Service Oriented Architecture

Een 7 lagen architectuur voor service oriëntatie

Master Thesis Informatiekunde
Radboud Universiteit Nijmegen
Nijmeegs Instituut voor Informatica en Informatiekunde (NIII)
Afstudeernummer: 43 IK

Thomas Dobbe

April 2007

- **Afstudeerbegeleider**

prof. dr. Mario van Vliet
Hoogleraar 'Management van softwareprojecten' aan de Radboud Universiteit Nijmegen
Vice president 'Supply Chain Management' bij Capgemini Consulting

- **Referent**

dr. Patrick van Bommel

- **Stagebegeleider**

drs. Diederik van Duuren
Manager SAP bij IDS Scheer Nederland
Projectmanager ‘VIEW’ (Essent)

Voorwoord

Voor u ligt een scriptie van het onderzoek naar Service Oriented Architecture (SOA) dat ik heb uitgevoerd ter afsluiting van de master informatiekunde aan de Radboud Universiteit Nijmegen. Het onderzoek is gecombineerd met een stage bij IDS Scheer, waar ik de mogelijkheid kreeg deel uit te maken van een project bij Essent. Hierin kon ik een praktijkvoorbeeld aanschouwen en beleven, kennis en kunde opdoen en mensen uit het project en de organisatie spreken.

Er is een aantal personen dat een bijdrage heeft geleverd aan het tot stand komen van deze scriptie. Graag wil ik mijn afstudeerbegeleider Mario van Vliet bedanken voor de prettige gesprekken die we gevoerd hebben gedurende de loop van het onderzoek en voor de feedback op mijn geschriften. Verder gaat mijn dank uit naar de begeleiders van IDS Scheer, Diederik van Duuren en Marcel Beelen. Ik heb met hen op een leuke manier samengewerkt tijdens de stage, de mogelijkheid gekregen mijn bijdrage te leveren aan het project, interessante gesprekken gevoerd met betrekking tot SOA en mijn onderzoek en feedback gekregen. Daarnaast wil ik de volgende personen bedanken voor hun tijd, uitleg en / of commentaar: Boyd Hendriksen, Wiemer Kuik, Edward van der Kust, Mark van Liefland, Lonneke Nouwen, Edo van de Velde en Huub Waterval.

Veel leesplezier gewenst!

Thomas Dobbe
Nijmegen, april 2007

Samenvatting

Deze scriptie is geschreven in het kader van de master informatiekunde aan de Radboud Universiteit Nijmegen. Het betreft een onderzoek naar de Service Oriented Architecture (SOA), een onderwerp dat zeer actueel is en waar men in de wereld van het bedrijfsleven en de IT niet meer omheen kan. SOA is een paradigma (een concept of abstractie) dat kan worden toegepast op een organisatie. Hierbij wordt IT-functionaliteit opgedeeld in onafhankelijke en herbruikbare delen, services, zodat deze gemakkelijk kunnen worden afgestemd op bedrijfsprocessen. Een organisatie zou hiermee beter in staat moeten zijn om in te spelen op veranderingen in markten waarin ze opereert.

Het onderzoek bestaat uit 4 hoofdvragen. De eerste vraag is welke verdeling in architectuurlagen het meest geschikt is voor SOA. Architectuurlagen vormen een abstractie van IT en organisatie en de wijze waarop ze gerelateerd zijn. Ten tweede worden deze architectuurlagen gebruikt om te beantwoorden in welke hoedanigheid SOA wordt toegepast bij het bedrijf Essent. Ten derde worden de Business Process Management Suites (BPMSs) ARIS en Cordys met elkaar vergeleken in een case study. Een BPMS is een tool waarmee processen in kaart kunnen worden gebracht en gekoppeld aan services. In de case study wordt aan de hand van de architectuurlagen onderzocht welke rol de tools spelen bij het toepassen van SOA. Tot slot wordt ingegaan op de toegevoegde waarde van SOA voor een organisatie.

De scriptie begint met een theoretisch deel, waarin de achtergrond en context van SOA worden behandeld en waarin wordt ingegaan op de manier waarop IT ondersteuning biedt aan bedrijfsprocessen. Verder wordt ingegaan op de ontwikkeling van bedrijfsprocessen in organisaties. Vervolgens wordt een 7 lagen architectuur voorgesteld, die geschikt is voor SOA. Achtereenvolgens bestaat deze uit een gebruikerslaag, procesmodellaag, procesexecutielaag, servicelaag, integratielaag, applicatielaag en datalaag. De servicelaag bevindt zich in het midden en bevat een overzicht (een service directory) van de beschikbare IT-functionaliteit in de vorm van services. Deze laag zorgt voor ontkoppeling tussen enerzijds een behoefte die voortkomt uit een bedrijfsproces en anderzijds de implementatie van een service. Dit is essentieel voor SOA, omdat men op die manier in staat is services in een gehele organisatie (en eventueel daarbuiten) te hergebruiken. Men is namelijk niet meer afhankelijk van bijvoorbeeld platform en taal.

Na de theorie volgt een deel over de praktijk van Essent en van de BPMSs ARIS en Cordys. Bij Essent blijkt dat SOA volgens de besproken theorie wordt toegepast. De architectuur van Essent verschilt echter met de 7 lagen architectuur. Ten eerste beschikt Essent niet over een aparte servicelaag. Ten tweede worden de processen ondergebracht in één laag, waar de 7 lagen architectuur een onderscheid maakt in twee lagen.
Verder is de rol die de BPMSs ARIS en Cordys spelen als volgt. ARIS biedt een oplossing voor het ontwikkelen en verbeteren van bedrijfsprocessen, gericht vanuit een organisatiecontext. Cordys biedt een integratie- en executieplatform, dat een integrale oplossing is voor meerdere architectuurlagen. In tegenstelling tot ARIS is het modelleren bij Cordys echter wel afhankelijk van de Process Execution Engine (een omgeving voor het uitvoeren van bedrijfsprocessen).

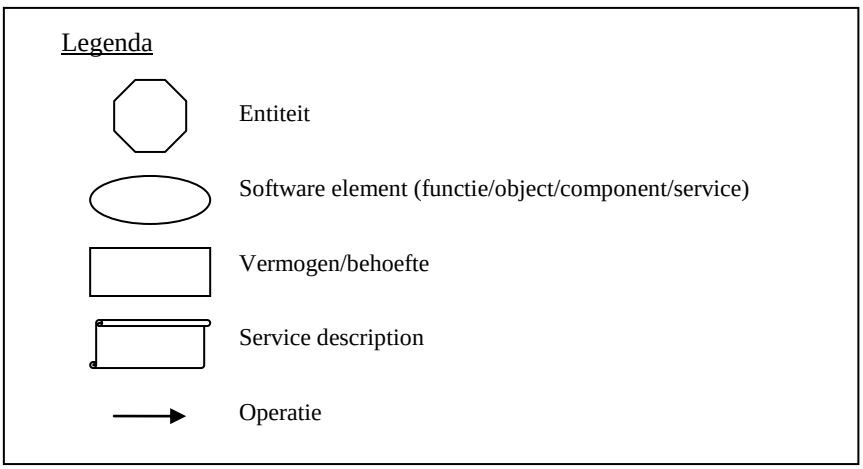
Het toepassen van SOA in organisaties is nog volop in ontwikkeling en zal nog tijd nodig hebben om zijn waarde te bewijzen. De verwachte voordelen zijn herbruikbaarheid, wendbaarheid en interoperabiliteit. De bijbehorende gevolgen zijn kortere ontwikkeltijden van services, lagere kosten, risicobeperking voor het maken van fouten, minder redundantie, kortere time-to-market, grotere beheersbaarheid van het IT-landschap en een hogere consistentie.

De verwachte nadelen zijn problemen en moeilijkheden die zich kunnen voordoen bij governance, met alle kosten van dien, en het pas op de lange termijn profiteren van voordelen.

Tot slot wordt geconcludeerd dat de 7 lagen architectuur een bijdrage kan leveren wanneer men SOA toepast. Het vormt een handvat bij het maken van een keuze voor een BPMS, het vormt een referentiekader dat services centraal stelt en het geeft de ontkoppeling aan tussen techniek en processen.

Inhoudsopgave

Figuren



Definities

1 Inleiding

Dit hoofdstuk vormt een introductie op het onderwerp en het onderzoek. Er wordt een theoretisch kader geschetst, waarna de probleemstelling en onderzoeksmethode volgen. Daarna wordt ingegaan op de stage, waar het onderzoek mee is gecombineerd, de verantwoording van het onderzoek en de opbouw van de scriptie.

1.1 Onderzoeksaanleiding

Het onderzoek is uitgevoerd in het kader van de masterscriptie informatiekunde aan de Radboud Universiteit Nijmegen en dient als afsluitende proeve van bekwaamheid. Het onderzoek is gecombineerd met een stage bij IDS Scheer, een software en consulting bedrijf. De probleemstelling is opgesteld in samenspraak met zowel de Radboud Universiteit als IDS Scheer.

1.2 Theoretisch kader

Ter inleiding op het onderwerp en de probleemstelling worden eerst enkele kernbegrippen toegelicht.

Service Oriented Architecture

Het onderzoeksonderwerp is Service Oriented Architecture (SOA), een term die momenteel zeer populair is. Het is een paradigma (een concept of abstractie, zie hoofdstuk 2) dat kan worden toegepast op een organisatie en de bijbehorende informatietechnologie(IT)-huishouding, waarbij men IT-functionaliteit herbruikbaar maakt, door deze aan te bieden in zo onafhankelijk mogelijke services. Deze services kunnen algemeen beschikbaar worden gesteld in de organisatie (en eventueel ook daarbuiten), worden gebruikt om delen van bedrijfsprocessen te ondersteunen en werken volgens afgesproken standaarden.

Bedrijfsprocessen

Door snel optredende veranderingen in markten waarin organisaties opereren, is het van belang dat organisaties hun bedrijfsprocessen hierop kunnen afstemmen. Een bedrijfsproces (afgekort: proces) is een verzameling van logisch gerelateerde taken met een gedefinieerd bedrijfsresultaat (zie hoofdstuk 4). Wanneer een bedrijf een nieuw product introduceert of zijn dienstverlening aanpast, moet de IT-ondersteuning hierin mee veranderen. SOA pretendeert hiervoor een oplossing te bieden door de IT-huishouding zo in te delen dat deze gemakkelijk afgestemd kan worden op processen.

Business Process Management

Zoals aangegeven is het voor SOA van groot belang dat business en IT op elkaar worden afgestemd. Processen (de business-kant) bepalen hoe een organisatie opereert en zullen zo goed mogelijk ondersteund moeten worden door services (de IT-kant). Bij deze afstemming kan gebruik gemaakt worden van een zogenaamde Business Process Management Suite (BPMS). Business Process Management (BPM) is de verzameling van activiteiten en het gebruik van methoden en software om bedrijfsprocessen te ontwikkelen, te beheren en continu te verbeteren (zie hoofdstuk 3). Een BPMS is een tool die mogelijkheden biedt om deze activiteiten te ondersteunen, zoals het in kaart brengen van processen en hierop afstemmen van IT-elementen. In het onderzoek zal een case study behandeld worden over de BPMSs ARIS (van IDS Scheer) en Cordys (van het gelijknamige bedrijf).

Architectuur

Architectuur is de fundamentele organisatie van een systeem, uitgedrukt in zijn componenten, hun relaties met elkaar en met de omgeving en de principes die ontwerp en evolutie leiden (zie hoofdstuk 4). Met systeem wordt zowel gedoeld op specifieke software systemen (applicaties, zie hoofdstuk 2) als op de wijze waarop processen,

producten en diensten samenhangen en ondersteund worden door IT. Organisaties gebruiken architectuur als abstractiemiddel om hun (vaak complexe) IT-huishouding in kaart te brengen. Dit kan met een verdeling in architectuurlagen, waarbij een architectuurlaag bestaat uit soortgelijke componenten. Zo kan men bijvoorbeeld een laag onderscheiden die te maken heeft met bedrijfsprocessen of met applicaties. In het onderzoek zal onder andere worden ingegaan op een verdeling in architectuurlagen en hun relaties, waarbij het minder gaat om de achterliggende principes.

1.3 Probleemstelling

In het onderzoek zal antwoord gegeven worden op vier hoofdvragen. Deze bevatten ieder enkele deelvragen, die helpen om de hoofdvragen te beantwoorden.

1. Welke verdeling in architectuurlagen is het meest geschikt voor het toepassen van SOA?

1.1. Wat is het SOA paradigma?

1.2. Welke architectuurlagen zijn te onderscheiden bij het toepassen van SOA?

1.3. Waardoor worden deze architectuurlagen gekenmerkt?

2. In welke hoedanigheid wordt het SOA paradigma toegepast bij Essent?

2.1. Wat zijn de verschillen en overeenkomsten tussen de voor SOA meest geschikte verdeling in architectuurlagen en de architectuur van Essent?

2.2. Wat zijn de verschillen en overeenkomsten tussen het SOA paradigma en de toepassing van SOA bij Essent?

3. Welke rol spelen de BPMSs ARIS en Cordys bij het toepassen van SOA?

3.1. Wat zijn de kenmerken van ARIS en Cordys?

3.2. Welke BPM functionaliteiten zijn relevant bij het toepassen van SOA?

3.3. Wat zijn de overeenkomsten en verschillen tussen ARIS en Cordys bij een vergelijking met de voor SOA meest geschikte verdeling in architectuurlagen?

4. Wat is de toegevoegde waarde van het toepassen van SOA voor het oplossen van bedrijfsproblematiek en het behalen van bedrijfsdoelstellingen?

4.1. Welke voor- en nadelen heeft het toepassen van het SOA paradigma?

4.2. Welke gevolgen hebben de voor- en nadelen voor een organisatie?

1.4 Onderzoeksmethode

Om de onderzoeksvragen te kunnen beantwoorden zijn de volgende activiteiten uitgevoerd.

Literatuurstudie
Allereerst is een literatuurstudie uitgevoerd naar het SOA paradigma vanuit de meest recente inzichten in de wetenschap. Ook is de literatuur geraadpleegd over de toegevoegde waarde van SOA.

Modelvorming
Er is een theoretisch model opgesteld dat definieert wat het SOA paradigma inhoudt.

Toetsing in de praktijk
Er zijn twee zaken getoetst met de praktijk. Allereerst is de voor SOA meest geschikte verdeling in architectuurlagen vergeleken met de situatie bij Essent; Essent werkt met de zogeheten '5 lagen architectuur'. Verder zijn ook het SOA model en de definitie van SOA vergeleken met de praktijk bij Essent; kortom, in welke hoedanigheid komen de elementen uit het model en uit de definitie tot uitdrukking bij Essent.

BPM case study
Bij Essent wordt ARIS gebruikt, dat een rol speelt bij het toepassen van SOA. Deze rol is nader onderzocht door als case study enkele processtappen van Essent uit te werken. Ook met Cordys is een case uitgewerkt.

Vergelijken BPMSs
Naast het modelleren van een proces zijn de BPMSs ARIS en Cordys vergeleken aan de hand van de voor SOA meest geschikte verdeling in architectuurlagen. Hierbij is ook ingegaan op de manier waarop business en IT gekoppeld worden.

1.5 Stage

De afstudeerstage is uitgevoerd bij IDS Scheer Nederland B.V. Het is een software en consulting bedrijf, dat betrokken is bij een project van Essent, waarbij SOA wordt toegepast. De stage bestond onder andere uit het inhoudelijk uitvoeren van het onderzoek, het bemannen van het projectbureau, het assisteren van de projectmanager, het ondersteunen van de procesmanager met het in kaart brengen van processen en het geven van trainingen aan toekomstige gebruikers.
De stage heeft mogelijkheid geboden om de literatuur in de praktijk te toetsen en om ervaring met ARIS op te doen.

1.5.1 IDS Scheer en ARIS

IDS Scheer is een Duits bedrijf en opgericht in 1984. De Nederlandse vestiging is in Den Haag. IDS Scheer is wereldwijd vertegenwoordigd door een netwerk van eigen vestigingen en business partners en heeft meer dan 6000 klanten in 70 landen. Wereldwijd zijn er 2800 werknemers en in 2006 is er een omzet van ruim 354 miljoen euro gerealiseerd. IDS Scheer maakt onderdeel uit van de groep bedrijven die worden verhandeld op de Frankfurt Stock Exchange, de TecDAX index.
Het bedrijf heeft een eigen BPMS, het ARIS platform. ARIS, dat staat voor *Architecture of Integrated Information Systems*, biedt een heel scala aan functionaliteiten, zoals modellering, analyse, implementatie, integratie en optimalisatie van bedrijfsprocessen. Volgens IDS Scheer is het platform wereldwijd de meest verkochte oplossing voor procesmodellering. [IDS Scheer 2007]

1.5.2 Essent en VIEW

Essent is leverancier van voornamelijk elektriciteit en gas, maar ook van kabelproducten zoals internet, televisie, radio en telefonie. Het bedrijf is opgericht in 1999. [Essent 2007]

Binnen Essent zijn in de loop der jaren veel verschillende applicaties ontstaan. Deze wirwar zorgt voor onoverzichtelijkheid voor werknemers, maakt het tijdrovend om mee te leren werken en zorgt voor een lange verwerkingstijd. Dit is de aanleiding geweest om het project VIEW te starten, wat staat voor Virtual Integrated Essent Workplace.

Het doel van VIEW is om een digitale werkplek of portal te creëren voor de werknemers van Essent. Alle 'oude' systemen (die blijven bestaan) worden hieraan gekoppeld, zodat de gebruikers hier niet meer direct mee te maken hebben, maar slechts met één gebruikersvriendelijke interface. Een bedrijfsproces waarbij dit wordt toegepast is het *hire-to-retire* proces, dat uitgevoerd moet kunnen worden via de portal. Dit omvat de processtappen *indienst*, *mutatie* en *uitdienst* die de gehele cyclus van een werknemer bij Essent omvatten. De portal wordt ingericht afhankelijk van de rol van een werknemer in de organisatie en bevat een takenlijst voor het uitvoeren van processtappen.

Naast het uitvoeren van processtappen via de portal wil men inzicht krijgen in het verloop ervan. Daarom moeten processtappen ook gevolgd kunnen worden middels een zogenaamde procesmonitor die de status en doorlooptijd van iedere processtap bijhoudt. Dit is essentieel voor BPM, omdat processen zo verbeterd kunnen worden.

5 lagen architectuur
De architectuur binnen Essent wordt aangeduid met de 5 lagen architectuur, die bestaat uit de presentatielaag, proceslaag, integratielaag, transactielaag en datalaag. De transactielaag en datalaag bestaan al en bevatten allerlei applicaties, bijvoorbeeld modules van softwareleverancier SAP, met achterliggende databases. De presentatielaag bevat onder andere de portal, waarmee de gebruiker werkt; deze interacteert met de achterliggende systemen. In de proceslaag worden bedrijfsprocessen gedefinieerd met behulp van ARIS en worden processen uitgevoerd. Vanuit de processen is er een koppeling via de integratielaag met de transactielaag (de applicaties) en datalaag (de databases). In hoofdstuk 6 wordt hier dieper op ingegaan.

1.6 Verantwoording

In de IT-branche is de laatste jaren steeds meer sprake van het begrip Service Oriented Architecture. Het is een nieuwe term die vaak in de mond wordt genomen, maar weinig mensen zijn werkelijk bekend met de betekenis van het concept [Marketcap 2006]. De Automatiseringgids zegt er het volgende over:

'Meer dan een derde van de mensen die in bedrijven verantwoordelijk zijn voor applicatiebeleid blijkt de term SOA wel te kennen, maar heeft geen goed beeld van het concept. De helft van die groep heeft niet meer dan een globaal idee van de inhoud van het SOA concept. Overall zegt maar 15 procent redelijk bekend te zijn met het begrip SOA: men heeft er ten minste iets over gelezen of over gehoord. Een selecte groep van 7 procent geeft aan goed bekend te zijn met de Service Oriented Architecture; men heeft zich naar eigen zeggen al in het onderwerp verdiept.' [Automatiseringgids 2006]

Kortom, het is van belang om SOA te benaderen vanuit een theoretisch oogpunt en te concretiseren wat men er in de praktijk mee kan. Voor het praktijkdeel is gebruik gemaakt van de situatie bij Essent en de BPMSs ARIS en Cordys.

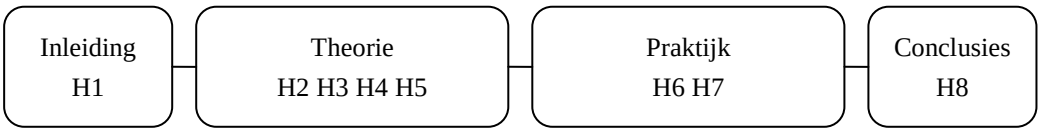
Op dit moment is er nog relatief weinig praktijkervaring met het toepassen van SOA. Juist hierom zal een organisatie wellicht wat terughoudend zijn. Wanneer men SOA wil toepassen zal men eerst moeten beschikken over een visie op zowel het gebied van organisatie als van architectuur. Men moet weten wat de organisatie wil

nastreven op de lange termijn en welke gevolgen dit heeft voor de IT-huishouding. In dit onderzoek wordt een verdeling in architectuurlagen voorgesteld die specifiek geschikt is voor SOA. Deze verschaft inzicht in de componenten die men daarvoor nodig heeft en hoe deze samenhangen.
Zoals eerder aangegeven kunnen BPMSs een belangrijke rol spelen bij het toepassen van SOA. Er is een vergelijking gemaakt tussen ARIS en Cordys aan de hand van de verdeling in architectuurlagen. Vanwege de grote verschillen tussen de beschikbare tools is het relevant om te weten in welke architectuurlagen zij functionaliteit bieden.
De keuze voor ARIS en Cordys is gebaseerd op de verschillen in nadruk van deze producten. Cordys is gericht vanuit de IT-kant en ARIS meer vanuit de business. Om de rol van een BPMS voor SOA aan te kunnen geven zijn twee platforms met grote verschillen juist interessant.

Verder is het voor een organisatie relevant om inzicht te krijgen in de toegevoegde waarde van SOA. Beslissingen hieromtrent hebben immers financiële impact en dienen goed onderbouwd te kunnen worden. Het gegeven dat er nog weinig praktijkervaring is maakt dit des te moeilijker. Bovendien richten softwareleveranciers, zoals SAP, Oracle en IBM, hun producten in op het gebruik van SOA. Naast de aandacht van de media is er dus ook een push vanuit de leveranciers waar te nemen. Des te belangrijker is het voor een organisatie om voldoende kennis te bezitten over SOA en over de mogelijke toegevoegde waarde voor de specifieke organisatiecontext. Zowel vanuit de situatie bij Essent als vanuit de literatuur is hier op ingegaan.

1.7 Opbouw scriptie

De scriptie bestaat uit 8 hoofdstukken. Hoofdstuk 1 is een inleidend hoofdstuk dat een introductie vormt op het onderwerp en het onderzoek. In hoofdstuk 2 wordt de context van SOA besproken, zodat de lezer bagage meekrijgt over de historie en achtergrond van het onderwerp. Hoofdstuk 3 gaat over de manier waarop processen werken in organisaties. In hoofdstuk 4 wordt ingegaan op architectuur en wordt een verdeling in architectuurlagen voorgesteld die geschikt is voor SOA. Het begrip SOA wordt verder toegelicht in hoofdstuk 5. In dit hoofdstuk is een theoretische benadering van het paradigma opgenomen en zijn de voor- en nadelen benoemd. Hoofdstuk 6 gaat over Essent en VIEW, waarin de architectuurlagen uit hoofdstuk 4 worden vergeleken met die van Essent en de toepassing van SOA bij Essent naast de theorie wordt gelegd. Hoofdstuk 7 is een case study over ARIS en Cordys, die eindigt met een vergelijking tussen de twee BPMSs aan de hand van de voorgestelde architectuurlagen. Hoofdstuk 8 rondt tot slot af met de conclusies.
Het geheel kan worden opgedeeld in een theoretisch en een praktisch deel. Dit is weergegeven in figuur 1.1.



Figuur 1.1: Opbouw scriptie

2 Context

In dit hoofdstuk wordt de context van SOA besproken. Eerst komt de historie van softwareontwikkeling aan bod in de vorm van paradigma's. Vervolgens wordt ingegaan op de koppeling tussen bedrijfsprocessen en IT, waarbij de begrippen Enterprise Application Integration en Business Process Integration toegelicht worden. Tot slot volgt een paragraaf over technische infrastructuur.

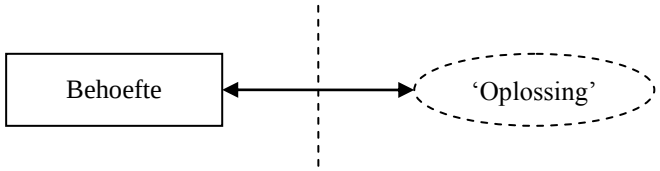
2.1 Historie

SOA is niet volledig nieuw. Wanneer je teruggaat in de historie van de softwareontwikkeling, zie je dat er steeds andere manieren geïntroduceerd worden om functionaliteit op te delen voor hergebruik. SOA kan hierbij gezien worden als de meest recente evolutiestap. De historie van de softwareontwikkeling kan toegelicht worden aan de hand van software paradigma's.

