Men beschrijft de relaties tussen de verschillende database schema's in een XML configuratiebestand. In deze architectuur worden enkel directe mappings ondersteund.

In sectie 6.3.4 bespraken we beperkingen van SPARQL. Kort samengevat zou het nuttig zijn voor onze implementatie als SPARQL ondersteuning bood voor transitieve sluiting en aggregatie op collecties. Volgens ons zijn deze gebreken belangrijk genoeg om de verspreiding van SPARQL binnen de industrie negatief te beïnvloeden.

Zoals we in sectie 6.3.1 aanhaalden zijn de configuratiebestanden eenvoudig aan te passen en uitbreidbaar. Enerzijds beschrijven ze relaties tussen verschillende multimedia metadata schema's (directe mappings), en anderzijds tussen algemene concepten in multimedia metadata (samengestelde mappings). Deze relaties vormen een ontologie die we als mogelijke uitbreiding zouden kunnen beschrijven in RDF Schema en/of OWL. Zo zouden ook andere applicaties over deze kennis kunnen redeneren. [HL01] demonstreert hoe relaties tussen metadata schema's (directe mappings) voor automatische mappings van verschillende elementen uit deze schema's kunnen zorgen.

Zoals we in sectie 4.3 beschreven ondersteunt deze query tool zoekopdrachten over verschillende metadata standaarden. Daarenboven is het mogelijk om meta-metadata relaties uit te buiten. De zoekopdrachten vereisen dus geen kennis over specifieke metadata schema's en kunnen gebruik maken van voorgedefinieerde of gepersonaliseerde relaties tussen metadata elementen. Het is dus duidelijk dat onze tool tot krachtigere queries zou kunnen leiden (zie hoofdstuk 5). Het is echter voorbarig om hier al conclusies over te trekken zonder eerst een grondige gebruikerstest uitgevoerd te hebben.

Tenslotte zou het interessant zijn om ondersteuning voor samengestelde mappings van een hoger niveau (zie sectie 4.2.5) toe te voegen aan de applicatie. Wegens de scheiding tussen de logica van de query tool en de metadata schema's zou deze uitbreiding niet al te veel problemen met zich mee mogen brengen.

Deel IV

Conclusie

In de thesis onderzochten we verschillende multimedia metadata standaarden. Hierin concludeerden we dat software ontwikkelaars die in hun product willen gebruik maken van metadata op dit moment best kiezen voor XMP wegens diens toegankelijke specificatie, uitbreidbaarheid, hergebruik van bestaande schema's en keuze voor RDF.

Verder werden de relevante standaarden met elkaar vergeleken en identificeerden we overlappende elementen. Overlappende elementen abstraheerden we tot dimensies. Verbanden tussen deze dimensies onderling werden gecombineerd in samengestelde mappings oftewel *meta-metadata*. Deze verbanden geven ons de mogelijkheid over verschillende metadata schema's te werken en flexibel om te gaan met onvolledige annotaties.

Ter illustratie van de nieuwe mogelijkheden op het gebied van metadata-querying werd een use case en bijbehorende applicatie uitgewerkt: een generieke metadata query tool. De querytool bestaat uit een server, webclient en PDA component. Er wordt gebruik gemaakt van SPARQL ter ondersteuning van queries. We stootten echter nog op een aantal problemen met SPARQL die volgens ons de verspreiding van SPARQL kunnen tegenwerken.

Er zijn echter nog een aantal interessante uitbreidingen mogelijk. Zo zouden we onze verbanden kunnen vastleggen in een RDFS/OWL ontologie waardoor er ook andere applicaties over deze kennis zouden kunnen redeneren. Verder zouden we het nut van onze geavanceerde zoekopdrachten kunnen staven d.m.v. een uitgebreide gebruikerstest. Tenslotte zouden we de samengestelde mappings onderling kunnen relateren om op deze manier tot metadata van het 3^{de} niveau te komen. Hiervoor zouden we dan uiteraard ook ondersteuning aan de applicatie toe moeten voegen.

Deel V

Appendices

Bijlage A

Detail Directe Mappings

Deze appendix geeft een detailweergave van de directe mappings die worden besproken in sectie 4.1.

