Re-engineering Legacy

in een veranderende software-architectuur

Universiteit van Amsterdam
Master Software Engineering
Masterproject

Marinus Geuze

Afstudeerdocent: Drs. H. Dekkers
Stagebegeleider: ing. B. Rebergen
Opdrachtgever: PinkRoccade Civility

5 augustus 2005

Samenvatting

Veel organisaties worden geconfronteerd met gedateerde software. Deze gedateerde software wordt ook wel legacy genoemd.

De steeds maar veranderde ICT-architecturen zorgen ervoor dat de legacy steeds verder gedateerd raakt. De kosten om de legacy te moderniseren en geschikt te maken voor nieuwe platformen en/of technieken worden steeds hoger.

Veel bedrijven passen daarom re-engineering toe op hun legacy software. Met re-engineering wordt bedoeld het onderzoeken, analyseren en veranderen van bestaande software zodat de functionaliteit, performance en implementatie van de software voldoet aan de huidig geldende ICT strategie.

PinkRoccade Civility ontwikkeld software voor overheden en heeft ook gedateerde software. Het doel van dit onderzoek is om een re-engineeringvoorstel te doen ten aanzien van deze legacy software. Hierbij rekening houdend met de huidige en nieuwe software-architectuur van PinkRoccade Civility.

PinkRoccade Civility is ontstaan uit het samengaan van een aantal bedrijven met ieder haar eigen software-architectuur. Hieruit is de huidige architectuur ontstaan die bestaat uit een samenstel van de verschillende softwareoplossingen.

De nieuwe software-architectuur van PinkRoccade Civility is servicegeoriënteerd en ontsluit data en functionaliteit via services. Verder spelen procesgeoriënteerde applicaties een belangrijke rol in de nieuwe architectuur. In procesgeoriënteerde applicaties staat het werkproces centraal. PinkRoccade Civility wil de nieuwe architectuur realiseren met het SAP NetWeaver en Microsoft platform. Het SAP NetWeaver platform staat centraal in dit onderzoek.

De onderzoeksvraag van dit project is:

Onderzoek hoe de data en functionaliteit van de legacy applicaties ontsloten kan worden met het SAP NetWeaver platform in de nieuwe servicegeoriënteerde architectuur van PinkRoccade Civility.

In dit project is gekozen voor een praktijk gerichte benadering. De stappen in het project zijn onderzocht en getoetst door middel van een prototype. De case betreft de dienst “aangifte van een geboorte”. Deze dienst wordt geleverd door de gemeentelijke afdeling Burgerzaken.

Het onderzoek heeft geresulteerd in een voorstel voor de migratie van de legacy van PinkRoccade Civility. Dit voorstel bestaat uit twee delen. Het eerste deel betreft een voorstel voor datageoriënteerde applicaties die bedoeld zijn ter vervanging van de legacy. Het tweede deel betreft een voorstel voor procesgeoriënteerde applicaties die gebruik gaan maken van deze nieuwe datageoriënteerde applicaties. Het eerste voorstel is zowel vanuit de theorie als in de praktijk getoetst, het tweede voorstel is alleen theoretisch uitgedacht.

Het voorstel heeft voordelen voor PinkRoccade Civility. De technische afhankelijkheid tussen de legacy en hun platformen (IBM iSeries en Unix platform) is geen beperkende factor meer. De mogelijkheden tot het integreren van applicaties nemen toe, dit zowel tussen eigen applicaties als met applicaties van externe leveranciers. Verder ondersteund dit een gefaseerde migratie van de legacy en zijn pilot projecten mogelijk met daarin zowel de bestaande legacy als de nieuwe applicatie. Ten slotte maakt het voorstel duidelijk dat procesgeoriënteerde applicatiebouw mogelijk is.

Een risico aan het voorstel is dat het SAP NetWeaver platform vrij nieuw is. Hierdoor is nog niet alle functionaliteit volledig toereikend of beschikbaar. Dit laatste zorgt ervoor dat het architectuurvoorstel voor procesgeoriënteerde applicaties slechts theoretisch aangetoond kan worden.

Tevens geeft het voorstel PinkRoccade Civility de mogelijkheid een migratiestrategie uiteenzetten voor de legacy. Dit in relatie tot het huidige en toekomstige productenportfolio. Op basis van de voordelen die behaald kunnen worden is de kans groot dat het voorstel wordt toegepast.

Inleiding

Dit afstudeerverslag beschrijft het masterproject van Marinus Geuze. Ik volg de opleiding Master Software Engineering aan de Universiteit van Amsterdam. Het masterproject is uitgevoerd in opdracht van PinkRoccade Civility. PinkRoccade Civility ontwikkelt applicaties voor gemeenten, provincies en waterschappen.

Het masterproject is gericht op het beantwoorden van de onderzoeksvraag, welke betrekking heeft op legacy applicaties van PinkRoccade Civility. Onderzocht wordt hoe de data en functionaliteit van de legacy applicaties ontsloten kan worden in de nieuwe servicegeoriënteerde architectuur van PinkRoccade Civility.

Het afstudeerverslag is als volgt opgebouwd. Hoofdstuk 1 bestaat uit een beschrijving van de aanleiding en achtergronden van de opdracht. De opdracht wordt nader geformuleerd in hoofdstuk 2. Vanuit de opdrachtformulering is het onderzoek uitgevoerd, deze staat beschreven in hoofdstuk 3. In hoofdstuk 4 zijn de resultaten van het onderzoek uitgewerkt. Als laatste is een lijst met literatuurverwijzingen opgenomen. In de bijlagen worden de volgende onderwerpen nader toegelicht: de servicegeoriënteerde architectuur, webservices, SAP NetWeaver, soorten legacy en technieken voor de analyse van legacy applicaties.

Ik heb met veel plezier aan mijn afstudeeropdracht bij PinkRoccade Civility gewerkt. Iedereen die daaraan heeft meegewerkt, wil ik bedanken; in het bijzonder mijn begeleiders Hans Dekkers en Bertil Rebergen.

Ik hoop dat u mijn afstudeerverslag met veel plezier leest.

Ing. Marinus Geuze

Inhoudsopgave

Samenvatting **2**
Inleiding **3**
Inhoudsopgave **4**
1. Aanleiding en achtergronden **5**
1.1. Aanleiding 5
1.2. Achtergronden 5
2. Opdrachtformulering **8**
2.1. Onderzoeksvraag 8
2.2. Doelstellingen 8
2.3. Aanpak 8
2.4. Scope 9
3. Uitvoering **10**
3.1. Analyse huidige architectuur, legacy en nieuwe architectuur 10
3.1.1. Huidige architectuur *10*
3.1.2. Legacy *12*
3.1.3. Nieuwe architectuur *13*
3.2. Ontsluiting legacy 15
3.2.1. Datalaag *15*
3.2.2. Business logicalaag *16*
3.2.3. Presentatielaag *16*
3.3. (On)mogelijkheden SAP NetWeaver platform 17
3.3.1. NetWeaver Developer Studio *17*
3.3.2. Java *18*
3.3.3. J2EE en Webservices *18*
3.3.4. Web Dynpro *19*
3.3.5. Composite Application Framework *19*
3.4. Data ontsluiting in SAP NetWeaver platform 21
3.4.1. J2EE Data persistence *21*
3.4.2. Keuze data persistence *22*
3.4.3. Bevindingen prototype *24*
3.5. Herbouw legacy functionaliteit in SAP NetWeaver platform 25
3.5.1. Bevindingen prototype *26*
3.6. Architectuurvoorstel 28
3.6.1. Datageoriënteerde applicaties *28*
3.6.2. Procesgeoriënteerde applicaties *29*
4. Resultaten **30**
Literatuurlijst **32**
Bijlagen **33**
Bijlage I - Servicegeoriënteerde architectuur 33
Introductie *33*
Wat is een Servicegeoriënteerde Architectuur? *33*
Hoe werkt de Servicegeoriënteerde Architectuur? *33*
Bijlage II - Webservices 35
UDDI *35*
WSDL *36*
SOAP *37*
Bijlage III - SAP NetWeaver 38
Bijlage IV - Legacy 40
Bijlage V - Analyse legacy 41

1. Aanleiding en achtergronden

1.1. Aanleiding

PinkRoccade Civility is ontstaan uit het samengaan van een aantal bedrijven. Deze bedrijven hadden elk hun eigen software-architectuur. Deze draaien zowel op het IBM iSeries platform als het Unix platform.

In 2000 heeft PinkRoccade Civility het besluit genomen om haar productlijnen voortaan op basis van het SAP platform of het Microsoft platform te realiseren. Naar aanleiding van dit besluit heeft PinkRoccade Civility een nieuw architectuurontwerp uitgewerkt. De nieuwe architectuur is servicegeoriënteerd en ontsluit data en functionaliteit via services.

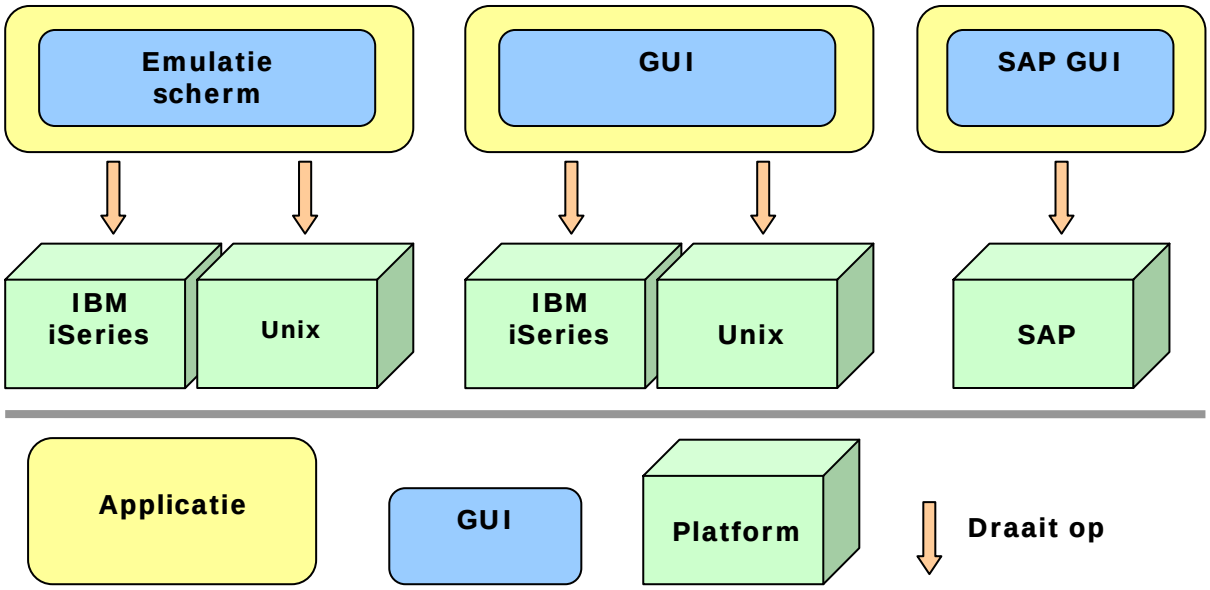
Een deel van de applicaties in de huidige architectuur is nog niet overgezet naar het SAP of Microsoft platform. Deze applicaties worden de legacy applicaties genoemd. Deze legacy applicaties zijn zeer belangrijk voor PinkRoccade Civility, maar zijn moeilijk te ontsluiten.

Dit stelt PinkRoccade Civility voor de vraag op welke manier de data en functionaliteit van de legacy applicaties het beste ontsloten kan worden in de nieuwe architectuur. Dit totdat deze applicaties zijn overgezet op het SAP of Microsoft platform. In dit onderstaat het SAP platform centraal.

1.2. Achtergronden

PinkRoccade Civility en haar voorgangers ontwikkelen sinds begin jaren ’80 applicaties voor gemeenten, provincies en waterschappen. Deze applicaties ondersteunen de ambtenaar bij de uitvoering van de verschillende werkprocessen binnen hun organisatie. Bij een gemeente kan hierbij gedacht worden aan Burgerzaken, Financiën en Belastingen. Deze (grotere) applicaties worden door PinkRoccade Civility taakspecifieke applicaties genoemd. Het aantal taakspecifieke applicaties dat door PinkRoccade Civility wordt aangeboden voor gemeenten ligt rond de 15, welke zijn opgebouwd uit verschillende deelsystemen. Daarnaast worden vele kleinere toepassingen ontwikkeld en in de markt aangeboden. De taakspecifieke applicaties voor de gemeenten staan centraal in deze opdracht. Ongeveer 300 gemeenten maken gebruik van één of meerdere van deze applicaties.

PinkRoccade Civility is door de jaren heen ervaren geworden in het ontwikkelen en beheren van taakspecifieke applicaties. Deze applicaties ondersteunen en sturen de werkprocessen van de gemeenten op een betrouwbare en eenduidige manier. De verantwoordelijkheden van de applicaties zijn duidelijk gescheiden over de verschillende werkterreinen. De schermen waarmee gewerkt wordt, zijn vertrouwd bij de betreffende ambtenaren. In de afgelopen jaren heeft PinkRoccade Civility haar legacy applicaties deels gemoderniseerd en voorzien van een moderne GUI (gebaseerd op screenpainting). Deze ontwikkeling is met open armen ontvangen en bereidt de klanten voor op de nieuwe toepassingen op basis van nieuwe architectuur. De applicaties worden van oudsher uitgeleverd op het IBM iSeries of Unix platform met bijbehorende ontwikkelomgeving. Sinds 2000 worden ook applicaties uitgeleverd op het SAP platform.



Figuur 1 - Huidige Architectuur

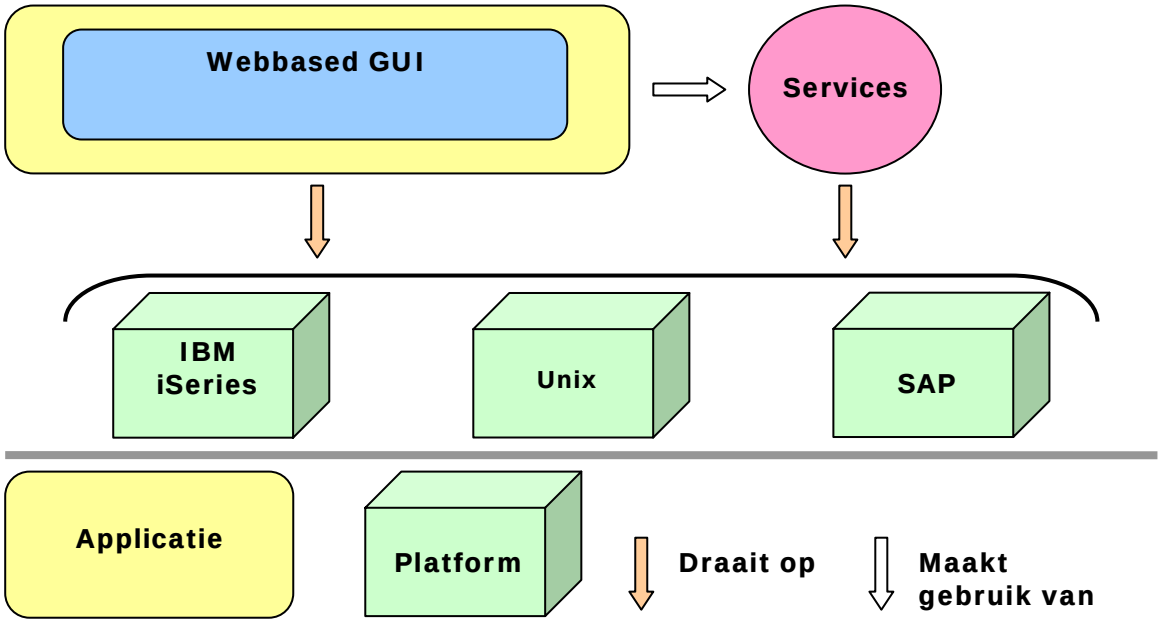
De applicaties voor het IBM iSeries en Unix platform zijn qua opbouw en functionaliteit vergelijkbaar, maar technisch strikt van elkaar gescheiden. Dit houdt in dat PinkRoccade Civility twee vergelijkbare productlijnen naast elkaar moet onderhouden. De kern van de applicaties in deze productlijnen is gedateerd. De applicaties worden ook wel legacy applicaties genoemd.