2.1.1 Software paradigma's

Een *paradigma* is een overkoepelend denkraamwerk, dat regels omvat die vertellen hoe je bepaalde problemen binnen gestelde grenzen kan oplossen. Het vormt een abstractie of een model van de werkelijkheid, om complexe vraagstukken beter te kunnen voorstellen en analyseren. Wanneer oplossingen worden gevonden die buiten de regels treden van een bestaand paradigma, is sprake van een paradigmaverschuiving.

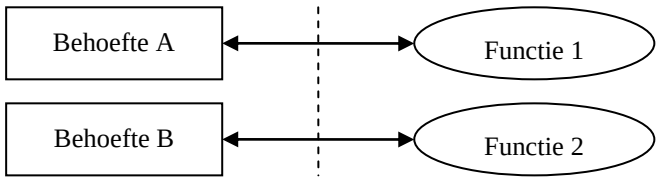
Software is vaak zeer complex door de omvang, de vele afhankelijkheden en de dynamiek van de omgeving. In de historie van de softwareontwikkeling zijn dan ook steeds nieuwe paradigma's ontstaan om complexiteit te reduceren door software-elementen te hergebruiken. Om de verschillen tussen paradigma's aan te kunnen geven worden ze besproken in termen van behoeftes en oplossingen. In elk paradigma bestaan behoeftes (bijvoorbeeld het berekenen van het totaalbedrag van een rekening of het beheren van de volledige salarisadministratie) die op een manier verbonden zijn met software-elementen die een oplossing vormen. In de hierna volgende figuren die de paradigma's illustreren staan steeds links de behoeftes en rechts de elementen die een oplossing representeren; zie figuur 2.1. De pijl geeft aan hoe behoefte en oplossing van elkaar afhangen; dit wordt later toegelicht.



Figuur 2.1: Software paradigma (algemene weergave)

- **Procedureel**

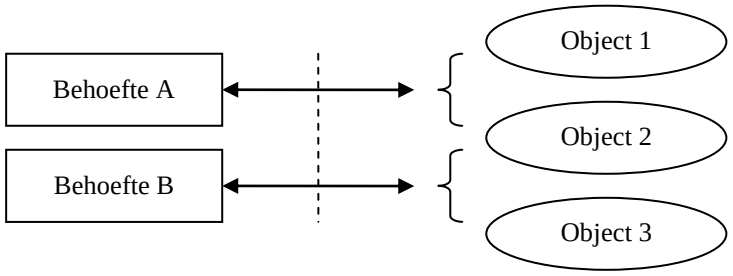
Bij procedureel ontwikkelen wordt hergebruik gerealiseerd door definitie van procedures, die alleen in het programma zelf kunnen worden aangeroepen. Software is dan vooral een verzameling instructies en functies. De behoefte bepaalt volledig hoe de oplossing eruit ziet. Behoefte en oplossing zijn nauw verbonden ('tightly coupled'); er is een 1 op 1 relatie. In de praktijk betekent dit dat er een functie wordt ontworpen die een oplossing vormt voor de behoefte.



Figuur 2.2: Procedureel

- **Object georiënteerd**

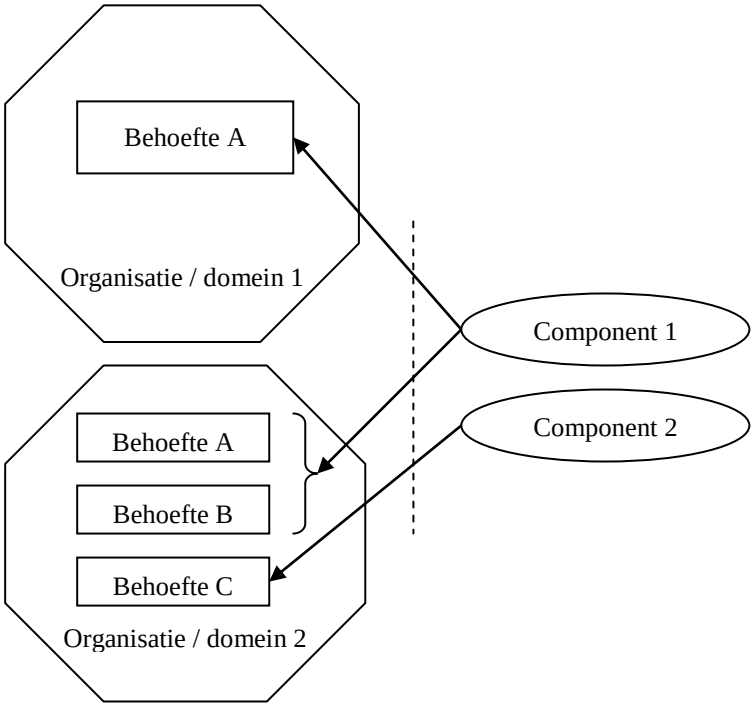
Bij object georiënteerd ontwikkelen worden klassen ontworpen met objecten. Een klasse vormt vervolgens de oplossing die gebruik kan maken van objecten uit andere klassen. Software wordt zo flexibeler en beter beheerbaar. Ook hier is de behoefte bepalend voor de oplossing. Echter, een verdeling in klassen biedt meer mogelijkheden tot hergebruik, waardoor oplossingen minder specifiek samenhangen met behoeftes.



Figuur 2.3: Object georiënteerd

- **Component gebaseerd**

Bij component gebaseerde softwareontwikkeling wordt gebruik gemaakt van kant-en-klare componenten. Een component is een software element dat een hoger abstractieniveau heeft dan een object en is in een ideale omgeving gemakkelijk vervangbaar. Componenten zijn namelijk zo min mogelijk afhankelijk van andere componenten en kunnen gebruikt worden zonder details te kennen van de implementatie. Bij componenten kan het echter wel zo zijn dat aannames gemaakt zijn over delen van bedrijfsprocessen, bijvoorbeeld de manier waarop een order wordt afgehandeld in een order-to-cash proces.
Een leverancier biedt een component, die een oplossing vormt voor een behoefte die in meerdere organisaties of domeinen kan bestaan. Een *domein* is een samenhangend deel (binnen een groter geheel, zoals een organisatie of branche). Er kan bijvoorbeeld een functionele samenhang zijn, zoals marketing of financiën. Een voorbeeld van een component is een Human Resources component die de mogelijkheden biedt om gegevens over werknemers, contracten en bedrijfsmiddelen te beheren en waar allerlei bewerkingen mee uitgevoerd kunnen worden. De component kan echter ook een oplossing bieden voor verschillende behoeftes en kan afhankelijk van de behoeftes in een domein of organisatie worden geconfigureerd. Zo kan een Human Resources component ook een oplossing bieden voor de behoeftes aan ondersteuning bij werving en selectie en voor personeelsplanning. Niet iedere klant heeft per se alle behoeftes waarvoor een component een oplossing biedt; een klant kan dan ook kiezen welke functionaliteiten gebruikt worden.
Bij dit paradigma wordt de oplossing bepaald door de leverancier van de component en minder door behoeftes van individuele klanten. De klant kan een component wel naar eigen wensen configureren.

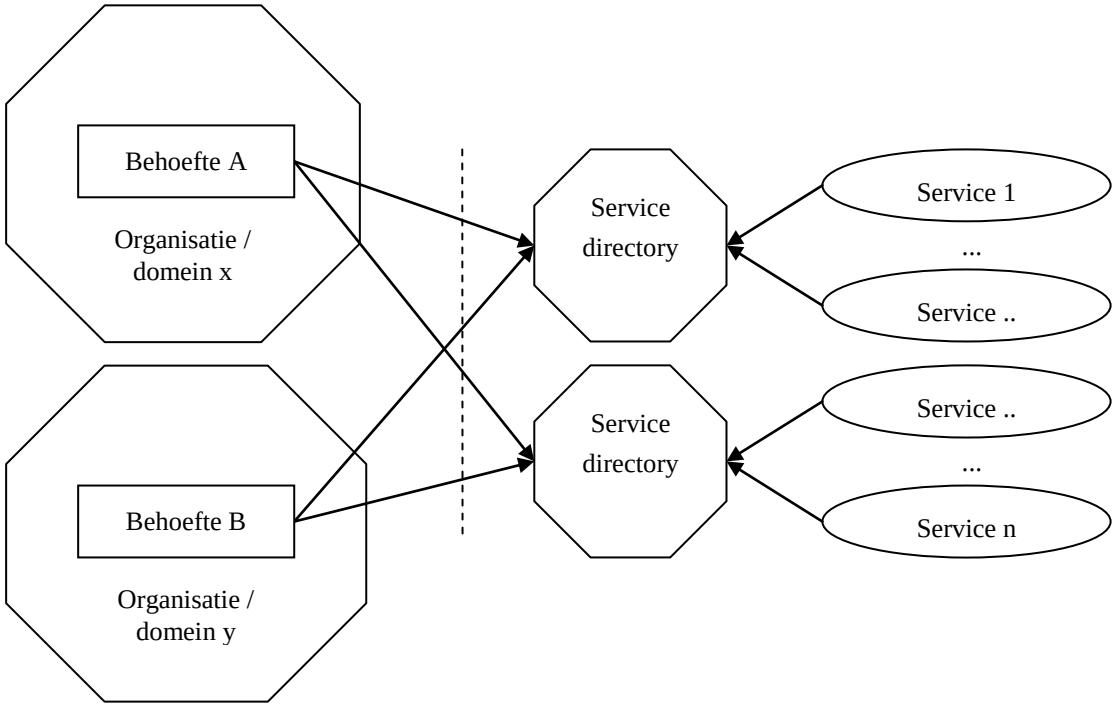


Figuur 2.4: Component gebaseerd

- **Service georiënteerd**

Bij service oriëntatie worden stukken functionaliteit aangeboden in services. Een oplossing bestaat vervolgens uit één of meerdere services. Services zijn zoveel mogelijk herbruikbaar en gebaseerd op standaarden. Het ontwikkelen van software gebeurt hier vanuit een ander perspectief, omdat de behoefte aan services met een bepaalde functionaliteit voortkomt uit bedrijfsprocessen. In een ideale service georiënteerde omgeving zoekt men bij het creëren van een bedrijfsproces eerst in een zogenaamde service directory (een overzicht van services, zie hoofdstuk 5) naar reeds bestaande services die delen van het proces kunnen ondersteunen. Pas daarna gaat men eventueel zelf services ontwikkelen. Dit is weergegeven in figuur 2.5. Ieder domein of iedere organisatie kan een eigen service directory hebben, die breder beschikbaar gesteld kan worden om services aan te bieden.

Services en componenten kunnen technisch gezien gelijk aan elkaar zijn. Ze verschillen in zoverre van elkaar dat bij services de mate van detaillering aangepast kan worden op de behoefte. Services zijn directer gerelateerd aan delen uit een bedrijfsproces. Verder kan een hiërarchie aangebracht worden bij services, waarbij services van elkaar gebruik maken en op die manier beter afgestemd kunnen worden op behoeftes.
Dit paradigma is bekend onder de naam Service Oriented Architecture, waar later uitgebreid op in wordt gegaan.



Figuur 2.5: Service georiënteerd

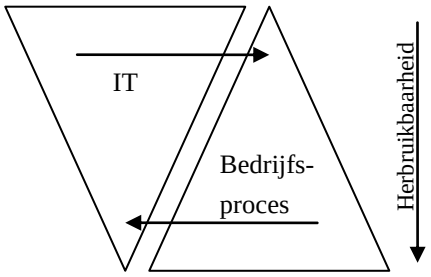
2.1.2 Verschillen en overeenkomsten

Paradigma's kunnen voor een deel naast elkaar bestaan. Procedureel en objectgeoriënteerd ontwikkelen zijn verschillende manieren om op laag niveau programmacode te structureren. Component gebaseerd ontwikkelen en service oriëntatie worden gezien vanuit een ander perspectief en kunnen elkaar overlappen. Een organisatie die componenten gebruikt kan deze namelijk best op een service georiënteerde manier koppelen met bedrijfsprocessen, maar dit hoeft niet. En bij service oriëntatie kan men gebruik maken van componenten, maar dit hoeft niet.

De pijlen in de figuren geven aan hoe de behoefte en de oplossing gerelateerd zijn. Bij procedureel ontwikkelen wordt een functie gebouwd voor een specifieke behoefte en kan niet voor andere behoeftes worden gebruikt. De relatie tussen behoefte en oplossing is hier 1 op 1. Bij objectoriëntatie worden objecten ook gecreëerd voor een specifieke behoefte en zijn iets meer herbruikbaar, maar nog steeds nauw verbonden. Bij component gebaseerd ontwikkelen wordt de oplossing bepaald door de leverancier. Deze kijkt wel naar behoeftes bij (potentiële) klanten, maar bepaalt zelf de (standaard)oplossing. Bij service oriëntatie tot slot, bepalen behoeftes (uit bedrijfsprocessen) hoe de oplossing eruit komt te zien, namelijk één of meerdere services.

In figuur 2.6 worden de paradigma's weergegeven met het bijbehorende herbruikbare element. Verder is te zien dat een verschuiving heeft plaatsgevonden van softwareontwikkeling vanuit de mogelijkheden die IT te bieden heeft naar de wensen voor bedrijfsprocessen in een organisatie. De herbruikbaarheid van software-elementen is hierbij steeds verhoogd.

Paradigma	Herbruikbaar element
Procedureel	Functie
Object georiënteerd	Object
Component gebaseerd	Component
Service georiënteerd	Service



Figuur 2.6: Paradigma's voor softwareontwikkeling

2.2 Koppeling bedrijfsprocessen en IT

In de vorige paragraaf zijn softwareparadigma's besproken. Bij component gebaseerd ontwikkelen en service oriëntatie spelen bedrijfsprocessen een belangrijke rol. Bij standaard componenten worden namelijk vaak aannames gedaan over (delen van) processen; bijvoorbeeld de manier waarop een bestelling van een product via internet wordt afgehandeld. En bij service oriëntatie vormen bedrijfsprocessen het uitgangspunt. Bij procedureel en object georiënteerd ontwikkelen ligt de focus echter op het indelen van programmacode. Bedrijfsprocessen hebben daar geen directe relatie.
Bij component gebaseerd ontwikkelen en service oriëntatie is er een koppeling tussen bedrijfsprocessen en IT. Deze koppeling is de manier waarop IT ondersteuning biedt aan bedrijfsprocessen; de manier waarop stappen in een bedrijfsproces worden uitgevoerd door IT. Hierbij zijn de begrippen Business Process Integration en Enterprise Application Integration erg belangrijk, die in de volgende subparagrafen besproken worden.

2.2.1 Business Process Integration

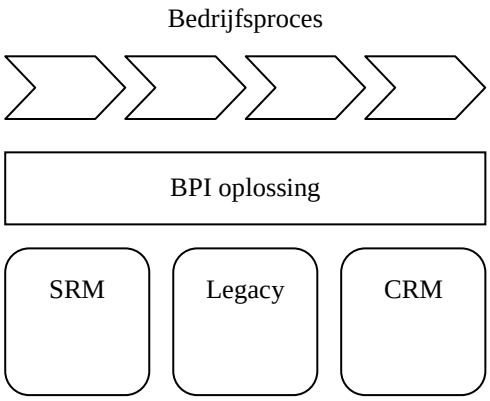
Business Process Integration (BPI) wordt als volgt gedefinieerd.

> **Definitie 2.1: Business Process Integration**
>
> *BPI is het mechanisme voor het managen van datastromen en het uitvoeren van bedrijfsprocessen in de juiste volgorde, waarbij processtappen worden ondersteund door applicaties (gebaseerd op [Linthicum 2003]).*

Eenvoudiger gezegd: een BPI oplossing integreert bedrijfsprocessen met applicaties (zie figuur 2.7). Bij de toepassing van software in organisaties wordt gesproken over systemen en applicaties. Beide termen worden als volgt gedefinieerd.

> **Definitie 2.2: Applicatie / systeem**
>
> *Een geheel of arrangement van elementen die zo zijn georganiseerd om door middel van het verwerken van informatie een bepaald doel te bereiken [Pressman 2001].*

BPI zorgt ervoor dat applicaties op de juiste manier worden ingezet om processen te ondersteunen, zoals deze in een organisatie verlopen. Een BPI oplossing vormt dus de koppeling tussen bedrijfsprocessen en applicaties.



Figuur 2.7: Business Process Integration (gebaseerd op [Meekels 2006])

Bij de manier waarop hierbij de communicatie plaatsvindt speelt Enterprise Application Integration een rol, dat in de volgende paragraaf wordt besproken.

Een voorbeeld waar BPI een rol speelt is het order-to-cash proces, dat begint bij het plaatsen van een bestelling door een klant en eindigt bij het ontvangen van de betaling. Dit proces heeft relaties met klantgegevens, logistiek, voorraad en financiën. De BPI oplossing bevat de proceslogica (hoe het proces verloopt) en zorgt dat de juiste applicaties van de betrokken departementen ingezet worden.

2.2.2 Enterprise Application Integration

Enterprise Application Integration (EAI) wordt als volgt gedefinieerd.

> **Definitie 2.3: Enterprise Application Integration**
>
> *Enterprise Application Integration is het onbeperkt delen van informatie tussen twee of meer applicaties in een organisatie. Het is een verzameling technologieën die uitwisseling van informatie verzorgen tussen verschillende applicaties en bedrijfsprocessen in en tussen organisaties [Linthicum 2003].*

Een EAI oplossing vormt de spil tussen verschillende applicaties, die vaak van verschillende leveranciers zijn en daarom niet vanzelfsprekend met elkaar kunnen communiceren. Een service die een processtap ondersteunt kan gebruik maken van meerdere applicaties. Het is in dat geval nodig dat de applicatie die de service levert communiceert met andere applicaties. Hierbij zorgt EAI voor de routering van berichten en de transformatie van de applicatiespecifieke berichten van de ene applicatie naar die van de andere.
Verder worden in applicaties vaak verschillende manieren gebruikt om bepaalde gegevens te representeren, zoals de datum, een werknemer of een product. *Master Data Management* (MDM) richt zich op het stroomlijnen van dergelijke gegevens met behulp van *master data*. Master data is een referentie van essentiële gegevens in een organisatie en wordt vastgelegd in een data model.

2.3 Technische infrastructuur

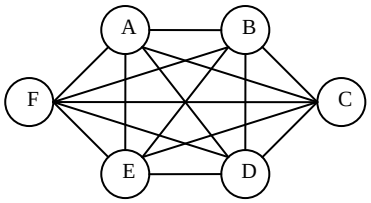
Een EAI oplossing is gebaseerd op een technische infrastructuur die het mogelijk maakt om berichten uit te wisselen tussen applicaties. Een infrastructuur kan op verschillende manieren worden opgebouwd die kunnen worden aangeduid met topologieën (communicatiemodellen).

2.3.1 Topologieën

De verschillende topologieën worden beschreven in termen van een netwerk met netwerkpunten die met elkaar communiceren. Het onderscheid zit in de manier waarop de communicatie plaatsvindt. De volgende topologieën zijn te onderscheiden [Meekels 2006, Shi 2001].

- **Point to point**

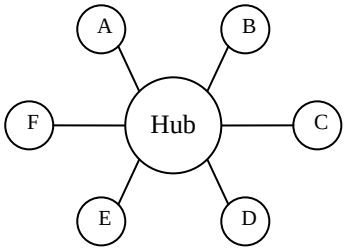
De point to point topologie is gedefinieerd door het verbinden van ieder punt met alle andere punten in het netwerk. Dit levert een zogenaamd 'spaghetti' netwerk op. Deze is gemakkelijk te implementeren, omdat ieder netwerkpunt zelf connecties kan aanleggen met andere netwerkpunten volgens eigen afspraken. Er is geen centrale entiteit die deze connecties beheert. Deze kenmerken maken een dergelijk netwerk moeilijk te onderhouden als er meer netwerkpunten zijn. Bij n netwerkpunten zijn n*(n-1)/2 connecties nodig.



Figuur 2.8: Point to point topologie

- **Hub & spoke**

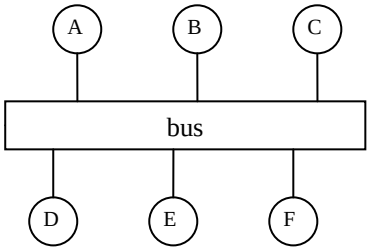
De hub & spoke (ook ster- of broker-) topologie is een structuur met een centrale hub, waarmee ieder netwerkpunt verbonden is. De hub zorgt voor de routering en transformatie van berichten. In de hub wordt dus de gehele integratie geregeld, zodat de netwerkpunten daar geen aandacht aan hoeven te besteden. Nadeel is dat de integratie volledig afhankelijk is van de hub (single point of failure). Bij n netwerkpunten zijn n connecties nodig.



Figuur 2.9: Hub & spoke topologie

- **Bus**

De bus topologie is gebaseerd op een gedistribueerde structuur (de bus) die zorgt voor de communicatie. Een netwerkpunt kan een bericht naar de bus sturen, die vervolgens naar andere netwerkpunten wordt doorgestuurd. Voor de data in de bus wordt gebruik gemaakt van één formaat, waardoor er gemakkelijk een netwerkpunt toegevoegd kan worden. Het belangrijkste verschil met de hub & spoke topologie is dat er niet één centraal punt is die de communicatie verzorgt. De bus omvat meerdere knooppunten, die samen de routering verzorgen. Mocht er een knooppunt uitvallen dan is niet per definitie de volledige communicatie verstoord. Kortom, de bus topologie is schaalbaar en betrouwbaar. Bij n netwerkpunten zijn n connecties nodig.



Figuur 2.10: Bus topologie

2.3.2 Enterprise Service Bus

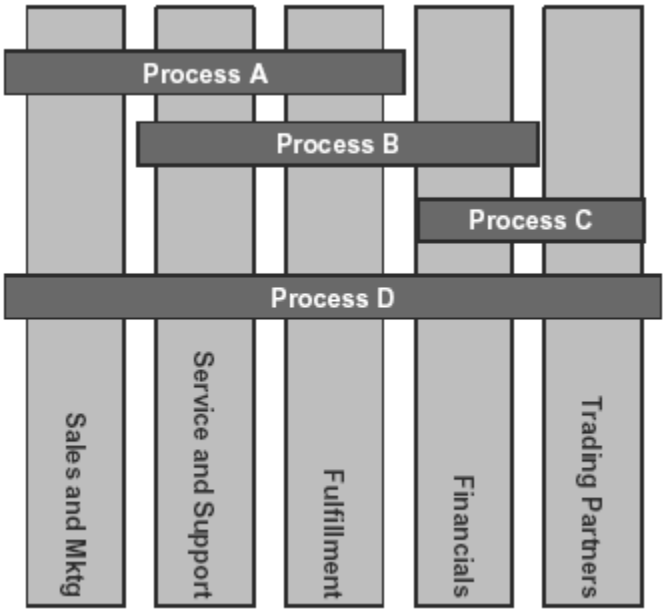
Sinds de opkomst van SOA komen er steeds meer producten op de markt met de naam *Enterprise Service Bus* (ESB). Een ESB is een gedistribueerde infrastructuur die gebruikt wordt voor integratie van applicaties en pretendeert speciaal te zijn toegespitst op SOA. Er bestaat geen algemeen geaccepteerde definitie van een ESB en er zijn dan ook verschillen tussen de producten. Over het algemeen kan gesteld worden dat een ESB gebaseerd is op de bus topologie, waardoor het een schaalbare oplossing is. Er kunnen dus gemakkelijk applicaties toegevoegd worden en nieuwe services beschikbaar worden gesteld. Verder regelt een ESB het berichtenverkeer en de datatransformatie tussen applicaties voor het gebruik van services en maakt gebruik van standaarden. Eventueel bevat een ESB ook een Process Execution Engine, een Rules Engine en een service directory en biedt het de mogelijkheid tot executable modeling (zie hoofdstukken 3 en 5). Een leverancier biedt vaak een platform met een ESB, waaraan allerlei functionaliteiten zijn toegevoegd die relevant zijn voor SOA. Een voorbeeld is Cordys, waar in hoofdstuk 7 op in wordt gegaan.
SOA en ESB worden nogal eens in één adem genoemd, terwijl de verschillen groot zijn. SOA is een paradigma; ESB is echter een infrastructuur die gebruikt kan worden bij het toepassen van SOA en gaat dus over techniek.

3 Bedrijfsprocessen

Zoals in hoofdstuk 2 uitgelegd is, komt de behoefte bij softwareontwikkeling steeds meer voort uit bedrijfsprocessen. In dit hoofdstuk zal dieper worden ingegaan op de achtergrond van bedrijfsprocessen en de manier waarop ze worden gemodelleerd en geëxecuteerd in een organisatie.

3.1 Organisaties, applicaties en processen

Een *bedrijfsproces* (business process) is een verzameling van logisch gerelateerde taken met een gedefinieerd bedrijfsresultaat [Davenport 1990]. Wat een proces onderscheidt van een simpele taak is dat de stappen in een proces uitgevoerd worden door meerdere individuen en applicaties. Hiertussen is afstemming van de uitvoering van processtappen en beheersing van gegevensstromen noodzakelijk. Hoewel processen de dagelijkse werkzaamheden van medewerkers bepalen, zijn organisaties vaak niet georganiseerd rond processen, maar op basis van bedrijfsfuncties. Zo zijn er meestal aparte organisatieonderdelen voor inkoop, financiën en personeel, die hun eigen hiërarchie en applicaties hebben (zie figuur 3.1). Processen lopen echter over de grenzen van functionele domeinen.