	DC	XMP	IPTC4 XMP	TEI	PRISM	DISC	EXIF	Z39_87	DIG35	IPOD	APE	ID3v2
Bestandsnaam		xmp:Label				Original File Name						TDFN
ID	Identifier	Identifier			Identifier		ImageUniqueID	Image-Identifier	UID	TrackID	ISRC, Catalog	UFID, ISRC
Titel	Title	Title		title	Title				FILE-NAME	Name		
Bestandsgrootte	Format:extent	extent			byteCount, extent	extent		FileSize		Size		
Bestandsversie									VERSION			
Bestandsformaat									FORMAT-TYPE			
MIME	Format, Format:medium	format, medium			Format, medium			MIMEType	MIME-TYPE	Kind		TFLT
ByteOrder								ByteOrder				
Checksum								CheckSum Value				

Detailweergave directe mapping: Bestandsinformatie

	DC	XMP	IPTC4XMP	TEI	PRISM	DISC	EXIF	Z39_87	DIG35	IPOD	APE	ID3v2
Creator	Creator	xmpDM:artist, Creator	Creator	author, principal	CorporateEntity, Creator	Creator	Artist	Image-Producer	IMAGE_CREATOR	Artist	Artist	TPE1
ContactInfo		Creator-ContactInfo	Creator-ContactInfo			Creator Contact Info						
Creator Title		CreatorJobTitle	CreatorJobTitle									
Contributie (alg)	Contributor	Contributor			Contributor							
Componist		xmpDM:composer								Composer	Composer	TCOM
Engineer		xmpDM:engineer										
Sponsor/Funder				sponsor, funder								
Dirigent											Conductor	TPE3
Andere												TPE2, TPE4, TOPE, TEXT, TOLY, TMCL, TIPL, TENC, WOAR
Verspreiding	Publisher	Publisher		publisher, distributor, authority	Distributor, Publisher						Publisher	TPUB

Detailweergave directe mapping: Auteur

	DC	XMP	IPTC4XMP	TEI	PRISM	DISC
Algemeen	Date	Date			Date	
Creatiedatum	Date:created	xmp:CreateDate, xmpDM:shotDate	DateCreated	CreationDate	creationDate	Date Created
Aangepast	date:modified	xmp:ModifyDate, xmpDM:metadataModDate, xmpDM:videoModDate, xmpDM:audioModDate, modified		ChangeDate	modificationDate	
Released		xmpDM:releaseDate				
Gedigitaliseerd		DateTimeDigitized				
PublicatieDatum	date:issued	date:issued			publicationDate, issued	
MetadataDatum		xmp:MetadataDate				
Gebruiksinterval	date:valid				embargoDate, expirationDate	
Andere	date:dateCopyrighted, date:available, date:dateAccepted, date:dateSubmitted	dateCopyrighted, available, dateAccepted, dateSubmitted			coverDate, coverDisplayDate, receptionDate, dateCopyrighted, available, dateAccepted, dateSubmitted	

	EXIF	Z39_87	DIG35	IPOD	APE	ID3v2
Algemeen						
Creatiedatum	DateTime, DateTimeOriginal	DateTime-Created	CAPTURE_TIME, CREATION_TIME	Date Added	Record Date	TDRC
Aangepast				Date Modified		
Released					Year	TDRL, TDOR
Gedigitaliseerd	DateTimeDigitized					TDEN
PublicatieDatum						
MetadataDatum						TDTG
Gebruiksinterval						
Andere						

Detailweergave directe mapping:: datum

	QDC	XMP	IPTC4XMP	TEI	PRISM	DISC	EXIF	DIG35	APE	ID3v2
Copyright	Rights	xmpDM:copyright, Rights	Copyright-Notice	Availability	Copyright	Copyright Notice	Copyright	COPYRIGHT	Copyright	TCOP, TPRO
Rechthouder	Rightsholder	xmpRights:Owner, Rightsholder, Source	Source		RightsAgent			IPR_PERSON, IPR_ORG, IPR_NAMEREF, CONTACT_POINT	Publication-right	
Voorwaarden	Rights:AccessRights	xmpRights:Usage-Terms, AccessRights	Rights-UsageTerms		prl:usage,prl:geography, prl:industry			USE-RESTRICTION, OBLIGATION, PROTECTION		USER
Link	Rights:License	xmpRights:Certificate, xmpRights:WebStatement, License			License			IDENTIFICATION		WCOP, WPUB
Andere								IPR_MGMT_SYS		

Detailweergave directe mapping: Rights

	DC	XMP	IPTC4XMP	PRISM	DISC	EXIF	DIG35	APE
Locatie	Coverage: Spatial	Spatial	Location	Location	Location		LOCATION	Record Location
Country/Prov/City		Country; Province, State; City	Country; Province; State City		Country, Province, State, City		COUNTRY, CITY, STATE	
CountryCode		CountryCode	CountryCode					
ZipCode							ZIPCODE	
Scene		xmpDM:scene, xmpDM: shotLocation	Scene				PERSON, THING, ORGANI-SATION, EVENT	
GPS		(overerving van EXIF)				GPSVersionID, GPSTimeStamp, GPSSatellites, GPSStatus, GPSMeasureMode, GPSDOP, GPSSpeedRef, GPSSpeed, GPSTrackRef, GPSTrack, GPSImgDirectionRef, GPSImgDirection, GPSMapDatum, GPSDestLatitudeRef, GPSDestLatitude, GPSDestLongitudeRef, GPSDestLongitude, GPSDestBearingRef, GPSDestBearing, GPSDestDistanceRef, GPSDestDistance, GPSProcessingMethod, GPSAreaInformation, GPSDateStamp, GPSDifferential	LATITUDE, LONGITUDE , ALTITUDE	