De code van deze legacy applicaties is moeilijk leesbaar en complex. Dit komt door de beperkingen die in de eerdere versies van de platformen bestonden en de vele wijzigingen en aanvullingen die op de code zijn aangebracht. Hierdoor wordt het steeds moeilijker en kostbaarder om aanpassingen of toevoegingen aan te brengen. Dit zorgt ervoor dat de kosten van het onderhoud toenemen en dat het maken van uitbreidingen op de legacy applicaties steeds minder rendabel wordt. Doordat uitbreidingen op de legacy applicaties kostbaar zijn, wordt het inspringen op nieuwe gebruikerseisen en nieuwe veelbelovende technologieën steeds moeilijker.

Verder is integratie tussen beide platformen en applicaties lastig. Dit komt omdat ze van oorsprong strikt zijn gescheiden. Deze integratie en de integratie met platformen en applicaties van andere leveranciers is voor PinkRoccade Civility erg belangrijk. Dit komt omdat de gemeenten steeds vaker kiezen voor een “multi vendor omgeving” (meerdere applicaties van verschillende leveranciers in één systeemlandschap). Hierdoor wordt connectiviteit en standaardisering steeds belangrijker.

De problemen hierboven geschetst zijn technisch gezien te verhelpen. Renovatieprojecten kunnen zorgdragen voor beter leesbare en minder complexe code. Tools om state of the art GUI’s te ontwikkelen zijn volop aanwezig. Verder is integratie tussen de verschillende platformen en applicaties mogelijk door applicatiespecifieke koppelingen te ontwikkelen. Los van het feit dat dit alles zeer kostbaar en tijdrovend zou zijn, biedt dit geen structurele oplossing. Namelijk bij alles moet rekening gehouden worden met het ontwerp dat al gedateerd is. Hierin zijn ontwerpbeslissingen opgenomen die vandaag de dag anders genomen zouden worden. Hierbij kan gedacht worden aan technologische beperkingen, datamodellering technieken, objectgeoriënteerd programmeren, procesgeoriënteerde applicatieontwikkeling, etcetera.
Verder dient bij een renovatie alles zo ontwikkeld te worden dat de beide platformen (IBM iSeries en Unix) ondersteund blijven.

Tegen deze achtergrond is PinkRoccade Civility gaan nadenken over een nieuwe architectuur. De nieuwe architectuur moest de technische afhankelijkheid tussen de legacy en hun platformen (IBM iSeries en Unix) wegnemen. De nieuw te ontwikkelen applicaties dienen dus zowel op het IBM iSeries als Unix platform te draaien, zonder dat daar tijdens de ontwikkeling hiermee rekening mee gehouden hoeft te worden. Verder moest het nieuwe platform webbased GUI's en de servicegeoriënteerde architectuur ondersteunen. Door het ondersteunen van webbased GUI's wordt het mogelijk om applicaties te ontwikkelen die op elke plek te gebruiken zijn waar een web browser is geïnstalleerd.
De servicegeoriënteerde architectuur maakt het onder andere mogelijk om de applicaties onderling te integreren. Dit wordt mogelijk gemaakt door applicaties als services aan te bieden en deze vervolgens in andere applicaties te gebruiken.



Figuur 2 - Nieuwe Architectuur

Het platform dat gekozen is om de nieuwe architectuur te implementeren is het SAP NetWeaver platform. Dit platform neemt de technische afhankelijkheid richting de onderliggende systemen weg en bovendien worden webbased GUI's en de servicegeorienteerde architectuur volledig ondersteund.

De nieuwe architectuur en het gekozen platform bieden veel mogelijkheden, maar tegelijkertijd is op een aantal cruciale onderdelen nog geen duidelijkheid. Zo zullen de nieuw te ontwikkelen applicaties gebruik gaan maken van de data en functionaliteit van bestaande applicaties, waaronder legacy applicaties. Uitgezocht moet worden hoe deze (legacy) applicaties gebruikt en opgenomen kunnen worden in de nieuwe architectuur. Verder dient uitgezocht te worden welke mogelijkheden en onmogelijkheden het SAP NetWeaver platform de nieuwe architectuur biedt.

De ICT-architecten en software-engineers van PinkRoccade Civility staan voor een geweldige professionele uitdaging. Deze uitdaging wordt enthousiast opgepakt en ik ben blij dat ik hier een steentje aan heb kunnen bijdragen.

2. Opdrachtformulering

2.1. Onderzoeksvraag

De onderzoeksvraag van dit project is:

Onderzoek hoe de data en functionaliteit van de legacy applicaties ontsloten kan worden met het SAP NetWeaver platform in de nieuwe servicegeoriënteerde architectuur van PinkRoccade Civility.

De onderzoeksvraag resulteert in de volgende onderzoeksgebieden:

- (Hoe) kan de data en functionaliteit van de legacy applicaties worden ontsloten?
- Welke (on)mogelijkheden biedt het SAP NetWeaver platform ten aanzien van de legacy ontsluiting?
- Welke invloed heeft de ontsluiting van de legacy met het SAP NetWeaver platform op de nieuwe architectuur?

2.2. Doelstellingen

PinkRoccade Civility heeft met het project de volgende doelstellingen:

- Het positioneren van de legacy in de nieuwe architectuur op het SAP NetWeaver platform.
- Het bepalen van een migratiestrategie voor de legacy in relatie tot het huidige en toekomstige productenportfolio. Dit op basis van de aangereikte onderzoeksresultaten.

2.3. Aanpak

Tijdens de start van dit project bleek geen duidelijke opdrachtomschrijving aanwezig te zijn. PinkRoccade Civility was begonnen met het opstellen van een nieuwe architectuur en het verkennen van het SAP NetWeaver platform. De vraag welke plaats de legacy innam in de nieuwe architectuur en platform was erg open. Besloten werd om deze vraag als uitgangspunt te nemen van het project.

Door het nieuw binnenkomen in de organisatie had ik geen kennis van de huidige architectuur, legacy en nieuwe architectuur. De hoeveelheid informatie die ik op moest nemen was dan ook enorm. Verder had ik een dergelijk project nog niet eerder uitgevoerd. Ten slotte was er een beperkte tijd (drie maanden) om het project tot een goed einde te brengen. Een goede aanpak van het project was van cruciaal belang.

De eerste stap in mijn project was het duidelijk krijgen van de daadwerkelijke vraagstukken en problemen rondom deze vraag. Ik ben begonnen met het achterhalen van de aanleiding en achtergronden die ten grondslag liggen van dit project. Dit was een uitdagend project op zichzelf. Deze uitdaging werd vooral veroorzaakt doordat er veel aspecten een rol spelen bij de legacy en de nieuwe architectuur. Achterhaald moest worden welke aspecten voor de opdrachtgever het belangrijkst waren.

Na het achterhalen van de daadwerkelijke vraagstukken en problemen ben ik begonnen met het formuleren van de opdracht. De opdracht heeft geresulteerd in de volgende projectstappen:

- Analyseren van de huidige architectuur, legacy en nieuwe architectuur;
- Onderzoeken hoe de legacy ontsloten kan worden in de nieuwe architectuur;
- Onderzoeken welke (on)mogelijkheden het SAP NetWeaver platform biedt;
- Onderzoeken hoe legacy data ontsloten kan worden in het SAP NetWeaver platform;
- Onderzoeken hoe legacy functionaliteit herbouwd kan worden in het SAP NetWeaver platform;
- Uitwerken van een architectuurvoorstel ten behoeve van de legacy ontsluiting.

In dit project is gekozen voor een praktijk gerichte benadering. De stappen in het project zijn onderzocht en getoetst door middel van een case, deze wordt uitgewerkt in een prototype. De case betreft de dienst “aangifte van een geboorte”. Deze dienst wordt geleverd door de gemeentelijke afdeling Burgerzaken.

2.4. Scope

De scope van het project bepaalt wat er al dan niet onderzocht gaat worden. Belangrijke vertrekpunten hierbij zijn de uitgangspunten van het project en de randvoorwaarden betreffende het project. Deze zijn opgesteld en opgenomen in het plan van aanpak.

Het onderzoek beperkt zich tot het volgende:

- Analyseproces “geboorteaangifte” in volgende applicaties:
  - CiPers (IBM iSeries legacy applicatie)
  - CiPers (Unix legacy applicatie)
- SAP NetWeaver Developer Studio onderdelen:
  - Composite Application Framework (CAF)
  - Web Dynpro
  - Webservices
  - J2EE
  - Java
- Analyse van complexiteit en omvang van de legacy.
- Conclusies betreffende de kwaliteit en de voorwaarden die verbonden zijn aan de uitgewerkte oplossing.

Het volgende wordt niet onderzocht in het project:

- Conclusies betreffende de risico’s, de tijdsbesteding en de kosten die verbonden zijn aan de uitgewerkte oplossing.
- Code van de legacy applicaties.

Het prototype betreft een proof of concept. Dit houdt in dat de werking van het prototype en de business rules in de business logica niet in detail correct hoeven te zijn.

Onderzoek naar de beste technieken en tools voor de analyse van de legacy valt buiten de scope van dit project. Voor de analyse van het proces “geboorteaangifte” wordt gebruik gemaakt van technieken in de categorie leveraging corporate knowlegde and experience [zie bijlage V].

3. Uitvoering

3.1. Analyse huidige architectuur, legacy en nieuwe architectuur

Het doel van deze stap is het analyseren van de huidige architectuur, legacy en nieuwe architectuur. Hierdoor wordt een goede basis gelegd voor de volgende stappen. Door de huidige architectuur en legacy te analyseren, wordt het mogelijk om stap 2 (het uitzoeken hoe de legacy ontsloten kan worden) uit te voeren. Door de nieuwe architectuur te analyseren wordt het mogelijk om duidelijk te krijgen welke invloed de (on)mogelijkheden van het SAP NetWeaver platform hebben op deze nieuwe architectuur.

De analyse heeft in de eerste plaats plaatsgevonden door gesprekken met mijn stagebegeleider en andere medewerkers van PinkRoccade Civility. Ook heeft een literatuurstudie plaatsgevonden. Hierin is informatie bestudeerd over de aspecten die verbonden zijn aan de huidige architectuur, legacy en nieuwe architectuur.

3.1.1. Huidige architectuur

PinkRoccade Civility ontwikkelt applicaties voor gemeenten. Deze applicaties ondersteunen de verschillende werkterreinen van een gemeente. Door de jaren heen heeft PinkRoccade Civility verschillende productlijnen gehad. Hieronder worden ze beschreven.

Gida lijn

De eerste reeks van applicaties van PinkRoccade Civility (destijds L+T Informatica genoemd) was de Gida lijn. De applicatie met de naam GidaB was bijvoorbeeld voor Burgerzaken bedoeld. Deze **Gida lijn** was alleen geschikt voor een IBM S36 platform. Kenmerken van de **Gida lijn** zijn: 5250 emulatie, RPGII, Cobol36, Flat Files (read/write statements).

Globit lijn

De opvolger van de **Gida lijn** was de **Globit lijn**, deze was alleen geschikt voor het IBM AS/400 platform, tegenwoordig IBM iSeries genoemd. De IBM AS/400 was ook geschikt voor de reeds bestaande **Gida lijn**. De applicaties in de **Globit lijn** bleven dan ook vaak nog gebruik maken van functionaliteit uit de bestaande **Gida lijn**.
Kenmerken van de **Globit lijn** zijn: 5250 emulatie, Cobol400, C en DB2/400 als RDBMS (SQL statements).

Gunim lijn

Voor de toepassingen op het Unix platform beschikt PinkRoccade Civility over de **Gunim lijn**. Deze lijn is oorspronkelijk ontwikkeld door Vuga en later verder uitgewerkt door Raet Decentrale Overheid.
Kenmerken van de **Gunim lijn** zijn: VT100 emulatie, MF-Cobol, C en C-ISAM als Flat Files formaat (read/write statements).

Deployment lijn

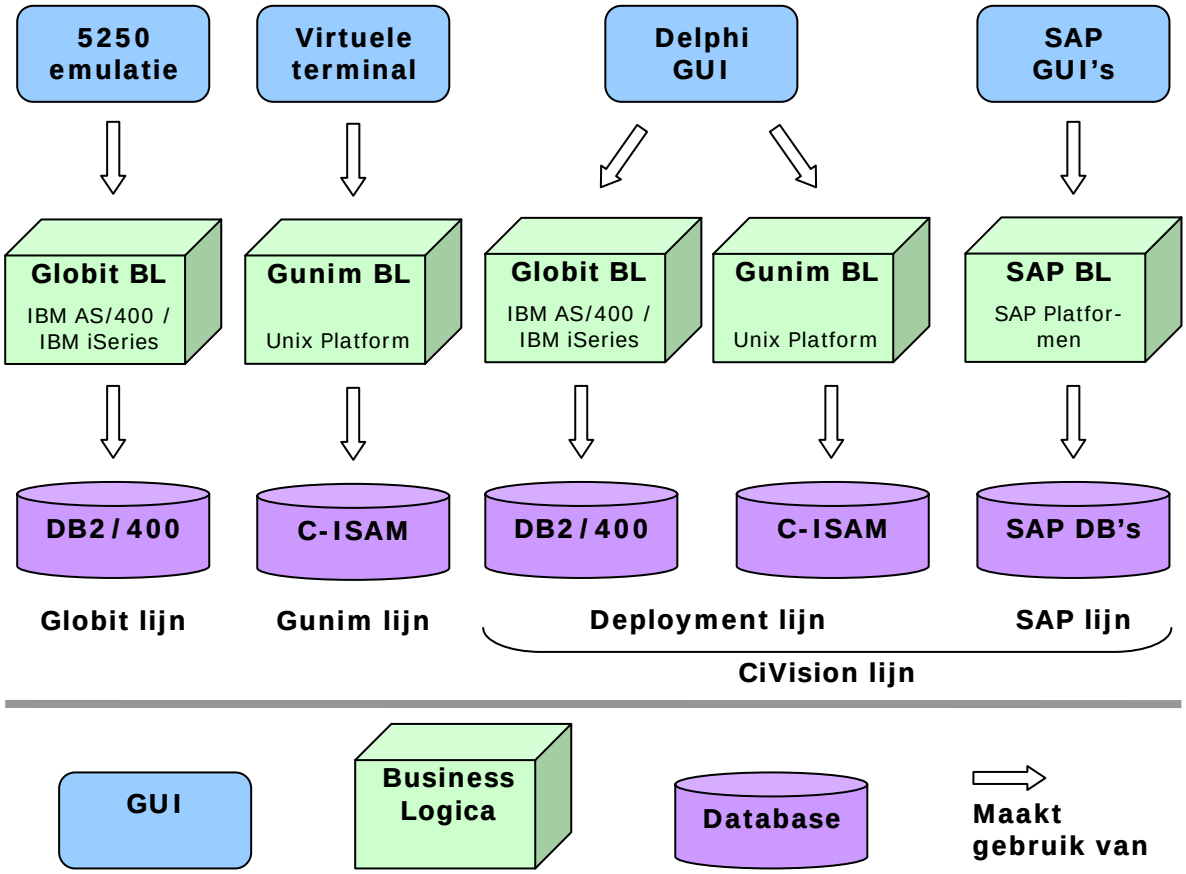
Inmiddels had PinkRoccade zowel L+T als Raet Decentrale Overheid ingelijfd en ondergebracht in de werkmaatschappij PinkRoccade Civility. De softwarelijnen **Globit** en **Gunim** kwamen hierbij samen. De applicaties in deze softwarelijnen zijn zowel in opbouw als in functioneel opzicht vergelijkbaar. Technisch zijn ze echter strikt gescheiden. PinkRoccade Civility heeft deze lijnen zoveel mogelijk naar elkaar toegebracht. Dit is gedaan door applicaties in deze lijnen client/server te maken, deze lijn wordt de **Deployment lijn** genoemd. De GUI van deze applicaties is in Delphi ontwikkeld en de database was DB2/400 of C-ISAM. Tussen de GUI en de database werd de reeds bestaande **Globit** of **Gunim** code geplaatst. Deze bestond uit de code van de oude **Glo-**