Figuur 3.1: Processen en functionele domeinen [Bruce Silver 2005]

Business Process Management is een discipline die gericht is op het denken in termen van bedrijfsprocessen in plaats van op de klassieke invalshoek met functionele domeinen en applicaties. In de volgende paragraaf wordt hier verder op ingegaan. [Bruce Silver 2005]

3.2 Business Process Management

Business Process Management (BPM) wordt als volgt gedefinieerd.

> **Definitie 3.1: Business Process Management**
>
> *Business Process Management is de verzameling van activiteiten en het gebruik van methoden en software om bedrijfsprocessen te ontwikkelen, te beheren en continu te verbeteren.*

Het geheel van activiteiten kan geplaatst worden in een cyclus. Deze BPM cyclus bevat de fasen Strategy, Business Process Modeling, Business Process Execution en Business Process Monitoring.

- **Strategy**

In een organisatie wordt begonnen met het bepalen van de strategie, waaruit een behoefte aan processen voortkomt. Een organisatie kan bijvoorbeeld besluiten om zich te richten op het primaire proces productie en de secundaire processen financiën en marketing uit te besteden. De nadruk komt dan te liggen op het zo efficiënt en effectief mogelijk inrichten van het productieproces. Bepaalde processtappen worden door de organisatie zelf uitgevoerd en kunnen mogelijk geoptimaliseerd worden. Andere stappen worden overgelaten aan experts.

- **Business Process Modeling**

Business Process Modeling is het op geformaliseerde wijze in kaart brengen van een bedrijfsproces. Wanneer je het ontwikkelen van een proces vergelijkt met het ontwikkelen van software, kun je zeggen dat het de ontwerpfase en de implementatiefase omvat. Bij Business Process Modeling lijken de grenzen hiertussen echter te vervagen ten opzichte van traditionele softwareontwikkeling. Toch is het belangrijk een onderscheid te maken tussen analytical modeling en executable modeling, omdat de verschillen ertussen groot zijn en er verschillende toolondersteuning voor bestaat [Bruce Silver 2005].

  - *Analytical modeling*

  Doorgaans wordt eerst een globaal model opgesteld van het proces met behulp van een (vaak grafische) procesnotatie; dit wordt analytical modeling genoemd en kan gezien worden als het ontwerp. Een analytical model is abstract in die zin dat activiteiten niet gebonden zijn aan een specifieke implementatie. De volgende activiteiten kunnen hierbij worden uitgevoerd.

  - Het modelleren van de *process flow* (procesverloop), waarbij de volgorde van processtappen wordt aangegeven met de bijbehorende rol. Er zijn twee soorten processtappen. Bij *interactieve* processtappen vindt interactie met een gebruiker plaats en bij *geautomatiseerde* processtappen wordt een software-element aangeroepen, zonder gebruikersinteractie.
  - Het definiëren van parameters van processtappen, zoals doorlooptijd, input en output.
  - Simulatie van procesexecutie op basis van verschillende scenario's.
  - Analyse van een proces en berekenen van optimale parameters.
  - Documentatie van een proces in een bepaald exporteerbaar formaat.

  - *Executable modeling*

  Na het opstellen van een analytical model wordt het proces geïmplementeerd met een zogenaamde procesexecutietaal; dit wordt executable modeling genoemd en kan gezien worden als de implementatie. In tegenstelling tot analytical modeling worden hierbij de technische details toegevoegd. Voor interactieve stappen bevat dit bijvoorbeeld de specifieke gebruikers en koppelingen naar gebruikersinterfaces. Voor geautomatiseerde stappen gaat het om verwijzingen naar executeerbare software-elementen (bijvoorbeeld een service) en procesdata.

- **Business Process Execution**

Nadat een proces is gemodelleerd wordt het proces in gebruik genomen. Wanneer een proces start wordt een instantie aangemaakt, die wordt geëxecuteerd. Dit wordt aangeduid met *Business Process Execution*. Om een proces te executeren is een runtime omgeving nodig die een *Process Execution Engine* wordt genoemd; dit is een omgeving die een proces uitvoert en indien nodig services aanroept.

- **Business Process Monitoring**

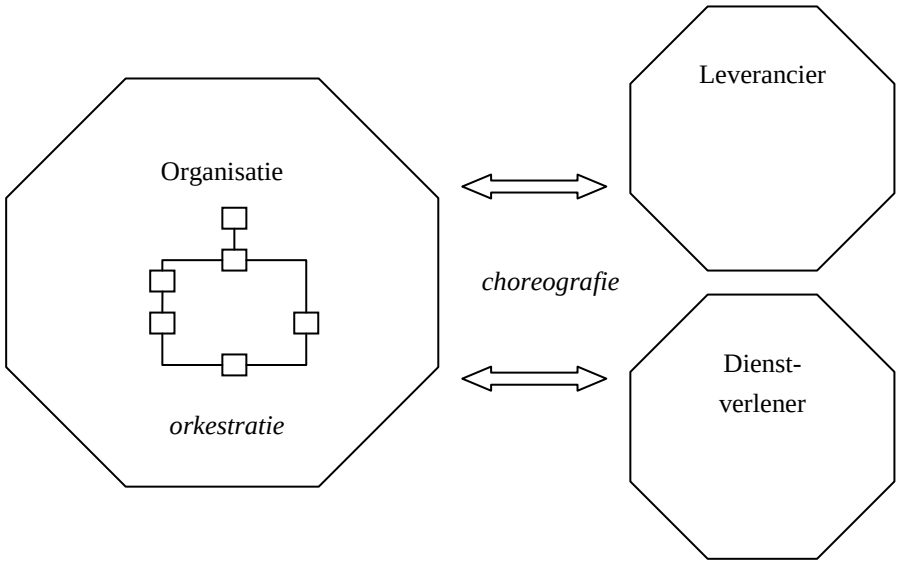
Bij de executie van een proces is het mogelijk dat het verloop gevolgd en gecontroleerd wordt. Informatie over bijvoorbeeld de doorlooptijd van processtappen en het aantal instanties per tijdseenheid kan zo gebruikt worden om processen te optimaliseren. Dit wordt *Business Process Monitoring* genoemd. Afhankelijk van het verloop van een proces kan de strategie of het model worden aangepast, waarna de cyclus opnieuw doorlopen wordt.

3.3 Orkestratie en choreografie

Bij Business Process Modeling in de vorige paragraaf is analytical en executable modeling behandeld. Het gaat daarbij over een proces in een bepaald domein. Wanneer interactie plaatsvindt met een externe partij, moet deze interactie worden afgestemd. Hierbij spelen de termen orkestratie en choreografie een rol, die nu worden toegelicht.

Orkestratie
Orkestratie geeft het verloop van een proces weer vanuit het perspectief van één domein of organisatie. Het geeft de volgorde van processtappen aan en welke services hierbij betrokken zijn.

Choreografie
Choreografie geeft het verloop van een proces weer, gezien vanaf een hoger niveau. Het beschrijft hoe processen in verschillende domeinen of organisaties op elkaar worden afgestemd.



Figuur 3.2: Orkestratie en choreografie

Een voorbeeld is de samenwerking tussen een organisatie, een leverancier en een dienstverlener. De organisaties hebben ieder hun eigen georkestreerde processen. Op één of meerdere momenten tijdens het uitvoeren van die processen zal er interactie plaatsvinden. Deze interactie moet afgestemd worden en dit heet choreografie. Zie figuur 3.2. Choreografie kan ook betrekking hebben op processen binnen een organisatie, zolang het maar om afstemming van interactie tussen verschillende processen gaat.
Orkestratie en choreografie hebben overigens niets te maken met executeerbaarheid door een Process Execution Engine.

3.4 Tools

Voor BPM zijn talloze producten van verschillende leveranciers beschikbaar, die worden aangeduid met *Business Process Management Suites* (BPMSs). Ieder product dekt echter andere delen van de BPM cyclus en heeft hiervoor eigen functionaliteiten. Tools die alleen functionaliteit bieden voor analytical en / of executable modeling worden aangeduid met *Business Process Modeling tools* (BP Modeling tools).
In de case study zal een vergelijking worden gemaakt tussen de BPMSs ARIS en Cordys.

3.5 Procesnotaties en executietalen

In deze paragraaf wordt ingegaan op verschillende manieren om processen te representeren. BPMSs en BP Modeling tools gebruiken verschillende manieren om analytical en executable modeling te ondersteunen. In deze paragraaf wordt uitgelegd welke representaties hiervoor bestaan.
Voor analytical modeling wordt een zogenaamde *procesnotatie* gebruikt, die gemakkelijk te interpreteren is door zowel IT- als businessmensen. Bij executable modeling wordt deze notatie vertaald naar een *procesexecutietaal*, die leesbaar is voor een Process Execution Engine. Met andere woorden, een procesnotatie is de business-representatie van een proces en een procesexecutietaal is de IT-representatie.

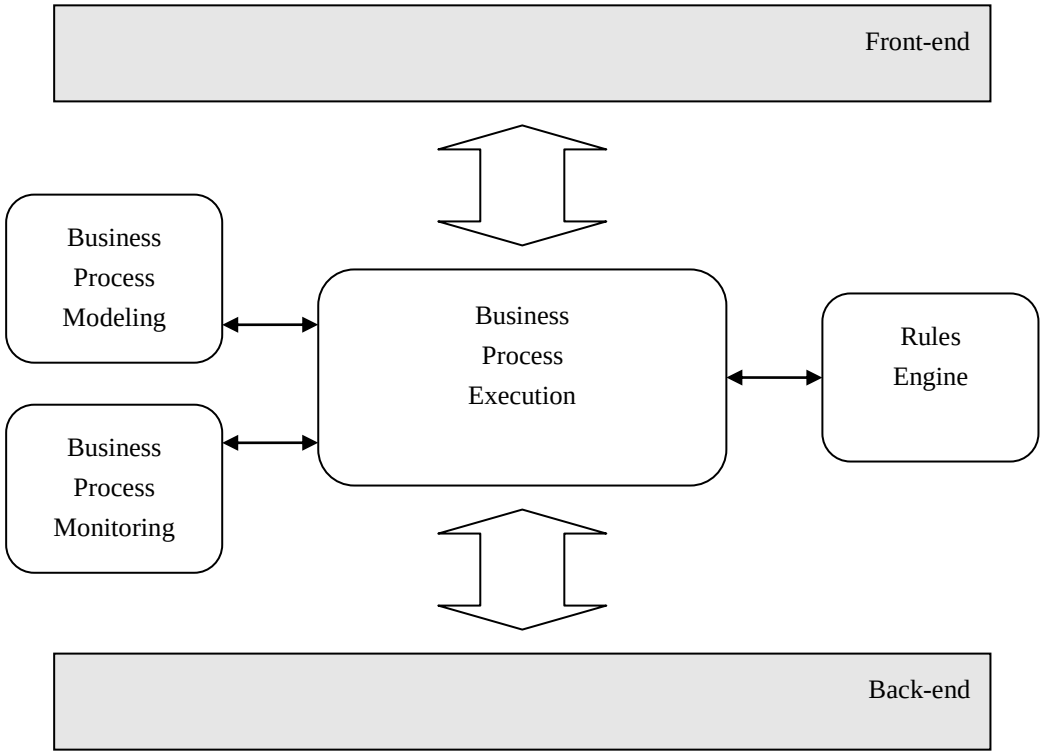
Huidige notaties en talen
Na pogingen van allerlei partijen om een universele notatie of taal te ontwikkelen zijn er nog enkele over (zie bijlage A). BPEL is veruit de meest gebruikte en door leveranciers ondersteunde standaard voor procesexecutie. Procesexecutietaal BPML wordt niet meer onderhouden, maar wordt nog wel gebruikt door Cordys. Voor procesnotatie zijn er meerdere mogelijkheden. BPMN (zowel choreografie als orkestratie) is een standaard die redelijk goed ondersteund wordt door leveranciers. Bovendien is er een transformatie mogelijk naar BPEL en BPML. Verder worden EPC diagrammen gebruikt door het ARIS platform. Vanuit deze notatie is een vertaling mogelijk naar BPEL. WS-CDL, tot slot, is een taal voor het beschrijven van de choreografie van Web Services (een Web Service is een standaard die het gebruik van services over een netwerk ondersteunt, zie bijlage B).

De wereld van notaties en talen is nog erg hard in ontwikkeling. De standaarden worden verbeterd en leveranciers proberen deze steeds beter te ondersteunen. Ook zijn er transformaties mogelijk tussen notaties en talen. Verder bieden leveranciers hun eigen interpretatie en representatie van standaarden, die niet per se volledig en compatibel met elkaar zijn.

3.6 BPI architectuur

Eerder is BPI kort geïntroduceerd. Nu bedrijfsprocessen, BPM en BPMSs zijn behandeld is het tijd om dieper in te gaan op de manier waarop een BPI oplossing processen koppelt aan IT. Dit is namelijk belangrijk voor een goed begrip van de werking van processen in een organisatie. Er is uitgelegd hoe Business Process Modeling, Business Process Execution en Business Process Monitoring met elkaar samenhangen. Verder is het eerst nodig om de term Rules Engine toe te lichten.

Bij Business Process Execution kan gebruik gemaakt worden van een *Rules Engine*, een beheersysteem voor business rules. *Business rules* (bedrijfsregels) leggen voorwaarden vast waaronder een bedrijf opereert. Ze bepalen welke keuzes gemaakt worden bij de uitvoering van processen. Een voorbeeld is dat je bij een bepaalde winkel een gekocht artikel alleen binnen 7 dagen mag ruilen met aankoopbon. Een ander voorbeeld is dat je bij een online bestelling 10% korting krijgt op de reguliere winkelprijs. Het kan verstandig zijn om business rules vast te leggen in een Rules Engine. Wanneer een processtap wordt uitgevoerd, waarbij een business rule betrokken is, wordt tijdens de executie een beslissing gemaakt door de Rules Engine. Het alternatief is dat business rules verwerkt zijn in de implementatie van services. Dit maakt een proces echter minder flexibel, omdat bij een verandering gesleuteld moet worden aan de implementatie van een service en omdat services hierdoor minder herbruikbaar zijn.



Figuur 3.3: BPI architectuur (gebaseerd op [Meekels 2006])

In figuur 3.3 zijn de genoemde begrippen weergegeven. Business Process Modeling is nodig voor het ontwerp en de implementatie van processen en speelt bij de procesexecutie dus geen directe rol. Wel kan men met behulp van Business Process Monitoring informatie over procesexecutie koppelen aan een analytical model, om het verloop van een proces per processtap weer te kunnen geven. Overigens zijn Business Process Monitoring en het gebruik van een Rules Engine niet noodzakelijk om een proces uit te voeren.
De *front-end* en de *back-end* zijn een abstractie voor applicaties die met gebruikers interacteren respectievelijk de achterliggende applicaties en databases.

4 Architectuur

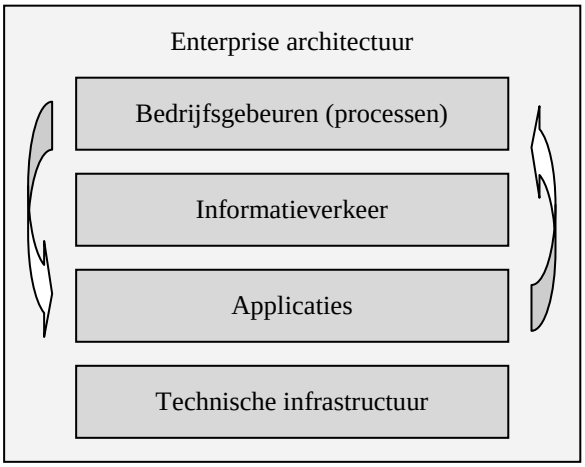
Om een beter beeld te krijgen van SOA in een organisatie zal in dit hoofdstuk worden ingegaan op architectuur. Er wordt een verdeling in architectuurlagen voorgesteld en er wordt hierbij een onderscheid gemaakt tussen designtime en runtime.

4.1 Architectuurraamwerk

Een breed geaccepteerde definitie van architectuur wordt gegeven door het IEEE[1].

> **Definitie 4.1: Architectuur**
>
> *Architectuur is de fundamentele organisatie van een systeem, uitgedrukt in zijn componenten, hun relaties met elkaar en met de omgeving en de principes die ontwerp en evolutie leiden [IEEE 2000].*

Er moet voor de volledigheid bij opgemerkt worden dat het architectuur betreft in de digitale wereld en niet in de fysieke wereld (zie [Rijsenbrij 2004]). Architectuur wordt in een organisatie gebruikt om structuur aan te brengen in de IT-huishouding, samenhang te creëren en complexiteit te reduceren. In de loop der jaren is in organisaties veelal een wildgroei ontstaan aan processen en applicaties. Architectuur kan een verhelderend hulpmiddel zijn, door de samenhang in kaart te brengen en principes op te stellen. Het kan als abstractie en als referentie dienen om de 'neuzen dezelfde richting op te krijgen' en te zorgen dat medewerkers in verschillende organisatieonderdelen toch volgens dezelfde principes werken. Als abstractiemiddel wordt doorgaans een raamwerk gebruikt om verschillende componenten te kunnen onderscheiden. In figuur 4.1 is een eenvoudig raamwerk zichtbaar met vier architectuurlagen, dat is gebaseerd op [Rijsenbrij 2004]. Een architectuurlaag bevat soortgelijke componenten en er bestaan relaties tussen de lagen.



Figuur 4.1: Architectuurraamwerk

De vier lagen representeren vier werelden, te weten het bedrijfsgebeuren, het informatieverkeer, de applicaties en de technische infrastructuur [Rijsenbrij 2004].

[1] IEEE (the Institute of Electrical and Electronics Engineers Inc.) is een internationaal instituut voor technici en wetenschappers (http://www.ieee.org).

De wereld van het bedrijfsgebeuren is die van het zakendoen van een organisatie en die van het aansturen van mensen en bedrijfsmiddelen. Het gaat om het produceren van producten of het aanbieden van diensten en de bedrijfsprocessen die hiervoor worden uitgevoerd.
In de wereld van het informatieverkeer bevinden zich informatiestromen, de documentstromen, de informatiebehoeftes, de informatiebronnen en de informatie-uitwisseling met de buitenwereld. Het is van groot belang dat de juiste mensen op tijd op de hoogte zijn van de juiste informatie. Dit geldt voor bestuurders, maar ook voor bijvoorbeeld medewerkers die informatie kunnen delen met geïnteresseerde collega's.
De applicatiewereld omvat alle applicaties en integratiemechanismen in een organisatie. De technische infrastructuur, tot slot, bevat de fundering voor de applicaties. Dit betreft hardware, netwerken, besturingssystemen en gemeenschappelijke software, zoals voor tekstverwerking.

Idealiter zouden de vier werelden in harmonie moeten zijn. Wanneer op bedrijfsniveau een verandering plaatsvindt kan dit gevolgen hebben voor de andere niveaus. De samenhang tussen de lagen mag niet uit het oog worden verloren, ofwel het geheel kan worden gezien als een holistisch systeem. Het geïntegreerde geheel van de vier werelden wordt aangeduid met *enterprise architectuur*.
In figuur 4.1 wordt met pijlen aangegeven dat er afstemming bestaat tussen het bedrijfsgebeuren en de applicaties. Deze relatie is specifiek relevant, omdat SOA een paradigma is waarin services, die door applicaties worden aangeboden, stappen uit processen ondersteunen in het bedrijfsgebeuren.

De verdeling in vier werelden heeft een hoog abstractieniveau en is vrij algemeen toepasbaar. Wanneer men naar SOA kijkt is een specifiekere verdeling in architectuurlagen meer geschikt, zodat een goed beeld kan worden verkregen van verschillende componenten en hoe ze zich tot elkaar verhouden. In de volgende paragraaf wordt hierop ingegaan.

4.2 Architectuurlagen

Wanneer men SOA toepast in een organisatie is het van groot belang dat eenieder begrijpt welke componenten erbij betrokken zijn, hoe ze samenhangen en welke rol ze spelen. Aan de hand van ervaringen en inzichten opgedaan bij Essent wordt een verdeling in 7 architectuurlagen voorgesteld die specifiek geschikt is voor SOA. De lagen, zichtbaar in figuur 4.2, worden één voor één besproken.

- **Gebruikerslaag**

De gebruikerslaag bevat componenten die te maken hebben met interactie met gebruikers, bijvoorbeeld door middel van GUI's (Graphical User Interfaces). Gebruikers bepalen hoe het proces doorlopen wordt, afhankelijk van keuzes die zij maken. Niet bij iedere processtap hoeft een gebruiker betrokken te zijn, maar wanneer dit het geval is kunnen er verschillende manieren zijn om gegevens te presenteren en / of input te vragen.

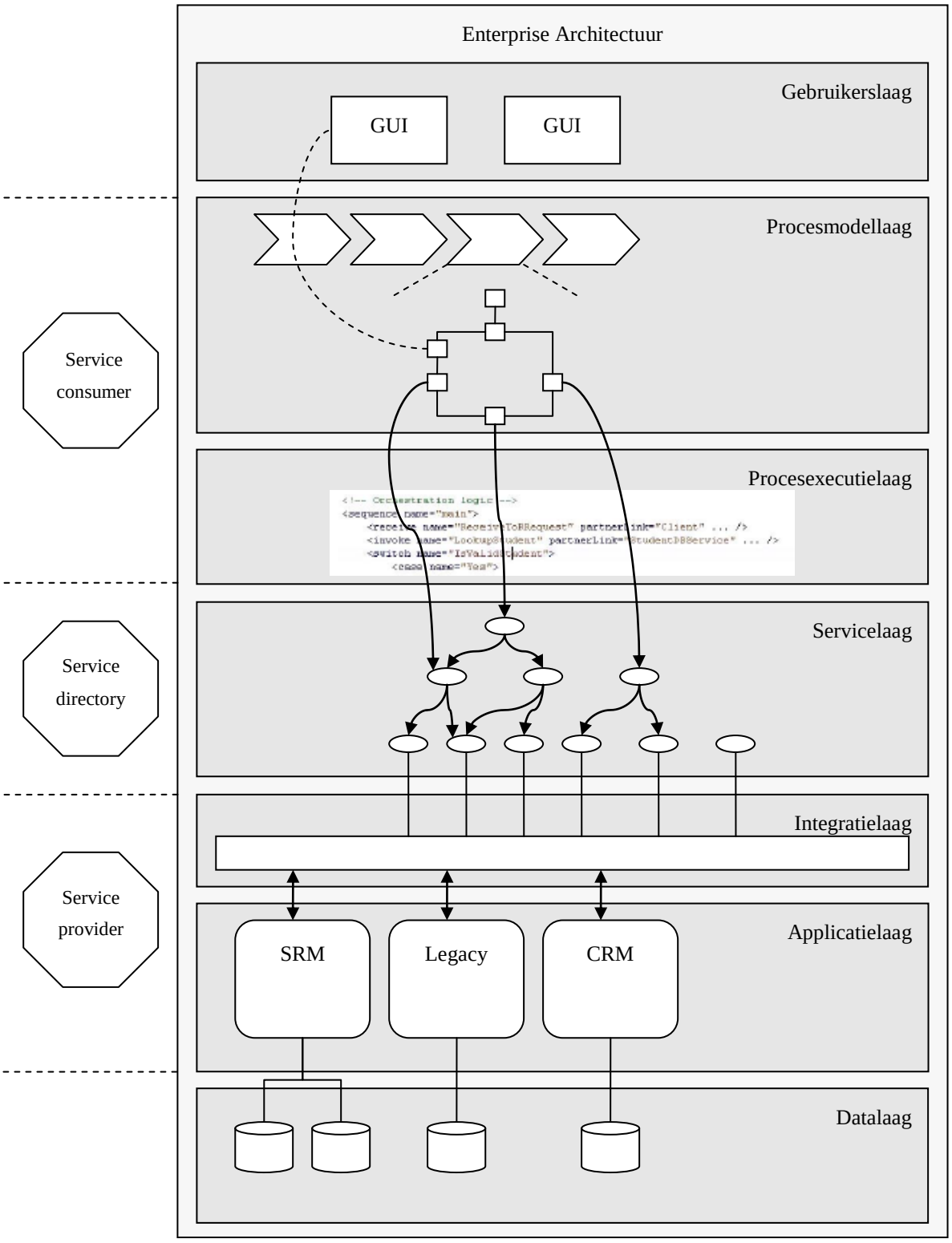
- **Procesmodellaag**

Een proces bestaat uit verschillende processtappen die in een bepaalde volgorde uitgevoerd moeten worden. Hiervan kan op hoog niveau een model worden gemaakt (een analytical model, met een procesnotatie); het is een vertaling van een proces zoals dit in de organisatie verloopt, dus niet de technische weergave. Dit vormt de procesmodellaag.

- **Procesexecutielaag**

Uiteindelijk moet een model uit de procesmodellaag vertaald worden naar uitvoerbare procestaal (een executable model, met procesexecutietaal). Dit is de technische weergave van een proces en valt in de procesexecutielaag.

De Process Execution Engine en de Rules Engine, zoals besproken in paragraaf 3.6, bevinden zich ook in deze laag.



Figuur 4.2: 7 lagen architectuur

- **Servicelaag**

De servicelaag omvat de services die gebruikt worden bij de procesexecutie. In deze laag kan een hiërarchie van services gecreëerd worden. Services kunnen zo worden gebouwd dat ze precies een processtap ondersteunen, maar er kunnen ook services zijn met een geringere granulariteit (zie hoofdstuk 5), die andere services ondersteunen. In de servicelaag bevindt zich een service directory (zie hoofdstuk 5), maar de daadwerkelijke uitvoering van services vindt plaats in de applicatielaag.