Detailweergave directe mapping: Locatie

	EXIF	IISO Z39_87	DIG35
Resolutie	ImageWidth, ImageLength, XResolution, Yresolution,	ImageWidth, ImageLength	Width, Height
Kleur Pixels	ColorSpace, PhotomtricInterpretation, PlanarConfiguration, YCbCrSubSampling, YCbCrPositioning, WhitePoint, PrimaryChromaticities, YCbCrCoefficients, ReferenceBlackWhite, BitsSample, SamplesPixel, StripOffsets, RowsPerStrip, StripByteCounts, TransferFunction, PixelXDimension, PixelYDimension, ComponentsConfiguration	Colorspace, ICCProfileName, ICCProfileURL, YCbCrSubSampling, YCbCrPositioning, YCbCrCoefficients, ReferenceBlackWhite, Colormap_Reference, Colormap_BitCodeValue, Colormap_RedValue, Colormap_GreenValue, Colormap_BlueValue, GrayResponseCurve, GrayResponseUnit, WhitePoint_Xvalue, WhitePoint_Yvalue, BitsPerSample, SamplesPerPixel, SegmentType, StripOffsets, RowsPerStrip, StripByteCounts, TileWidth, TileLength, TileOffsets, TileByteCounts, PlanarConfiguration, Orientation, DisplayOrientation, XTargetedDisplayAR, YTargetedDisplayAR, Methodology, SamplingFrequencyPlane, SamplingFrequencyUnit, XSamplingFrequency, YSamplingFrequency, Source_Xdimension, Source_XdimensionUnit, Source_Ydimension, Source_YdimensionUnit, ExtraSamples	Profile_Name, Premultiplied, Component, Num_Components
Compressie			
Camera	Model, Make, Software	CompressionScheme, CompressionLevel	Compression, Comp_Size
Instellingen	SpectralSensitivity, ISOSpeedRatings, OECF, ShutterSpeedValue, ApertureValue, BrightnessValue, ExposureBiasValue, MaxApertureValue, MeteringMode, LightSource, SubjectArea, SpatialFrequencyResponse, SubjectLocation, ExposureIndex, FileSource, SceneType, CFAPattern, CustomRendered, ExposureMode, WhiteBalance, SceneCaptureType, GainControl, Contrast, Saturation, Sharpness, DeviceSettingDescription, SubjectDistanceRange, Fnumber, DigitalZoomRatio, FlashEnergy, ExposureProgram, ExposureTime, SubjectDistance, FocalPlaneXResolution, FocalPlaneYResolution,Orientation	DigitalCameraModel, DigitalCameraManufacturer Brightness, ExposureBias, MeteringMode, SceneIlluminant, ColorTemp, Flash, FlashReturn, BackLight, ExposureIndex, AutoFocus, XPrintAspectRatio, YprintAspectRatio, Sensor, Fnumber, FlashEnergy, ExposureTime, SubjectDistance, FocalLength	Model, Org_Name, Software_Info, Image_Source Lens_Info, Sensor_Technology, Focal_Plane_Resolution, Spectral_Sensitivity, ISO_Saturation, ISO_Noise, Spatial_Freq_Response, CFA_Pattern, OECF, Brightness_Value, Exposure_Bisa, Metering_Mode, Scene_Illuminant, Color_Temp, Flash, Flash_Return, Back_Light, Subject_Position, Exposure_Index, Auto_Focus, Special_Effects, Camera_Location, Orientation, Accessory, F_Number, Flash_Energy, Exposure_Time, Exp_Program, Subject_Distance, Subject_Distance, Focal_Length

detailweergave directe mapping: Foto

Scanner		DeviceSource, HostComputer, OS, OSVersion, ScannerManufacturer, ScannerModelName, ScannerModelNumber, ScannerModelNumber, ScanningSoftware, ScanningSoftwareVersionNo	Scanner_Info, Software_Info
Instellingen		PixelSize, XphysScanResolution, YphysScanResolution	Pixel_Size, Scan_Res
Audio	RelatedSoundFile		Audio
PrefRes		Preferred Presentation	Pref_Presentation_Param
History		SourceData, ProcessingAgency, ProcessingSoftwareName, ProcessingSoftwareVersion, ProcessingActions, Previous image metadata	Img_Summary, History, Img_Processing_Hings, Captured_Item, Operator_Org, Opertator_ID

detailweergave directe mapping: Foto (2)