bit (en **Gida**) of **Gunim** lijn. Dit werd gedaan omdat deze bestaande code zeer complex en betrouwbaar was en qua functionaliteit nog steeds voldeed. Door gebruik te maken van de bestaande schermroutines kon de Delphi GUI gebruik maken van met de bestaande **Globit** of **Gunim** functionaliteit.
Applicaties die volgens deze opzet zijn opgebouwd zijn CiPers (Burgerzaken), CiVision Welzijn en Civision Basisregistratie.

SAP lijn
De huidige applicatieopzet van PinkRoccade Civility maakt gebruik van SAP. De nieuwe applicaties worden ontwikkeld door middel van customizing (gebruik maken en aanpassen van bestaande SAP componenten) of door middel van nieuwbouw (ontwikkelen van SAP applicatie gebruikmakend van ABAP, de interne programmeertaal van SAP). Met de komst van het SAP NetWeaver platform kan bij nieuwbouw ook gebruik gemaakt worden van de programmeertaal Java.
De volgende applicaties zijn gebouwd met SAP: Middelen, Innen (beide customizing) en Heffen, Kadaster, Waarderen en Subsidies (allen nieuwbouw).

CiVision lijn
De **SAP lijn** en de **Deployment lijn** zijn samengebracht en worden nu de **CiVision lijn** genoemd.

De huidige architectuur van PinkRoccade Civility ziet er schematisch als volgt uit.



Figuur 3 - Huidige Architectuur

3.1.2. Legacy

De Globit, Gunim en Deployment lijnen worden ook wel de legacy van PinkRoccade Civility genoemd. De code in deze lijnen is gedateerd. De kern van de legacy bestaat uit een datamodel en een spreekwoordelijke brei van code. Behoudens wettelijke wijzigingen wordt deze code niet of zelden gewijzigd. Enerzijds omdat PinkRoccade Civility haar legacy applicaties wil vervangen door applicaties volgens de nieuwe architectuur, maar ook omdat de gevolgen van wijzigingen niet goed in te schatten zijn. Dit komt onder andere door het feit dat er geen of weinig documentatie over deze code aanwezig is.

Een andere zeer belangrijke oorzaak is het feit dat de legacy code moeilijk leesbaar en complex is. Dit is onder andere veroorzaakt door het feit dat snelheidswinst vroeger (in verband met beperkingen platform) alleen behaald kon worden door de code te optimaliseren. Optimalisatie werd bereikt door de code te herschrijven en/of betere maar ook complexere algoritmes toe te passen. Dit was belangrijker dan leesbaarheid.
De leesbaarheid en complexiteit zijn verder verslechterd door de vele wijzigingen in de code die, al dan niet onder tijdsdruk, in veel gevallen minder logisch zijn gerealiseerd. Men ging op zoek naar een werkend geheel in plaats van het oorspronkelijke ontwerp van de applicatie te volgen. Door de vele iteraties ontstond hierdoor de eerder genoemde brei aan code.

Duidelijk is dat de legacy zeer belangrijk is, maar tegelijkertijd een zeer grote last vormt voor PinkRoccade Civility. Afgevraagd kan worden waarom de legacy nog niet herbouwd is. Hiervoor zijn verschillende redenen te noemen. Een belangrijke reden is economisch van aard. De kosten om de legacy te herbouwen zijn namelijk erg hoog. Doordat de herbouw van de legacy op kortere termijn geen voordeel oplevert, maar wel een grote investering is, is nog niet aan herbouw van alle legacy applicaties begonnen. Deze herbouw vindt gefaseerd plaats.

Een andere reden waarom de legacy nog niet herbouwd is, komt door het feit dat de overheid grote veranderingen in de wetgeving heeft aangekondigd. Hierbij kan gedacht worden aan de modernisering van de GBA, het overheidsthema “Andere Overheid” (hierin veel aandacht voor digitale dienstverlening en authentieke registraties), de mogelijke afschaffing van de WOZ en de mogelijke centralisatie van de verstrekking van uitkeringen. Niet duidelijk is wat precies gewijzigd gaat worden en welke impact deze wijzigingen gaan hebben. Het is in die gevallen dus onverstandig om vervangingen in gang te zetten.

Daarnaast speelt de slechte leesbaarheid en de complexiteit van de legacy code een rol om vervangingsinvesteringen uit te stellen. Wanneer de legacy herbouwd moet worden zal het moeilijk zijn om alle functionaliteit te achterhalen en opnieuw te ontwikkelen. Het risico op analyse-, ontwerp-, implementatiefouten en niet voorziene fouten is dan aanwezig.

De legacy, afkomstig uit twee afzonderlijke productlijnen (Globit en Gunim lijn), draait op verschillende systeemplatformen. De Globit lijn draait op het IBM iSeries platform, de Gunim lijn op het Unix platform.

De legacy op het IBM iSeries platform is hoofdzakelijk ontwikkeld in Cobol400. De legacy data wordt opgeslagen in een DB2/400 database. Het aantal actieve taakspecifieke applicaties is ongeveer 10 en het aantal regels code hiervan wordt geschat op ongeveer 10 miljoen.

De legacy op het Unix platform (HP-UX 11) is hoofdzakelijk ontwikkeld in MF-Cobol. Als database is gebruik gemaakt van C-ISAM (Informix) database. Het aantal actieve taakspecifieke applicaties is ongeveer 10 en het aantal geschatte regels code hiervan is 10 miljoen.

De legacy van PinkRoccade Civility, zowel de IBM iSeries en Unix legacy, valt onder de categorie online transaction programs [zie bijlage IV]. Een karakteristieke eigenschap van dit soort programma's is dat de presentatie-, business logica- en datalaag met elkaar verweven zijn.

3.1.3. Nieuwe architectuur

PinkRoccade Civility is vanuit de huidige architectuur gaan nadenken over een nieuwe architectuur. In de nieuwe architectuur staat het te ondersteunen gebruikersproces centraal. Voor de ontwikkeling van applicaties betekent dat er naast de ontwikkeling van datageoriënteerde applicaties ook procesgeoriënteerde applicaties ontwikkeld gaan worden.

Datageoriënteerde applicaties zijn gefocust op de data. De functionaliteit bestaat uit gegevensbewerkingen die gedaan worden op dataobjecten, zoals bijvoorbeeld een persoon of gebouw. Het verloop van het proces wordt in dergelijke applicaties door de gebruiker zelf bepaald. De legacy applicaties van PinkRoccade Civility zijn typische datageoriënteerde applicaties.

Procesgeoriënteerde applicaties zijn gefocust op het proces in plaats van op de data. Het proces bestaat uit processtappen en in deze stappen zijn gegevensbewerkingen nodig op de verschillende dataobjecten. Deze bewerkingen worden echter aangestuurd vanuit het proces en niet door de gebruiker zelf.

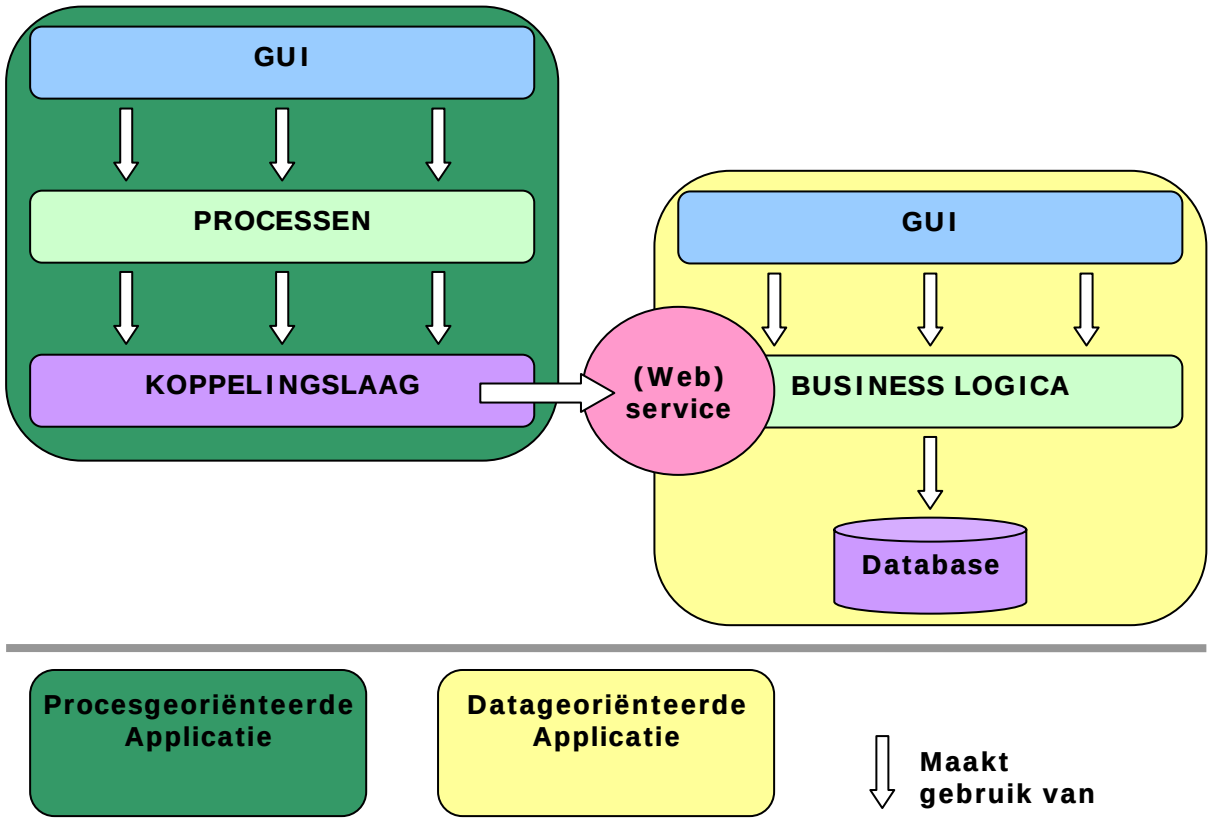
In de nieuwe architectuur wil PinkRoccade Civility naast deze datageoriënteerde applicaties ook procesgeoriënteerde applicaties ontwikkelen. Het ontwikkelen van deze procesgeoriënteerde applicaties verloopt geheel anders dan de traditionele datageoriënteerde applicatiebouw. Bij datageoriënteerde applicatiebouw is uitgegaan van drie lagen: data-, business logica- en presentatielaag. Alle applicaties bij PinkRoccade Civility zijn volgens dit principe opgebouwd.

Bij procesgeoriënteerde applicatiebouw wordt begonnen met het modelleren van procesmodellen. Vanuit deze modellen wordt de grafische interface ontwikkeld en indien mogelijk (gedeeltelijk) gegenereerd.

De grafische interface bevat geen business logica. De business logica die nodig is voor een bepaalde stap in een bepaald proces wordt gedefinieerd in het procesmodel. De benodigde business logica staat in nieuw te ontwikkelen of bestaande datageoriënteerde applicaties. Dit kunnen SAP applicaties zijn, maar ook legacy applicaties.

De koppeling tussen het procesmodel en de datageoriënteerde applicaties wordt gelegd in een koppelingslaag. Om de koppelingen in deze laag aan te kunnen leggen moeten de datageoriënteerde applicaties ontsloten worden als (web)services. Deze (web)services ontsluiten data en functionaliteit die aanwezig is in de betreffende applicatie.

De nieuwe architectuur ziet er vereenvoudigd als volgt uit:



Figuur 4 - Nieuwe Architectuur

De geschetste architectuur wordt de **Service-oriented Architecture** genoemd. Meer over de servicegeoriënteerde architectuur in bijlage I en II.

De servicegeoriënteerde architectuur wordt door PinkRoccade Civility opgezet met het SAP NetWeaver platform. Onderdeel van dit platform zijn Java, J2EE, Webservices, Web Dynpro en het Composite Application Framework (CAF).

SAP NetWeaver is een integratie- en applicatieplatform en heeft als basis de servicegeoriënteerde architectuur. Het platform is gebaseerd op Java en J2EE en is compatibel met Microsoft .NET en ondersteunt internet standaarden zoals HTTP, XML en webservices.
Een onderdeel van SAP NetWeaver is het Composite Application Framework. Dit framework maakt de integratie mogelijk tussen processen, grafische user interfaces en objecten. De procesmodellen en de koppelingslaag worden opgesteld in dit framework.
Web Dynpro, ook onderdeel van SAP NetWeaver, is een framework voor het ontwikkelen van user interfaces. Een user interface kan in Web Dynpro op een gebruiksvriendelijke manier worden gemodelleerd en vervolgens gegenereerd.
Een verdere beschrijving van het SAP NetWeaver platform en de genoemde onderdelen is te vinden in paragraaf 3.3 en bijlage III.

3.2. Ontsluiting legacy

De analyse van de nieuwe architectuur heeft duidelijk gemaakt dat de data en functionaliteit van de IBM iSeries en Unix legacy als services ontsloten worden. Uitgezocht moet worden hoe dit gerealiseerd kan worden. Hierbij moet rekening gehouden worden met het feit dat de druk toeneemt om een structurele oplossing te vinden voor de legacy. Dit doordat het onderhoud op en het aanpassen van de legacy steeds moeilijker is te realiseren. De oplossing zal het bij voorkeur mogelijk moeten maken om de legacy gefaseerd uit te faseren. Hierdoor zullen de als maar toenemende kosten voor het onderhoud op en het aanpassingen van de legacy afnemen.