- **Integratielaag**

De integratielaag zorgt voor integratie tussen de verschillende applicaties (EAI). Vanuit deze laag kunnen services worden aangeboden in de servicelaag.

- **Applicatielaag**

In de applicatielaag bevinden zich de verschillende applicaties, waar de uitvoering van de services plaatsvindt.

- **Datalaag**

In de datalaag bevinden zich databases met gegevens, waar de applicaties gebruik van maken.

4.2.1 Architectuurlagen en SOA

In het voorgaande zijn de 7 lagen en hun samenhang beschreven. Het geïntegreerde geheel van de 7 lagen wordt weergegeven met enterprise architectuur (zie figuur 4.2). Het betreft de relaties tussen gebruikers, processen, services, applicaties en data.

De 7 lagen architectuur is specifiek geschikt voor SOA. Allereerst omdat de servicelaag centraal staat en hiervan het belang aangeeft. Services zijn essentieel voor SOA en moeten dan ook goed zichtbaar zijn in de architectuur. Een service directory moet ervoor zorgen dat functionaliteit beschikbaar en vindbaar is. De servicelaag staat óók centraal in de lagenverdeling, omdat het de processen ontkoppelt van de applicaties (ontkoppeling, zie hoofdstuk 5). Wanneer een behoefte bestaat in een proces, kan men zoeken naar een service die hierin voorziet. Enerzijds beschikt een service over informatie over beschikbare functionaliteit en anderzijds over informatie over welke applicatie deze levert. De manier waarop services gebruikt kunnen worden is echter onafhankelijk van de implementatie.
Verder is de integratielaag zeer belangrijk als intermediair naar en tussen de applicaties. In een complexe omgeving met veel verschillende bedrijfsapplicaties is integratie onmisbaar. Wanneer men een service wil gebruiken kan dit via de integratielaag, die daarna de koppeling verzorgt met de applicaties.
Boven de servicelaag bevinden zich de procesmodel- en procesexecutielaag. Het is essentieel deze lagen te scheiden, omdat er verschillende activiteiten in plaatsvinden, mogelijk met verschillende toolondersteuning. Voordat men een proces modelleert wordt de strategie bepaald en wordt op het niveau van het bedrijfsgebeuren een plan gemaakt. Hierbij komt in eerste instantie nog geen IT kijken. Om deze reden is het dan ook goed om pas later de ondersteuning van IT te beschouwen in plaats van zich erdoor te laten leiden.

Tot slot kan het volgende opgemerkt worden. SOA is een paradigma dat interessant is wanneer het wordt toegepast in een complexe omgeving. In een klein bedrijf met twee applicaties die gegevens uitwisselen is het niet nodig om SOA toe te passen. Door de eenvoudige situatie is er in een oogopslag overzicht over de gehele IT-huishouding en zal SOA het geheel alleen onnodig ingewikkeld maken. In een grote organisatie, met veel applicaties van verschillende leveranciers en met complexe processen, kan SOA wel meerwaarde opleveren.

4.3 Designtime versus runtime

Om de rol van architectuurlagen beter toe te kunnen lichten wordt een onderscheid gemaakt tussen designtime (ontwerp en implementatie van een proces) en runtime (daadwerkelijk executeren van een instantie van een proces).

4.3.1 Designtime

In designtime wordt een proces gemodelleerd en geïmplementeerd. Het modelleren valt in de procesmodellaag en implementeren valt in de procesexecutielaag. Bij de implementatie worden processtappen gekoppeld aan services. De services worden gebouwd in de applicatielaag en aangeboden in de servicelaag. De integratielaag verzorgt de koppeling tussen de servicelaag en de applicatielaag.
Verder kunnen GUI's worden gerealiseerd (gebruikerslaag) en kunnen applicaties en hun achterliggende gegevenssystemen worden gebouwd of aangepast (applicatielaag en datalaag).

In figuur 4.2 zijn ook de drie entiteiten uit het SOA model (zie hoofdstuk 5) weergegeven. Een service consumer bevindt zich in de procesmodel- en procesexecutielaag. In de servicelaag is een service directory beschikbaar, waarin tijdens designtime gezocht kan worden. Een service kan ook tijdens designtime geregistreerd worden bij de service directory. Verder bevinden service providers zich in het geheel van de integratie- en applicatielaag.

4.3.2 Runtime

In runtime wordt daadwerkelijk een instantie van een proces uitgevoerd in de procesexecutielaag. Hierbij worden de verschillende processtappen uitgevoerd en worden services gebruikt uit de servicelaag. De uitvoering van de services vindt plaats in de applicatielaag, met behulp van de datalaag. De koppeling wordt hierbij verzorgd door de integratielaag. De interactie met gebruikers gebeurt in de gebruikerslaag.

Met het oog op het SOA model (zie hoofdstuk 5) kun je zeggen dat de interactie tussen de service participanten in runtime gebeurt. Er zijn zelfs al toepassingen om ook het zoeken in een service directory runtime te laten plaatsvinden. Een voorbeeld is het runtime zoeken naar een printer, het ophalen van de service description en het uitvoeren van een printopdracht.

5 SOA

In dit hoofdstuk wordt SOA eerst benaderd vanuit een theoretisch oogpunt. Vervolgens wordt het paradigma in een organisatiecontext geplaatst, waarna enkele belangrijke definities worden gegeven. Verder wordt ingegaan op verwachte voor- en nadelen van SOA.

5.1 Theoretische benadering

Om later in te kunnen gaan op de toepassing is het eerst van belang om SOA te benaderen vanuit een theoretisch oogpunt. Deze is voor een deel gebaseerd op [OASIS 2006]. In eerste instantie wordt juist niet op de toepassing van SOA ingegaan, omdat in de meeste literatuur al impliciet aannames worden gedaan over beschikbare technische mogelijkheden die SOA ondersteunen.

OASIS[2] heeft een zogenaamd referentiemodel opgesteld, waarin de relaties tussen de relevante begrippen rondom SOA gedefinieerd zijn. Dit model wordt in de literatuur vrij algemeen beschouwd als uitgangspunt. OASIS definieert SOA op zeer abstracte wijze.

OASIS
Service Oriented Architecture is een paradigma voor het organiseren en gebruiken van gedeelde vermogens die beheerd kunnen worden door verschillende domeinen [OASIS 2006].

Omdat deze definitie erg algemeen is opgesteld is deze geschikt als uitgangspunt. Er wordt impliciet aangenomen dat er gedeelde vermogens zijn in verschillende domeinen. Om hierop voort te bouwen worden eerst definities gegeven van de begrippen entiteit, vermogen en behoefte.

Definitie 5.1: Entiteit

Een entiteit is 'iets dat wezenlijk bestaat' (Van Dale); in dit onderzoek wordt gedoeld op een persoon, organisatie(deel) of systeem(deel).

Definitie 5.2: Vermogen

Een vermogen is dat waartoe een entiteit in staat is, een bekwaamheid.

Definitie 5.3: Behoefte

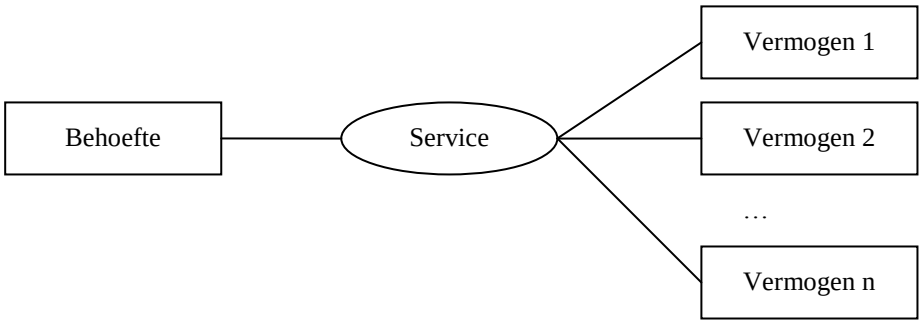
Een behoefte is iets dat een entiteit nodig heeft.

Een entiteit kan over vermogens beschikken om problemen op te lossen. Idealiter worden behoeftes die in een organisatie bestaan gedekt door de aanwezige vermogens om deze op te lossen. De waarde van SOA wordt gezien in het leveren van een krachtig raamwerk om behoeftes en vermogens op elkaar af te stemmen.

[2] OASIS (Organization for the Advancement of Structured Information Standards) is een internationaal consortium dat standaarden ontwikkelt voor e-business. Er zijn ruim 600 organisaties in vertegenwoordigd (http://www.oasis-open.org).

5.1.1 Model

Bij SOA staan services centraal. De relatie tussen enerzijds behoeftes en vermogens en anderzijds services is als volgt. Een service is een mechanisme dat toegang geeft tot een of meerdere vermogens. Een entiteit met een behoefte kan een service aanroepen, die vervolgens gebruik maakt van een of meerdere vermogens. Een service brengt vermogens en behoeftes samen (zie figuur 5.1). Dit betekent dus dat SOA zelf *geen* (elementen van) oplossingen bevat voor problemen, maar de koppeling verzorgt tussen vermogens en behoeftes.



Figuur 5.1: Een service verbindt behoefte met vermogen(s)

Een entiteit die een service aanbiedt wordt een *service provider* genoemd. Deze entiteit kan zelf over een vermogen beschikken en deze tegelijkertijd aanbieden in een service. Het kan echter ook voorkomen dat de service gebruik maakt van een of meerdere vermogens van andere entiteiten. De service provider beschikt dus niet per definitie over een vermogen. Degene met een behoefte die gebruik maakt van een service wordt een *service consumer* genoemd. Service providers en service consumers worden gezamenlijk aangeduid met *service participanten*.

Definitie 5.4: Service provider

Een service provider is een entiteit die een service levert.

Definitie 5.5: Service consumer

Een service consumer is een entiteit die een service gebruikt.

In de praktijk is het niet relevant of een service provider zelf over een vermogen beschikt of hiervoor gebruik maakt van andere entiteiten. Een service consumer is namelijk niet geïnteresseerd in de werking van de service, maar alleen in het resultaat. Voor de volledigheid wordt het hier wel vermeld.

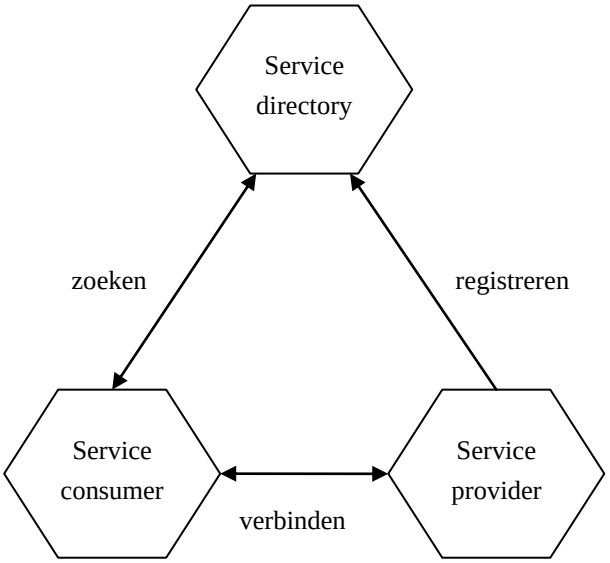
Om services vindbaar te maken voor entiteiten uit verschillende domeinen kan een service directory (ook service registry, service repository of service catalog genoemd) worden aangelegd. Daarin is van iedere service een *service description* te vinden, waarin vermeld staat wat de service doet en hoe de interactie plaatsvindt. Een service consumer kan zo zoeken en kiezen welke service mogelijk aan een behoefte voldoet.

Definitie 5.6: Service directory

Een service directory is een verzameling van service descriptions, waarin service consumers kunnen zoeken.

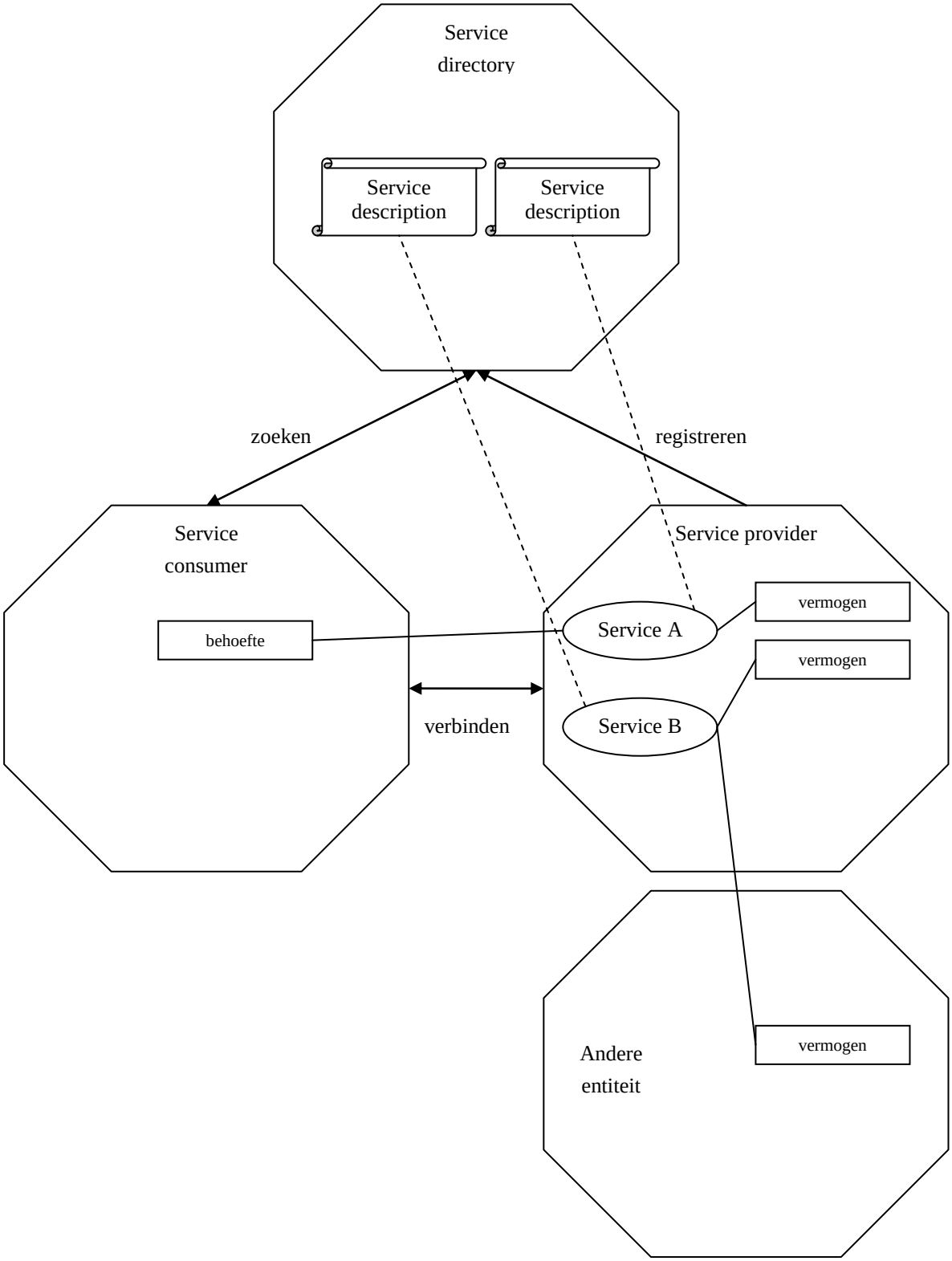
Een service provider kan een service aanroepen met een of meer effecten tot gevolg. Dit kan het leveren van informatie zijn, de verandering van de staat van een entiteit of een combinatie daarvan. Later zal dieper op genoemde termen worden ingegaan.

In figuur 5.2 wordt een model van SOA weergegeven, waarin de relatie tussen de zojuist geïntroduceerde termen zichtbaar is. Een service provider registreert de service bij de service directory. Een service consumer kan zoeken in de directory en in de service descriptions lezen of een service mogelijk geschikt is om een behoefte te vervullen. Een service description biedt ook de informatie om een service te gebruiken. Op basis hiervan kan een service consumer een service aanroepen, waarbij interactie plaatsvindt met de service provider.



Figuur 5.2: SOA model [Specht 2005]

In figuur 5.3 staat het voorgaande in zijn geheel weergegeven. De service participanten en de service directory interacteren zoals weergegeven in figuur 5.2.
De service consumer heeft een behoefte. De service provider heeft meerdere vermogens en biedt deze aan in de vorm van een service. Een van de services maakt ook gebruik van een vermogen van een andere entiteit. Van beide services is een service description beschikbaar in de service directory, die gevonden kan worden door de service consumer.



Figuur 5.3: Compleet SOA model

5.1.2 Voorbeeld

Om een wat concreter beeld te krijgen wordt nu een voorbeeld gegeven van een situatie waar gebruik wordt gemaakt van het SOA paradigma. Het is met opzet geen IT-voorbeeld om de algemene toepasbaarheid weer te geven.

De situatie betreft het boeken van een vakantie. Het verhaal begint met een vakantieganger (service consumer), die graag op vakantie wil (behoefte). De vakantieganger besluit een vakantie te gaan boeken (service: vakantie boeken) en gaat op zoek naar een entiteit die de service kan leveren, een reisbureau (service provider). Een reisbureau kan een reis boeken, een locatie vastleggen, vervoer ter plaatse regelen en dagtripjes boeken (vermogens). Vereenvoudigd weergegeven kan een vakantie worden geboekt met (een combinatie van) deze vier vermogens.
Eerst gaat de vakantieganger op zoek naar een reisbureau, bijvoorbeeld door te zoeken in de Gouden Gids (service directory). Hieruit wordt een reisbureau geselecteerd, die benaderd wordt om een vakantie te boeken (service aanroep).

5.1.3 Service concepten

Er is al globaal ingegaan op services met betrekking tot behoeftes en vermogens. Hier wordt verder ingegaan op enkele belangrijke concepten en eigenschappen. Er zijn drie fundamentele concepten die belangrijk zijn om te begrijpen hoe de interactie met services plaatsvindt: de zichtbaarheid tussen service providers en service consumers, de interactie ertussen en het effect van de interactie.

Zichtbaarheid
Voor de interactie tussen service participanten is het essentieel dat zij elkaar kunnen zien of vinden. Bij bijvoorbeeld object georiënteerd programmeren kan men zoeken in libraries en zijn de beschikbare onderdelen bekend. Bij SOA is dit niet per se het geval. Juist omdat er over grenzen van domeinen en organisaties heen gekeken kan worden is het niet evident dat service participanten elkaar kunnen zien. Er zijn drie voorwaarden waar aan voldaan moet zijn om zichtbaarheid mogelijk te maken.

- *Bewustzijn*. Een service consumer moet zich bewust zijn van het bestaan en de toepassing van een service provider. Dit kan bijvoorbeeld met behulp van een service description en registratie bij een service directory.
- *Bereidheid*. De service participanten moeten beiden bereid zijn tot interactie. Bij bereidheid gaat het erom of de interactie plaatsvindt volgens de afspraken (deze worden vastgelegd in de service description).
- *Bereikbaarheid*. De service participanten moeten de mogelijkheid hebben met elkaar te communiceren.

Interactie
De interactie tussen service participanten vindt plaats door het uitwisselen van berichten. Voor de wijze waarop dit gebeurt zijn de volgende begrippen relevant.

- *Informatiemodel*. Het informatiemodel karakteriseert de informatie die uitgewisseld wordt. Het gaat hierbij om de syntax en semantiek. Syntax zegt iets over de representatie of structuur van informatie en berichten. Semantiek zegt iets over de interpretatie of de betekenis die eraan gegeven wordt.
- *Gedragmodel*. Het gedragmodel karakteriseert de wijze van interactie tussen service participanten. Het bevat de acties die een service kan uitvoeren en de manier waarop berichten worden uitgewisseld, zoals een protocol.

Effect
Zoals al eerder aangegeven heeft een service als effect het leveren van informatie, de verandering van de staat van een entiteit of een combinatie daarvan. In het voorbeeld over het boeken van een vakantie kun je zeggen dat het tonen van beschikbare vluchten een effect is, namelijk het leveren van informatie. Het daadwerkelijk boeken van een vlucht is ook een effect en wel de verandering van de staat van de vluchtgegevens van de betreffende vliegmaatschappij.

Overige relevante begrippen met betrekking tot SOA zijn als volgt.

Service description
Een service description bevat kritieke informatie die een service consumer nodig heeft om te beslissen of deze een service kan gebruiken. Er bestaat niet één juiste invulling van een service description; de volgende punten dient het in ieder geval te bevatten.

1. Dat de service bestaat en bereikbaar is;
2. Dat de service een bepaalde functie of set van functies uitvoert;
3. Dat de service opereert onder een gespecificeerde set van voorwaarden;
4. Hoe interactie plaatsvindt (de service interface).

Service interface
De service interface is het middel voor interactie met een service. Het maakt onderdeel uit van de service description en bevat de specifieke protocollen, commando's en informatie-uitwisseling, waarmee acties worden uitgevoerd die in een effect resulteren. Voor SOA is het essentieel dat interfaces worden beschreven in dezelfde syntax. Deze staat gedefinieerd in het informatiemodel.

Ontkoppeling
Voor een optimale onafhankelijkheid van platform en taal is het van belang om de implementatie te scheiden van de service interface. Dit wordt *ontkoppeling* genoemd ('loose coupling'). Een service consumer hoeft nu alleen te kunnen communiceren volgens de standaarden voor interactie, zonder dat deze te maken heeft met de implementatie van een service. De processen, die de services gebruiken, worden zo ontkoppeld van de techniek, ofwel de applicaties die de services aanbieden.
Ontkoppeling kan ook betrekking hebben op de context, waarbij de service zo min mogelijk afhankelijk is van zijn context om breed toepasbaar te zijn, en op de locatie, waarbij het gebruik onafhankelijk is van de locatie van de service.

Granulariteit
Granulariteit is de mate van detaillering en de omvang van de functionaliteit van een service.

5.2 Definities

Tot op dit punt is er besproken hoe een service behoeftes en vermogens verbindt, is de relatie tussen service consumer, provider en directory aangegeven met behulp van een model en zijn enkele relevante concepten rond het begrip service uitgelegd. Nu wordt het tijd naar definities van service en SOA toe te werken.

De benadering van OASIS is een goede basis, maar is voor dit onderzoek te weinig contextspecifiek. Wanneer je SOA beschouwt in een organisatiecontext komen behoeftes voort uit onderdelen van bedrijfsprocessen en zijn vermogens er om in deze behoefte te voorzien. Een service ondersteunt dus een (deel van een) processtap. De nu volgende definities worden geplaatst in een organisatiecontext.

5.2.1 Service

Zoals aangegeven verschaft een service toegang tot vermogens om in een behoefte te voorzien. Voor het SOA paradigma is dit van belang, om aan te kunnen geven dat het gericht is op het koppelen van vermogens en behoeftes. De entiteit die het vermogen levert hoeft dus niet dezelfde te zijn als de entiteit die de service uiteindelijk biedt. Voor het gemak wordt aangenomen dat een service zelf over vermogens beschikt en deze aanbiedt in een service.

In het onderzoek wordt gebruik gemaakt van een definitie van de Open Group[3].

Definitie 5.7: Service

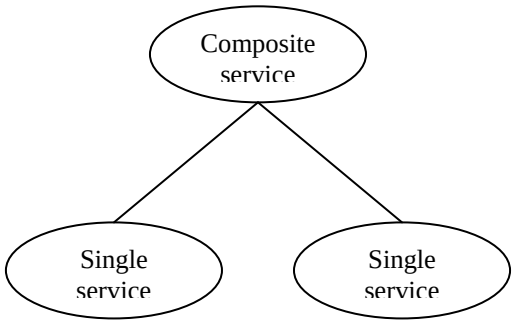
Een service is een logische representatie van een herhaalbaar bedrijfsproces, dat een specifieke uitkomst heeft. Typische kenmerken van een service zijn:

- *Een service opereert zelfstandig en is onafhankelijk (self-contained);*
- *Een service kan zijn opgebouwd uit andere services of kan hiervan gebruik maken;*
- *Een service is een 'black box' voor gebruikers van de service.*

[Open Group 2006]

In deze definitie wordt niet gerept over behoeftes en vermogens, maar wordt impliciet aangenomen dat een service vermogens gebruikt voor een behoefte die voortkomt uit een bedrijfsproces. Deze definitie is dan ook bruikbaar voor een organisatiecontext. Belangrijk is ook dat het gaat om een herhaalbaar bedrijfsproces; dit benadrukt de mogelijkheid tot hergebruik dat voor SOA essentieel is. Verder heeft een service een specifieke uitkomst, zoals eerder behandeld bij het effect van een service.
De zelfstandigheid en onafhankelijkheid beklemtonen dat een service een los geheel is, met zo min mogelijk afhankelijkheden van andere services en van techniek. Zelfstandigheid wil ook zeggen dat een gebruiker zich geen zorgen hoeft te maken over de meest recente versie van de service. De service provider draagt zorg voor het beheer.
Verder kan een service andere services gebruiken en werkt hierbij als 'black box' voor gebruikers.