	ID3v2	APE	IPOD	XMP
Tempo	TBPM, SYLT		BitRate, SampleRate	xmpDM:numberOfBeats, xmpDM:tempo
Lengte	TLEN		Total Time	xmpDM:duration
Toonaard	TKEY			xmpDM:key
Taal	TLAN	Language		language
Genre	TCON, TMOO	Genre	Genre	xmpDM:genre
HerkomstMedium	TSSE, TALB, TOAL, TPOS, TRCK, TSST, TMED, WOAS, WOAF, WORS, MCDI, TRSN, TRSO	Album, Debut Album, Track, EAN/UPC, ISBN, Media	Album, Disc Number, Disc Count, Track Number, Track Count	xmpDM:album, xmpDM:trackNumber, xmpDM:instrument
Audiosignaal				xmpDM:audioSampleRate, xmpDM:audioSampleType, xmpDM:audioChannelType
Playlist Informatie	POPM, TDLY, TSOA, TSOP, TSOT, PCNT		Rating, PlayCount, Play Date, Play Date UTC	xmpDM:introTime, xmpDM:loop, xmpDM:outCue
Events	ETCO			xmpDM:relativeTimestamp
Playback	RVA2, SYTC, EQU2, RVRB			xmpDM:stretchMode, xmpDM:timeScaleParams, xmpDM:resampleParams, xmpDM:beatSpliceParams, xmpDM:timeSignature
Andere	AENC, WCOM, WPAY, COMR, APIC, GEOB, USLT	Introplay		xmpDM:speakerPlacement

Detailweergave directe mapping: Audio

Bijlage B

XMP output van Webclient

In deze appendix wordt de output van de XMP SDK Packet sniffer, toegepast op een foto die werd gegenereerd door de Webclient (sectie 6.3) weergegeven.

```
// Dumping raw input for "C:\ambiorix.jpg" (193..3258)
<?xpacket begin="n++" id="W5M0MpCehiHzreSzNTczkc9d"?>
<x:xmpmeta xmlns:x="adobe:ns:meta/" x:xmptk="Public XMP Toolkit 3.2">
   <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
      <rdf:Description rdf:about=""
            xmlns:dc="http://purl.org/dc/elements/1.1/">
         <dc:format>image/jpeg</dc:format>
         <dc:title>Standbeeld</dc:title>
         <dc:description>Ambiorix</dc:description>
         <dc:date>2006-05-04T15:13:12Z</dc:date>
         <dc:creator>http://didactiek.edm.uhasselt.be/thesis/luk.vloemans/
             people/Luk</dc:creator>
      </rdf:Description>
      <rdf:Description rdf:about=""
            xmlns:luk="http://didactiek.edm.uhasselt.be/
            thesis/luk.vloemans/">
         <luk:proximity>Marc</luk:proximity>
      </rdf:Description>
      <rdf:Description rdf:about=""
            xmlns:exif="http://ns.adobe.com/exif/1.0/"
         <exif:GPSLatitude>50.9467389</exif:GPSLatitude>
         <exif:GPSLongitude>5.4375649</exif:GPSLongitude>
      </rdf:Description>
   </rdf:RDF>
</x:xmpmeta>
<?xpacket end="w"?>
```

Bijlage C

Samengestelde Mappings

In deze bijlage bespreken we de uitgewerkte samengestelde mappings uit hoofdstuk 4.

C.1 Portretfoto

In sectie 4.2.2 bespraken we de samengestelde mapping die een portretfoto uitdrukt. In deze sectie tonen we een uitgebreid voorbeeld van deze samengestelde mapping. Tabel C.1 geeft aan welke dimensies van deze samengestelde mapping kunnen worden uitgedrukt in DIG35, NISO Z39.87 en EXIF. DISC werd in dit voorbeeld niet opgenomen omdat het geen enkele dimensie van deze samengestelde functie bevat. Verder werd ook XMP niet opgenomen omdat XMP de elementset van EXIF overerft. Tabel C.1 toont de configuratie van 11 digitale foto's.

	NISO Z39.87	DIG35	EXIF
Camera mode	√	√	√
Resolutie	√	√	√
Pitch	¬	√	¬
Roll	¬	√	¬
Collections	¬	√	¬
Distance	√	√	√

Tabel C.1: Dimensies voor portretfoto in beschouwde standaarden.