Tijdens het scherp krijgen van de opdrachtformulering werd duidelijk dat het onderzoek twee kanten op kon. Enerzijds kon onderzocht worden hoe de legacy applicaties ontsloten kunnen worden als services. Anderzijds kon uitgezocht worden hoe de legacy applicaties het beste (gedeeltelijk) herbouwd kunnen worden.

Onderzoek naar de manier waarop de legacy applicaties als services ontsloten kunnen worden, bijvoorbeeld door wrapping, was een onderzoek op zich. Dergelijk onderzoek zou waarschijnlijk de gehele afstudeerperiode in beslag nemen.

Bij onderzoek naar de manier waarop de legacy applicaties (gedeeltelijk) herbouwd kunnen worden, dient nagegaan te worden op welke manier en welke onderdelen van een legacy applicatie het beste herbouwd kunnen worden in de nieuwe architectuur. Met onderdelen wordt bedoeld de data, functionaliteit en de GUI van een legacy applicatie.

In gezamenlijk overleg met de belanghebbenden is besloten dat onderzoek naar hoe de legacy herbouwd kan worden het meest interessant is voor PinkRoccade Civility. Deze keuze is het resultaat van het onderzoek naar de aanleiding en achtergronden die ten grondslag liggen aan dit project. De keuze houdt niet in dat het ontsluiten van de legacy als service geen juist alternatief kan bieden. Verder onderzoek moet dit uitwijzen, maar onderzoek hierna maakt geen onderdeel uit van dit project.

De analyse van de legacy maakte duidelijk dat de legacy van PinkRoccade Civility valt onder de categorie online transaction programs [Sneed02]. Een karakteristieke eigenschap van dit soort programma's is dat de presentatie-, business logica- en datalaag met elkaar verweven zijn. Dit heeft gevolgen voor het onderzoek naar hoe en welke onderdelen van de legacy het beste herbouwd kunnen worden.

3.2.1. Datalaag

De datalaag van de legacy bestaat uit twee delen. Één deel is de programmastatements met daarin de statements voor het ophalen en wegschrijven van de data (read/write of SQL statements). Het andere deel bestaat uit tabellen (IBM iSeries legacy) en databestanden (Unix legacy). Doordat de verschillende lagen met elkaar verweven zijn, is het deel met daarin de programmastatements niet bruikbaar. De tabellen en de databestanden zijn daarentegen redelijk gedocumenteerd en redelijk leesbaar.

Het voorstel voor de datalaag is als volgt. De nieuwe datageoriënteerde applicaties die ontwikkeld worden als vervanging van een legacy applicatie gaan gebruikmaken van de bestaande legacy tabellen en databestanden. Hierdoor kunnen andere legacy applicaties, die gebruikmaken van de betreffende legacy data, zonder aanpassingen blijven draaien. Ook wordt het hierdoor bijzonder goed mogelijk om pilot projecten op te starten. Gemeenten kunnen namelijk de nieuwe applicatie naast de bestaande legacy applicatie gebruiken. Implementatieproblemen kunnen hierdoor geminimaliseerd worden.

De statements voor het ophalen en wegschrijven van de data zullen in de nieuwe applicatie opnieuw ontwikkeld moeten worden. Hierbij moet rekening gehouden worden dat de data weggeschreven moet kunnen worden naar de IBM iSeries tabellen of de Unix databestanden. In de nieuwe applicatie zal uitgegaan moeten worden van een nieuw datamodel. Vanuit dit nieuwe datamodel wordt dan data weggeschreven naar één van beide bestaande datamodellen.

Stapsgewijze migratie van de legacy applicaties wordt door deze aanpak mogelijk. Wanneer alle afhankelijkheden van de legacy data zijn verdwenen, kan overgestapt worden naar een nieuwe (door SAP ondersteunde) database. Bij deze nieuwe database wordt uitgegaan van het nieuwe datamodel.

3.2.2. Business logicalaag

De business logica is door de verweving met de statements van de presentatie- en datalaag moeilijk te gebruiken. Los van deze verweving wordt alles nog verder bemoeilijkt door de slechte leesbaarheid en complexiteit van de code.

Het voorstel voor de business logica is om deze te herbouwen in de nieuwe datageoriënteerde applicatie. Tijdens het ontwerpen en bouwen van deze nieuwe business logica kan gebruik gemaakt worden van kennis die aanwezig is in de code en bij de onderhoudsmedewerkers van de betreffende legacy applicatie. Hierdoor moeten de risico's op analyse-, ontwerp-, implementatiefouten en niet voorziene fouten bij de herbouw van de legacy applicatie zoveel mogelijk ondervangen worden. Verder onderzoek naar hoe deze risico's ondervangen kunnen worden en hoe het beste de kennis uit de bestaande code en uit de onderhoudsmedewerkers gehaald en gebruikt kan worden, is zeker wenselijk maar valt buiten de scope van dit project.

3.2.3. Presentatielaag

De presentatielaag van de legacy applicatie is ook door de eerder genoemde verweving niet bruikbaar. Daarnaast is dit ook niet wenselijk, omdat PinkRoccade Civility toe wil naar een moderne en uitnodigende gebruikersinterface. De op terminal gebaseerde interface van de bestaande legacy past hierin niet thuis.

Voorgesteld wordt om de grafische user interface van de nieuwe datageoriënteerde applicatie opnieuw te ontwikkelen. Hierbij gebruikmakend van een webbased interface die zoveel mogelijk gemodelleerd en gegenereerd wordt.

3.3. (On)mogelijkheden SAP NetWeaver platform

In deze fase wordt onderzocht welke (on)mogelijkheden het SAP NetWeaver platform biedt. Hierbij wordt specifiek gekeken naar de tools en technieken die gebruikt worden bij het ontsluiten van de legacy data en de herbouw van de legacy functionaliteit en GUI.

Door de omvang van het SAP NetWeaver platform is het verkrijgen van antwoorden niet eenvoudig. Het platform bestaat namelijk uit vele onderdelen, tools en technieken. Tijdens de afbakening van het project is besloten alleen de volgende tools en technieken van het SAP NetWeaver platform te betrekken in het onderzoek: Java, J2EE, Webservices, Web Dynpro en CAF. Deze tools en technieken zijn allen onderdeel van de NetWeaver Developer Studio.

3.3.1. NetWeaver Developer Studio

De NetWeaver Developer Studio is de ontwikkelomgeving van het SAP NetWeaver platform. De Developer Studio bestaat uit drie delen [SAP]:

- Een integrated development environment (IDE);
- Een modeling environment;
- Een Java Development Infrastructuur.

De Developer Studio is gebaseerd op het IBM Eclipse Platform. Dit platform vormt een basis waarop ontwikkeltools ontwikkeld en gebruikt kunnen worden. Eclipse maakt het mogelijk om al deze verschillende ontwikkeltools samen te gebruiken, een soort toolkit dus. In de Developer Studio kunnen naast de ontwikkeltools geleverd door Eclipse en SAP ook ontwikkeltools van derden worden gebruikt.

Integrated development environment

De Developer Studio is onder andere een integrated development environment (IDE). Een IDE is een programma of set van tools dat de programmeur in staat stelt om een programma te schrijven, bewerken, compileren en in sommige gevallen te testen en debuggen [Apple]. Dit vanuit een geïntegreerde en interactieve omgeving.

De Developer Studio van SAP stelt een programmeur in staat om Java/J2EE programmatuur voor het SAP NetWeaver platform te ontwikkelen.

Modeling environment

Naast de mogelijkheid voor programmeren stelt de Developer Studio een programmeur in staat om te modelleren. Modelleren houdt in dat een implementatie op een hoger niveau wordt beschreven. Hierdoor wordt het mogelijk vanuit deze beschrijving de implementatie te laten genereren. Hierdoor kan men veel handmatig werk voorkomen.

Het automatisch genereren van code levert tijdsbesparing en minder fouten op. Doordat er in de implementatie de mogelijkheid is om eigen code toe te voegen aan de genereerde code kunnen ook specifieke, niet modelleerbare zaken geïmplementeerd worden.

Een mogelijk nadeel van gegeneerde code is dat een overhead in de code ontstaat waardoor de performance vermindert. Andere nadelen zijn; minder flexibiliteit tijdens het ontwikkelen en bij fouten of beperkingen in de generatietool ontstaat er een volledige afhankelijkheid ten aanzien van de bereidwilligheid van de leverancier om de fout of beperking in de tool te verhelpen.

Voorbeelden van implementaties die gemodelleerd kunnen worden zijn grafische user interfaces, processen, data objecten en relaties. Het modelleren kan worden gedaan in

een visuele omgeving, de Visual Composer genaamd. De Visual Composer is onderdeel van de Developer Studio. Met de Visual Composer kunnen de gemodelleerde representaties aangemaakt, gewijzigd, verwijderd, versleept en geplaatst worden. Web Dynpro en het Composite Application Framework (CAF) maken gebruik van de Visual Composer.

Het modelleren van een grafische user interface wordt in de Developer Studio gedaan met Web Dynpro. Door te modelleren in Web Dynpro hoeft een programmeur voor het maken van een grafische user interface minder code te schrijven en minder tijd te besteden aan het oplossen van fouten. Verder kunnen veranderingen in de user interface makkelijker en sneller worden gerealiseerd.

Het modelleren van processen, grafische interfaces en services en het onderling integreren van de gemodelleerde representaties wordt gedaan in het Composite Application Framework.

Modelleren gaat een steeds belangrijkere rol spelen in het SAP NetWeaver platform.

Java Development Infrastructuur
De Java Development Infrastructuur (JDI) ondersteunt de ontwikkelorganisatie met het beheren van de ontwikkelde code. De JDI biedt een source-code management systeem, een component build service en een change management service aan.

3.3.2. Java
Het Java gedeelte van de Developer Studio maakt gebruikt van de standaard Eclipse functionaliteit. Java applicaties of Java code kan ontwikkeld worden met de standaard Java Development Kit (JDK) van Sun. Voordelen en nadelen die kleven aan Eclipse en Java zijn ook van kracht op applicaties die ontwikkeld worden in het SAP NetWeaver platform. SAP heeft bepaalde nadelen willen ondervangen. Dit is gedaan door toevoegingen te maken op deze standaard functionaliteit.

Een belangrijke toevoeging is de “seamless access to data”. Seamless access to data wordt gerealiseerd door Open SQL. Open SQL biedt functies aan in Java die gebruikt kunnen worden om data te benaderen en te bewerken. De programmeur hoeft hierbij geen rekening te houden met de database die wordt gebruikt. Dit wordt namelijk afgehandeld door Open SQL en het SAP NetWeaver platform.

Open SQL is alleen te gebruiken voor de databases die ondersteund worden door het SAP NetWeaver platform. De databases die relevant zijn voor dit project (IBM’s DB2/400 en de C-ISAM datafiles) worden niet ondersteund.

Een andere toevoeging is bijvoorbeeld de Java Dictionary. Deze speelt geen actieve rol in het project en wordt daarom niet verder behandeld.

3.3.3. J2EE en Webservices
Het J2EE gedeelte van de Developer Studio maakt ook gebruik van de standaard Eclipse en Sun functionaliteit. Ook hier heeft SAP gezorgd voor extra functionaliteit.

Een belangrijke toevoeging met betrekking tot het project is het genereren van een webservice. Vanuit de in J2EE geprogrammeerde functionaliteit kan zonder extra inspanning een webservice worden gegenereerd. De SOAP berichten en WSDL beschrijvingen hoeven niet geprogrammeerd te worden. Dit neemt de ontwikkelaar veel werk uit handen. De (her)generatie van een webservice is veel gebruikt tijdens de ontwikkeling van het prototype.

Een andere toevoeging is het SAP Component model, deze speelt geen actieve rol in het project.

3.3.4. Web Dynpro

De tool voor het ontwikkelen van een Grafische User Interface (GUI) is Web Dynpro. Web Dynpro is geen standaard Eclipse of Sun onderdeel, maar is ontwikkeld door SAP. Web Dynpro maakt het mogelijk om door middel van modellering een user interface te genereren. De gegenereerde grafische user interface is webbased en in de verschillende internet- en mobiele browsers te benaderen.

Door standaard grafische user interface elementen aan te bieden en gebruik te maken van patterns kan een eenduidige grafische user interface voor een applicatie worden ontwikkeld. Waarin onder andere standaard foutafhandeling en caching mechanismen zijn opgenomen.

Verder maakt Web Dynpro een duidelijk onderscheid tussen de laag waarin de grafische user interface gemodelleerd wordt (design-time environment) en de laag waarin de daadwerkelijke uitvoering van een actie plaatsvindt (runtime environment). Hierdoor wordt het mogelijk om wijzigingen door te voeren in de runtime environment zonder dat daarvoor wijzigingen doorgevoerd hoeven te worden in de design-time environment. Hierdoor wordt het ook mogelijk om in een applicatie gebruik te maken van de diverse platformen, zoals J2EE, ABAP en Microsoft.NET. Het gebruik laten maken van de verschillende platformen wordt onder andere mogelijk gemaakt door het gebruik van webservices en RFC's (Remote Function Calls).

De scheiding tussen de design-time en runtime environment wordt bereikt doordat Web Dynpro gebaseerd is op het MVC pattern. Dit houdt in dat een in Web Dynpro ontwikkelde grafische user interface is opgebouwd uit een model, view en controller. Het model kan hierbij gebaseerd zijn op een J2EE, ABAP of .NET applicatie.

3.3.5. Composite Application Framework

Het composite application framework (CAF) maakt het mogelijk om nieuwe J2EE applicaties te creëren door middel van modellering, dit in plaats van codering. Het idee hierachter is dat de programmeur aangeeft op een abstracter niveau wat een applicatie moet doen en dat het CAF vervolgens de applicatie genereert.

Een belangrijk uitgangsprincipe van het CAF is dat gebruik gemaakt kan worden van functionaliteit uit bestaande applicaties. Bestaande applicaties bieden hun functionaliteit aan als services. De programmeur kan vervolgens aangeven waar deze services gebruikt moeten worden in de applicatie welke/die gemodelleerd wordt in het CAF. Het CAF maakt het dus mogelijk dat functionaliteit van verschillende applicaties, in verschillende programmeertalen, op verschillende platformen worden samengebracht in één applicatie.

Centraal bij het modelleren in het CAF staan de processen, grafische user interfaces, services, objecten en hun onderlinge relaties. Processen en de grafische user interfaces kunnen gebruik maken van services en objecten. Processen kunnen gekoppeld worden aan grafische user interfaces en kunnen zodoende door de gebruiker doorlopen worden. Dit alles wordt in het CAF mogelijk gemaakt door:

- User interface patterns en frameworks;
- Role- en process-based modeling;
- Guided procedures;
- Services en objecten.

User-interface patterns en frameworks

Patterns en frameworks worden veel gebruikt in software. Patterns in CAF zijn vaste patronen, sjablonen of modellen die gebruikt kunnen worden voor het genereren van grafische user interfaces. Deze patterns kunnen gebruik maken van gemodelleerde

rollen, processen, services en objecten. In het framework van CAF kunnen verschillende patterns met elkaar samenwerken. Vanuit deze (combinatie van) patterns kan de grafische user interface van een applicatie worden gegenereerd.