Dit laatste kenmerk vraagt nog verdere toelichting. Wanneer een service gebruik maakt van een of meer andere services wordt deze een *composite service* genoemd. Anders heet het een *single service* (zie figuur 5.4). Dit onderscheid heeft overigens niets met het aantal vermogens te maken. Zo zijn in figuur 5.3 service A en B beide een single service.



Figuur 5.4: Composite en single services

5.2.2 SOA

Voordat een eigen definitie wordt gegeven is het eerst van belang twee invalshoeken van SOA te belichten.

- *SOA als paradigma.* Hierbij wordt SOA gezien als een concept dat toegepast kan worden op een organisatie.

[3] De Open Group is een internationaal consortium dat onafhankelijk is van leveranciers en technologie en open standaarden ontwikkelt voor informatie-integratie binnen en tussen bedrijven (http://www.opengroup.org).

- *SOA als architectuur of architectuurstijl.* Hierbij wordt SOA gebruikt als aanduiding van de architectuur van een organisatie met bepaalde kenmerken (een service georiënteerde architectuur).

Beide invalshoeken worden gebruikt in de literatuur, maar het is belangrijk om te beseffen welke wordt bedoeld. De invalshoeken worden vaak door elkaar gebruikt zonder een onderscheid te maken. Dit kan verwarring veroorzaken, aangezien SOA geen algemeen geaccepteerde definitie heeft.

Geen van de invalshoeken is goed of fout, maar de paradigmabenadering doet meer recht aan de betekenis van SOA. Fundamenteel gezien is het namelijk een concept voor het samenbrengen van behoeftes en vermogens. In organisatiecontext wordt dit gerealiseerd door services die bedrijfsprocessen ondersteunen. In het onderzoek wordt dan ook uitgegaan van de paradigma-invalshoek.

Om naar een eigen definitie toe te werken volgen eerst enkele definities van relevante autoriteiten.

Open Group
SOA is een architectuurstijl die service oriëntatie ondersteunt.
- Service oriëntatie is een manier van denken in termen van services en servicegebaseerde ontwikkeling en het resultaat van services.
- Een service is een logische representatie van een herhaalbaar bedrijfsproces, dat een specifieke uitkomst heeft.
Typische kenmerken van een service zijn:

- Een service opereert zelfstandig en is onafhankelijk (self-contained).
- Een service kan zijn opgebouwd uit andere services of kan hiervan gebruik maken.
- Een service is een ‘black box’ voor gebruikers van de service.

- Een architectuurstijl is de combinatie van kenmerkende eigenschappen, waarmee architectuur tot uitdrukking komt.
[Open Group 2006]

Object Management Group[4] (OMG)
SOA is een architectuurstijl voor een gemeenschap van service providers en consumers om wederzijdse waarde te creëren, die:
- service participanten de mogelijkheid biedt om samen te werken met minimale afhankelijkheid van elkaar en van techniek;
- de contracten specificeert waar organisaties, mensen en technologie aan moeten voldoen om te mogen deelnemen in de gemeenschap;
- bedrijfswaarde oplevert en het mogelijk maakt bedrijfsprocessen te realiseren door de gemeenschap;
- het toestaat om verscheidene technologieën te gebruiken voor de interactie in de gemeenschap.
[OMG 2006]

Deze twee definities spreken van SOA als een architectuurstijl. Toch komt in de definitie van de Open Group ook de paradigmabenadering terug, namelijk ‘een manier van denken’. Deze benadering zou terug moeten komen in een definitie, omdat bij toepassen van SOA het gaat om het realiseren van een visie op een organisatie. Het overstijgt de architectuur en architectuurstijl.

[4] De Object Management Group is een internationaal consortium dat standaarden ontwikkelt voor de computer industrie. Er zijn honderden bedrijven in vertegenwoordigd uit allerlei branches (http://www.omg.org).

De definitie van de Open Group bestaat uit de termen service oriëntatie, service en architectuurstijl. De OMG benadrukt de samenwerking tussen service participanten in een gemeenschap. Beide definities zijn op zich bruikbaar. Een aparte definitie van service is echter beter voor de leesbaarheid. Verder is het in de definitie van de OMG onduidelijk wat er met 'bedrijfswaarde opleveren' wordt bedoeld. Beide noemen daarnaast onafhankelijkheid van services, waar ontkoppeling en het gebruik van standaarden ontbreken. Een zelf aangescherpte definitie is dan ook als volgt.

Definitie 5.8: Service Oriented Architecture

SOA is een paradigma dat kan worden toegepast op een organisatie, waarbij:

- *Services worden gebruikt om bedrijfsprocessen te ondersteunen;*
- *Services uit verschillende domeinen kunnen worden aangeboden;*
- *Services herbruikbaar zijn;*
- *De service interface ontkoppeld is van de implementatie van de service. Dit betekent dat de implementatie op ieder platform en in iedere taal mag, maar dat de interactie verloopt volgens standaarden.*

5.3 Voor- en nadelen

In de inleiding is aangegeven dat SOA een oplossing *pretendeert* te bieden voor het aanpassingsvermogen van de IT-huishouding. Dit wordt bewerkstelligd door IT zo in te richten dat deze makkelijk afgestemd kan worden op processen. Vanwege de geringe ervaring bevat de literatuur vooral veel verwachte en beoogde voordelen en nadelen.

Voordelen

De belangrijkste verwachte voordelen zijn als volgt [Channabasavaiah 2004, OASIS 2006, Sogeti 2007].

- **Hergebruik**

SOA zorgt voor hergebruik van IT-functionaliteit, zodat men binnen en eventueel buiten een organisatie integraal gebruik kan maken van dezelfde services. Niet iedere organisatie-eenheid hoeft zijn eigen oplossingen te verzinnen om te voorzien in generieke behoeftes. Dit zorgt voor kortere ontwikkeltijden en lagere kosten. Door gebruik te maken van dezelfde services zal er ook minder redundantie optreden en is er minder risico op het maken van fouten. Men maakt immers op gelijke wijze gebruik van data en verwerkt deze op dezelfde manier.

- **Wendbaarheid**

Bij SOA wordt gebruik gemaakt van herbruikbare services die gebaseerd zijn op standaarden en is de service interface ontkoppeld van de implementatie. Dit zorgt ervoor dat de IT-huishouding is ingedeeld in losse, onafhankelijke 'brokken' die men in de gehele organisatie en eventueel daarbuiten kan gebruiken. Voor een organisatie leidt dit tot een grotere wendbaarheid. Het gaat erom dat men beter in staat is om de IT-huishouding af te stemmen op het bedrijfsgebeuren. Wanneer men een proces wil realiseren of aanpassen kan men gebruik maken van bestaande functionaliteit en indien nodig services ontwikkelen. Bij hergebruik van services hoeft niet meer het gehele ontwikkelingstraject doorlopen te worden. Dit voordeel komt beter tot zijn recht wanneer er meer services zijn, wat tevens meer mogelijkheden biedt voor het creëren van composite services.
Een grotere wendbaarheid heeft tot gevolg dat het inrichten van een proces korter duurt. Bij het introduceren van een dienst of product betekent dit een kortere *time-to-market*. Het hergebruik van functionaliteit zal verder leiden tot lagere kosten.

- **Interoperabiliteit**

Ontkoppeling van functionaliteit en implementatie zorgt voor interoperabiliteit. Overal in een organisatie en eventueel daarbuiten kan men services op dezelfde manier gebruiken volgens gemaakte afspraken. Essentieel hierbij zijn de service interface en het gebruik van master data. Er worden zo min mogelijk aannames gedaan over de omgeving en de techniek, wat ervoor zorgt dat nieuwe functionaliteit gemakkelijk geïntegreerd kan worden in een bestaand IT-landschap. Dit is bijvoorbeeld relevant bij het ontwikkelen van nieuwe services, maar ook bij een fusie of overname. Verder hoeven oude applicaties niet vervangen te worden, maar is het mogelijk om functionaliteit uit een applicatie aan te bieden als een service. Deze services kunnen eventueel weer gebruikt worden om andere services te ondersteunen.
Interoperabiliteit heeft als gevolg dat het IT-landschap beter beheersbaar wordt en zorgt voor een hogere consistentie.

Nadelen

Mogelijke nadelen zijn als volgt.

- **Governance**

Bij het introduceren en toepassen van SOA is het belangrijk dat men toeziet op het naleven van opgestelde principes en regels en het beschikbaar stellen en gebruiken van services. Dit wordt ook wel *governance* genoemd. Hergebruik heeft zijn voordelen, maar er moet ook een eigenaar aangesteld worden die verantwoordelijk is voor bijvoorbeeld de beschikbaarheid, juist gebruik en het verzorgen van updates. Juist omdat services niet meer aan een specifieke afdeling of specifiek organisatieonderdeel gebonden zijn is het belangrijk hier goede afspraken over te maken. Wanneer men start met SOA is het verder essentieel dat er wordt bewaakt dat nieuwe initiatieven zich conformeren aan regels en standaarden, eventueel gebruik wordt gemaakt van beschikbare services en dat generieke services daadwerkelijk geregistreerd worden bij de service directory. De activiteiten die voor governance moeten worden uitgevoerd of moeilijkheden die zich voordoen kunnen leiden tot extra kosten.

- **Lange termijn resultaten**

Het toepassen van SOA vergt tijd en werpt pas na enige tijd zijn vruchten af. Bij de introductie van SOA is er nog niet of nauwelijks sprake van hergebruik en zal er veel geïnvesteerd moeten worden in het opzetten van een service directory en het beschikbaar stellen van services. Ook moet men de gehele organisatie bewust maken van de voordelen en zorgen dat eenieder meedoet. Voor een organisatie zal de drempel dus hoog zijn, omdat de voordelen pas op de lange termijn tot hun recht komen.

6 Essent

Essent is een praktijkvoorbeeld waar SOA wordt toegepast. In dit hoofdstuk wordt dieper ingegaan op de manier waarop dit vorm wordt gegeven in het VIEW project en wordt een vergelijking gemaakt met de theorie.

6.1 VIEW project

Essent is een bedrijf dat is ontstaan door een aantal fusies en overnames. Hierdoor zijn in de loop der jaren veel verschillende applicaties ontstaan. In 2003 werden de eerste organisatiebrede applicaties gerealiseerd. Vanaf 2004 kwamen er steeds meer zogenaamde self-service scenario's beschikbaar, zodat allerlei handelingen in de support functies door de betreffende medewerkers zelf konden worden uitgevoerd. Aan het einde van dat jaar werden ook de eerste concepten uitgewerkt voor end-to-end processen. Een end-to-end proces omvat alle stappen die nodig zijn om een proces te doorlopen, dus vanaf de 'trigger' tot het volledig afronden van het proces.

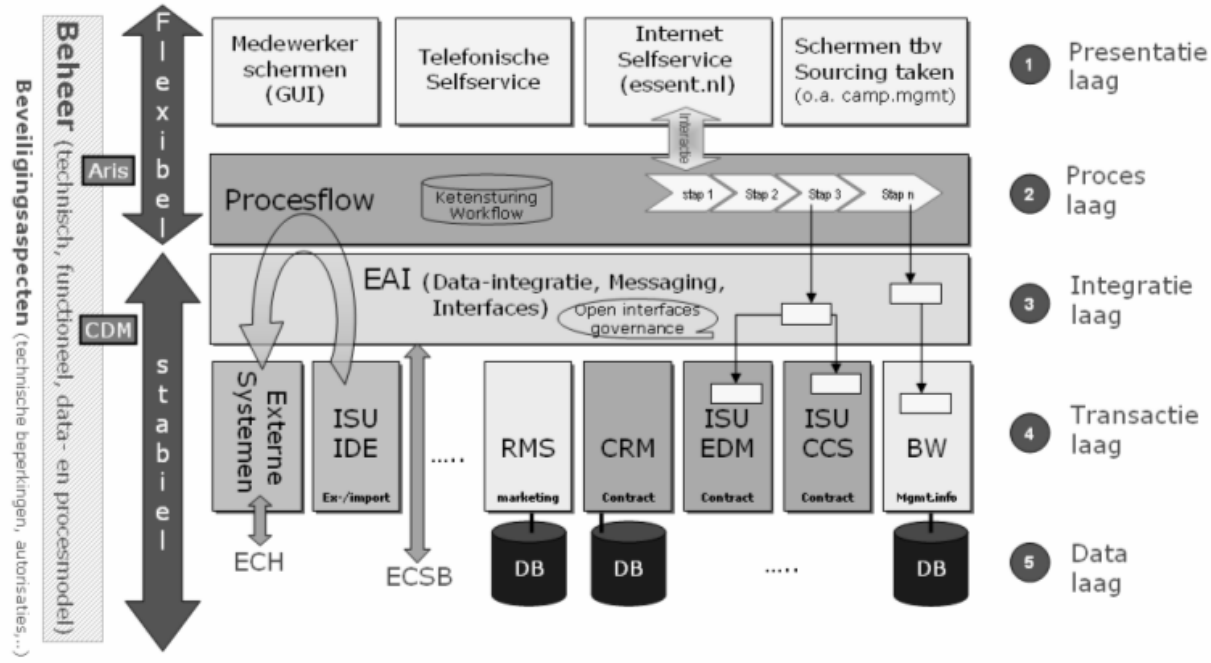
Kortom, de geschiedenis van Essent kan gezien worden als een verschuiving van 'functionele silo's' (losse applicaties voor verschillende functionele domeinen) naar gebruikersvriendelijke end-to-end processen (die over grenzen van functionele domeinen heenlopen). Zie figuur 3.1, waarin deze verandering ook wordt beschreven.

In deze verschuiving kan het VIEW project geplaatst worden. VIEW heeft als doel een digitale werkplek te realiseren (een laag over de bestaande applicaties). Hiermee kunnen werknemers bijvoorbeeld ruimtes reserveren, telefoonnummers opvragen en processtappen uitvoeren. In het eerste deel van het project is een portal ingericht die de digitale werkplek van Essent vormt. Het vervolg van het project heeft als doel om een geoptimaliseerd hire-to-retire proces te ontwerpen en implementeren. Dit proces moet uitgevoerd kunnen worden via de portal. Met geoptimaliseerd wordt bedoeld: met een bepaalde maximale doorlooptijd (de leadtime) en zo laag mogelijke kosten (in termen van de workload, de hoeveelheid berekeningen die applicaties moeten uitvoeren). Dit is gekwantificeerd in een Business Case, die inzichtelijk maakt dat de leadtime x % korter wordt en de workload y % minder. Het proces omvat de processtappen *indienst*, *mutatie* en *uitdienst* die de gehele cyclus van een werknemer bij Essent omvatten. In de toekomst zal zoveel mogelijk andere functionaliteit beschikbaar moeten komen via de portal.

Zoals eerder aangegeven bevat de portal verder een takenlijst voor processtappen en wordt het verloop van processen gevolgd met behulp van Business Process Monitoring. Alle medewerkers die een rol in een proces hebben kunnen via de procesmonitor zicht krijgen op het verloop van een proces en mogelijke knelpunten opmerken. Dit vormt de basis voor procesverbetering.

6.2 5 lagen architectuur

Essent heeft een eigen architectuur, die wordt aangeduid met de 5 lagen architectuur. Deze bestaat uit de presentatielaag, proceslaag, integratielaag, transactielaag en datalaag (zie figuur 6.1). De architectuur wordt organisatiebreed gebruikt, maar bij de nu volgende bespreking wordt deze toegespitst op het VIEW project.



Figuur 6.1: 5 lagen architectuur Essent

- **Presentatielaag**

De presentatielaag is de ‘gebruiksvriendelijke schil’ rondom de systemen. Deze zorgt voor interactie met de eindgebruiker en schermt de complexiteit van de onderliggende systemen af voor de buitenwereld. Er kunnen verschillende manieren gebruikt worden voor interactie, die relatief eenvoudig kunnen veranderen, zonder dat de back-end systemen aangepast moeten worden.

- **Proceslaag**

De proceslaag zorgt voor het verloop van een volledig end-to-end proces. Hierbij wordt de volledigheid, tijdigheid en volgorde van processtappen bewaakt. Zowel het ontwerp en de implementatie als de executie van processen vallen in deze laag. Voor het ontwerp wordt gebruik gemaakt van ARIS. De procesexecutie vindt plaats in SAP XI[5]. Een proces kan relatief eenvoudig veranderen, zonder dat de back-end systemen aangepast moeten worden. In het VIEW project wordt overigens nog geen gebruik gemaakt van een automatische transformatie van EPC diagrammen naar BPEL.

- **Integratielaag**

De integratielaag zorgt voor een verbinding tussen de verschillende systemen. Hierbij gaat het om uitwisseling van berichten en transformatie van data. Hiervoor bestaat een Common Data Model (CDM), een model waarin gedefinieerd staat hoe (in welk formaat) gegevens gerepresenteerd moeten worden. Bij VIEW wordt voor deze laag ook gebruik gemaakt van SAP XI. Naast het leggen van koppelingen met bestaande services kunnen in SAP

[5] SAP XI staat voor SAP Exchange Infrastructure; onderdeel van de SAP NetWeaver group. Het is een platform dat functionaliteit biedt op het gebied van EAI én Business Process Execution.

XI verder zelf services gebouwd worden. Het gebruik van services kan vervolgens meteen verwerkt worden in procesexecutietaal.

- **Transactielaag**

De transactielaag omvat de back-end systemen en is de laag waarin de individuele transacties daadwerkelijk worden uitgevoerd, aangestuurd door de proceslaag. Bij VIEW gaat het hier bijvoorbeeld om SAP HRM, dat functionaliteit bevat met betrekking tot personeel. Wanneer een service wordt aangeroepen vanuit een proces, wordt een transactie uitgevoerd in één of meerdere back-end systemen.

- **Datalaag**

De datalaag omvat de databases waarin alle gegevens zijn opgeslagen. Hier maken de systemen uit de transactielaag gebruik van.

Eigenschappen

De architectuur als geheel bezit verder enkele relevante eigenschappen. De presentatielaag en proceslaag zijn flexibel, terwijl de onderste drie lagen stabiel zijn (zie figuur 6.1). Dit wil zeggen dat de geïntegreerde back-end systemen (de integratie-, transactie- en datalaag) een basis vormen, waarbij het veel tijd en geld kost om deze aan te passen. De processen en de interactie met de gebruikers daarentegen worden zo gebouwd dat deze relatief eenvoudig zijn aan te passen. Dit komt doordat er gebruik gemaakt wordt van herbruikbare services, die de processtappen ondersteunen. Het aanpassen van de services zelf kost meer moeite dan het opstellen van een nieuwe orkestratie van services.
Verder worden beheer en beveiliging integraal geregeld. En er wordt gebruik gemaakt van standaarden, namelijk Web Services.

6.3 Bedrijfsvoordelen VIEW

Voordat VIEW werd gestart is er uitgebreid onderzoek gedaan naar de voordelen voor de organisatie. Eerst is een Business Case opgesteld, een onderbouwing voor de start van VIEW, waarmee aangetoond werd dat het starten van het project voordelen zou opleveren. Sterker nog, het niet starten van het project zou zelfs nadelen tot gevolg hebben. Verder is er een Proof of Concept uitgevoerd, een test op kleine schaal om de werking in de praktijk aan te tonen. Volgens de Business Case zou VIEW de volgende bedrijfsvoordelen hebben.

- **Working faster**

Processen zullen inzichtelijker worden en beter te volgen en er zullen minder fouten gemaakt worden. De hoeveelheid werk (workload) zal verkleinen en de doorlooptijd (leadtime) zal korter worden. Een hogere werknemertevredenheid draagt hier ook aan bij.

- **Working cheaper**

Het samenbrengen van user interfaces van verschillende applicaties naar één standaard zal leiden tot lagere support-, onderhouds- en leerkosten. Ook gecentraliseerde master data en self-service scenario’s zorgen voor minder kosten.

- **Working smarter**

VIEW zal zorgen voor een hoge usability. De portal is rol-gebaseerd en krijgt een consistente navigatie. Werknemers krijgen meer autonomie, omdat zij meer zelf kunnen regelen. Bovendien loopt alle communicatie binnen Essent via één kanaal.

- **Working more motivated**

Meer autonomie voor werknemers zal ook leiden tot een hogere werknemertevredenheid. Via de portal wordt het mogelijk beter en sneller informatie te zoeken, wordt kennisdeling mogelijk en worden mogelijkheden tot samenwerking, zoals video conferencing en web cast, aangeboden.

De Business Case beschrijft ook het scenario bij de situatie dat VIEW niet uitgevoerd zou worden en er niets ondernomen wordt. In dit geval zal Essent onnodige kosten moeten maken, omdat er een wildgroei aan systemen zal ontstaan. Zonder architectuur zal er geen visie zijn en bedenkt iedere business unit zijn eigen oplossingen, wat samenwerking zal bemoeilijken.

In vergelijking met de in hoofdstuk 5 genoemde voordelen van SOA wordt hier vooral ingegaan op de voordelen voor het gebruik van VIEW. De voordelen van SOA die terugkomen zijn een hogere consistentie en lagere kosten.

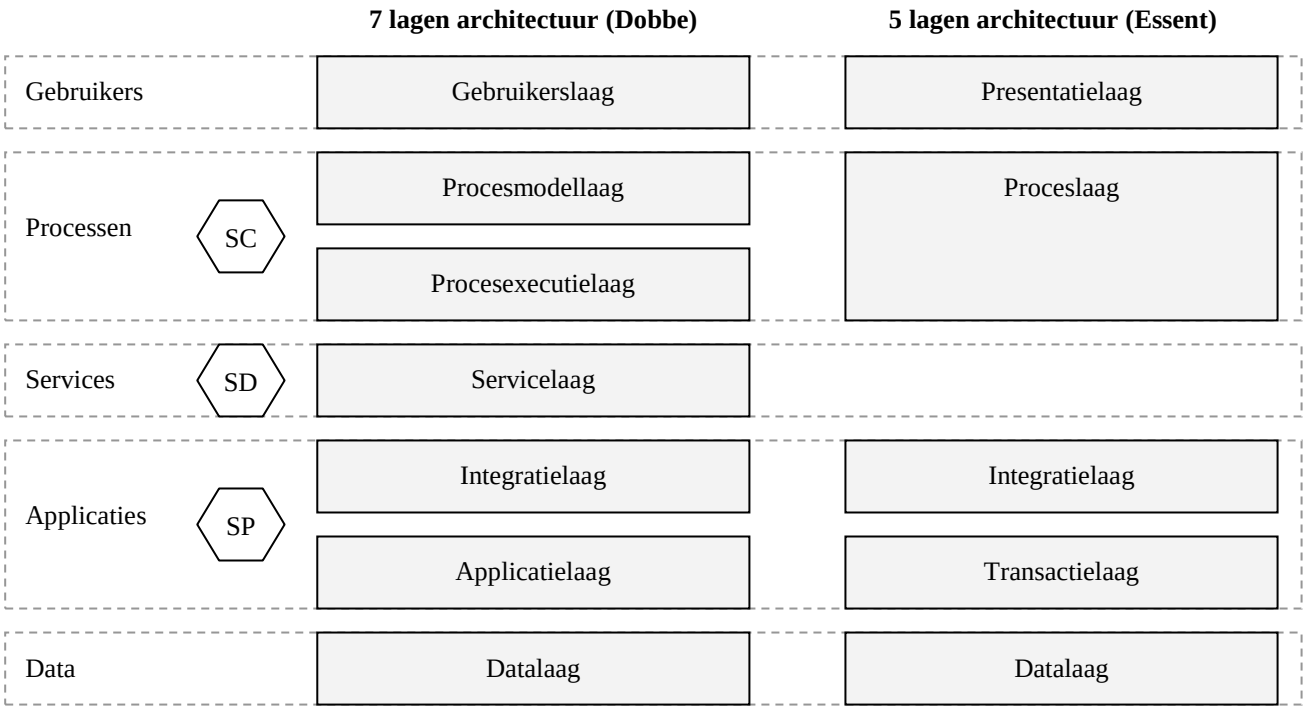
6.4 Governance

De genoemde voordelen en het alternatieve scenario geven aan dat VIEW verder reikt dan de functionaliteiten voor het hire-to-retire proces. Het is de bedoeling dat buiten VIEW ieder initiatief zich ook conformeert aan de afgesproken regels en standaarden. De 5 lagen architectuur is het uitgangspunt en er zal zoveel mogelijk gebruik gemaakt worden van de portal. De afdeling binnen Essent die zich bezighoudt met EAI draagt hier zorg voor. Deze beheert ook een service directory en zorgt ervoor dat generieke (en dus herbruikbare) services beschikbaar komen voor de gehele organisatie. Dit geldt niet voor de (domein)specifieke services.

De service directory is in kaart gebracht met een modelleertool. De directory bestaat uit een verzameling van services met bijbehorende relevante gegevens, zoals de functionaliteit en de betrokken applicaties. Dit kan gezien worden als de service description (zie hoofdstuk 5). Verder is een service hiërarchie opgesteld waarmee aangegeven is hoe services gebruik maken van andere. Tot slot worden er afspraken gemaakt over het beheer van services. Er zijn verschillende business domeinen die verantwoordelijk zijn voor een deel van de services en de EAI afdeling zorgt ervoor dat iedereen zich houdt aan het CDM.