De laatste kolom van tabel C.1 bevat het resultaat van het gewogen gewicht. Een lege cel in de tabel betekent dat de overeenkomstige dimensie niet kan worden uitgedrukt in die metadata standaard. Het symbool [−] duidt op een ontbrekende waarde. Om het gewogen gewicht te bekomen kozen we voor de evaluatiefuncties de volgende gewichten: Camera Mode: 0.7; Resolutie: 0.7; Pitch: 1.0; Afstand: 0.5; Roll: 0.15; Collections: 1; Exposure: 1.

De Collections dimensie stellen we abstract voor. Wanneer een bepaald item wordt beschreven in de collections noteren we enkel het type van de collectie. Meer uitleg over deze dimensie is beschreven in sectie 4.2.2. We geven een korte uitleg bij elke foto:

Foto 1 is een perfect geannoteerde, die op elke schaal maximaal scoort. Foto 2 is een bijgesneden portretfoto. Foto 3 heeft perfecte afmetingen, maar werd getrokken op een afstand van 3 meter. Foto 4 heeft vierkante afmetingen. Foto 5 heeft bevat geen metadata over de resolutie, maar werd op een aanvaardbare afstand, pitch en roll getrokken. Foto 6 krijgt een lagere score omdat de breedte reeds iets groter is dan de hoogte van de foto. Dit is ongewenst in het geval van een portretfoto. Foto 7 scoort slecht omdat hij werd getrokken met een

hoge exposure. Foto 8 heeft perfecte afmetingen, maar getrokken met de lens naar de grond. Foto 9 heeft eveneens perfecte afmetingen, maar bevat een gebeurtenis en werd getrokken op een afstand van 6 meter. Foto 10 heeft een volledig liggend formaat (foto's 6 en 4 werden minder gediscrimineerd o.w.v. een betere verhouding tussen breedte en hoogte). Foto 11 werd horizontaal getrokken met landschapsinstellingen.

		Mode	Pitch	Dist	Roll	Collec	Exp	W	H	
1	DIG35	portrait	0	1	[-]	person	[-]	480	640	10
2	EXIF	portrait		3			0.01	800	700	9
3	DIG35	normal	[-]	6	[-]	person	[-]	480	640	8
4	NISO	normal		[-]			[-]	600	600	8
5	DIG35	[-]	30	2	70	[-]	[-]			8
6	NISO	normal		[-]			[-]	800	700	6
7	EXIF	[-]		[-]			5	600	800	4
8	DIG35	normal	-75	[-]	[-]	[-]	[-]	480	640	4
9	DIG35	normal	[-]	6	[-]	event	[-]	480	640	3
10	NISO	[-]		[-]			[-]	800	600	0
11	EXIF	landscape		70			[-]	1400	1050	0

C.2 Muziek Voorkeur

In sectie 4.2.3 bespraken we de samengestelde mapping voor *muziek voorkeur*. In deze sectie passen we deze samengestelde functie toe op een aantal geannoteerde muziekbestanden. Tabel C.2 geeft aan welke dimensies van deze samengestelde mapping kunnen worden uitgedrukt in ID3v2, IPOD, XMP en APEv2. De dimensie creatiedatum werd toegevoegd omdat de evaluatiefunctie TimesPlayed in deze mapping hiervan gebruik maakt. We gebruiken de hasmaps

	ID3	APEv2	IPOD	XMP
Genre	√	√	√	√
Language	√	√	¬	√
Mood	√	¬	¬	¬
Release Date	√	√	√	√
TimesPlayed	√	¬	√	¬
CreationDate	√	√	√	√
Rating	√	¬	√	¬

Tabel C.2: Dimensies voor muziek voorkeur in de beschouwde standaarden.

die werden geïntroduceerd in sectie 4.2.3. Deze hashmappen beschrijven een gebruiker die o.a. graag luistert naar opgewekte (Happy) Nederlandstalige of Franstalige rockmuziek uit de zomer van 1969. De gebruiker heeft een hekel aan Engelstalige, romantische (Romantic) of droevige (Sad) muziek, en muziek die werd gemaakt na het jaar 2000. De volledige beschrijving van de gebruikte hashmappen van dit voorbeeld staan beschreven in sectie 4.2.3. Voor de evaluatiefunctie TimesPlayed veronderstellen we dat het vandaag de 19de mei 2006 is en nemen voor parameter d waarde tien (dagen). Voor de gewichten van deze samengestelde mapping kozen we: Genre 1.0; Taal: 1.0; Mood: 1.0; Release Date: 1.0; Playcount: 1.0; Rating: 5.0. Tabel C.2 toont negen uitgewerkte voorbeelden van geannoteerde muziekbestanden die we kort overlopen:
Fragment 1 is volledig afgestemd op de voorkeur van de gebruiker. Fragment 2 scoort iets

