Gebruikscenario's blijven van grote waarde

Metadata-repository: bouwen of kopen?

Malcolm Chisholm

Door de jaren heen stellen klanten mij steeds weer de vraag of ze een metadata-repository moeten kopen of er zelf een bouwen. Dat is een lastige vraag, omdat de vragenstellers vaak de term 'metadata' niet goed begrijpen.

Druk van vendors, presentaties op congressen en artikelen in de vakbladen overtuigen organisaties ervan dat ze een metadata-repository nodig hebben. Zo ontstaat de noodzaak om de afweging te maken tussen kopen en bouwen. Helaas vergeet men hierbij meestal de vraag te beantwoorden waarom er *überhaupt* een metadata-repository nodig is. Dat is jammer, omdat veel organisaties een prangende behoefte hebben aan veel meer repository-functionaliteit dan waar ze nu over beschikken. Het probleem schuilt hierin, dat de manier waarop mensen denken over metadata ze ervan weerhoudt de behoefte te begrijpen, laat staan een weloverwogen besluit te nemen over kopen of bouwen.

Definitie metadata en repository

Het begint allemaal met een definitie van metadata. Het gebruikelijke antwoord luidt dan: "metadata zijn data over data". Daar schieten we niet echt mee op. Het is een erg onvolledige definitie die het onmogelijk maakt de problemen rond informatiemanagement op te lossen. Ik suggereer een andere definitie:

Metadata is the data that describes all aspects of an enterprise's information assets, and enables the organization to effectively use and manage these assets.

Belangrijk aspect is dat metadata niet noodzakelijk iets van doen hoeven te hebben met computers of databases. Zoals de beroemde opmerking van Dr. Samuel Johnson, die in de 18e eeuw het eerste Engelse woordenboek schreef : "Knowledge is of two kinds.
We know a subject ourselves, or we know where we can find information upon it". Dus zelfs in de tijden dat papier en inkt de enige voorhanden media waren om informatie op te slaan, was het toch nodig te beschikken over metadata, zoals definities van termen en de toegangswegen naar opgeslagen informatie.
Dit staat in contrast met mijn ervaringen in discussies over metadata. Het blijkt dat veel mensen metadata zien als een verzameling gedefinieerde informatie die eindig, afgebakend en conceptueel tamelijk klein is. Ze zien metadata als een 'black box': ze weten niet precies wat er in zit, maar ze nemen aan dat daar lang geleden allemaal goed over is nagedacht door de IT-professionals. Ga je uit van één goed gedefinieerde en afgebakende verzameling metadata, dan is het logisch om te denken dat je maar één repository hoeft te implementeren om het te kunnen managen. Daaruit volgt dan weer de vraag of die repository moet worden gekocht of gebouwd.
En wat is een repository? In de praktijk is dat een gewoon een database met bijbehorende functionaliteit, gebruikt om metadata op te slaan en te managen. Een database slaat data op; een repository slaat metadata op.

Slechts één repository?

Als je over metadata denkt als een eindige verzameling informatie met goed gedefinieerde grenzen, is de verleiding dus groot om te denken dat je maar één repository per onderneming hoeft te hebben. Als dat zo zou zijn, dan zou kopen zeker de voorkeur hebben boven het zelf bouwen van de repository. Dan zouden we op de markt een aantal vendors zien die allemaal min of meer dezelfde repository-functionaliteit zouden aanbieden, met maar minieme verschillen. Dergelijke producten zouden dan alle metadatabehoeften lenigen. Zo ziet het er op de markt echter helemaal niet uit. Er zijn enkele algemene repository's, maar naar mijn mening kan geen van deze aan alle metadata-behoeften voldoen.
Hoe kan van één repository worden verwacht dat deze bijvoorbeeld zowel metadata van vertrouwelijke klantinformatie als metadata over de netwerkarchitectuur kan managen? Dat kan maar op één manier, namelijk door de eisen en wensen zodanig te veralgemeniseren dat er nauwelijks iets meer overblijft dan document management.