6.5 Vergelijking architectuur

Nu de praktijk bij Essent is besproken is het interessant om deze te vergelijken met de theorie. De 5 lagen architectuur van Essent wordt vergeleken met de eerder besproken 7 lagen architectuur. In figuur 6.2 zijn de twee verschillende modellen naast elkaar geplaatst. De lagen zijn in horizontale banen geplaatst, waar ook de drie SOA entiteiten zichtbaar in zijn. De gebruikerslaag en presentatielaag hebben te maken met gebruikers. De procesmodel-, procesexecutie- en proceslaag hebben te maken met processen; de service consumer (SC). De servicelaag heeft te maken met services en het beheer ervan; de service directory (SD). De applicatielaag en integratielaag hebben te maken met de applicaties; de service provider (SP). En de datalaag heeft te maken met data, gegevensverzamelingen.



Figuur 6.2: Vergelijking architectuurlagen

Overeenkomsten

De bovenste laag van beide modellen komt overeen. In de voorgestelde 7 lagen verdeling wordt deze echter gebruikerslaag genoemd, omdat deze betrekking heeft op de interactie met gebruikers en niet slechts op de presentatie van gegevens. Bovendien sturen gebruikers uiteindelijk een proces aan.
Verder zijn de onderste drie lagen van beide modellen ook gelijk, met uitzondering van de naam applicatielaag in plaats van transactielaag.

Verschillen

Als eerste verschil omvat de proceslaag bij Essent zowel de procesmodellaag als de procesexecutielaag van de 7 lagen architectuur. Het is belangrijk deze te scheiden, omdat er verschillende activiteiten plaatsvinden, namelijk analytical modeling en executable modeling. Kortom, de procesmodellaag staat voor de business kant van een proces en de procesexecutielaag staat voor de technische kant van een proces. Verder worden er ook verschillende tools voor gebruikt. Bij Essent wordt ARIS gebruikt voor analytical modeling en wordt SAP XI gebruikt voor executable modeling. Verder wordt SAP XI in dit geval ook voor de integratielaag ingezet en worden er services gebouwd, wat verwarrend kan zijn. Het is daarom veel overzichtelijker twee architectuurlagen voor de processen te onderscheiden.

Het tweede en belangrijkste verschil is dat Essent geen aparte servicelaag heeft. Deze komt niet voor in de 5 lagen architectuur, maar er is wel een service directory. Het benoemen van services in de architectuur benadrukt beter het gebruik en belang ervan. Services staan immers centraal bij SOA en behoren dan ook in een laag genoemd te worden. Zoals eerder aangegeven worden behoeftes en vermogens verbonden middels services en biedt SOA een raamwerk voor de afstemming hiervan. De servicelaag zorgt ook voor ontkoppeling; het scheidt de functionaliteit en toepassing van een service van de techniek. Tot slot zal men bij invoering van SOA in een organisatie wellicht sceptisch zijn. Het is dan ook essentieel services te benoemen in een aparte architectuurlaag,

zodat de gehele organisatie wordt betrokken en men gebruik maakt van en bijdraagt aan de mogelijkheden tot hergebruik.

In de praktijk zal de servicelaag een service directory bevatten, die overzicht biedt over de beschikbare IT-functionaliteit binnen een organisatie of domein. Verder is governance essentieel. Men moet principes vastleggen en afspraken maken over de verantwoordelijkheid voor services. Bijvoorbeeld dat services daadwerkelijk beschikbaar worden gesteld voor anderen en dat ze voldoen aan afgesproken standaarden. Ook afspraken over financiën horen hierbij. Hoe wordt bijvoorbeeld omgegaan met kosten die een business unit maakt om services te ontwikkelen, terwijl andere business units hier ook gebruik van kunnen maken.

Services worden bij Essent niet per definitie alleen gebruikt in de proceslaag. Het kan ook voorkomen dat een applicatie gebruik maakt van een service van een andere applicatie. Er is dan sprake van gebruik van services op het niveau van integratie- en applicatielaag. In dat geval treedt een applicatie op als service consumer. Services kunnen dus in meerdere lagen een rol spelen, maar toch is het essentieel deze in een laag midden in de architectuur te plaatsen om aan te geven dat services voor ontkoppeling zorgen van processen en applicaties.

6.6 Vergelijking toepassing SOA

Essent heeft een organisatiebrede service directory, die reeds de nodige services bevat. De service directory wordt beheerd door de EAI afdeling en is nog in ontwikkeling. Er kan gezegd worden dat het eerder beschreven SOA model gebruikt wordt bij Essent. De service consumers zijn de processtappen in bijvoorbeeld het hire-to-retire proces. Zoals eerder aangegeven kunnen services ook door applicaties worden gebruikt. Tijdens de procesexecutie wordt vanuit SAP XI een aanroep gedaan naar de betreffende services. SAP XI is naast service consumer ook service provider, omdat de meeste services in dat platform gebouwd zijn.
Als we kijken naar de definitie van SOA komen alle elementen ervan terug. (De meeste) services ondersteunen bedrijfsprocessen; services worden vanuit verschillende domeinen aangeboden en centraal beschikbaar gesteld in een service directory; services zijn zoveel mogelijk herbruikbaar en de service interface is ontkoppeld van de implementatie van de service.

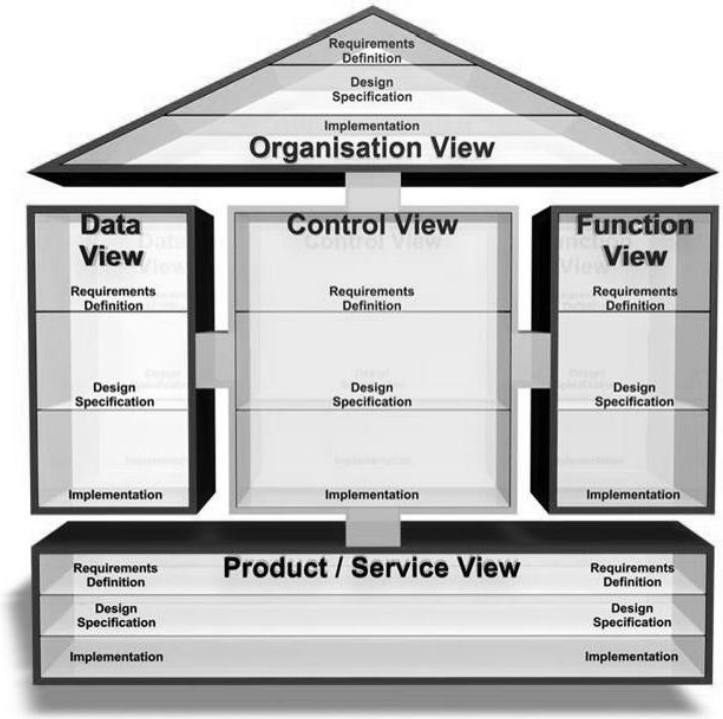
7 Case study BPMSs

In dit hoofdstuk wordt een case study behandeld over de twee BPMSs ARIS en Cordys. Eerst zal uitgelegd worden wat de kenmerken en functionaliteiten zijn van de platforms. Vervolgens worden uitgewerkte procesmodellen besproken van het indienst proces. Met de case study zal geïllustreerd worden wat de rol is van ARIS en Cordys in het toepassen van SOA. Dit gebeurt met een vergelijking van de twee BPMSs aan de hand van de 7 lagen architectuur.

7.1 ARIS

7.1.1 Architectuurraamwerk

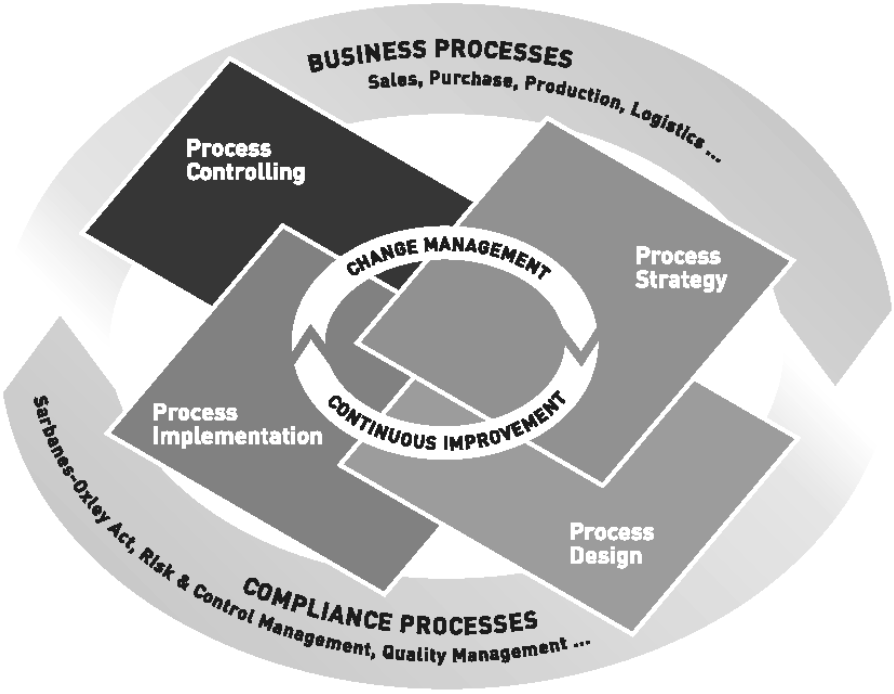
ARIS [ARIS 2007] staat voor Architecture of Integrated Information Systems en biedt een architectuurraamwerk als uitgangspunt. In dit raamwerk kan allerlei informatie worden 'opgeborgen' om een organisatie in kaart te brengen. Het raamwerk, aangeduid met 'het ARIS huis' bevat vijf views (perspectieven), namelijk de data view, function view, organisation view, product / service view en de control view. Verder heeft iedere view drie description levels (beschrijvingsniveaus); de requirements definition, de design specification en de implementation description. Deze description levels geven aan dat iedere view betrekking heeft op drie belangrijke fasen die men doorloopt, bij het ontwikkelen van een proces.
Het raamwerk wordt gebruikt als uitgangspunt om processen op een geordende manier te kunnen modelleren. De organisation view bevat de organisatiestructuur met bijvoorbeeld rollen en organisatieonderdelen die van belang zijn bij een processtap. De data view bevat de informatieobjecten die een rol spelen bij een proces en de relaties ertussen. Bijvoorbeeld een klant of een bestelling. De function view bevat de processtappen die uitgevoerd worden en de volgorde. De product / service view bevat de producten en diensten van een organisatie. Tot slot worden alle views geïntegreerd in de control view, met als resultaat een ARIS procesmodel.



Figuur 7.1: Architecture of Integrated Information Systems

7.1.2 Procesontwikkelmethode

Om processen in kaart te brengen heeft IDS Scheer ARIS Value Engineering (AVE) ontwikkeld. Dit is een methode die helpt om de levenscyclus van een proces te doorlopen en business en IT te koppelen met behulp van zowel het ARIS platform als kennis en ervaring van IDS Scheer. De basis hiervoor vormt de BPM levenscyclus, weergegeven in figuur 7.2. De vier vierhoeken in het midden stellen de vier onderdelen van het ARIS platform voor. Deze volgen elkaar op in een levenscyclus van continue verbetering en verandering.
Eerst wordt de strategie vastgelegd, bijvoorbeeld in termen van doelstellingen, met behulp van het Strategy Platform. Vervolgens wordt een ontwerp gemaakt op basis van de strategie, ondersteund door het Design Platform. Dan wordt een koppeling gelegd met de implementatie van de processen, ondersteund door het Implementation Platform. Bij de procesexecutie kan vervolgens informatie over de executie gebruikt worden om bij te sturen; dit kan met behulp van het Controlling Platform.



Figuur 7.2: BPM levenscyclus

7.1.3 Platform

ARIS heeft vier platformonderdelen, waarvan de belangrijkste functionaliteiten hier besproken worden.

- **Strategy platform**

Met het Strategy platform zijn allerlei mogelijkheden om bedrijfsdoelen en strategie in kaart te brengen en om de dagelijkse uitvoering van processen te analyseren. Om strategie vast te leggen biedt het ARIS platform de Balanced Score Card methode, waarin Key Performance Indicators (KPI's) kunnen worden aangegeven. Verder zijn er onder andere mogelijkheden om oorzaak-gevolg relaties te bepalen, 'what-if scenario's' af te spelen, ondersteuning bij het maken van 'make-or-buy' decisions en het identificeren van best practices. Kortom, allerlei ondersteuning om de strategie te bepalen en vastleggen en deze af te stemmen op de uitvoering van de organisatie.

- **Design platform**

Met het Design platform kunnen bedrijfsprocessen worden afgestemd op de strategie. Allereerst kan de enterprise architectuur worden vastgelegd op basis van het ARIS huis, waarbij organisatie, data, functies, producten/diensten en processen in gemodelleerd kunnen worden. Verder kunnen processen worden gesimuleerd, voor een analyse van de huidige situatie en het opmerken van mogelijke verbeterpunten. Daarnaast kan met indicatoren worden aangegeven hoe processen zouden moeten verlopen; hier zijn ook industriespecifieke waarden voor beschikbaar. Daarnaast is er een tool voor kwaliteitsmanagement op basis van de ISO 9000 standaard. Tot slot zijn er best practices, gebaseerd op ITIL (Information Technology Infrastructure Library); dit is een verzameling beste praktijkoplossingen die een referentiekader vormen bij het inrichten van processen voor organisaties in verschillende branches.

- **Implementation platform**

Het implementation platform biedt mogelijkheden om een vertaalstap te maken van de gemodelleerde bedrijfslogica naar procesexecutietaal. Het implementation platform bevat zelf geen Process Execution Engine. ARIS kan procesmodellen in EPC of BPMN transformeren naar BPEL, deze bewerken en exporteren. Men kan zelf bepalen hoeveel informatie over de ondersteunende techniek in ARIS wordt gemodelleerd. De geëxporteerde BPEL kan vervolgens geïmporteerd kunnen worden in een Process Execution Engine, waar de processen indien nodig kunnen worden aangevuld.

Verder zijn er afspraken gemaakt met leveranciers van producten met een Process Execution Engine, waarvan de belangrijkste SAP is. Bij het opstellen van een model kunnen in dat geval meteen relaties worden gelegd met SAP elementen voor een goede transformatie naar procesexecutietaal. En tot slot kunnen business rules worden opgesteld en beheerd en kunnen technische requirements worden vastgelegd in UML (Universal Modeling Language).

- **Controlling platform**

Het controlling platform omvat metingen over de efficiëntie van processen en controlesystemen voor wetten en normen. Metingen vinden plaats op basis van de procesexecutie op een Process Execution Engine. Met de ARIS Process Performance Manager kan men processen analyseren en visualiseren. Er worden bijvoorbeeld doorlooptijden per processtap gemeten en vergeleken met normen. Verder biedt ARIS nog een tool voor metingen voor de mate waarin een organisatie voldoet aan wetten zoals de Sarbanes-Oxley Act en een tool voor risicomanagement.

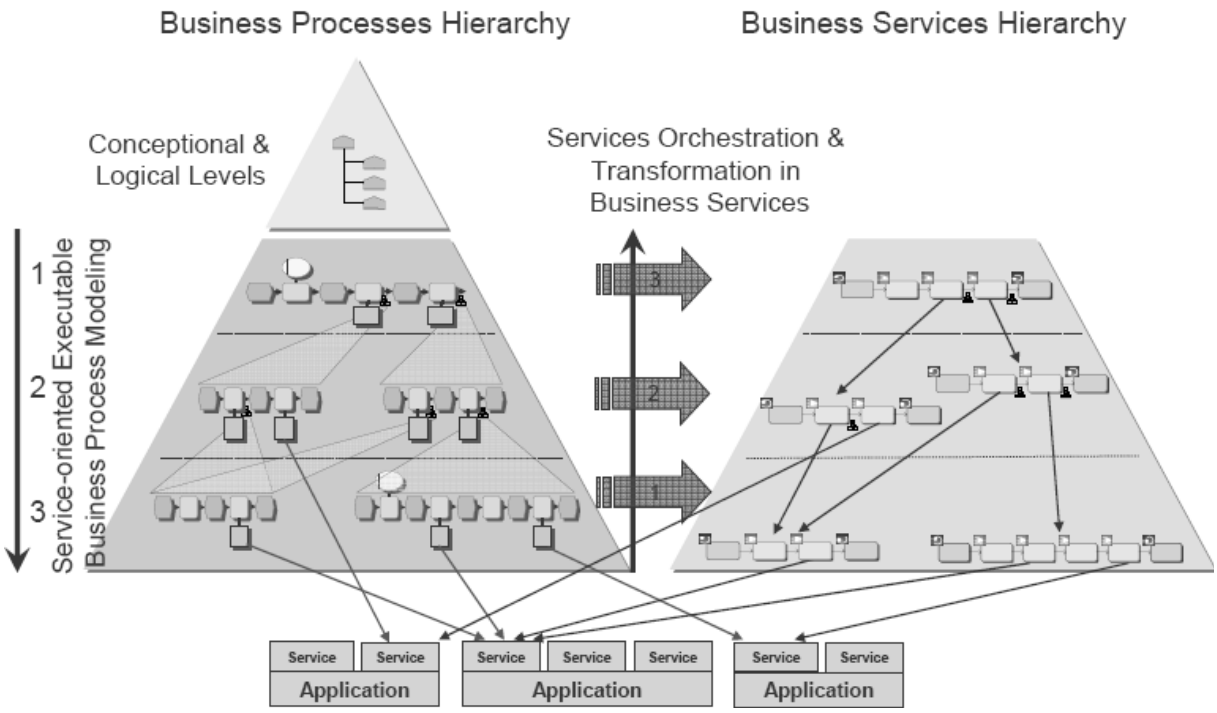
7.1.4 Modelleren

Om bedrijfsprocessen te modelleren maakt ARIS gebruik van EPC (Event-driven Process Chain) diagrammen en van BPMN.

Proceshiërarchie en EPC

Een EPC diagram is een procesnotatie die bestaat uit events (gebeurtenissen) en functions (functies). Functions kunnen worden verrijkt met aanvullende informatie, zoals wie de uitvoerder is, wat de input en output is en welke user interface erbij betrokken is. Functions kunnen ook een subproces zijn; in dat geval wordt verwezen naar een ander EPC diagram, die een deel van het proces beschrijft. Hierdoor biedt EPC de mogelijkheid om een proces hiërarchisch op te delen en het overzichtelijk te houden.
Deze hiërarchie is van belang bij het vertalen van business naar IT. Services hebben namelijk een verschillende mate van granulariteit en zullen zo goed mogelijk afgestemd moeten worden op de behoefte uit processen. In figuur 7.3 is te zien dat er verschillende niveaus van processen onderscheiden worden in de linker driehoek.

Links wordt de procesmodellen in EPC getoond en rechts de uitvoerbare versies in BPEL (de grafische BPEL weergave in ARIS). Onderaan de figuur bevinden zich services die worden aangeboden door applicaties. De EPC diagrammen bevatten koppelingen naar deze services. Bij de vertaling naar BPEL moeten onder andere de technische details (de service interfaces) van de services worden toegevoegd om de processen uitvoerbaar te maken.



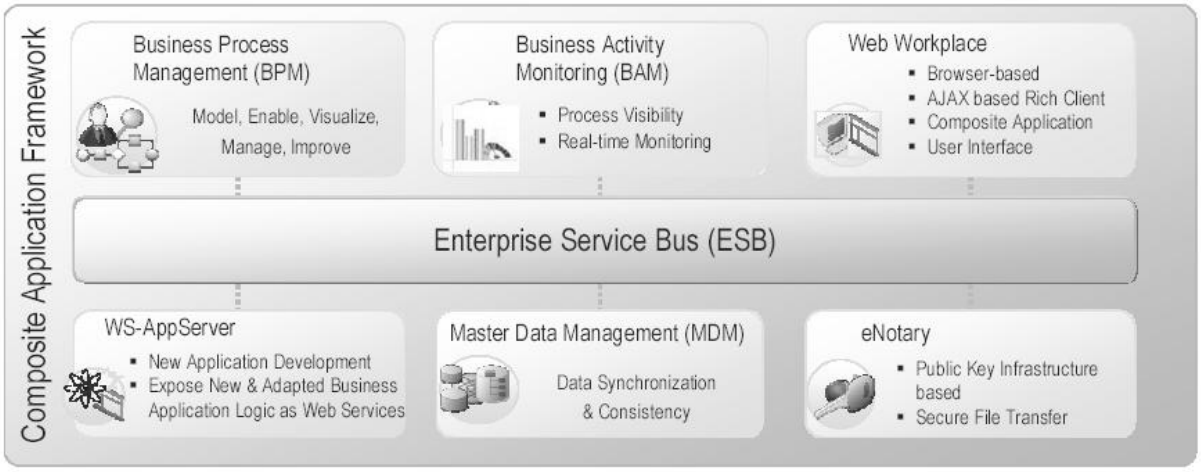
Figuur 7.3: Proceshiërarchie en transformatie naar BPEL in ARIS

BPMN

Een proces wordt in BPMN weergegeven met onder andere functions (functies), gateways (beslissingspunten) en een start- en eindpunt. Wanneer men een interactie wil aangeven tussen meerdere processen worden zogenaamde *pools* gebruikt. Een pool is een grafische weergave van een organisatieonderdeel, dat een proces bevat. De interactie tussen processen uit verschillende pools kan zo worden gemodelleerd. Binnen een pool kan ook nog een onderscheid worden gemaakt in bijvoorbeeld rollen van processtappen, dat tot uitdrukking gebracht kan worden met behulp van *lanes*. Een lane bevat dan alle processtappen met een bepaalde rol. In paragraaf 7.3 wordt een voorbeeld besproken.

7.2 Cordys

Cordys [Cordys 2007] noemt zijn product het Cordys Composite Application Framework (zie figuur 7.4).

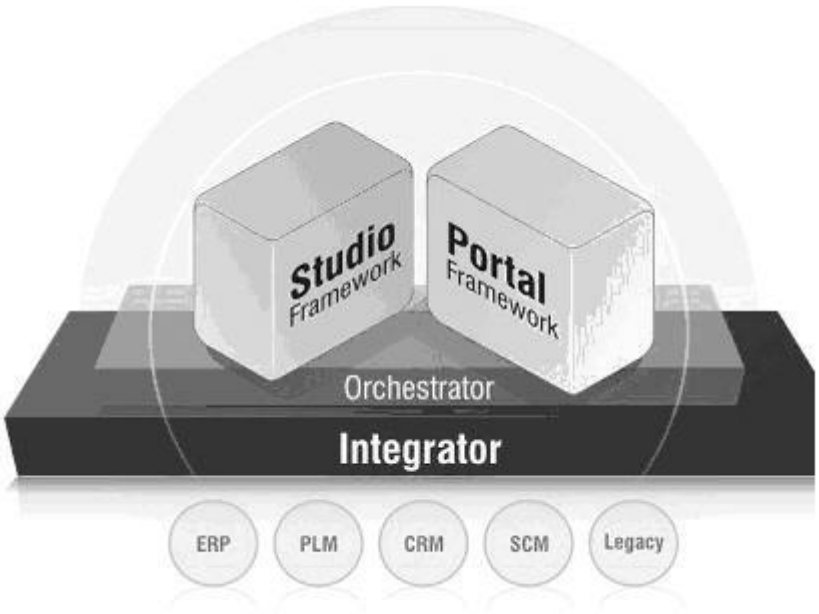


Figuur 7.4: Cordys Composite Application Framework

Het product van Cordys is opgebouwd rondom een ESB, gebaseerd op Web Services. Het is één omgeving die wordt gebruikt voor zowel designtime als runtime. Analytical modeling (in BPMN) heeft een directe koppeling naar executable modeling (in BPML). Het platform biedt namelijk naast een modelleergedeelte ook een Process Execution Engine en integratie functionaliteit met back-end systemen. Software elementen uit back-end systemen kunnen op die manier in Cordys beschikbaar gesteld worden als Web Services, zodat deze direct aan processtappen gekoppeld kunnen worden. Tot slot biedt de suite mogelijkheden voor Business Process Monitoring en Master Data Management en biedt het functionaliteit voor security services voor bijvoorbeeld toegang, authenticatie, encryptie en Public Key Infrastructure (PKI) en voor ondersteuning van processtappen met behulp van een takenlijst.

7.2.1 Platform

In deze paragraaf wordt een overzicht gegeven van de elementen van het Cordys platform. Zie figuur 7.5.



Figuur 7.5: Cordys platform

- **Cordys Studio**

Cordys Studio biedt gebruikers de mogelijkheid om processen te modelleren (in BPMN) met de Business Modeling Studio en om gebruikersinterfaces te ontwerpen en ontwikkelen met de Application Modeling Studio.

- **Cordys Integrator**

In de Integrator worden methoden uit de back-end systemen beschikbaar gesteld als Web Services. Iedere Web Service bevat een WSDL en een XSD bestand, zodat deze aangeroepen kan worden door een service consumer. Een WSDL bestand (Web Service Description Language) beschrijft de service interface en een XSD bestand (XML Schema Definition) definieert welke elementen het WSDL bestand dient te bevatten.

- **Cordys Orchestrator**

De Orchestrator biedt functionaliteit om processen te implementeren (in BPML). Andere functionaliteiten zijn Business Process Monitoring, een Business Rules Engine, een notificatie service en datatransformatie. De notificatie service biedt de mogelijkheid om berichten te sturen naar gebruikers met taken, bij het voordoen van 'business events'. Een gebruiker krijgt bijvoorbeeld een taak in zijn inbox om een bestelling te doen bij een leverancier, wanneer een klant een bestelling heeft geplaatst. Datatransformatie betreft EAI functionaliteit en kan nodig zijn om datastromen tussen verschillende organisaties af te stemmen.