minder omdat het een nummer uit de jaren '90 is. Fragment 3 is enkel geannoteerd met het type genre, pop-muziek wordt in de hashfunctie van genre positief gevalueerd. Fragment 4 scoort redelijk goed, maar wordt bevat een droevige mood, en werd geproduceerd na het jaar 2000.
Fragment 5 is pas toegevoegd aan de verzameling (in minder dan 10 dagen). Om die wordt haar TimesPlayed dimensie genegeerd. Verder bevat fragment 5 een onbekende mood (Funny). Mood heeft geen *else* waarde in haar hashmap, dus fragment 5 wordt enkel bepaald door haar genre (mapt op *else*) en jaar van uitgave.
Fragment 6 stelt een conflictsituatie voor. Enerzijds bevat het een voorkeurstaal en werd het geproduceerd voor 2000. Anderzijds heeft de gebruiker een hekel aan droevige soulmuziek.
Fragment 7 wordt naar onvoldoende geëvalueerd o.w.v. haar lage rating. Fragment 8 scoort onvoldoende omdat het nog maar 12 maal werd afgespeeld. Tenslotte scoort fragment 9 barslecht omdat we enkel weten dat het Engelstalig is en werd opgenomen na 2000.

		Genre	Lang	Mood	RelDT	#	CreaDT	Rate	
1	ID3v2	rock	NL	Happy	1969-07-13	122	2002-04-29	100	10
2	APEv2	rock	[-]		1994-08-20		[-]		9
3	XMP	pop	[-]		[-]		[-]		8
4	ID3v2	[-]	FR	Sad	2004-08-14	132	2004-04-30	90	8
5	ID3v2	humor	[-]	Funny	1976-08-16	7	2006-05-10	[-]	7
6	ID3v2	soul	NL	Sad	1999-08-15		2004-05-01	[-]	5
7	IPOD	blues			1969-07-15	121	1992-12-03	1	4
8	IPOD	techno			[-]	12	1992-12-03	[-]	4
9	APEv2	[-]	EN		2004-08-19				0

Bijlage D

Bounding Box voor geopunt

In deze appendix geven we de bereking voor een bounding box, op input van latitude, longitude en aantal kilometers. De resultaten van deze berekening werden gebruikt in sectie 6.3.3. We berekenen de linkerbovenhoek en rechteronderhoek van een rechthoek rondom het geopunt. Als we deze hoeken kennen kunnen we eenvoudig in SPARQL een filter genereren (met basis wiskunde $<$ en $>$ operaties). Voor het berekenen van de geopoint-rechthoek maken we gebruik van de formule van afstand tussen twee punten op een bol:

$$D = R * BgCos[sin(y_i)sin(y_j) + cos(y_i)cos(y_j)cos(x_j - x_i)]$$

Hierbij staat R voor de straal van de cirkel, (x_i,y_i) en (x_j,y_j) voor de hoeken van twee punten op deze bol en D de afstand tussen deze twee punten. Wanneer we deze formule toepassen op de aarde (met straal 6374 kilometer) krijgen we[1]:

$$D = 6374 * BgCos[sin(lat_i)sin(lat_j) + cos(lat_i)cos(lat_j)cos(lon_j - lon_i)] \tag{D.1}$$

Voor de berekening van een rechthoek rond een geopunt kennen we D, lat_i en lon_i. De verticale longitude linker en-rechtergrens worden bekomen door lat_j gelijk te stellen aan lat_i in vergelijking (3.1). We bekomen dan devolgende vergelijking:

$$cos(\frac{D}{6374}) = sin^2(lat_i) + cos^2(lat_i)cos(lon_j - lon_i)$$

Dit stelsel moet worden uitgeschreven naar lon_j, mits de andere variabelen allen een waarde hebben:

$$\frac{cos(\frac{D}{6374}) - sin^2(lat_i)}{cos^2(lat_i)} = cos(lon_j - lon_i)$$

De boogcosinus introduceert twee oplossingen voor longitude linkergrens en rechtergrens:

$$lon_j = \pm acos(\frac{cos(\frac{D}{6374}) - sin^2(lat_i)}{cos^2(lat_i)}) + lon_i \tag{D.2}$$

De horizontale latitude onder en-bovengrens worden bekomen door lon_i gelijk te stellen aan lon_j in vergelijking D.1. Hierdoor bekomen we volgende vergelijking:

$$D = 6374 * BgCos[sin(lat_i)sin(lat_j) + cos(lat_i)cos(lat_j)cos(0)]$$

[1]Merk op: we veronderstellen dat de aarde een perfecte bol is.