Role- en process-based modeling
Tijdens het modelleren van een applicatie in CAF kan de rol (role) van de gebruiker in een proces of de stappen in een proces centraal staan. De keuze voor het centraal zetten van de rol of de stappen in een proces is moeilijk. De volgende globale richtlijn zou aangehouden kunnen worden. Wanneer het gaat om een klein aantal rollen en de gebruiker heeft een centrale rol in het proces dan is role-based modeling aan te raden. Wanneer niet de gebruiker en zijn rollen de centrale rol spelen, maar eerder het proces, dan is een process-based applicatie aan te raden.

Guided procedures
De guided procedures geven de mogelijkheid om het proces in een composite application te modelleren. Guided procedures zijn opgebouwd in fases (phases) en deze fases zijn weer opgebouwd uit acties (actions). Een voorbeeld van een fase is de aangifte van een geboorte en een afgeleide actie daaruit is het invullen van de gegevens van de ouders van het kind.
Tijdens de uitvoering van een fase kan de guided procedure verwijzingen naar objecten, services, documenten en andere guided procedures verzamelen. Achter elke actie zit een service. Deze service kan bijvoorbeeld geleverd worden door een webservice.

Services en objecten
De services en objecten van de applicaties in en de applicaties verbonden met het SAP NetWeaver platform kunnen gebruikt worden in het CAF. CAF is volledig ingesteld om deze services en objecten gemakkelijk met elkaar te verbinden om zodoende een nieuwe composite applications te creëren.

Business objecten vormen de basis functionaliteit van de composite applications. Business objecten kunnen zelf functionaliteit bevatten of geven toegang tot functionaliteit die aangeboden wordt door externe services. Een voorbeeld van een business object is een persoon. Een persoon kan aangemaakt, gewijzigd en verwijderd worden, maar een persoon kan bijvoorbeeld ook een kind krijgen (geboorte aangifte). Dit alles kan gerealiseerd worden met business objecten.

Business engines zijn andere soorten objecten. Ze leveren een bepaalde functionaliteit, maar er bestaan geen meerdere kopieën van, zoals bij een business object bijvoorbeeld wel mogelijk is (meerdere personen). Een business engine kan bijvoorbeeld een BTW-berekening doen.

Business processes vormen een uitbreiding op business objecten. Een business proces maakt het mogelijk dat bepaalde procesgerelateerde informatie gekoppeld blijft aan het business object. Een voorbeeld hiervan is een geboorteakte van een persoon.

Objecten kunnen gebruikt worden in grafische user interface patterns en guided procedures. De objecten kunnen gebruik maken van webservices, maar ook zelf ontsloten worden als webservice.

3.4. Data ontsluiting in SAP NetWeaver platform

De data van de legacy wordt ontsloten in het SAP NetWeaver platform. Het ontsluiten van de data zal gebeuren met J2EE technologie. Het ophalen en wegschrijven van de data wordt in J2EE termen “data persistence” genoemd. Voor data persistence in het J2EE platform zijn drie technieken beschikbaar [Gabhart 03]: JDBC, Entity Beans en JDO.

3.4.1. J2EE Data persistence

JDBC wordt ingezet in combinatie met een Session Bean. Bij deze techniek moeten de persistence statements en persistence transacties worden geprogrammeerd. De programmeur zelf is dus verantwoordelijk voor de gehele cyclus van het ophalen en wegschrijven van data.

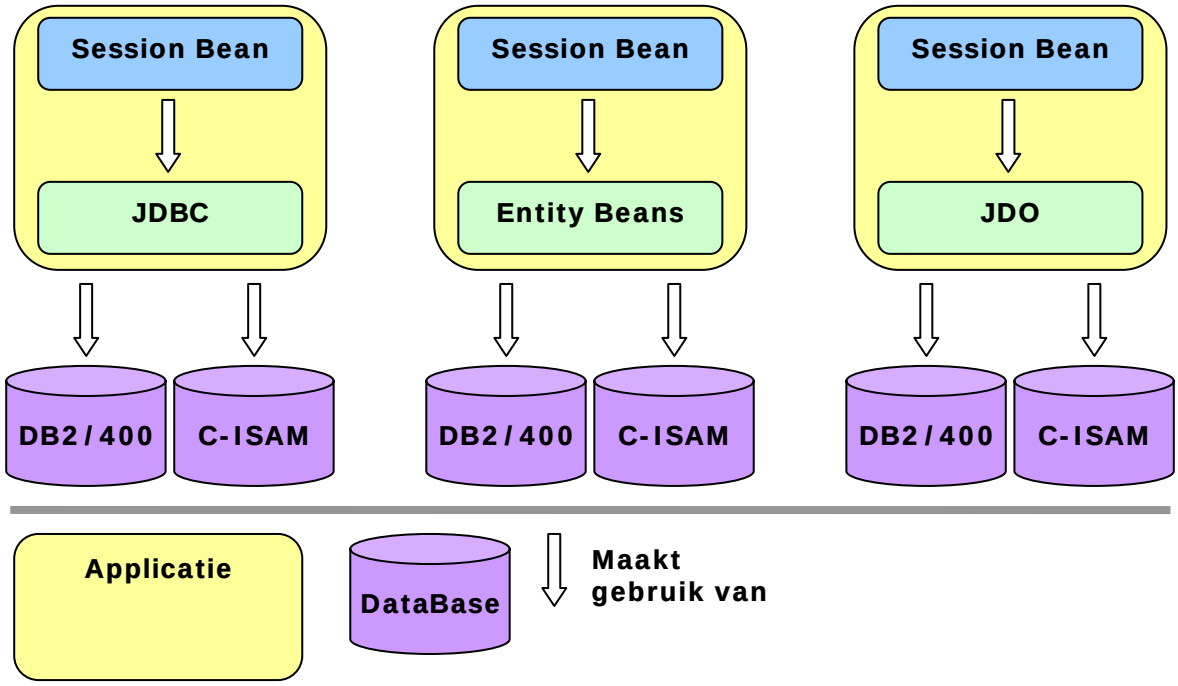
Entity Beans is een robuuste vorm van data persistence. Bij gebruik van Entity Beans wordt zorg gedragen voor data integrity, resource management and concurrency. Er zijn twee toepassingen van Entity Beans. Bean Managed Persistence (BMP) Entity Beans is een vorm waar de persistence statements geprogrammeerd moeten worden. De persistence transacties worden door de container afgehandeld. Bij Container Managed Persistence (CMP) Entity Beans worden zowel de persistence statements als persistence transacties door de container afgehandeld.

JDO (Java Data Objects) is een relatief nieuwe vorm van data persistence. JDO is opgezet vanuit de objectgeoriënteerde gedachte. De data wordt gezien als objecten. Deze objecten kunnen gebruikt worden om data op te vragen, te wijzigen of weg te schrijven.

Elke techniek heeft zijn eigen voordelen en nadelen. Onderstaand een tabel met een vergelijking van de drie technieken.

Persistence Type	Voordelen	Nadelen
JDBC	Ontwerp eenvoud Vergaande controle Volwassenheid Snelheid	Implementatie complexiteit Transacties handmatig Persistence statement handmatig
Entity Beans	Standaardisatie Container-managed Transparant Transactie ondersteuning Component gebaseerd ontwerp	Ontwerp complexiteit Lange build cyclus Responsetijden Resourcegebruik
JDO	Ontwerp eenvoud Vergaande controle Ontwikkeleenvoud Verbeterde productiviteit Objectgeoriënteerde persistence	Onvolwassen Transacties alleen via Session Bean beschikbaar

De oplossingen zien er schematisch als volgt uit:



Figuur 5 - Data ontsluiting

Het soort persistence heeft gevolgen voor de herbouw van de functionaliteit. Wanneer gekozen wordt voor JDBC dan zal al de business logica geplaatst worden in de Session bean. Wanneer gekozen wordt voor de objectgeoriënteerd technieken dan wordt de business logica verdeeld over de Session beans en Entity beans.

Duidelijk is dat er drie soorten data persistence technieken zijn: JDBC, Entity Beans en JDO. De Entity Beans techniek kan worden onderverdeeld in CMP en BMP.

3.4.2. Keuze data persistence

Onderzoek naar de data persistence in het SAP NetWeaver platform maakt duidelijk dat Entity Beans (CMP) en JDO geen opties zijn voor de ontsluiting van de legacy data. De reden hiervoor is dat de DB2/400 en C-ISAM databases niet ondersteund worden door de persistence laag van SAP. Alle databases die niet door SAP ondersteund worden, dienen buiten de persistence laag om te gaan. De opties die hierdoor overblijven zijn Entity Beans (BMP) en JDBC.

De keuze voor Entity Beans (BMP) heeft als voordeel dat overgestapt kan worden naar Entity Beans (CMP). Dit houdt in dat wanneer de legacy databases vervangen worden door een nieuwe database (welke ondersteund wordt door SAP), eenvoudig overgestapt kan worden naar Entity Beans (CMP). CMP heeft als groot voordeel dat geen rekening gehouden hoeft te worden met de onderliggende database. Want de persistence statements en transacties worden afgehandeld door de container. Dit zorgt voor tijdsbesparing en er kan gemakkelijk overgestapt worden tussen de verschillende databases. Een nadeel van BMP is dat voor ontsluiting van de huidige legacy data alles persistence statements geprogrammeerd moeten worden. Andere nadelen van zowel CMP als BMP zijn volgens [Gabhart 03] de slechte responsetijden en het resourcegebruik.

Een keuze voor JDBC heeft de volgende voor- en nadelen. Een groot voordeel is de snelheid van databewerking die de techniek biedt. Een groot nadeel is dat alle persistence statements en transacties handmatig geprogrammeerd dienen te worden. Verder dienen, wanneer overgestapt wordt op een nieuwe database, alle persistence statements en transacties herschreven te worden voor de nieuwe database.

[Gabhart 03] geeft vier keuzecriteria om een keuze tussen beide technieken te kiezen. Deze zijn: wijzigbaarheid dataontwerp, transactie ondersteuning, time to market, resourcegebruik en responsetijden.

Wijzigbaarheid dataontwerp
Met dit criteria wordt bedoeld hoe vaak het dataontwerp wijzigt. Wanneer dit vaak gebeurt, kan beter gekozen worden voor Entity Beans (vooral CMP). Bij een enkele wijziging in de zoveel tijd kan volgens [Gabhart 03] beter gekozen worden voor JDBC.

Transactie ondersteuning
Wanneer de standaard persistence voorzieningen van Entity Beans (CMP) ten aanzien van transacties voldoende is, dan valt de keuze voor CMP. Wanneer meer controle gewenst is, kan worden gekozen voor BMP. Voor volledige controle over de transacties moet gekozen worden voor JDBC.

Time to market
De snelste manier van ontwikkelen kan worden bereikt met CMP. Met welke techniek daarna het snelste ontwikkeld kan worden, is niet duidelijk vast te stellen. JDBC en BMP ontlopen elkaar namelijk niet veel.

Resourcegebruik en responsetijden
Wanneer dit criteria van belang is, moet gekozen worden voor JDBC. Het resourcegebruik is bij Entity Beans namelijk veel hoger en de responsetijden zijn veel langer.

Door een belang aan de keuzecriteria toe te kennen, kan bepaald worden welke techniek het meeste in aanmerking komt voor PinkRoccade Civility. Dit zal worden gedaan voor CMP, BMP en JDBC. CMP is zoals gezegd momenteel niet mogelijk. Maar wanneer de legacy volledig herbouwd zal zijn, zal overgestapt worden naar een door SAP ondersteunde database. Een eenvoudige switch van BMP naar CMP is dan mogelijk.

Aan de keuzecriteria wordt een belang gekoppeld. Aan het belang kan een waarde van 1 tot en met 5 toegekend worden. Hoe hoger de waarde hoe hoger het belang van PinkRoccade Civility. Aan een techniek, kijkend naar het betreffende keuzecriterium, kan een waarde van 1 tot en met 10 worden toegekend. Deze waarde wordt vermenigvuldigd met het belang van het betreffende keuzecriteria.

De toekenning van het belang is gedaan in samenwerking met de medewerkers van PinkRoccade Civility. Het resultaat is uitgewerkt in de onderstaande tabel.

Keuzecriteria	Belang
Wijzigbaarheid dataontwerp (WD)	2
Transactie ondersteuning (TO)	3
Time to market (TTM)	4
Resourcegebruik en responsetijden (RR)	5

Uit de bepaling van het belang blijkt dat de performance van een applicatie en de tijd om een applicatie op de markt te zetten, het meeste van belang zijn. De mate waarin de transactie ondersteund wordt en de wijzigbaarheid van het dataontwerp zijn van minder belang.

De volgende waardes zijn toegekend aan de technieken.

Techniek	WD (x2)	TO (x3)	TTM (x4)	RR (x5)	Totaal
JDBC	1	10	5	10	102
BMP	5	5	1	5	54
CMP	10	1	10	1	68

De uitslag van de technieken, welke/die afgezet zijn tegen de keuzecriteria plus belang, is duidelijk. Het toepassen van JDBC heeft de meeste voorkeur. Dit komt vooral omdat JDBC goed/hoog scoort op de keuzecriteria performance en transactieondersteuning. BMP komt slecht uit de score, terwijl dit het enige alternatief is voor JDBC.

De overweging om voor BMP te kiezen omdat deze na vervanging van de legacy omgezet kan worden naar CMP is op basis van de score niet interessant. Wanneer de legacy databases vervangen worden door een nieuwe database, dient op dat moment de keuze gemaakt te worden om eventueel alsnog CMP toe te passen.

Op basis van de literatuur wordt gekozen voor JDBC voor de ontsluiting van de legacy data. Het prototype moet uitwijzen of deze keuze juist was.

3.4.3. Bevindingen prototype

In het prototype zijn de technieken JDBC en BMP geïmplementeerd. Hierbij is rekening gehouden dat de data weggeschreven moet kunnen worden naar twee verschillende database platformen, namelijk DB2/400 en C-ISAM. Verder moest rekening gehouden worden met het feit dat na de vervanging van de legacy de bestaande legacy databases vervangen moeten worden door een door SAP ondersteunde database.

Het wegschrijven naar twee verschillende databases is mogelijk gemaakt met het factory pattern [Brown 02]. Dit pattern houdt het volgende in. De concrete implementaties van de classes die de data wegschrijven naar één van de databases worden verborgen achter een interface class. Deze interface class wordt gebruikt in de Session Bean. Tijdens het opstarten van de applicatie wordt in de Session Bean de interface geïnitialiseerd met de concrete implementatie die is gewenst. Door te communiceren met een interface hoeft de Session Bean niet aangepast te worden wanneer van database wordt veranderd. Het toepassen van dit pattern maakt het ook goed mogelijk om een nieuwe database toe te voegen aan de applicatie.

Het mogelijk maken dat de legacy databases vervangen worden door een SAP ondersteunende database is gerealiseerd door in het prototype uit te gaan van een nieuw datamodel. In de concrete implementatie classes voor het wegschrijven van de data wordt dit nieuwe datamodel vertaald naar het specifieke datamodel van de betreffende legacy database.

In het prototype is het switchen tussen databases en het vervangen van de legacy database door een SAP ondersteunende database getest. Dit verliep moeiteloos en de voorgestelde oplossingen werken dan ook in de praktijk. Hierbij is overigens niet de migratie van de data in de legacy database naar de SAP database meegenomen. Maar uit het feit dat vanuit het nieuwe datamodel data opgehaald en weggeschreven kan worden naar de legacy database kan gesteld worden dat dit mogelijk moet zijn.

Ten slotte maakte het prototype duidelijk dat BMP een onacceptabele performance levert. JDBC daar in tegen levert een goede performance. Ook werd duidelijk dat de snelheid van ontwikkelen met JDBC hoger ligt dan met BMP. Dit wordt veroorzaakt door de in de theorie genoemde ontwerpcomplexiteit, maar ook omdat Entity Beans aan een bepaalde vaste opbouw moeten voldoen. Dit zorgt voor extra werk tijdens het ontwikkelen.

3.5. Herbouw legacy functionaliteit in SAP NetWeaver platform

De functionaliteit van de legacy wordt herbouwd in het SAP NetWeaver platform. Hierbij zal gebruik gemaakt gaan worden van Java, J2EE, Webservices, Web Dynpro en het Composite Application Framework (CAF).

Java zal gebruikt worden om de data persistence te bouwen, hierbij gebruikmaken van JDBC. Deze Java code kan gebruikt worden in het J2EE gedeelte van de nieuwe applicatie.

In J2EE zullen vervolgens een of meerdere Session Beans worden aangemaakt waarin functies worden aangeboden die benodigd zijn voor de betreffende applicatie. Deze Session Beans maken gebruik van de Java classes die zorgen voor de data persistence. Wanneer gekozen was voor CMP of BMP dan hadden de Session Beans gebruik gemaakt van Entity Beans.

Alle business logica van de applicatie wordt geplaatst in het J2EE gedeelte. Hierdoor bevindt alle business logica zich op één plaats. Het voordeel hiervan is dat de business logica niet verspreid raakt over de data-, business logica- en presentatielaag. Een beschouwing van hoe/met betrekking tot op welke manier de business logica het beste ontworpen en ontwikkeld kan worden in het J2EE platform, valt buiten de scope van dit project.

De tools CAF en Web Dynpro maken gebruik van webservices om hun data en functionaliteit te verkrijgen. Deze data en functionaliteit van de webservice wordt geleverd door de Session Bean. Het creëren van een webservice op basis van een Session Bean wordt gedaan met behulp van de standaard functionaliteit in de Developer Studio. De WSDL's en SOAP berichten hoeven niet geprogrammeerd te worden, maar worden gegenereerd. Doordat alle business logica zich bevindt in de webservice, kan deze webservice door verschillende applicaties worden gebruikt. Hiermee wordt de servicegeoriënteerde gedachte gerealiseerd.

De combinatie van Web Dynpro en de webservice is ideaal voor het ontwikkelen van datageoriënteerde applicaties. De webservice (gebaseerd op de Session Bean) biedt de functionaliteit en data die nodig is voor de Grafische User Interface. Hiermee is een duidelijke scheiding bereikt/aangebracht tussen de presentatie- en de business logicalaag. De scheiding tussen de business logica- en datalaag is minder scherp. Deze scheiding dient duidelijk aangebracht te worden in het technisch ontwerp.

De combinatie van CAF en de webservice is ideaal voor het ontwikkelen van procesgeoriënteerde applicaties. Dit komt door het onderdeel Guided Procedures van de CAF tool. In de Guided Procedures kunnen processen gemodelleerd worden. Hierbij kan worden aangegeven welke data en functionaliteit in een bepaalde processtap benodigd is. Deze data en functionaliteit wordt geleverd door de webservice. Met het inzetten van CAF en de webservice kan één van de doelstellingen die PinkRoccade Civility met de nieuwe architectuur heeft worden bereikt, namelijk het ontwikkelen van procesgeoriënteerde applicaties.

3.5.1. Bevindingen prototype

De bevindingen van het prototype kunnen onderverdeeld worden in drie delen, namelijk het Java/J2EE/Webservice, Web Dynpro en CAF gedeelte.

Java / J2EE / Webservice

Naast dat in het J2EE gedeelte geëxperimenteerd is met data persistence, is ook gekeken/nagegaan hoe de databewerking gestructureerd kon worden. Hierbij is gekeken naar data locking, validatie en de commit of rollback van een transactie.

Door hun jarenlange ervaring met het ontwikkelen van datageoriënteerde applicaties hebben de medewerkers van PinkRoccade Civility veel kennis opgedaan op dit gebied. Deze kennis is gebruikt bij het ontwikkelen van het prototype.

Om data locking, validatie en de commit en rollback principes te realiseren is er een bepaalde structuur bedacht. Deze structuur is gerealiseerd in de ABAP omgeving. Deze structuur gaat uit van de volgende principes:

- De functies voor databewerking zijn: display, insert, update, delete en mget (geeft een lijst van objecten terug).
- Elke functie heeft een actie. De acties zijn: get4… (met deze actie worden een functie begonnen), save en cancel.
- Data locking vindt plaats voordat wordt begonnen met het wijzigen of verwijderen van de data (actie: *get4Update* of *get4Delete*).
- Validatie heeft plaats voor de data bewerkingsactie (actie: save). Bij een correcte validatie wordt een data bewerkingsactie uitgevoerd. Bij een niet correcte validatie worden één of meer foutmeldingen teruggegeven.
- Bij het succesvol uitvoeren van de data bewerkingsactie wordt een commit uitgevoerd, bij een niet succesvolle actie wordt een rollback uitgevoerd.
- De get4… actie kan geannuleerd worden met de actie cancel. In de actie cancel worden de data locks opgeheven.

De genoemde structuur met bijbehorende principes is gerealiseerd in het prototype. De structuur was redelijk eenvoudig te realiseren en werkt naar tevredenheid. In het prototype is hierbij de validatie daadwerkelijk gerealiseerd. De plaatsen waar de locking en commit/rollback moesten worden uitgevoerd, zijn geïdentificeerd maar niet gerealiseerd. Dit in verband met een gebrek aan tijd. Wel blijkt uit SAP documentatie dat data locking en commit/rollback functionaliteit beschikbaar is in JDBC. Bij de verdere ontwikkeling van het prototype moet deze functionaliteit daadwerkelijk geïmplementeerd worden. Een belangrijk punt van aandacht hierbij is dat de data locking mechanismen van de nieuwe en oude legacy applicatie met elkaar moeten samenwerken. Beide applicaties moeten namelijk naast elkaar gebruikt kunnen worden.

Web Dynpro

De realisatie van de datageoriënteerde applicatie voor het proces “Geboorteaangifte” was met de ontwikkeling van de webservice niet afgerond. De GUI moest ook nog ontwikkeld worden. Dit is gerealiseerd met Web Dynpro. De werking van Web Dynpro was relatief eenvoudig. Het MVC principe komt heel duidelijk terug tijdens het ontwikkelen van de GUI. Na het eigen maken van het principe kan snel en gemakkelijk ontwikkeld worden. Het meeste van de GUI kon worden gemodelleerd en gegenereerd. Speciale eisen aan de GUI werden gerealiseerd door code toe te voegen aan de gegenereerde code. Dit werd gedaan in de daar voor aangewezen plekken/plaatsen. Wanneer extra business logica gewenst was, werd deze toegevoegd aan de webservice. Er is dan ook geen business logica terug te vinden in de Web Dynpro code. De Web Dynpro code bevat enkel schermlogica.

Een belangrijke constatering die uit de bouw van het prototype getrokken kan worden, is dat de functionaliteit van Web Dynpro om gebruik te maken van webservices nog niet volledig toereikend is. Bijvoorbeeld een reimport van een webservice is niet mogelijk. Bij een wijziging van de webservice dient dus eerst het model (gecreëerd op basis van de webservice) verwijderd te worden en vervolgens opnieuw aangemaakt te worden. Vervolgens dient de connectie met de controller handmatig hersteld te worden. Dit probleem en andere zijn gemeld aan SAP. Uit de terugkoppeling vanuit SAP kan geconstateerd worden dat de verschillende problemen in toekomstige versies verholpen zijn. Deze versies kunnen binnen enkele maanden tot een jaar verwacht worden. De problemen hadden overigens niet tot gevolg dat het prototype niet ontwikkeld kon worden.

Indrukken van het ontwikkelde prototype zijn op te doen in bijlage VI.

CAF
Tijdens de ontwikkeling van het prototype was er geen commerciële versie van CAF beschikbaar. Hierdoor was het niet mogelijk om een procesgeoriënteerde versie van het prototype te ontwikkelen.

Doordat er wel een niet werkende versie van CAF beschikbaar was, was het wel mogelijk om een beeld van CAF te vormen. Hieruit is gebleken dat de verdere ontwikkeling van CAF nauwlettend in de gaten gehouden dient te worden en wanneer mogelijk verder moet worden verkend. Dit in de eerste plaats van belang omdat het procesgeoriënteerde applicatieontwikkeling mogelijk maakt. Dit was een van de doelstellingen die PinkRoccade Civility met de nieuwe architectuur wil bereiken.

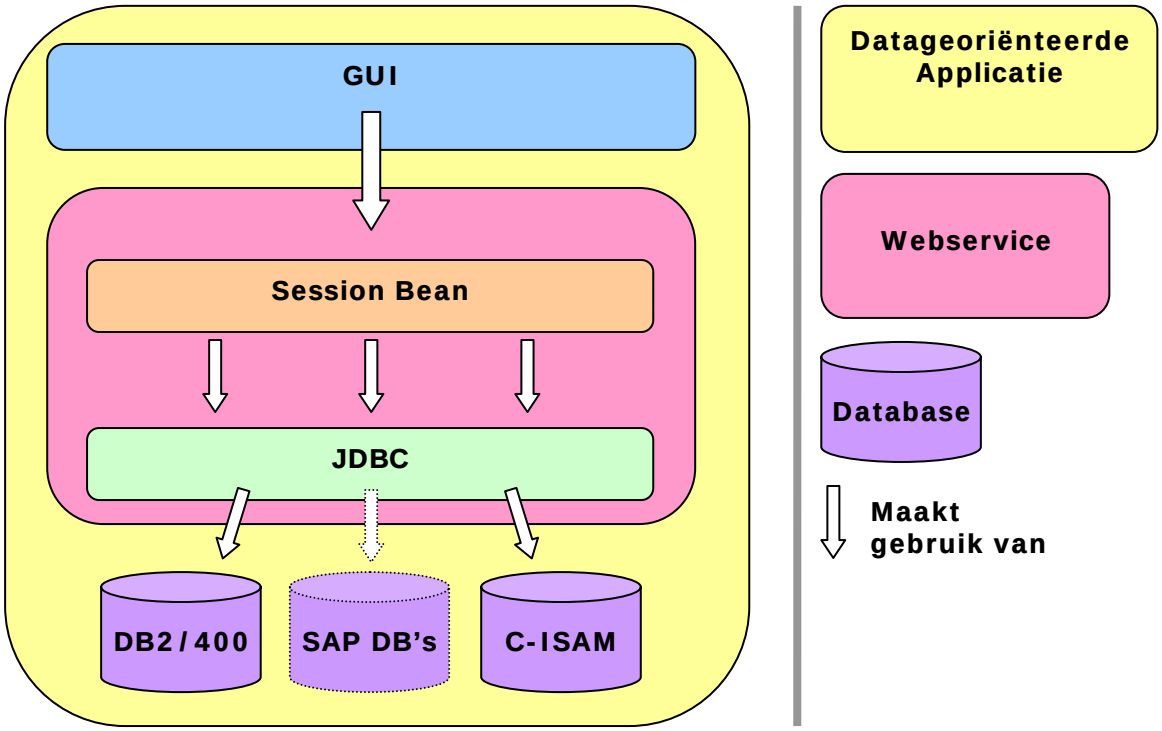
Verder is uit de verkenning van de niet werkende versie van CAF gebleken dat er de volgende verwachte voordelen zijn:
- Snellere software-ontwikkeling door modelleren in plaats van programmeren;
- Minder fouten door het modelleren in plaats van programmeren;
- GUI generatie vanuit het procesmodel voor delen van de applicatie. Naar het zich laat aanzien kan deze genereerde code worden bewerkt en aangevuld met Web Dynpro;
- Het kunnen samenvoegen van verschillende services (vanuit de verschillende datageoriënteerde applicaties) in één applicatie.

3.6. Architectuurvoorstel

De (on)mogelijkheden van het SAP NetWeaver platform ten aanzien van de legacy ontsluiting zijn onderzocht. Deze hebben invloed op de nieuwe architectuur. Het voorstel voor de nieuwe architectuur bestaat uit twee delen. Het eerste deel betreft een voorstel voor datageoriënteerde applicaties die bedoeld zijn ter vervanging van de legacy. Het tweede deel betreft een voorstel voor procesgeoriënteerde applicaties die gebruik gaan maken van deze nieuwe datageoriënteerde applicaties.

3.6.1. Datageoriënteerde applicaties

Het architectuurvoorstel voor datageoriënteerde applicaties ziet er schematisch als volgt uit:



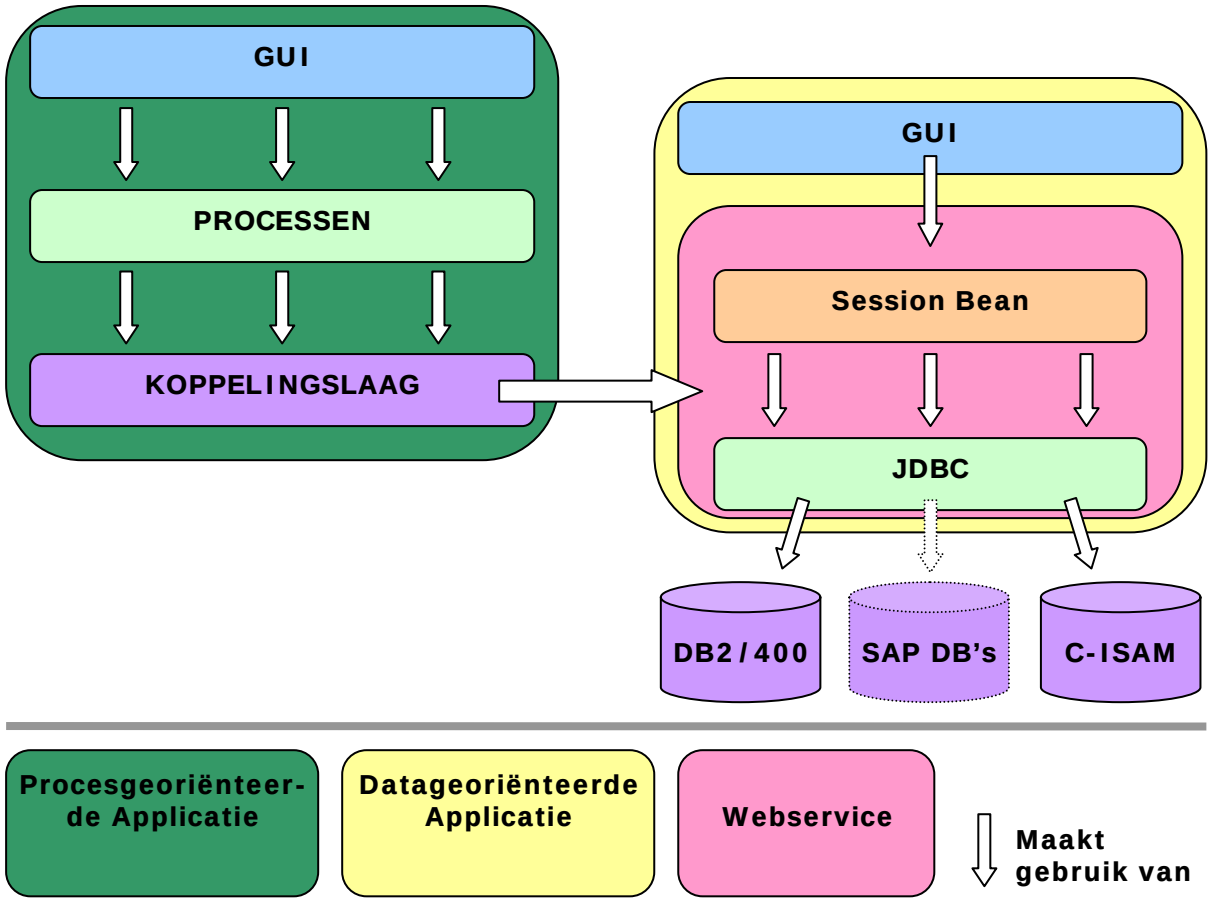
Figuur 6 - Architectuur datageoriënteerde applicaties

De voorgestelde architectuur voor datageoriënteerde applicaties is zowel theoretisch (gebaseerd op literatuurstudie) als in de praktijk mogelijk. De GUI wordt in Web Dynpro ontwikkeld en maakt gebruik van een webservice. De webservice is gegenereerd vanuit de Session Bean welke alle business logica bevat. Deze business logica maakt gebruik van de Java/JDBC laag om data op te halen en te bewerken. Door gebruik te maken van het factory pattern [Brown 02] kan tijdens de implementatie van applicatie aangegeven worden welke legacy database bij een betreffende gemeente aanwezig is. Wanneer alle afhankelijkheid van de data van de betreffende legacy applicatie is vervallen kan de data gemigreerd worden naar een door SAP ondersteunde database. Deze migratie is mogelijk omdat in de nieuwe applicatie is uitgegaan van een nieuw datamodel. Met de migratie is de laatste fase van de vervanging een legacy applicatie voltooid.

De voorgestelde architectuur is getest in het prototype. Vanuit het prototype kan geconcludeerd worden dat er geen technische beperkingen kleven aan de voorgestelde architectuur.

3.6.2. Procesgeoriënteerde applicaties

Het architectuurvoorstel voor procesgeoriënteerde applicaties ziet er als volgt uit:



Figuur 7 - Architectuur procesgeoriënteerde applicaties

De voorgestelde architectuur voor procesgeoriënteerde applicaties is mogelijk, zo blijkt uit de literatuur. Of de voorgestelde architectuur in de praktijk ook mogelijk is, moet in de toekomst blijken. Voor de ontsluiting van de functionaliteit wordt gebruik gemaakt van één of meerdere webservices die door de verschillende datageoriënteerde applicaties worden aangeboden.

In het Composite Application Framework (CAF) wordt de procesgeoriënteerde applicatie gemodelleerd en waar nodig aangevuld met Java code. Het modelleren begint bij het proces. Per proces wordt aangegeven uit welke processtappen het bestaat. Vervolgens wordt per processtap aangegeven welke data en functionaliteit nodig is. De koppeling tussen de benodigde data en functionaliteit en de webservices wordt gelegd in de koppelingslaag.

Vanuit de gemodelleerde processen kan vervolgens een GUI worden gegenereerd. Aangenomen mag worden, maar dit valt niet met zekerheid te zeggen, dat deze gegenereerde GUI aangepast en gewijzigd kan worden in Web Dynpro.

4. Resultaten

De resultaten van het onderzoek worden afgemeten aan de mate waarin de doelstellingen van het project zijn behaald. De doelstellingen van het project zijn:

- Het positioneren van de legacy in de nieuwe architectuur op het SAP NetWeaver platform.
- Het bepalen van een migratiestrategie voor de legacy in relatie tot het huidige en toekomstige productenportfolio. Dit op basis van de aangereikte onderzoeksresultaten.

Ad 1. Het onderzoek naar de positionering van legacy heeft geresulteerd in een duidelijk voorstel ten aanzien van de migratiestrategie van de legacy. Dit voorstel houdt in dat de datalaag van de legacy applicaties wordt ontsloten en de business logica- en presentatielaag worden herbouwd.

De legacy applicaties worden herbouwd naar nieuwe datageoriënteerde applicaties op het SAP NetWeaver platform. De ontsluiting van de datalaag wordt gerealiseerd met Java en JDBC. De business logica wordt ontwikkeld in J2EE met behulp van Session Beans en wordt ontsloten als webservice. Met Web Dynpro wordt vervolgens de presentatielaag gemodelleerd en gegenereerd.

Doordat de business logica wordt ontsloten als webservice, wordt het servicegeoriënteerde uitgangspunt van de nieuwe architectuur gerealiseerd. De strategie voor de migratie van de legacy applicaties naar de nieuwe datageoriënteerde applicaties is zowel vanuit de theorie als praktijk (prototype) getoetst. Hieruit blijkt dat de voorgestelde strategie uitvoerbaar/realiseerbaar is.

Op basis van het Composite Application Framework (CAF), onderdeel van het SAP NetWeaver platform, kunnen procesgeoriënteerde applicaties worden ontwikkeld. CAF kan hierbij gebruikmaken van de webservices die geleverd worden door de nieuwe datageoriënteerde applicaties. Hiermee is bereikt dat de data en functionaliteit van de legacy ontsloten is voor de nieuw te ontwikkelen procesgeoriënteerde applicaties.

Doordat geen werkende versie van CAF beschikbaar was tijdens het project is de strategie voor de ontwikkeling van procesgeoriënteerde applicaties alleen theoretisch onderbouwd. Er is dus niet met zekerheid te stellen dat de voorgestelde strategie in de praktijk uitvoerbaar is.

De manier waarop de nieuwe datageoriënteerde en procesgeoriënteerde applicaties ontwikkeld kunnen worden zijn uitgewerkt een architectuurvoorstel. Hiermee zijn open punten in de nieuwe architectuur ingevuld en is de eerste twee doelstelling behaald.

Ad 2. Tijdens het onderzoek zijn een aantal voor- en nadelen ten aanzien van de migratie van de legacy naar voren gekomen. Hierdoor kan iets gezegd worden over de te verwachten kwaliteit en de voorwaarden verbonden aan het architectuurvoorstel.

Het eerste voordeel is dat de legacy gemigreerd kan worden naar het SAP NetWeaver platform. Hiermee vervalt de technische afhankelijkheid tussen de legacy en hun platformen (IBM iSeries en Unix platform).

Verder kan de data en functionaliteit, afkomstig uit de legacy, gebruikt worden in andere applicaties. Dit zowel in eigen applicaties als in applicaties van externe leveranciers. Dit wordt bereikt doordat de nieuwe datageoriënteerde applicaties hun business logica ontsluiten met webservices. Door gebruik te maken van webservices hoeven geen applicatiespecifieke koppelingen meer te worden ontwikkeld, maar kunnen generieke koppelingen worden gerealiseerd.

Ook maakt het architectuurvoorstel een gefaseerde migratie van de legacy mogelijk. Hiermee worden de hoge kosten voor onderhoud en voor het maken van aanpassingen stapsgewijs lager.

Een ander voordeel van het architectuurvoorstel is dat het pilot projecten met daarin zowel de bestaande legacy als nieuwe applicatie mogelijk maakt. Hierdoor kunnen gemeenten de oude en nieuwe applicatie voor een korte of langere periode naast elkaar gebruiken.

Ten slotte maakt de voorgestelde architectuur procesgeoriënteerde applicatiebouw mogelijk. Hierdoor wordt een belangrijke doelstelling van PinkRoccade Civility met de nieuwe architectuur behaald.

Er kleven ook nadelen aan het architectuurvoorstel, namelijk het Java, J2EE, Webservice en Web Dynpro gedeelte in de NetWeaver Developer Studio is vrij nieuw. Dit bleek ook uit het feit dat de functionaliteit ten aanzien van Web Dynpro niet volledig toereikend is. Verder is er nog geen commerciële versie van CAF beschikbaar. Dit zorgde ervoor dat het architectuurvoorstel voor procesgeoriënteerde applicaties niet in de praktijk is getoetst.

Met het achterhalen van de voor- en nadelen is de tweede doelstelling van het project behaald. Namelijk er is inzicht verkregen in de te verwachte kwaliteit en de voorwaarden verbonden aan het architectuurvoorstel. Op basis hiervan kan PinkRoccade Civility een migratiestrategie bepalen voor de legacy in relatie tot het huidige en toekomstige productenportfolio.

Kortom met het behalen van de doelstellingen en het aantonen dat een succesvolle migratie van de legacy mogelijk is, kan gesteld worden dat het project succesvol is verlopen.

Signalen binnen PinkRoccade Civility wijzen erop dat de kans groot is dat mijn voorstel wordt overgenomen.

Buiten de behaalde resultaten is PinkRoccade Civility erg tevreden over mijn inzet en wijze waarop ik mijn werk heb uitgevoerd.

Literatuurlijst

[Apple]
http://developer.apple.com/documentation/Darwin/Conceptual/KernelProgramming/Glossary/chapter_21_section_1.html

[Brown 02] K. Brown. *Solve application issues with the factory pattern.* http://builder.com.com/5102-6386-1045719.html. 2002

[Gabhart 03] K. Gabhart. *J2EE pathfinder: Persistent data management.* http://www-128.ibm.com/developerworks/java/library/j-pj2ee3.html en http://www-128.ibm.com/developerworks/java/library/j-pj2ee4.html. 2003.

[He 03] H. He. *What is Service-Oriented Architecture?*. http://webservices.xml.com/pub/a/ws/2003/09/30/soa.html. 2003.

[Müller] H.A. Müller, J.H. Jahnke, D.B. Smith, MA. Storey, S.R. Tilley, K. Wong. *Reverse Engineering: A Roadmap.*

[SAP] D. Woods, J. Word. *SAP NetWeaver.* Wiley Publishing, Inc. 0-7645-6883-3.

[Sellink 98] A. Sellink, H. Sneed, C. Verhoef. *Restructuring of COBOL/CICS Legacy Systems*. http://www.cs.vu.nl/~x/cics/cics.html. 1998

[Sneed 02] H. Sneed. *Wrapping Legacy COBOL Programs behind an XML-Interface*. 2002.

[SOA] http://www.service-architecture.com/web-services/articles/service-oriented_architecture_soa_definition.html

Bijlagen

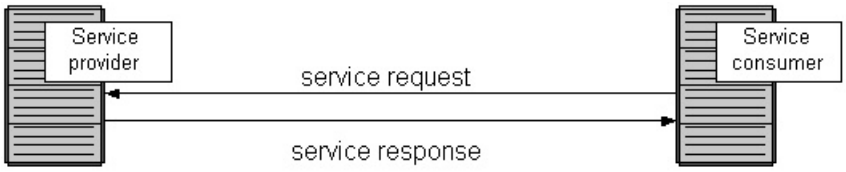
Bijlage I - Servicegeoriënteerde architectuur

Introductie

De servicegeoriënteerde architectuur (Service-Oriented Architecture - SOA) is ontstaan doordat de behoefte ontstond om de verschillende software systemen met elkaar te koppelen. De behoefte is ontstaan omdat bedrijven functionaliteit van bestaande systemen wil gebruiken in andere of nieuw te bouwen systemen. Dit resulteert in het feit dat het ene systeem afhankelijk is van functionaliteit in het andere systeem. Dit wordt een feitelijke afhankelijkheid (Real dependency) genoemd. Er is ook een technische afhankelijkheid (Artificial dependency); namelijk de verschillende systemen moeten technisch met elkaar communiceren. Het doel van de SOA is om deze technische afhankelijkheid zo ver mogelijk weg te nemen, dit wordt loose coupling genoemd. [He 03] stelt het volgende: “Artificial dependencies should be reduced to the minimum but real dependencies should not be altered.”

Wat is een Servicegeoriënteerde Architectuur?

De SOA is een architectuurstijl welke als doel heeft om loose coupling te bewerkstelligen tussen software systemen. In deze architectuur wordt de functionaliteit van de verschillende software systemen ontsloten als services. Een service kan een service aanbieden of een service aanroepen. De ene service kan dus een rol als aanbieder (provider) van gewenste functionaliteit vervullen, terwijl een andere service de rol als afnemer (consumer) van de gewenste functionaliteit vervult (zie figuur 1).



Figuur 8 – Service provider en service consumer [SOA]

Een voorbeeld van een SOA in het allerdaagse leven is het eten in een restaurant. Het restaurant biedt als service een maaltijd. Een klant in een restaurant geeft een order en verwacht als resultaat de bestelde maaltijd. De rollen van aanbieder en afnemer worden vervolgens ook omgedraaid. Want wanneer de maaltijd op is, verwacht het restaurant dat er betaald wordt, en is het daarmee dus een afnemer van het geld want de klant (provider) betaalt.

Hoe werkt de Servicegeoriënteerde Architectuur?

Om loose coupling tussen software systemen te bereiken heeft de SOA de volgende twee uitgangspunten [He 03]:

- Services hebben een makkelijke en voor één uitleg vatbare interface. Deze interfaces zijn op dezelfde manier opgebouwd en zijn universeel voor alle providers en consumers.
- De services communiceren met beschrijvende boodschappen (messages). De opbouw en structuur van en voorwaarden aan de beschrijvende boodschappen zijn algemeen bepaald. De boodschappen beschrijven geen, of een minimum aan, systeemgedrag.

Architecturen die voldoen aan de bovenstaande uitgangspunten zijn niet automatisch een SOA. Aan de volgende regels moet ook voldaan worden:

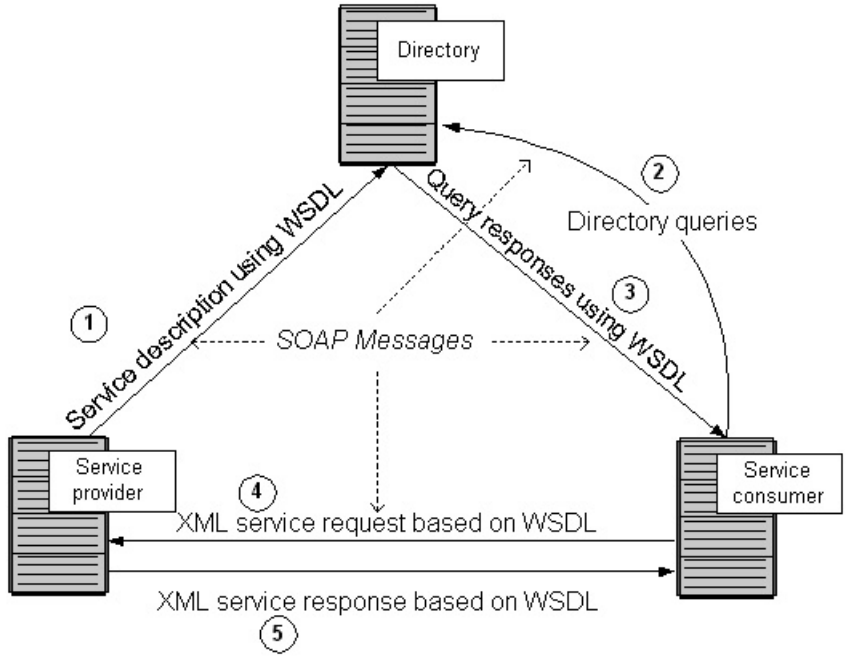
- De boodschappen moeten beschrijvend en niet instruerend zijn. Dit omdat de service zelf verantwoordelijk is voor het uitvoeren van de aanvraag. Kijkend naar het voorbeeld van het restaurant. Wanneer een order wordt opgenomen, geeft de klant aan wat hij wil bestellen en wat zijn voorkeur is, maar hij geeft niet aan hoe het vervolgens gekookt moet worden.
- De boodschappen moet qua opbouw, structuur en taal begrepen worden door alle services. Limitering van de functionaliteit van een taal waarin de boodschappen geschreven worden, is bepalend voor een efficiënte communicatie.
- De taal waarin de boodschappen wordt geschreven, moet uitbreidbaar zijn. Dit is nodig om nieuwe eisen en wensen aan de communicatie tussen de services mogelijk te maken. Deze regel is in tegenspraak met de voorgaande regel. Deze regels zijn dus conflicterend. Een goede balans dient gevonden te worden.
- De architectuur moet een mechanisme aanbieden om het voor een consumer mogelijk te maken een provider te vinden die een service biedt die gewenst is door de consumer. Dit mechanisme moet flexibel zijn: een centraal register moet dus niet verplicht zijn.