- **Web Workplace (portal)**

De Web Workplace is een portal waarin de prestaties van een organisatie weergegeven kunnen worden en waarmee organisatiebreed gecommuniceerd kan worden. Aan de hand van informatieanalyse over procesverloop en KPI's kunnen metingen gedaan worden. Verder zijn er allerlei mogelijkheden voor communicatie in een organisatie, zoals discussionboards en polls en er is een integratie met MS Outlook en MS Sharepoint. Bovendien is het (in designtime) mogelijk gebruikersinterfaces te construeren.

7.2.2 Modelleren

Cordys biedt op hoog niveau een soort beschrijvende modellen om processen binnen een organisatie en met handelspartners weer te geven. In zogenaamde Value Chain Models worden de relaties met partnerorganisaties gemodelleerd en de processen die daarbij een rol spelen. Bij Business Context Models worden processen binnen een organisatie gecatalogiseerd op basis van functionele domeinen.

Voor analytical modeling biedt Cordys BPMN aan. Hierbij kan een directe koppeling gelegd worden tussen Web Services en processtappen en kan het proces, wanneer het gereed is, meteen uitgevoerd worden (in BPML). Een proces begint en eindigt met respectievelijk een start- en eindpunt. Processtappen worden weergeven met activiteiten en meerdere processtappen kunnen worden ondergebracht in een subproces. Verder kunnen onder andere berichten en conditities met betrekking tot tijd worden gemodelleerd.

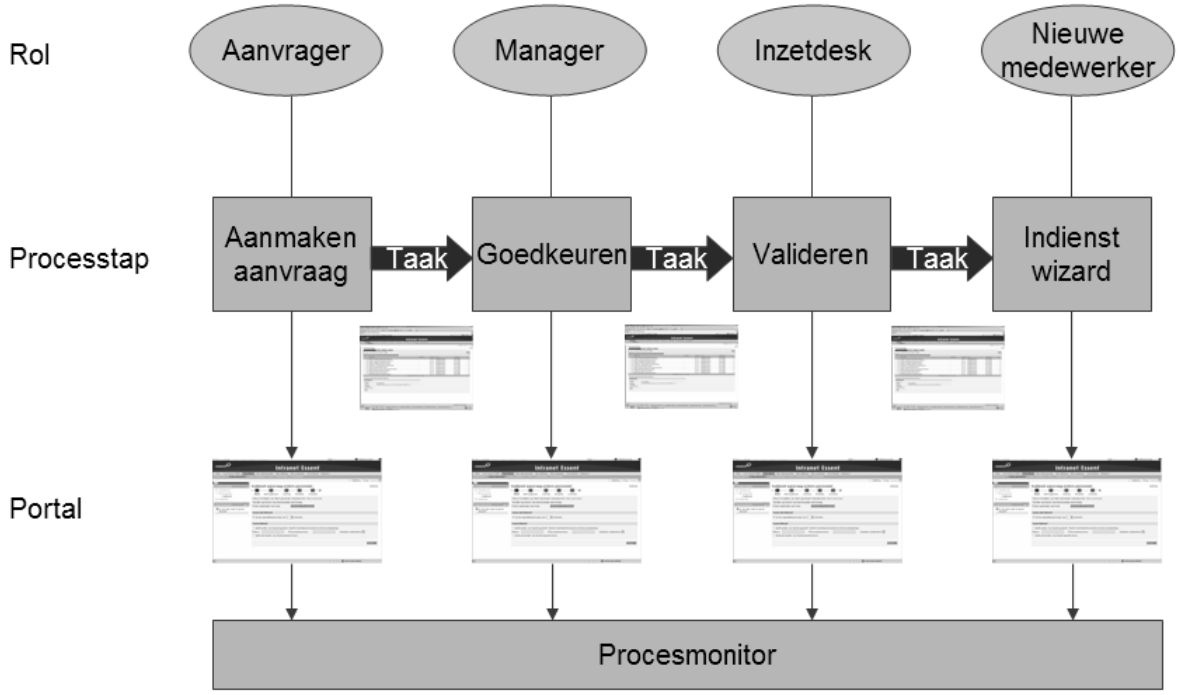
7.3 Het indienst proces

In deze paragraaf wordt een vereenvoudigde weergave gegeven van het indienst proces bij Essent, dat dient als illustratie van analytical modeling. Het betreft alleen het indienst gedeelte van het gehele hire-to-retire proces en niet het mutatie en uitdienst deel. Het beschreven proces is gemodelleerd in ARIS en Cordys.

7.3.1 Procesbeschrijving

Een vereenvoudigde weergave van het indienst proces is te zien in figuur 7.6. Hierin is aangegeven dat processtappen aangestuurd worden door taken, dat gebruikers (met een bepaalde rol) taken uitvoeren via de portal en dat het verloop wordt bijgehouden door de procesmonitor. In de figuren 7.7 tot en met 7.9 zijn

modellen zichtbaar van ARIS en Cordys, waarin alle processtappen zichtbaar zijn.



Figuur 7.6: Indienst proces

Het proces start met een behoefte aan een nieuwe werknemer. Het gaat om tijdelijk extern personeel. De eerste stap ‘Creëer inhuur aanvraag’ wordt uitgevoerd door een medewerker van een afdeling, bijvoorbeeld een secretaresse. In deze processtap wordt een formulier ingevuld, waar gegevens als het functieprofiel en de inhuurperiode aangegeven kunnen worden. Iedere aanvraag is slechts voor één werknemer. Nadat een aanvraag is ingediend wordt deze voorgelegd aan de manager van de afdeling waar de toekomstige werknemer werkzaam zal zijn. Deze kan de aanvraag goedkeuren of afwijzen. Bij afwijzing ontvangt de aanvrager hier een notificatie van; bij goedkeuring belandt de aanvraag bij een medewerker van Human Resources (HR), dit wordt de ‘inzetdesk’ genoemd. Deze beheert de verschillende aanvragen en is op de hoogte van afspraken met een uitzendbureau (UZB). In de processtap ‘UZB aanvraag’ wordt een aanvraag verstuurd naar het uitzendbureau. Deze verwerkt de aanvraag en het resultaat hiervan is een terugkoppeling met de gegevens van een werknemer die aan de aanvraag voldoet. Het proces vervolgt met het creëren van de master data van de nieuwe inhuurkracht door een medewerker van HR. Op de eerste werkdag van de nieuwe medewerker wordt hem/haar gevraagd om de zogenaamde indienst wizard te doorlopen. Hierin geeft hij/zij aan of de juiste bedrijfsmiddelen (laptop, muis, etcetera) zijn ontvangen en of medewerkersverklaringen (bijvoorbeeld voor geheimhouding) zijn ondertekend. Hiermee is het indienst proces afgerond.

7.3.2 Indienst proces in SOA termen

Na de procesbeschrijving in natuurlijke taal is het interessant om de voor SOA relevante elementen hieruit te abstraheren. Hierbij wordt gebruik gemaakt van de in hoofdstuk 5 geïntroduceerde termen, zoals behoeftes en vermogens.

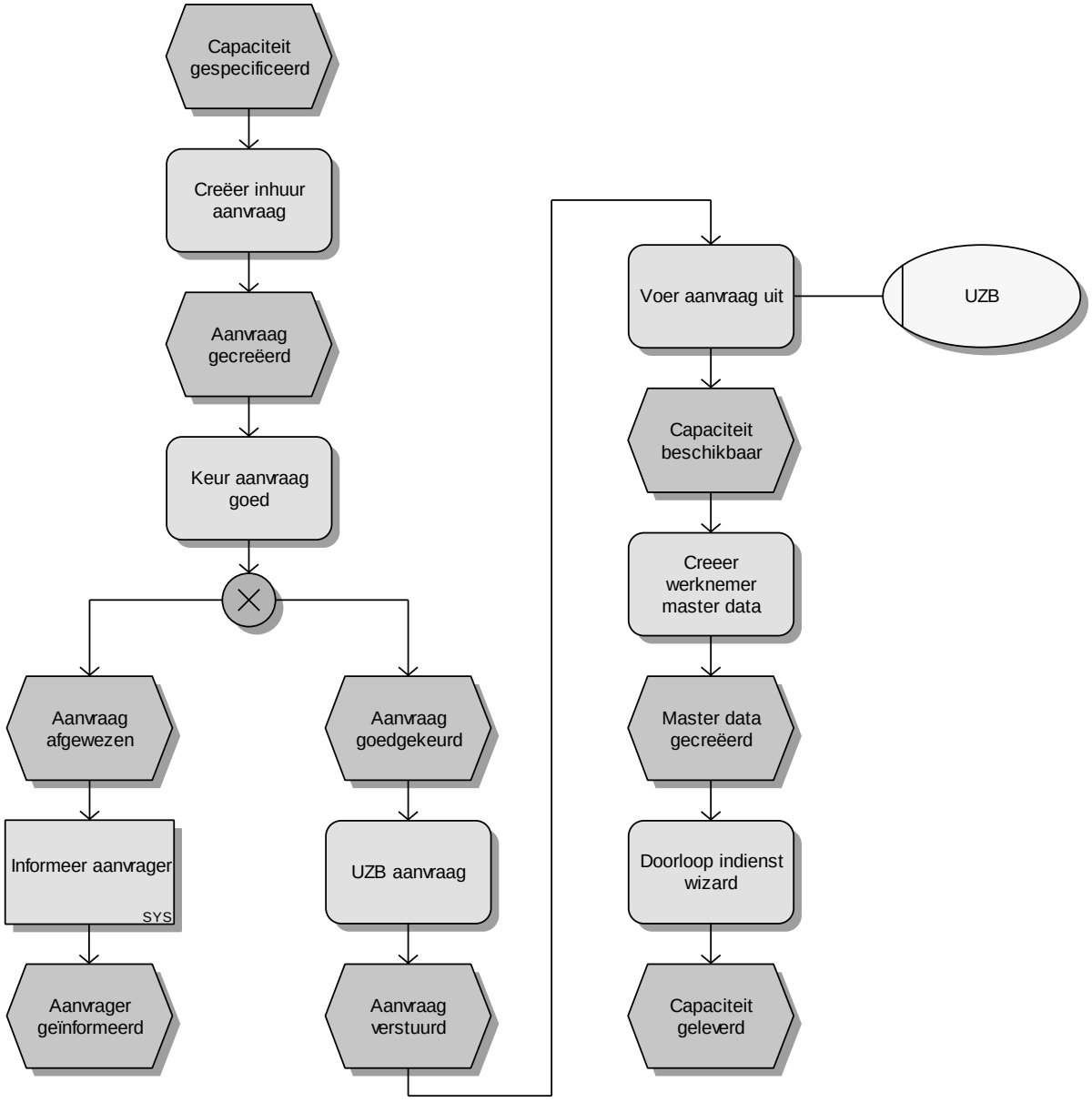
De ‘trigger’ is een behoefte aan een nieuwe werknemer. Om in deze behoefte te voorzien zijn vermogens nodig. Bij het ontwerp van het indienst proces heeft men onderzocht in hoeverre er reeds services beschikbaar waren met bepaalde vermogens. Omdat men bij Essent net is begonnen met SOA waren er nog geen relevante services

beschikbaar in de service directory; deze moesten dus zelf ontwikkeld worden om de stappen in het indienst proces te ondersteunen. De services zijn ontwikkeld in SAP XI; dit is de entiteit die de services levert, ofwel de service provider. De procesexecutie vindt ook plaats in SAP XI; dit is de entiteit die de service gebruikt, ofwel de service consumer. SAP XI vervult in dit geval een dubbelrol. Wanneer er in de toekomst meer services uit verschillende domeinen beschikbaar komen, die organisatiebreed gebruikt worden, kunnen de mogelijkheden die SOA biedt beter tot hun recht komen.

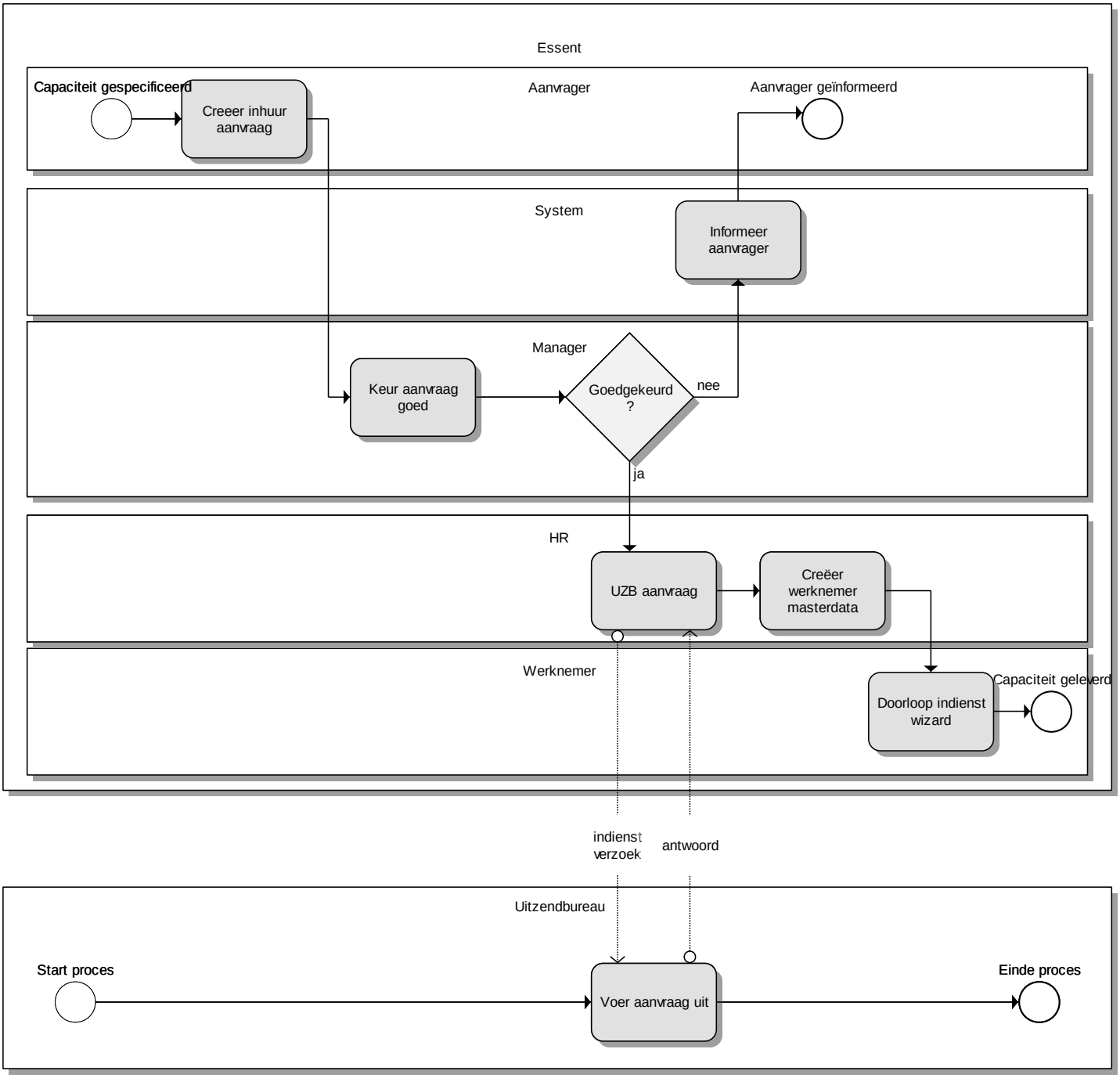
Het indienst proces bestaat uit meerdere processtappen, die ieder eigen behoefte hebben. Aan de andere kant zijn er services met een vermogen die geheel of gedeeltelijk in een behoefte kunnen voorzien. Deze worden op elkaar afgestemd.
Voor enkele processtappen zijn portalschermen nodig met formulieren. Voor ieder formulier zijn weer specifieke gegevens of mogelijkheden vereist, die geleverd kunnen worden door services. Voorbeelden van zulke services bij Essent zijn *getEmployee* (haalt alle werknemergegevens op uit een database), *getFunctionProfiles* (haalt alle functieprofielen op uit een database) en een kalender om een datum te selecteren. Deze services kunnen hergebruikt worden in formulieren voor bijvoorbeeld de inhuur aanvraag, het goedkeuren, de UZB aanvraag en de indienst wizard. Een compleet formulier dat de werknemergegevens of contractgegevens presenteert kan op zijn beurt ook hergebruikt worden, namelijk bij het creëren van een inhuur aanvraag en bij de indienst wizard. Kortom, er kan een hiërarchie opgesteld worden van services die een vermogen bieden, om te voorzien in een behoefte uit een processtap. In figuur 7.3 is dit weergegeven voor ARIS.

7.3.3 Indienst proces in ARIS en Cordys

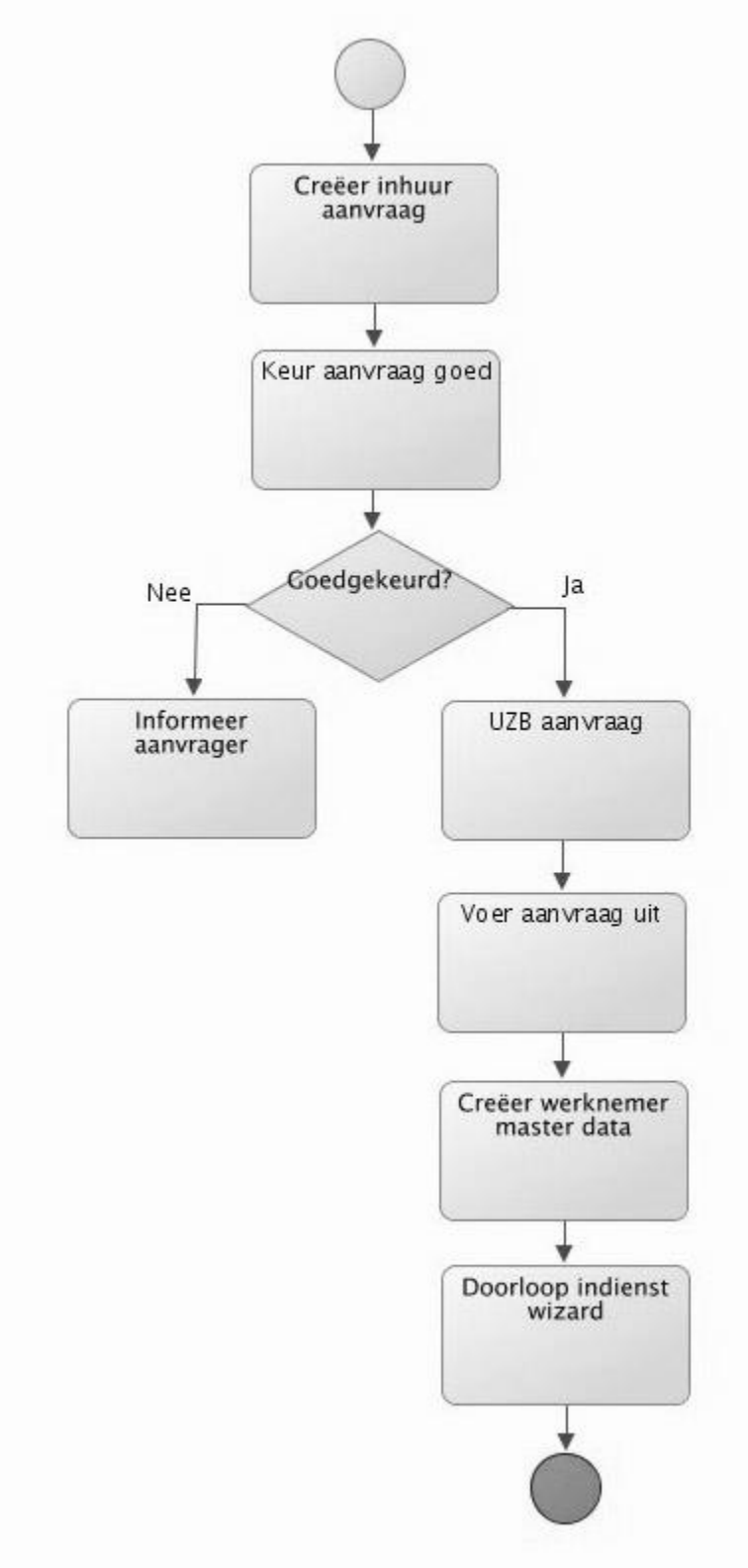
In de figuren 7.7 tot en met 7.9 is het indienst proces te zien dat is gemodelleerd in ARIS en Cordys. In ARIS is het proces zowel in EPC als in BPMN weergegeven en in Cordys alleen in BPMN.
In EPC worden functions steeds afgewisseld met events, die niet zijn weergegeven bij BPMN (zowel bij ARIS als Cordys). In figuur 7.8 is te zien welke processtap wordt uitgevoerd door welke rol. In de andere modellen kan dit ook worden aangegeven door aan iedere processtap een rol te koppelen, maar voor de overzichtelijkheid is dit weggelaten. De indienst wizard wordt door een gebruiker uitgevoerd en wordt tevens ondersteund door enkele services. De stap ‘Voer aanvraag uit’ wordt uitgevoerd door het uitzendbureau. Dit is weergegeven in EPC door de organisatorische rol ‘UZB’ aan de processtap te koppelen. Het uitzendbureau voert een stap in het proces uit. In BPMN (ARIS) is het zichtbaar doordat de processtap in de pool ‘uitzendbureau’ is geplaatst. Hierbij vindt interactie plaats met het uitzendbureau, waarna het proces vervolgt.
De meeste processtappen zijn interactief (er is een gebruiker bij betrokken), alleen de processtap ‘Informeer aanvrager’ is geautomatiseerd (er is geen gebruikersinteractie). In figuur 7.7 is deze systeemstap aangegeven met rechte hoeken en de afkorting ‘SYS’.



Figuur 7.7: Indienst proces ARIS (EPC)



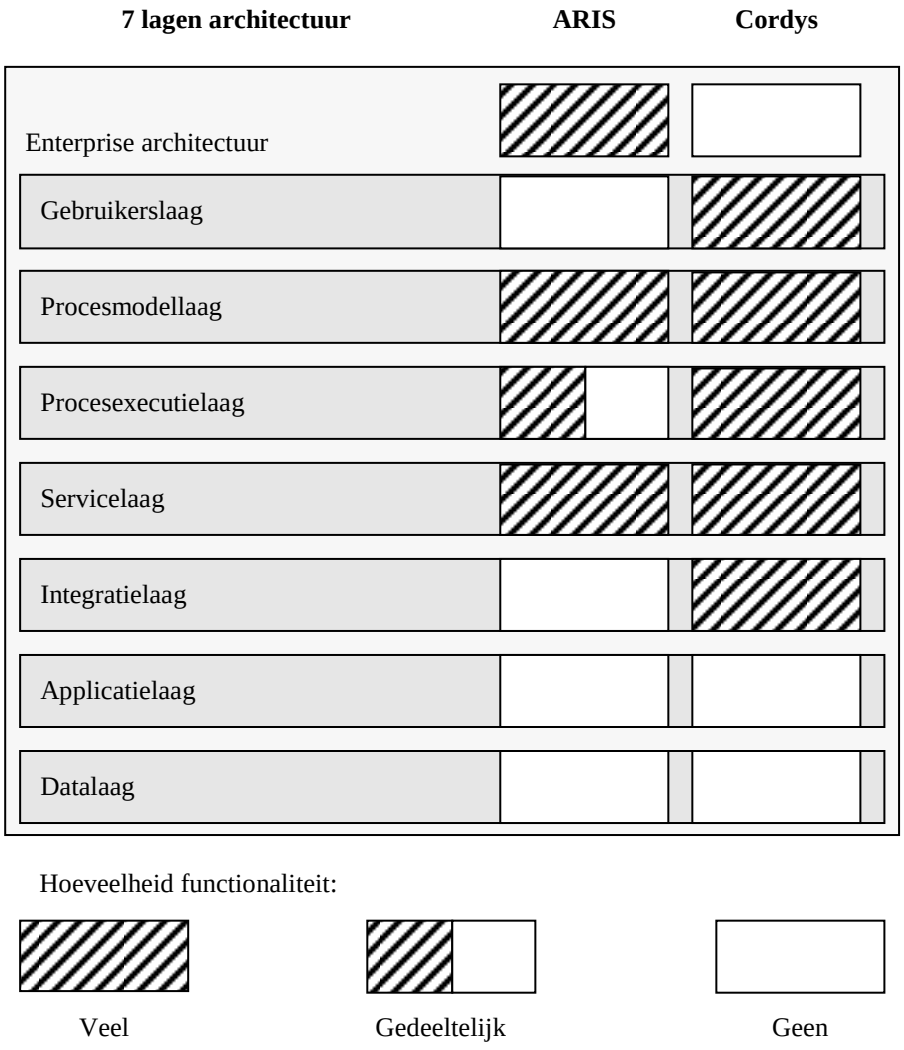
Figuur 7.8: Indienst proces ARIS (BPMN)



Figuur 7.9: Indienst proces Cordys (BPMN)

7.4 Vergelijking ARIS en Cordys

Om de verschillen en overeenkomsten tussen ARIS en Cordys goed te kunnen aangeven is een vergelijking gemaakt met behulp van de 7 lagen architectuur. Per laag wordt vergeleken in hoeverre de BPMSs functionaliteit aanbieden. De gearceerde delen in figuur 7.10 geven de hoeveelheid functionaliteit aan.



Figuur 7.10: Vergelijking ARIS en Cordys

7.4.1 ARIS

Aan het modelleren in ARIS ligt een architectuurraamwerk ten grondslag, waarin men alles dat in een organisatie betrekking heeft op processen kan opbergen. De platformonderdelen sluiten hier op aan en passen ook bij de procesontwikkelmethode die ARIS biedt. Kortom, op het gebied van enterprise architectuur heeft ARIS uitgebreide mogelijkheden.
Op het gebied van gebruikers heeft ARIS nauwelijks iets te bieden. Voor interactie met gebruikers kan wel worden gemodelleerd welke gegevens in bepaalde velden in een formulier moeten verschijnen. Ook kunnen gebruikers, rollen en betreffende organisatieonderdelen gekoppeld worden. Het bouwen van een GUI is echter niet mogelijk.

In de procesmodellaag heeft ARIS een zeer goede ondersteuning. Analytical modeling kan met zowel EPC als BPMN en er kunnen talloze relaties worden gelegd tussen processtappen en andere elementen uit het ARIS huis. Vervolgens is het mogelijk om een transformatie te maken naar BPEL. Wanneer men beschikt over de service interfaces, kunnen deze toegevoegd worden, zodat de link wordt gelegd tussen een processtap en de technische uitvoering. Er kan dan BPEL geëxporteerd worden, zodat men dit kan gebruiken op een platform dat over een Process Execution Engine beschikt. Kortom, ARIS beschikt over een gedeeltelijke hoeveelheid functionaliteit in de procesexecutielaag, omdat wel transformatie naar BPEL mogelijk is, maar er geen Process Execution Engine is.
In de servicelaag biedt ARIS de mogelijkheid om een service directory aan te leggen. Dit is een overzicht van de beschikbare services, eventueel gecatalogiseerd per domein. Er kan allerlei informatie worden toegevoegd, zoals een beschrijving, service interface en eventueel gebruik van andere services. Services kunnen van elkaar gebruik maken, waardoor een service hiërarchie ontstaat.
ARIS heeft geen functionaliteit in de integratie-, applicatie- en datalaag.

7.4.2 Cordys

Cordys is niet opgebouwd vanuit een architectuurraamwerk, maar meer vanuit de techniek. Op het gebied van enterprise architectuur heeft het platform dan ook geen functionaliteit.
Voor de gebruikerslaag heeft Cordys mogelijkheden om gebruikersinterfaces te bouwen. Elementen waarin gegevens of keuzemogelijkheden geplaatst moeten worden, kunnen direct gekoppeld worden aan services die de input of output verzorgen. Verder kunnen gebruikers met de ‘Web Workplace’ allerlei algemene functionaliteiten, zoals een tekstverwerker gebruiken en kunnen processtappen uitgevoerd worden middels een takenlijst.
Met betrekking tot de procesmodellaag heeft Cordys de mogelijkheid om processen te modelleren in BPMN. Deze modellen worden getransformeerd naar BPML en geëxecuteerd in een eigen Process Execution Engine. De procesexecutielaag wordt dus ook goed ondersteund.
Cordys biedt tevens functionaliteit in de integratielaag, die ervoor zorgt dat elementen uit applicaties beschikbaar komen als Web Services. Deze kunnen direct gekoppeld worden aan processtappen bij het modelleren.
Cordys heeft geen functionaliteit in de applicatie- en datalaag.

7.4.3 Conclusie

Figuur 7.10 kan slechts gezien worden als een indicatie van de focus van de twee BPMSs. De gearceerde delen kunnen niet simpelweg opgeteld worden om een conclusie te trekken. Wel kan op basis van de vergelijking een belangrijk verschil aangestipt worden.

- Bij ARIS ligt de nadruk op procesverbetering. IDS Scheer geeft zelf aan dat haar product is gericht op ‘Business Process Excellence’, het zo effectief en efficiënt mogelijk laten verlopen van bedrijfsprocessen. De methodiek AVE en de best practices dragen hier ook sterk aan bij.
- Cordys daarentegen, is een integratie- en procesexecutieplatform, waar het uitgebreide mogelijkheden voor biedt. Men kan ook processen modelleren en monitoren, maar hierop ligt niet de focus.

Verder koppelt Cordys processen direct aan de techniek, ofwel aan ondersteunende services. Bij ARIS kan men dit indirect doen, door service interfaces te koppelen aan processtappen. Men kan er ook voor kiezen om dit in een executieplatform te doen. ARIS is hiermee onafhankelijk van een Process Execution Engine.
Onderzoeksbureau Forrester geeft aan dat een organisatie flexibeler is, indien processen niet verbonden zijn aan een executieplatform. In grote organisaties heeft men vaak meerdere executieplatforms, die hun eigen, vaak verschillende mogelijkheden hebben om processen te modelleren. Wanneer men het modelleren scheidt van

executie is men onafhankelijk van de implementatie en van de voorwaarden en beperkingen van de afzonderlijke executieplatforms [Forrester 2006].

Kortom, men kan ARIS het best gebruiken in de volgende gevallen:

- wanneer men bedrijfsprocessen wil ontwikkelen en verbeteren met als uitgangspunt de organisatie als geheel met bijbehorende enterprise architectuur;
- wanneer men bedrijfsprocessen wil ontwikkelen onafhankelijk van een executieplatform;

en men kan Cordys het best gebruiken in de volgende gevallen:

- wanneer men een integrale oplossing wil voor integratie, procesexecutie en procesmodelleren;
- wanneer het geen probleem is dat het ontwikkelen van bedrijfsprocessen afhankelijk is van de implementatie en de voorwaarden en beperkingen van een executieplatform.

8 Conclusies

Het SOA paradigma en de toepassing ervan in organisaties is in opmars en zal nog tijd nodig hebben zijn waarde te bewijzen. Ten eerste zullen de voordelen nog moeten blijken uit praktijksituaties waar men met SOA is begonnen. Dit zal nog zeker enkele jaren duren, aangezien het toepassen van SOA niet over één nacht ijs gaat, men een goed doordachte visie moet hebben en herbruikbaarheid pas tot zijn recht komt wanneer er integraal aan wordt bijgedragen. Ten tweede is de term SOA nog niet ingeburgerd en eenduidig.
Dit onderzoek vormt een bijdrage aan het structureren van de veelheid aan informatie rond SOA en een handvat aan de toepassing ervan in een organisatie.

Allereerst is het paradigma in zijn context geplaatst door deze te vergelijken met andere paradigma's in de softwareontwikkeling. Er is een model opgesteld van SOA, waarin de drie-eenheid service consumer, service provider en service directory is beschreven en er is een definitie gegeven van het paradigma.
Vervolgens is uitgelegd dat SOA de afstemming verzorgt tussen behoeftes die voortkomen uit bedrijfsprocessen en vermogens die worden aangeboden door services. Services zijn zo opgebouwd dat ze onafhankelijk zijn van implementatie. Op die manier zorgen ze voor ontkoppeling van processen en applicaties.

Om alle elementen in een organisatie en de bijbehorende architectuur die relevant zijn voor SOA een plaats te kunnen geven is een voorstel gedaan voor een architectuur in 7 lagen. Deze bevat een servicelaag die de ontkoppeling aangeeft tussen enerzijds processen en anderzijds applicaties.
Vervolgens is onderzocht in welke hoedanigheid SOA wordt toegepast bij het bedrijf Essent. Daaruit blijkt dat de elementen uit de definitie en uit het model terug te vinden zijn in de organisatie. Verder zijn er twee belangrijke verschillen tussen de 5 lagen architectuur van Essent en de voorgestelde 7 lagen architectuur. Ten eerste beschikt Essent niet over een aparte servicelaag. Ten tweede worden analytical modeling en executable modeling ondergebracht in één laag, waar de 7 lagen architectuur er twee heeft.

Bij het toepassen van SOA is een belangrijke rol weggelegd voor een BPMS. Een dergelijk platform is in staat om processen en IT op elkaar af te stemmen. In het onderzoek zijn de twee BPMSs ARIS en Cordys met elkaar vergeleken met behulp van de 7 lagen architectuur. ARIS biedt een oplossing voor het ontwikkelen en verbeteren van bedrijfsprocessen, gericht vanuit een organisatiecontext. Cordys biedt een integratie- en executieplatform, dat een integrale oplossing is voor meerdere architectuurlagen. In tegenstelling tot ARIS is het modelleren bij Cordys echter wel afhankelijk van de Process Execution Engine.

Zoals aangegeven zullen de voordelen nog moeten blijken uit praktijksituaties, maar er zijn wel verwachtingen. De verwachte voordelen zijn herbruikbaarheid, wendbaarheid en interoperabiliteit. De bijbehorende gevolgen zijn kortere ontwikkeltijden van services, lagere kosten, risicobeperking voor het maken van fouten, minder redundantie, kortere time-to-market, grotere beheersbaarheid van het IT-landschap en een hogere consistentie. De verwachte nadelen zijn problemen en moeilijkheden die zich kunnen voordoen bij governance, met alle kosten van dien, en het pas op de lange termijn profiteren van voordelen.

Tot slot kan geconcludeerd worden dat de 7 lagen architectuur de volgende bijdragen kan leveren wanneer men SOA toepast.

- Het vormt een handvat bij het maken van een keuze voor een BPMS. Wanneer men inzicht heeft in de huidige ondersteuning van IT in elk van de 7 lagen kan men een goede keuze maken voor een BPMS.
- Het vormt een referentiekader dat services centraal stelt. Als een organisatie start met het toepassen van SOA zal de visie die is ontwikkeld overgebracht moeten worden op de medewerkers. Iedereen zal moeten

meewerken aan het realiseren van herbruikbaarheid. De 7 lagen architectuur zorgt ervoor dat men hetzelfde beeld voor ogen heeft van de IT-huishouding en de centrale servicelaag geeft het belang van services aan.

- Het geeft de ontkoppeling aan tussen techniek en processen met behulp van de servicelaag. Hierin worden service descriptions vastgelegd in een service directory en kan een hiërarchie van services worden opgesteld. De servicelaag dient ook als uitgangspositie voor een data model en SOA governance.

Literatuur

[ARIS 2007] ARIS, http://www.aris.com, geraadpleegd 1 maart 2007.

[Automatiseringgids 2006] Automatiseringgids, *SOA bekend maar nog onbegrepen*, 22 juni 2006.

[BPMN 2006] Object Management Group, *BPMN specification*, 1 februari 2006, http://www.omg.org/cgi-bin/apps/doc?dtc/06-02-01.pdf.

[Bruce Silver 2005] Bruce Silver Associates, BPMInstitute.org, *The 2006 BPMS Report*, http://www.bpminstitute.org/bpmsreport.html, 2005.

[Channabasavaiah 2004] Channabasavaiah, K., Holley, K., Tuggle, E.M., *Migrating to a service-oriented architecture*, IBM whitepaper, april 2004.

[Cordys 2007] Cordys B.V., http://www.cordys.com, geraadpleegd 1 maart 2007.

[Davenport 1990] Davenport, T.H., Short, J.E., *The New Industrial Engineering: Information Technology and Business Process Redesign*, MIT Sloan Management Review, Volume 31, Number 4, 1990, p. 11-27.

[Essent 2007] Essent N.V., http://www.essent.nl, geraadpleegd 8 maart 2007.

[Forrester 2006] Forrester Research, *The Forrester Wave: Business Process Modeling Tools, Q3 2006*, Tech Choices, 29 september 2006.

[IDS Scheer 2006] IDS Scheer B.V., http://www.ids-scheer.com, geraadpleegd 8 maart 2007.

[IEEE 2000] The Institute of Electrical and Electronics Engineers, *IEEE Standard 1471-2000: IEEE recommended practice for architecture description of software-intensive systems*, ISBN: 0738125180, 2000.

[Linthicum 2003] Linthicum, D.S., *Next Generation Application Integration: from Simple Information to Web Services*, Addison-Wesley Longman Ltd., First edition, 2003.

[Marketcap 2006] Marketcap, *SOA & EAI survey*, maart 2006, gepresenteerd op het Computable SOA seminar op 21 september 2006.

[Meekels 2006] Meekels, S., *Business Process Modeling in BPI*, Master Thesis informatica, Radboud Universiteit Nijmegen, augustus 2006.

[OASIS 2006] OASIS OPEN, *Reference Model for Service Oriented Architecture 1.0, Committee Specification 1*, 2 augustus 2006, http://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf.

[OMG 2006] Object Management Group, *Definition of SOA*, april 2006, http://colab.cim3.net/cgi-bin/wiki.pl?SoaGlossary.

[Open Group 2006] The Open Group, *Definition of SOA*, 8 juni 2006, http://www.opengroup.org/projects/soa/doc.tpl?CALLER=index.tpl&gdid=10632.

[Pressman 2001] Pressman, R.S., *Software Engineering: A practitioner's approach*. McGraw-Hill New York, Fifth International Edition, 2001.

[Rijsenbrij 2004] Rijsenbrij, D.B.B., *Collegedictaat 'Inleiding Digitale Architectuur'*, Radboud Universiteit Nijmegen, http://www.digital-architecture.net/collegedictaat.htm.

[Shi 2001] Shi, P., Gandhi, S., *Enterprise Application Integration*, DiamondCluster International, Inc., Viewpoint volume 2 number 3, 2001, http://www.diamondconsultants.com/PublicSite/ideas/perspectives/downloads/viewpointv2n3EAI.pdf.

[Sogeti 2007] Sogeti B.V., *Service Oriented Architecture*, http://www.dya.info/Home/architectuur/index.jsp, geraadpleegd 30 maart 2007.

[Specht 2005] Specht, T., Drawehn, J., Thränert, M., Kühne, S., *Modeling Cooperative Business Processes and Transformation to a Service Oriented Architecture*, Proceedings of the Seventh IEEE International Conference on E-Commerce Technology, München 2005.

[WSBPEL 2006] OASIS OPEN, *Web Services Business Process Execution Language Version 2.0,* Public Review Draft, 23 augustus 2006, http://docs.oasis-open.org/wsbpel/2.0/wsbpel-specification-draft.pdf.

[WS-CDL 2005] W3C, *Web Service – Choreography Description Language version 1.0*, W3C Candidate Recommendation 9 november 2005, http://www.w3.org/TR/ws-cdl-10/.

Bijlagen

A. Huidige procesnotaties en executietalen

Op dit moment zijn er de volgende relevante procesnotaties en executietalen, die al dan niet gestandaardiseerd zijn door een standaardenorganisatie.

BPEL

De meest gebruikte procesexecutietaal op dit moment is BPEL, dat staat voor Business Process Execution Language. Het is een orkestratietaal die oorspronkelijk is ontstaan uit eigen talen van Microsoft en IBM en inmiddels is gestandaardiseerd door OASIS. BPEL is een XML (Extensible Markup Language) gebaseerde taal voor Web Services. De nieuwste versie, die is verschenen in september 2006, heet WS-BPEL 2.0 (Web Services BPEL). [WSBPEL 2006]

BPML

BPML is een procesexecutietaal die ontwikkeld is door het BPMI, een voormalige standaarden organisatie voor BPM, die inmiddels is gefuseerd met de OMG. BPML (Business Process Management Language) lijkt op BPEL en wordt niet meer onderhouden. Het Cordys platform maakt als een van de weinige leveranciers nog gebruik van BPML.

BPMN

BPMN staat voor Business Process Management Notation en is in beheer van de OMG. Het is een grafische procesnotatie met als doel processen inzichtelijk te maken voor business mensen. BPMN ondersteunt zowel choreografie als orkestratie. [BPMN 2006]

EPC

Een EPC (Event-Driven Process Chain) diagram is een grafische procesnotatie. Het is een notatie die afkomstig is van het ARIS platform van IDS Scheer. Processen worden hierbij beschreven met gebeurtenissen en functies (events en functions). [ARIS 2007]

WS-CDL

WS-CDL (Web Services – Choreography Description Language) is een choreografietaal voor Web Services en is ontwikkeld door het W3C. De taal is gebaseerd op XML en is niet grafisch. WS-CDL heeft als doel om de interactie tussen processen in verschillende organisaties te stroomlijnen en is onafhankelijk van de executietaal van de afzonderlijke processen. WS-CDL is niet executeerbaar, maar definieert hoe de interactie tussen participanten plaatsvindt. [WS-CDL 2006]

B. Begrippen

ANALYTICAL MODELING

Analytical modeling is het modelleren van de technische kant van een bedrijfsproces met behulp van een procesexecutietaal.

APPLICATIE / SYSTEEM

Een applicatie of systeem is een geheel of arrangement van elementen die zo zijn georganiseerd om door middel van het verwerken van informatie een bepaald doel te bereiken.

ARCHITECTUUR

Architectuur is de fundamentele organisatie van een systeem, uitgedrukt in zijn componenten, hun relaties met elkaar en met de omgeving en de principes die ontwerp en evolutie leiden.

BACK-END

De back-end is een abstractie voor applicaties en databases die op de achtergrond acteren en niet direct te maken hebben met gebruikersinteractie.

BEDRIJFSPROCES

Een bedrijfsproces is een verzameling logisch gerelateerde taken met een gedefinieerd eindresultaat.

BEHOEFTE

Een behoefte is iets dat een entiteit nodig heeft.

BUSINESS PROCESS EXECUTION

Business Process Execution is het uitvoeren van een instantie van een proces met behulp van een Process Execution Engine.

BUSINESS PROCESS INTEGRATION (BPI)

Business Process Integration is het mechanisme voor het managen van datastromen en het uitvoeren van bedrijfsprocessen in de juiste volgorde, waarbij processtappen worden ondersteund door applicaties.

BUSINESS PROCESS MANAGEMENT (BPM)

Business Process Management is de verzameling van activiteiten en het gebruik van methoden en software om bedrijfsprocessen te ontwikkelen, te beheren en continu te verbeteren.

BUSINESS PROCESS MANAGEMENT SUITE (BPMS)

Een Business Process Management Suite is een platform dat de BPM cyclus gedeeltelijk of geheel ondersteunt en meer functionaliteit biedt dan alleen Business Process Modeling.

BUSINESS PROCESS MODELING

Business Process Modeling is het op geformaliseerde wijze in kaart brengen van een bedrijfsproces.

BUSINESS PROCESS MODELING TOOL (BP MODELING TOOL)

Een Business Process Modeling tool is een platform voor analytical en / of executable modeling.

BUSINESS PROCESS MONITORING
Business Process Monitoring is het volgen en meten van het verloop van een proces en kan gebruikt worden voor procesverbetering.

BUSINESS RULES
Business rules (bedrijfsregels) leggen voorwaarden vast waaronder een bedrijf opereert. Ze bepalen welke keuzes gemaakt worden bij de uitvoering van processen.

CHOREOGRAFIE
Choreografie geeft het verloop van een proces weer, gezien vanaf een hoog abstractieniveau. Het beschrijft hoe processen in verschillende domeinen of organisaties op elkaar worden afgestemd.

COMPOSITE SERVICE
Een composite service is een service die gebruik maakt van één of meer andere services.

DESIGNTIME
Designtime is de tijd waarin een proces wordt ontworpen en geïmplementeerd.

DOMEIN
Een domein is een samenhangend deel (binnen een groter geheel, zoals een organisatie of branche).

ENTERPRISE APPLICATION INTEGRATION (EAI)
Enterprise Application Integration is het onbeperkt delen van informatie tussen twee of meer applicaties in een organisatie. Het is een verzameling technologieën die uitwisseling van informatie verzorgen tussen verschillende applicaties en bedrijfsprocessen in en tussen organisaties.

ENTERPRISE ARCHITECTUUR
Enterprise architectuur is het geïntegreerde geheel van 4 werelden: het bedrijfsgebeuren, het informatieverkeer, de applicaties en de technische infrastructuur.

ENTERPRISE SERVICE BUS (ESB)
Een Enterprise Service Bus is een gedistribueerde infrastructuur die gebruikt wordt voor integratie van applicaties en pretendeert speciaal te zijn toegespitst op SOA.

ENTITEIT
Een entiteit is ‘iets dat wezenlijk bestaat’; in dit onderzoek wordt gedoeld op een persoon, organisatie(deel) of systeem(deel).

EXECUTABLE MODELING
Executable modeling is het modelleren van de business kant van een bedrijfsproces met behulp van een procesnotatie.

FRONT-END
De *front-end* is een abstractie voor applicaties die met gebruikers interacteren.

GEAUTOMATISEERDE PROCESSTAP
Een geautomatiseerde processtap is een processtap zonder gebruikersinteractie.

GOVERNANCE
Governance is het toezien op de afspraken die men heeft gemaakt met betrekking tot SOA, zoals het conformeren aan standaarden, het gebruiken en beschikbaar stellen van services en het toewijzen van eigenaren aan services.

GRANULARITEIT
Granulariteit is de mate van detaillering en de omvang van de functionaliteit van een service.

INTERACTIEVE PROCESSTAP
Een interactieve processtap is een processtap met gebruikersinteractie.

MASTER DATA
Master data is een referentie van essentiële gegevens in een organisatie. Het gaat om gegevens over bijvoorbeeld klanten, producten en werknemers, die in meerdere applicaties of organisaties wordt gebruikt.

MASTER DATA MANAGEMENT (MDM)
Master Data Management is de discipline die zich richt op het beheren van master data en het toezien op het juist gebruik ervan.

ONTKOPPELING
Ontkoppeling (‘loose coupling’) duidt op het scheiden van processen en applicaties, van service interface en implementatie. Ontkoppeling kan ook betrekking hebben op de context, waarbij de service zo min mogelijk afhankelijk is van zijn context om breed toepasbaar te zijn, en op de locatie, waarbij het gebruik onafhankelijk is van de locatie van de service.

ORKESTRATIE
Orkestratie geeft het verloop van een proces weer vanuit het perspectief van één domein of organisatie. Het geeft de volgorde van processtappen aan en welke services hierbij betrokken zijn.

PARADIGMA
Een paradigma is een overkoepelend denkraamwerk, dat regels omvat die vertellen hoe je bepaalde problemen binnen gestelde grenzen kan oplossen.

PROCESEXECUTIETAAL
Een procesexecutietaal is de IT-representatie van een proces en wordt gebruikt bij executable modeling.

PROCESNOTATIE
Een procesnotatie is de businessrepresentatie van een proces en wordt gebruikt bij analytical modeling.

PROCESS EXECUTION ENGINE
Een Process Execution Engine is een runtime omgeving die instanties van processen executeert.

RULES ENGINE

Een Rules Engine is een beheersysteem voor business rules

RUNTIME

Runtime is de tijd waarin een instantie van een proces wordt geëxecuteerd.

SERVICE

Een service is een logische representatie van een herhaalbaar bedrijfsproces, dat een specifieke uitkomst heeft. Typische kenmerken van een service zijn:

- Een service opereert zelfstandig en is onafhankelijk (self-contained);
- Een service kan zijn opgebouwd uit andere services of kan hiervan gebruik maken;
- Een service is een 'black box' voor gebruikers van de service.

SERVICE CONSUMER

Een service consumer is een entiteit die een service gebruikt.

SERVICE DESCRIPTION

Een service description is een beschrijving van wat de service doet en hoe de interactie plaatsvindt.

SERVICE DIRECTORY

Een service directory is een verzameling van service descriptions, waarin service consumers kunnen zoeken.

SERVICE INTERFACE

De service interface beschrijft hoe de interactie plaatsvindt met een service. Het maakt onderdeel uit van de service description en bevat de specifieke protocollen, commando's en informatie-uitwisseling, waarmee acties worden uitgevoerd die in een effect resulteren.

SERVICE ORIENTED ARCHITECTURE (SOA)

SOA is een paradigma dat kan worden toegepast op een organisatie, waarbij:

- Services worden gebruikt om bedrijfsprocessen te ondersteunen;
- Services uit verschillende domeinen kunnen worden aangeboden;
- Services herbruikbaar zijn;
- De service interface ontkoppeld is van de implementatie van de service. Dit betekent dat de implementatie op ieder platform en in iedere taal mag, maar dat de interactie verloopt volgens standaarden.

SERVICE PARTICIPANT

Een service participant is een service consumer of service provider.

SERVICE PROVIDER

Een service provider is een entiteit die een service levert.

SINGLE SERVICE

Een single service is een service die geen gebruik maakt van andere services.

SOA MODEL

Met het SOA model wordt het samenspel bedoeld tussen service consumer, service provider en service directory.

VERMOGEN

Een vermogen is dat waartoe een entiteit in staat is, een bekwaamheid.

WEB SERVICE

Web Services zijn gedefinieerd door het W3C (World Wide Web Consortium), een internationaal consortium dat werkt aan standaarden voor internet. De service interface wordt beschreven met WSDL (Web Service Description Language) en de interactie tussen service participanten vindt plaats met SOAP (Simple Object Access Protocol), beide op basis van XML.

C. Afkortingen

AVE	ARIS Value Engineering
BPI	Business Process Integration
BPM	Business Process Management
BPMS	Business Process Management Suite
CDM	Common Data Model
CRM	Customer Relationship Management
EAI	Enterprise Application Integration
ESB	Enterprise Service Bus
GUI	Graphical User Interface
IT	Informatietechnologie
KPI	Key Performance Indicator
MDM	Master Data Management
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
SRM	Supplier Relationship Management
UML	Universal Modeling Language
WSDL	Web Service Description Language
XML	eXtensible Markup Language
XSD	XML Schema Definition