We ontleden de vergelijking naar sin(lat_j):

$$cos(\frac{D}{6374}) - sin(lat_i)sin(lat_j) = cos(lat_i)\sqrt{1 - sin^2(lat_j)}$$

We vereenvoudigen deze verlijking door de substitutie van constante waarden (M = cos(D/6374) , S = sin(lat_i) en C =cos(lat_i))) als volgt:

$$M - S * sin(lat_j) = C * \sqrt{1 - sin^2(lat_j)}$$

Dit kunnen we schrijven als vierkantsvergelijking naar lat_j:

$$(S^2 + C^2)sin(lat_j) - 2 * M * S * sin(lat_j) + (M^2 - C^2) = 0$$

De discriminant van deze vergelijking is steeds groter dan nul, tenzij de orginele afstand D = 0 (in dit geval kan SPARQL eenvoudig met de vergelijkingsoperator latitudes en longitudes van geopunten vergelijken en hebben we geen nood aan een omliggende rechthoek). Om deze reden vinden we dus steeds twee oplossingen voor sin(lat_j), nl:

$$Discriminant = 4 * [M^2S^2 - (S^2 + C^2)(M^2 - C^2)]$$

$$sin(lat_j) = \frac{MS \pm \sqrt{Discriminant}}{(M^2 + S^2)} \tag{D.3}$$

De boogsinus van bovenstaande vergelijking levert de boven en-ondergrens voor lat_j. Merk op dat de boogsinus opnieuw twee oplossingen introduceert. De nieuw geïntroduceerde oplossingen komen mogelijk voor in het Westelijk wereldhalfrond. Afhankelijk van de bronpositie is een afweging noodzakelijk welke oplossing van de boogsinus relevant is. Voor coördinaten in centraal Europa volstaat de eerste oplossing (kortste hoek) zonder conversie. Met vergelijkingen D.2 en D.3 kunnen we dus de vier hoeken berekenen die we nodig hebben in de webclient.

Bibliografie

[Ado05] Adobe Systems Inc. *XMP Specification*, 1.0 edition, june 2005. `http:/www.adobe.com`. 27

[BH02] Jan Bormans and Keith Hill. *MPEG-21 Overview*, 5 edition, october 2002. `http://www.chiariglione.org/mpeg/standards/mpeg-21/mpeg-21.htm`. 24

[Bor04] Niel Bornstein. *Hacking the iTunes Music Library*, november 2004. `http://www.idealliance.org/proceedings/xml04/papers/80/paper.html`. 48

[CDES05] Eugene Inseok Chong, Souripriya Das, George Eadon, and Jagannathan Srinivasan. An Efficient SQL-based RDF Querying Scheme. In *31st VLDB Conference, Trondheim, Norway, 2005*, 2005. 10

[Dub05] Dublin Core Metadata Initiative. *DCMI Metadata Terms*, june 2005. `http://dublincore.org/documents/2005/06/13/dcmi-terms/`. 23

[Fed05] Digital Library Federation. Metadata Encoding and Transmission Standard Schema v1.5. `http://www.loc.gov/standards/mets/`, april 2005. 36

[GvOH05] Joost Geurts, Jacco van Ossenbruggen, and Lynda Hardman. Requirements for practical multimedia annotation. In *2nd European Semantic Web Conference*, June 2005. 49

[HL01] Jane Hunter and Carl Lagoze. Combining rdf and xml schemas to enhance interoperability between metadata application profiles. In *WWW '01: Proceedings of the 10th international conference on World Wide Web*, pages 457–466, New York, NY, USA, 2001. ACM Press. 76, 92

[ID303] *ID3v2 Informal standard*, 2.4.0 edition, november 2003. `http://www.id3.org`. 45, 66

[IDE05a] IDEAlliance. DISC Metadata for digital submission - final draft for public comment, july 2005. 44

[IDE05b] IDEAlliance. *Guide to PRISM Aggregator Document Type Definition*, 1.3 edition, january 2005. `http://www.prismstandard.org/`. 39

[IDE05c] IDEAlliance. *PRISM Specification*, 1.3 edition, october 2005. `http://www.prismstandard.org/`. 37

[Int01] International Imaging Industry Association (I3A). *DIG35 specification*, 1.1 edition, june 2001. `http://www.i3a.org/i_dig35.html`. 3, 41, 43

[IPT99] IPTC. *Information Interchange Model specification*, 4.1 edition, july 1999. http://www.iptc.org/std/IIM/4.1/specification/IIMV4.1.pdf. 30