In het algemeen geldt dat enkelvoudige metadata-repository's een illusie zijn, want het is niet waarschijnlijk dat de specifieke metadata-behoefte kan worden ingevuld met een of ander generiek commercieel pakket, en dus zal repository-functionaliteit moeten worden gebouwd. Zulke repository's worden geacht om te kunnen gaan met uiteenlopende eisen, zoals database-specificaties vastgelegd in datamodellen, of de afkomst van data uit ETL-componenten in datawarehouse-projecten.
In werkelijkheid hebben metadata een oneindig bereik – het betreft dus zeker geen gefixeerde verzameling van tevoren bekende constructies. Denk aan alle nieuwe vormen van metadata

die zich voordoen als gevolg van het internet. Omdat de economie steeds meer informatiegeoriënteerd wordt, zullen de scope en de kwantiteit van te managen metadata daar naar verwachting in meegroeien.

Metadata en metamodellen

Als metadata in aanleg een onbeperkt bereik hebben, als één enkele repository niet alle metadata kan onderbrengen, en als veel metadata helemaal niets te maken hebben met andere metadata, dan moeten we uitgaan van een specifieke ontwerpbehoefte voor elke repository. Omdat het bij metadata om niet meer gaat dan een bijzonder perspectief van bepaalde soorten data, is het logisch een metadatamodel voor elke potentiële repository te ontwikkelen.
Tegenwoordig zou er nauwelijks meer iemand een database-implementatie durven plegen zonder eerst een datamodel van de database te ontwikkelen. Daarom moeten ook repository's worden gebaseerd op datamodellen. Datamodellen van repository's worden metamodellen genoemd, maar het zijn nog steeds datamodellen. Een datamodel voor een repository is een enorme hulp, of je nu een commercieel product koopt of van plan bent zelf te gaan bouwen. Bij een commercieel product dient het datamodel als referentiepunt, van waaruit 'gap' analyses kunnen worden uitgevoerd op kandidaat-producten. Het wordt veel eenvoudiger om te bepalen of een product dicht genoeg bij de gewenste repository-structuur zit om te worden aangeschaft. Als geen van de producten op de markt in de buurt komt, dan kan het datamodel gebruikt worden als basis voor het bouwen van de gewenste functionaliteit. Kortom, een datamodel maakt het besluit om te kopen of te bouwen een stuk gemakkelijker.

Een ander interessant punt van datamodellen voor repository's is dat ze eenvoudig kunnen laten zien waar metadata overbodig worden gedupliceerd. Al jarenlang beklagen data-administrators zich over het feit dat veel data onnodig worden gerepliceerd in zogenaamde silo's, vertegenwoordigd door stand-alone systemen. Een groot gedeelte van de initiatieven op het gebied van Master Data Management probeert deze situatie onder controle te krijgen en te houden. Het zou daarom treurig zijn als de behoefte om metadata te managen ertoe zou leiden dat de repository-silo's als paddestoelen uit de grond schieten. Dat gevaar is niet denkbeeldig. Het zal zich ook sneller voordoen als er gekochte producten worden geïmplementeerd. Veel repository's hebben bijvoorbeeld informatie nodig over entiteiten en attributen (of tabellen en kolommen). Als zulke metadata onafhankelijk van elkaar worden onderhouden in de vele repository's binnen een onderneming, dan is het onvermijdelijk dat de data steeds minder gesynchroniseerd raken.
Uiteindelijk zal dit problemen geven. Aangezien een kleine hoeveelheid metadata als hefboom werkt voor heel veel data, zullen die problemen hoogstwaarschijnlijk ernstige gevolgen hebben. Implementatie van commerciële producten, vooral die zonder open architectuur, leiden bijna zeker tot deze situatie.

De aanschaf van commerciële repository's op basis van 'need-by-need' kan gerechtvaardigd zijn in termen van individuele behoefte en in termen van de informatievereisten uitgedrukt in termen van datamodellen. Hoe dan ook, in het algemeen kan de aanschaf van zulke 'black box' repository's lange-termijn-problemen veroorzaken in het managen van metadata in een onderneming. Het goede nieuws is dat het ontwikkelen van repository-datamodellen om een besluit te kunnen nemen tussen kopen en bouwen, de beslissers al in een vroegtijdig stadium attendeert op deze problemen.

Gebruikscenario's

Ook de functionele eisen kunnen over het hoofd gezien worden, als de dataprofessionals zich op het standpunt stellen dat metadata begrensd zijn en dat één repository alle metadata-behoeften lenigt. Helaas is het beschrijven van de functionele eisen net zo nodig voor een repository als voor welk ander informatiesysteem dan ook; het is echt onmogelijk ze te omzeilen. Gelukkig raken IT-professionals meer bekend met *use cases* of gebruikscenario's, een uitstekende manier om de vereisten te definiëren. Als het goed opgestelde gebruikscenario's zijn, beschrijven ze tot in detail elke functionaliteit nodig in een repository. Commitment aan deze scenario's dwingt iedere betrokkene bij het repository-project tot helder denken en tot in detail wat ze van het project verwachten. Er zijn verschillende modellen voor de documentatie van gebruikscenario's, maar ze dienen allemaal het zelfde doel. Als er eenmaal gebruikscenario's zijn opgesteld, kunnen ze worden gebruikt bij de gap-analyse ten opzichte van commerciële producten, of als basis voor de eigen ontwikkeling van de repository-functionaliteit.

> Een database slaat data op, een repository slaat metadata op

Door hun specifieke karakter kunnen gebruikscenario's er ook voor zorgen dat de aanschaf of bouw van een té generieke repository-functionaliteit wordt vermeden. Dat is een reëel gevaar, dat zich eerder voordoet bij het kopen dan het bouwen van een repository. Ik heb bij verschillende opdrachtgevers gewerkt die zeer generieke repository's hadden gekocht, geïmplementeerd en gevuld met metadata. Na de lange en erg kostbare implementatiestadia, kwamen ze erachter dat hun repository's niet werden gebruikt. Ze hadden verzuimd de eisen en wensen vanuit de verschillende bedrijfsonderdelen te verzamelen en gebruikscenario's te ontwikkelen, en hun aanpak gebaseerd op de aanname dat alle metadata hetzelfde zijn, net als de eisen die aan het gebruik gesteld worden; het enige dat dus nog te doen stond, was de aankoop van een op de markt beschikbaar product.

Scope creep

Gebruikscenario's doen ook nog een paar andere dingen.
Ze voorkomen *scope creep*, zeg maar een erin sluipende focus-

verschuiving, dat altijd een gevaar vormt bij het bouwen van een repository, omdat metadata voor vele doeleinden gebruikt kunnen worden. De definities van entiteiten en attributen kunnen bijvoorbeeld worden toegepast in een Data Dictionary, of een business rules-applicatie bedoeld voor specialistische financiële transacties. Alleen als de functionaliteit van de repository strikt wordt gehouden aan de oorspronkelijke bedoeling, kan de verleiding om meer en meer eisen toe te voegen, worden weerstaan. Het is mijn ervaring, dat het gevaar van scope creep groter is bij zelfbouw-projecten dan bij koop. Ik ben echter ook situaties tegengekomen waarbij de vendor was ingegaan op functionaliteits-eisen die de aanvankelijk gestelde overtroffen, en te verklaren dat hun product aan die aanvullende eisen kon voldoen. Begrijpelijk, want geen enkele vendor wil zichzelf diskwalificeren in de slag om de klant, en technisch gesproken hadden de vendors gelijk waar het om de toepassingsmogelijkheden van hun product ging. Het is mijn ervaring echter dat op de markt beschikbare repository-producten zelden kunnen worden gebruikt voor andere doeleinden dan om invulling te geven aan de basiseisen waarvoor ze zijn bedoeld.
Een verzameling tevoren gedefinieerde gebruikscenario's, voorafgaand aan de beslissing te kopen of te bouwen, voorkomt scope creep. Nog belangrijker: gebruikscenario's zijn een grote hulp bij het nemen van de beslissing tussen koop en bouw, en zijn zelfs nadat de beslissing is gevallen van grote waarde.

Uitbreiding van repository's

Een ander probleem dat zich voordoet bij repository's is dat in het algemeen wordt aangenomen dat ze gemakkelijk kunnen worden uitgebreid. Bij nadere beschouwing is het niet moeilijk om te begrijpen dat dat niet zo is. Het feit dat we een database hebben geïmplementeerd voor bijvoorbeeld de crediteurenadministratie, betekent niet dat we die database ook kunnen gebruiken voor data van personeelszaken. Toch wordt er zo meestal niet over metadata gedacht. Ik heb veel klanten die denken dat, omdat een bepaalde repository een bepaalde klasse metadata beheert, deze gemakkelijk kan worden uitgebreid om ook andere klassen metadata te managen – simpelweg omdat het toch allemaal metadata zijn. Nogmaals: erg generieke repository's, meestal commerciële producten, zouden dit kunnen doen. Ze zijn echter zo algemeen van aard dat ze niet voldoen aan de specifieke eisen die de meeste ondernemingen stellen.
Met dit alles wordt niet gezegd dat repository's niet kunnen worden uitgebreid. Het is echter veel lastiger een commercieel product uit te breiden dat een ongedocumenteerd of moeilijk te begrijpen metamodel heeft. In het uiterste geval kunnen commerciële producten 'black boxes' zijn, waarvan het interne database-ontwerp niet wordt vrijgegegeven aan de kopers. Zulke producten zijn buitengewoon moeilijk uit te breiden, dus als vermoed wordt dat er in de toekomst additionele eisen zullen worden gesteld, is het niet verstandig een dergelijke repository aan te schaffen. Andere producten zijn opener en kunnen op maat worden uitgebreid. Als een organisatie echter besluit dit te doen, bestaat de kans dat ze de mogelijkheid verspelen om de repository te upgraden naar nieuwe versies. Dit is een bekend probleem met ERP-software, dat bij repository's moet worden vermeden. Er zijn nog meer problemen. Zelfs als een vendor het interne database-ontwerp van een repository-product vrijgeeft, kan een groot deel van het ontwerp schuilgaan achter de wijze waarop de code-tabellen het product feitelijk configureren. Het datamodel van het metamodel kan ongeschikt zijn voor maatwerk en uitbreiding van de repository.
In tegenstelling daarmee hebben repository's die op maat gebouwd zijn voor specifieke doeleinden, meer kans op uitbreiding. Het is waarschijnlijker dat de kennis over het ontwerp van zo'n repository binnen de onderneming aanwezig is, dan bij gekochte producten. Dat biedt echter geen garantie voor uitbreidbaarheid van de repository.

De behoefte aan algemene metadata-klassen voor gebruik in verschillende repository's is een belangrijke afweging. Zoals eerder gesteld kan bijvoorbeeld informatie over de definities van entiteiten en attributen nodig zijn in een groot aantal verschillende repository's. De implementatie van meerdere repository's die onafhankelijk van elkaar algemene metadata managen, gaat zeker problemen opleveren. Het is echter onmogelijk dat één repository alle vereiste functionaliteit levert. Dat betekent dat ondernemingen in de toekomst op zoek zullen gaan naar een federatieve repository-architectuur, waarin de algemene metadata op één plek wordt geactualiseerd en vervolgens gedeeld met andere repository's. Het is belangrijk om leveranciers te vragen of hun producten een dergelijke architectuur kunnen ondersteunen, bijvoorbeeld door het eenvoudig importeren van entiteits- en attribuut-definities.
Wordt besloten de repository-functionaliteit zelf te bouwen, dan moet de onderneming ervoor waken onafhankelijke eilanden van metadata te creëren, en moet rekening houden met een overall metadata-architectuur.

Tot slot

Er zijn veel afwegingen te maken als een keuze moet worden gemaakt een repository-functionaliteit te kopen of zelf te bouwen. Het belangrijkste is dat wordt begrepen dat metadata en repository-functionaliteit in aanleg een oneindige scope hebben en dat een focus nodig is op de details van de eisen.
Datamodellen en gebruikscenario's kunnen in dit opzicht erg behulpzaam zijn. Een verrassend groot aantal repository's wordt nog in eigen huis gebouwd en dat zal waarschijnlijk in de nabije toekomst zo blijven. Aan de andere kant groeit het aantal hoogwaardige commerciële producten, die uitstekende oplossingen kunnen bieden als de onderneming de producten zorgvuldig evalueert.

In geval van discussies geeft de originele Engelstalige tekst van dit artikel de doorslag. Deze tekst is te vinden op onze website www.dbm.nl in het hoofdmenu onder Specials/Extra materiaal.

Malcolm Chisholm is directeur van Askget.com Inc te New Jersey.