Om de communicatie tussen de software systemen in de SOA mogelijk te maken, wordt de gewenste functionaliteit van een systeem ontsloten als webservice.

Bijlage II - Webservices

WSDL (Web Service Description Language) vormt de basis van webservices. In figuur 2 wordt een schematische weergave gegeven van de werking van webservices. De weergave toont een service provider en consumer. De stappen voor het aanbieden en afnemen van een service zijn [SOA]:

- Een service provider beschrijft de service die hij levert, hierbij gebruikmakend van WSDL. Deze beschrijving wordt bekend gemaakt in een directory van servicebeschrijvingen. Deze directory maakt gebruik van UDDI (Universal Description, Discovery, and Integration), maar andere soorten directories kunnen ook worden gebruikt.
- Een service consumer verzendt een of meerdere verzoeken naar de directory om zo de locatie van de service en de wijze van communiceren met de service te achterhalen.
- Een deel van de beschrijving, in WSDL, van de betreffende service provider wordt door de directory teruggeven aan de service consumer. De beschrijving geeft de service consumer informatie over welke verzoeken (requests) en antwoorden (responsen) verwacht kunnen worden van de service provider.
- De service consumer kan vervolgens WSDL gebruiken om een verzoek (request) te verzenden naar de service provider.
- De service provider geeft vervolgens een antwoord (response) aan de service consumer, hierbij gebruikmakend van WSDL.



Figuur 9 - Webservices en WSDL [SOA]

UDDI

UDDI staat voor Universal Description, Discovery, and Integration. UDDI wordt gebruikt als register om bij te houden welke webservices een organisatie gebruikt. Een UDDI kan ook in een bredere zin ingezet worden. UDDI kan namelijk ook gebruikt worden om webservices die beschikbaar zijn op het internet, te registeren. Hierdoor wordt het mogelijk om services af te nemen van een organisatie die aan de andere kant van de wereld is gevestigd.

WSDL

Web Service Description Language (WSDL) wordt gebruikt om de interface van een webservice te beschrijven. WSDL bestaat uit drie delen [SOA]:

- definities;
- operaties;
- service bindingen.

Definities

De definities zijn onder te verdelen in twee typen: de datatype definities en de boodschap (message)definities. De boodschapdefinities maken gebruik van de datatype definities. De definities worden meestal geschreven in XML (eXtensible Markup Language), maar andere talen zijn ook toegestaan.

Operaties

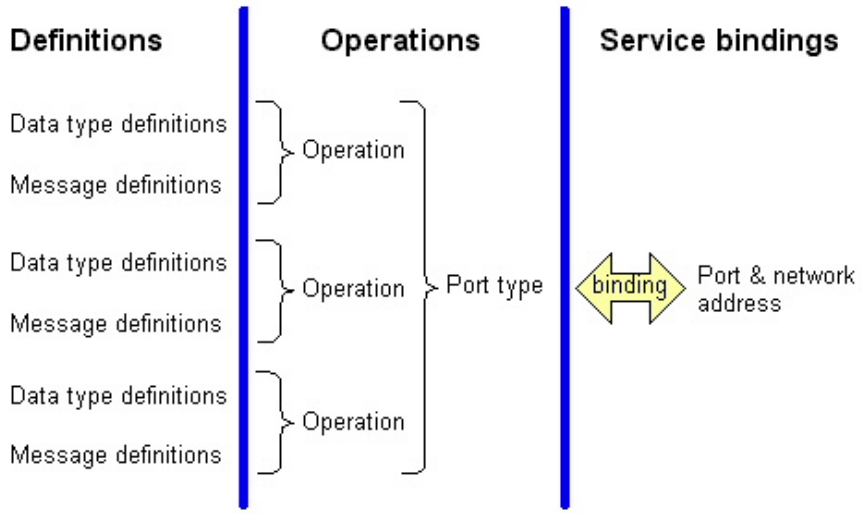
Operaties beschrijven het type boodschap (message). Er zijn twee soorten operaties:

- Eenzijdige boodschappen: Een boodschap wordt verzonden zonder dat een antwoord verplicht is.
- Verzoek/Antwoord-boodschappen: De zender zendt een verzoek en ontvangt een antwoord.

Andere soorten operaties zijn in de nabije toekomst te verwachten.

Operaties worden gegroepeerd in poorttypen. Een poorttype in combinatie met een netwerkadres vormt een poort. Dit wordt gedaan door servicebindingen. Een servicebinding kan worden gecreëerd met SOAP, maar andere vormen zijn ook mogelijk, hierbij kan gedacht worden aan IIOP, DCOM, .NET, etc. Alle servicebindingen samen vormen de webservice.

Figuur 3 geeft de samenwerking tussen definities, operaties en service bindingen schematisch weer.

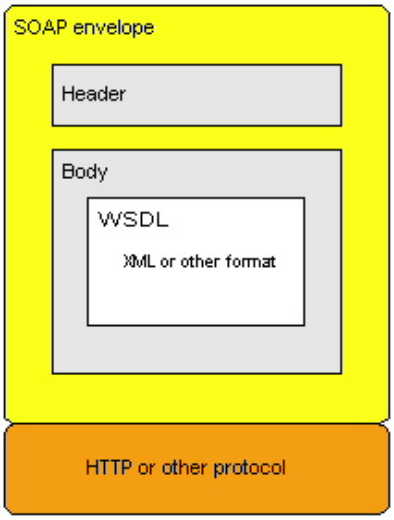


Figuur 10 – WSDL [SOA]

SOAP

SOAP kan gezien worden als het mechanisme dat het mogelijk maakt om webservice boodschappen (messages) over het netwerk of internet te versturen. SOAP verzendt de boodschappen als het ware via “enveloppen”. Een “envelop” bestaat uit twee gedeelten (zie figuur 4):

Het eerste gedeelte is een optionele header welke/die informatie verschaft over authenticatie, encryptie of hoe de ontvanger de SOAP boodschap zou moeten verwerken. Het tweede gedeelte, de body, bevat de boodschap. De boodschappen kunnen geschreven zijn in WSDL.
SOAP “enveloppen” worden normaal gesproken verstuurd over een HTTP protocol, maar kunnen ook verstuurd worden over andere protocollen. SOAP kan gebruikt worden om servicebeschrijvingen te versturen, maar ook voor het versturen van data.



Figuur 11 - SOAP envelop [SOA]

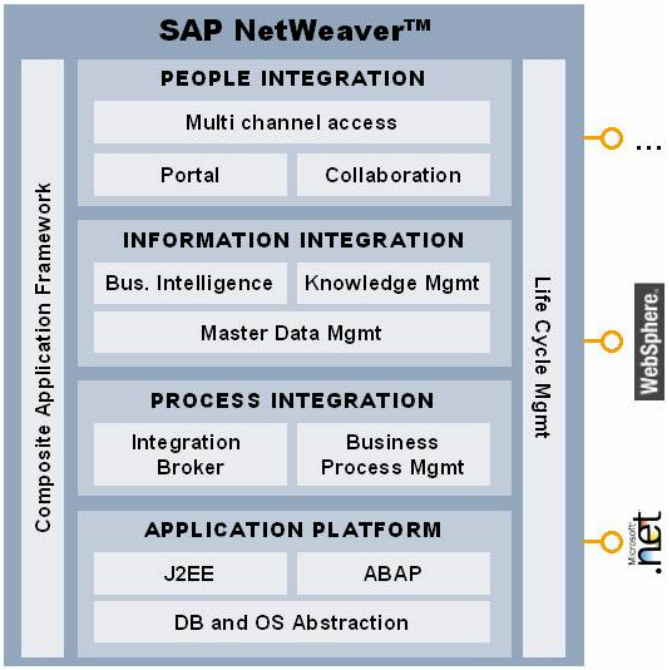
Opmerking: SOAP stond eerder voor het Simple Object Access Protocol. Sinds versie 1.2 hebben de letters in de afkorting geen bepaalde betekenis meer.

Bijlage III - SAP NetWeaver

Het SAP NetWeaver is een integratie- en applicatieplatform en kan als volgt worden gedefinieerd [SAP]:

SAP NetWeaver biedt een set aan componenten en tools om applicaties samen te kunnen laten werken en om nieuwe applicaties te ontwikkelen, die gebruik kunnen maken van bestaande applicaties. Dit wordt mogelijk gemaakt door de componenten en tools in het SAP NetWeaver platform, die ontworpen zijn om nauw te kunnen samenwerken.

Schematisch kan SAP NetWeaver als volgt worden weergegeven:



Figuur 12 - SAP NetWeaver [SAP]

De grafische weergave toont dat SAP NetWeaver als platform mensen (people), informatie (information) en processen (process) integreert. De integratie van mensen houdt in dat het platform het mogelijk maakt dat mensen efficiënter kunnen samenwerken. Informatie-integratie betekent dat informatie vanuit de verschillende locaties worden samengebracht. Procesintegratie zorgt voor de coördinatie van de workflow van de afdelingen, divisies en organisaties onderling. SAP NetWeaver maakt het ook mogelijk om deze integraties onderling te integreren.

De integratie wordt mogelijk gemaakt door verschillende componenten en tools. Voorbeelden van deze componenten zijn de SAP Enterprise Portal en de SAP Web Application Server en voorbeelden van tools zijn de NetWeaver Developer Studio en het SAP Composite Application Framework.
De SAP Enterprise Portal zorgt voor één centrale plek waaruit alle applicaties met één en dezelfde interface ontsloten kunnen worden. De SAP Web Application Server vormt de basis van SAP NetWeaver. Het is als het ware de motor achter alle SAP applicaties. Applicaties voor de SAP Web Application Server kunnen ontwikkeld worden in de interne SAP taal ABAP en de programmeertaal Java. Het programmeren zelf vindt plaats in de SAP NetWeaver Developer Studio. Integratie met andere platformen wordt mogelijk gemaakt door het feit dat NetWeawer compatibel is met Microsoft .NET en het Java 2 Platform (J2EE) en de ondersteuning van internet standaarden zoals HTTP, XML en webservices.

De servicegeoriënteerde architectuur is volledig verwerkt in het SAP NetWeaver platform, gebruikmakend van UDDI, WSDL, XML en SOAP. SAP applicaties kunnen ontsloten worden als en gebruik maken van webservices. Hiervoor maakt SAP NetWeaver gebruik van het Java 2 Platform. Vanuit de servicegeoriënteerde gedachte introduceert SAP een nieuw soort type applicatie, composite applications genoemd. Composite applications worden mogelijk gemaakt door het Composite Application Framework.

Bijlage IV - Legacy

Software legacy's zijn er in vele soorten en maten. Op het gebied van administratie legacy systemen, ongeacht de programmeertaal, zijn drie basistypen programma's te onderscheiden [Sneed 02]:

- online transaction programs;
- batch processing programs;
- general subprograms.

De legacy van PinkRoccade Civility valt onder de noemer online transaction programs. In een online transactional program staat de teleprocessing monitor centraal. Een teleprocessing monitor is een complex programma die op een mainframe systeem alle taken op zich neemt om mainframe programma's te kunnen laten draaien. Onder de taken valt het afhandelen van de communicatie met de remote en local terminals, het afhandelen van user verzoeken, het afhandelen van de database toegang, het afhandelen van de foutmeldingen, het bijhouden van logboeken en meer [Sellink 98]. Er zijn twee soorten teleprocessing monitors. Één soort is dat het programma de teleprocessing monitor aanstuurt, de andere soort is dat het programma een subprogramma is van de teleprocessing monitor en dat de teleprocessing monitor het programma aanstuurt. De legacy bij PinkRoccade Civility is zo opgezet dat de programma's de teleprocessing monitor aansturen.
Een karakteristieke eigenschap van online transactional programs is dat de presentatie-, business logica- en databaselaag volledig in elkaar vermengd zijn. De business logica in de programmacode is dus moeilijk te achterhalen, omdat de code die de business logica bevat vermengd is met de code voor de aansturing van de user interface en het ophalen en wegschrijven van de data. Deze eigenschap is ook karakteristiek voor de legacy van PinkRoccade Civility.

Bijlage V - Analyse legacy

Technieken om inzicht te verkrijgen in software systemen zijn onder te verdelen in drie categorieën [Müller]:

- Unaided browsing;
- Leveraging corporate knowlegde and experience;
- Computer-aided techniques like reverse engineering.

Unaided browsing

Unaided browsing houdt in dat een software engineer handmatig de code analyseert. Dit kan gedaan worden door de code uit te printen of door middel van een code editor. Deze aanpak is volledig afhankelijk van de kwaliteit van de engineer en mate van details die deze engineer kan bevatten.

Leveraging corporate knowlegde and experience

De analyse bij leveraging corporate knowlegde and experience vindt plaats door begeleiding van een mentor of het uitvoeren van informele interviews. Deze interviews vinden plaats met de medewerkers die (domein)kennis hebben van het betreffende systeem. Het analyseren van een systeem op deze manier kan zeer succesvol zijn als er genoeg medewerkers in de organisatie aanwezig zijn die ver- en diepgaande kennis hebben van het systeem. Door op deze manier de analyse uit te voeren kan een goed/juist beeld gevormd worden van het systeem als geheel of het betreffende onderdeel dat is onderzocht.

Computer-aided techniques like reverse engineering

De bovenstaande technieken zijn niet altijd mogelijk. De medewerkers met ver- en diepgaande kennis van het systeem kunnen de organisatie verlaten hebben of de code van het systeem kan erg moeilijk en complex zijn. In deze gevallen kunnen computer-aided techniques helpen. Een voorbeeld van een computer-aided technique is reverse engineering. Reverse engineering maakt het mogelijk om inzicht in een systeem te verkrijgen door informatie op een hoger abstractieniveau uit de systemen te extraheren. Een combinatie van de technieken in de verschillende categorieën is mogelijk.