[IPT05] IPTC. *IPTC Core Schema for XMP specification*, 1.0 edition, march 2005. http://www.iptc.org/IPTC4XMP/. 30

[ISO02] ISO-639, Codes for the representation of names of languages, 2002. 66

[ISO04] ISO-8601, Representation of dates and times, 2004. 54, 66, 77

[Jap02] Japan Electronics and Information Technology Association. *Exchangeable Image File Format Specification*, 2.2 edition, april 2002. http://www.exif.org. 40

[Kle] Frank Klemm. APE Tag version 2.0. http://www.personal.uni-jena.de/~pfk/mpp/sv8/apetag.html. 46

[Mar04] José M. Martnez. *MPEG-7 Overview*. ISO/IEC, 1.0 edition, october 2004. http://www.chiariglione.org/MPEG/standards/mpeg-7/mpeg-7.htm. 3, 16, 19, 21

[Mel05] Jim Melton. SQL, XQuery, and SPARQL, What's Wrong With This Picture? http://www.idealliance.org/proceedings/xml05/ship/185/XML2005_185.HTML, 2005. Sandy, UT, USA. 13

[Mic06] Microsoft. Windows Media Metafiles Specification. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/wmplay10/mmp_sdk/asx_elementsintro.asp, 2006. 30

[mpe05] MPEG21 Part 2: Digital Item Declaration, October 2005. 24

[NIS02] NISO/AIIM. *NISO Z39.87 specification*, 1.0 edition, june 2002. http://www.loc.gov/standards/mix/. 39

[RDF] Rdf, Resource Description Framework. Web page at http://www.w3.org/RDF/. 9

[REG99] Reggie: The Metadata Editor http://metadata.net/dstc/, 1999. 84

[Sik01] Thomas Sikora. The MPEG-7 Visual Standard for Content Description An Overview. In *IEEE Transactions on circuits and systems for video technology*, pages p696 – p702, 2001. 3, 20, 22

[SMB02] C. M. Sperberg-McQueen and L. (eds.) Burnard. TEI P4: Guidelines for electronic text encoding and interchange., 2002. 33

[SPA06] SPARQL, SPARQL query language for RDF. Web page at http://www.w3.org/TR/rdf-sparql-query/, April 2006. 11

[VDO99] Bart Verhoeven, Erik Duval, and Henk J. Olivie. A generic metadata query tool. In *WebNet (2)*, pages 1122–1127, 1999. 84, 92

[Wau98] Andrew Waugh. Specifying metadata standards for metadata tool configuration. In *WWW7: Proceedings of the seventh international conference on World Wide Web 7*, pages 23–32, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V. 84

[XMI98] XML Metadata Interchange (XMI), october 1998. 69

[XML] XML Schema. Web page at `http://www.w3.org/XML/Schema`. 8

Auteursrechterlijke overeenkomst

Opdat de Universiteit Hasselt uw eindverhandeling wereldwijd kan reproduceren, vertalen en distribueren is uw akkoord voor deze overeenkomst noodzakelijk. Gelieve de tijd te nemen om deze overeenkomst door te nemen en uw akkoord te verlenen.

Ik/wij verlenen het wereldwijde auteursrecht voor de ingediende eindverhandeling:
Multimedia Metadata Description Standards and Applications
Richting: **Licentiaat in de informatica** Jaar: **2006**
in alle mogelijke mediaformaten, - bestaande en in de toekomst te ontwikkelen - , aan de Universiteit Hasselt.

Deze toekenning van het auteursrecht aan de Universiteit Hasselt houdt in dat ik/wij als auteur de eindverhandeling, - in zijn geheel of gedeeltelijk -, vrij kan reproduceren, (her)publiceren of distribueren zonder de toelating te moeten verkrijgen van de Universiteit Hasselt.

U bevestigt dat de eindverhandeling uw origineel werk is, en dat u het recht heeft om de rechten te verlenen die in deze overeenkomst worden beschreven. U verklaart tevens dat de eindverhandeling, naar uw weten, het auteursrecht van anderen niet overtreedt.

U verklaart tevens dat u voor het materiaal in de eindverhandeling dat beschermd wordt door het auteursrecht, de nodige toelatingen hebt verkregen zodat u deze ook aan de Universiteit Hasselt kan overdragen en dat dit duidelijk in de tekst en inhoud van de eindverhandeling werd genotificeerd.

Universiteit Hasselt zal u als auteur(s) van de eindverhandeling identificeren en zal geen wijzigingen aanbrengen aan de eindverhandeling, uitgezonderd deze toegelaten door deze licentie

Ik ga akkoord,

Luk VLOEMANS

Datum: