Omgaan met veranderende requirements in outsourcing-projecten

Marco Lormans Hylke van Dijk
Delft University of Technology
The Netherlands
{M.Lormans,H.W.vanDijk}@ewi.tudelft.nl

Arie van Deursen
CWI and Delft University of Technology
The Netherlands
Arie.van.Deursen@cwi.nl

Eric Nöcker
LogicaCMG, Technical Software Engineering, The Netherlands
Eric.Nocker@logicacmg.com

Samenvatting

Dit artikel beschrijft de ervaringen opgedaan bij het opzetten van een requirements-managementsysteem. We beschouwen met name de problemen bij het consistent beheren van de requirements in geval van gedistribueerde ontwikkeling.

1. Introductie

In outsourcing-projecten (uitbestedings-projecten) worden de requirements (eisen) voor een nieuw product doorgaans aangeleverd door een externe stakeholder, de klant. Deze heeft behoefte aan een nieuw product, maar wil dat niet zelf ontwikkelen. Om het risico van kostenoverschrijding te vermijden dringt de klant vaak aan op een contract met een vaste prijs. Een wezenlijk ondereel van het contract is het requirements-specificatie document dat de resultaten van de door de klant (uitbestedende partjij) uitgevoerde requirements-identificatieproces vastlegd.

Eén van de voornaamste problemen bij outsourcing is de voortdurende verandering van de requirements (evolutie). Hoe doordacht de requirements-specificatie ook is opgezet, de requirements voor een niet-triviaal systeem zullen altijd aan verandering onderhevig zijn, zowel tijdens als na de implementatie van het systeem. Deze evolutie kent vele oorzaken, waaronder veranderende bedrijfsbelangen, marktontwikkelingen, of technologische vooruitgang. Tevens leidt het proces van ontwerpen, implementeren en testen tot voortschrijdend inzicht, met als mogelijk gevolg aanpassingen in de verzameling van requirements.

Een groot risico bij de evolutie van requirements zijn inconsistenties in de verzameling van requirements. Evolutie kan leiden tot misvattingen en onjuiste aannames van ontwikkelaars, tot een systeem dat niet implementeerbaar is of tot een systeem dat niet aan de eisen van de klant voldoet.

Het managen van evoluerende requirements is, naast het specificeren van requirements, een van de twee hoofdactiviteiten van requirements-engineering [4]. De specificatie van requirements betreft het identificeren, analyseren, documenteren en valideren van de requirements. Bij requirementsmanagement gaat het erom de requirements op een gestructureerde manier te beheren en de onvermijdelijke veranderingen correct te verwerken. Veranderingen kunnen betrekking hebben op de requirements zelf, op de relaties tussen de requirements en op de relaties tussen requirements en requirements-documenten of andere artefacten die geproduceerd zijn tijdens het ontwikkeltraject. Requirements management is in de eerste plaats een ondersteunend proces gedurende het gehele software-ontwikkeltraject [2].

Het groeiende aantal requirements en de dynamiek van de requirements heeft de ontwikkeling van requirementsmanagementsystemen gestimuleerd. Dergelijke systemen ondersteunen de communicatie gedurende het requirementsmanagementproces, en helpen bij het bewaken van de consistentie van de requirements-documentatie.

In dit artikel bespreken we de invloed van outsourcing op het beheren van requirements. In het bijzonder kijken we naar de methoden en technieken die LogicaCMG heeft gebruikt bij het ontwikkelen van een systeem om verkeersstromen te volgen. We beschrijven de ervaringen met een requirements-managementsysteem opgedaan tijdens het ontwikkelen van dit systeem.

2. Het Verkeersvolgsysteem

Onze casestudie heeft betrekking op een verkeersvolgsysteem (VVS) dat onderdeel is van een groter verkeersregelsysteem. Het VVS-onderdeel is verantwoordelijk voor het bijhouden en vastleggen van de posities van voertuigen op een verkeersnet. Deze data is invoer voor een logistiek-optimalisatieproces en wordt gebruikt voor het maken van dienstregelingen.

Het VVS verzamelt data door middel van een serie metingen. De taken van het VVS omvatten het distribueren van deze informatie ten behoeve van de sturing van de bezetting

van het netwerk. Het VVS voorziet cliëntsystemen met real-time, consistente en niet-ambigue informatie over voertuigposities. Het houdt voertuigbewegingen bij (identificatie en volgorde) op de grenzen van gebieden die gevolgd worden.

De eisen aan het VVS zijn opgesteld door de eigenaar van het systeem. Het ontwerp en de implementatie is uitgevoerd door LogicaCMG. Voor het ontwerp zijn verschillende UML-modellen opgesteld, voor de implementatie is gebruik gemaakt van de programmeertaal C++.

2.1. Documentstructuur en Gereedschappen

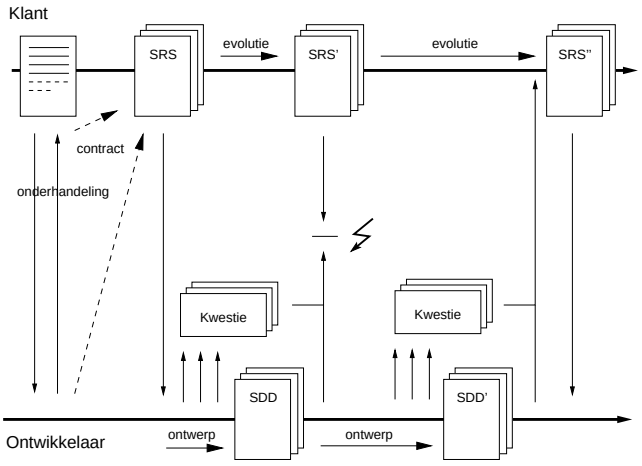
De interactie tussen de eigenaar van het VVS, de uitbestedende partij, en LogicaCMG is georganiseerd rond de MIL-std 498 standaard. Onze casestudie gebruikt slechts een deel van de MIL-std 498 geïmplementeerd (Zie Sectie 4.1).

LogicaCMG heeft voor Rational^{TM} tooling gekozen voor de ondersteuning van de diverse ontwikkelactiviteiten. Rational RequisitePro wordt gebruikt voor het managen van de requirements. Deze tool beheert een repository van requirements. De data wordt opgeslagen in een externe gegevensbank, in onze casestudie een Microsoft Access Database. Deze is goed geïntegreerd met Microsoft Word, de tekstverwerker waarin de requirements gedocumenteerd worden. Voor het maken van de verschillende UML-modellen wordt Rational Rose gebruikt. De gebruikte modellen in onze casestudie zijn activiteitsdiagrammen, collaboratiediagrammen en klassediagrammen. Ze beschrijven het ontwerp van het systeem en ze worden ook gebruikt om details van de requirements nader toe te lichten. Voor het genereren van rapporten wordt Rational SoDa gebruikt. Rational SoDa haalt de benodigde data uit Rational RequisitePro en Rational Rose. Tot slot wordt Telelogic Synergy gebruikt voor het beheren van de verschillende configuraties. Alle documenten en repositories worden hierin opgeslagen en beheerd.

3. Outsourcing

Het uitbesteden (outsourcing) van systeemontwikkeling, systeemintegratie en onderhoudswerkzaamheden is een belangrijke ontwikkeling op het gebied van IT-diensten en stelt speciale eisen aan het beheren van requirements en het gebruik van tools. Voor de uitbestedende partij is outsourcing een kostenbesparende manier om een systeem of dienst te laten ontwikkelen. Het systeem wordt geleverd volgens een vaste prijs en een vooraf gedefinieerde kwaliteit. De specificatie van de requirements is een belangrijk onderdeel van het contract tussen de uitbestedende partij en de aannemende partij.

De details en de structuur van de aangeleverde requirements-specificatie heeft een grote invloed op de manier van werken bij de systeemontwikkelaar. Op de korte termijn moet de ontwikkelaar op een zo goedkoop mogelijke manier voldoen aan de verplichtingen van het contract,



Figuur 1. Het requirements-proces in een outsourcing-project

d.w.z. voldoen aan de requirements en slagen voor de acceptatietests. Voor de lange termijn wil de ontwikkelaar ook het onderhoudscontract binnenhalen en vergelijkbare opdrachten voor het ontwikkelen van toekomstige systemen.

Andere kwesties waar rekening mee gehouden moet worden zijn: wie heeft de leiding over het proces, wie is eigenaar van de verschillende opgeleverde artefacten, hoe ga je om met de verschillende ontwikkelparadigma's, hoe garandeer je de kwaliteit en hoe ga je om met de systeemdecompositie?

3.1. Het Requirements-Engineering-Proces

In een outsourcing context zijn de verantwoordelijkheden van het requirements-engineering-proces verdeeld over de uitbestedende partij (klant) en de aannemende partij (ontwikkelaar). In Figuur 1 is dit proces weergegeven.

De klant verzamelt en documenteert de requirements in een requirements-specificatie (System Requirements Specification, SRS). Deze SRS vormt de basis van het contract. Gegeven de SRS gaat de ontwikkelaar het ontwerp van het systeem uitwerken en documenteert deze in een systeemontwerp (System Design Description, SDD).

Tijdens de ontwikkeling van een systeem loopt de ontwikkelaar vaak tegen conflicten of onduidelijkheden in de requirements aan. Deze worden vastgelegd in 'kwesties' en moeten overlegd worden met de klant. Ondertussen ontwikkelt de klant nieuwe ideeën die ook geïmplementeerd moeten worden (SRS′). De synchronisatie van deze parallelle activiteiten is een potentieel probleem. In principe zijn hier twee mogelijke oplossingen voor; de ontwikkelaar wacht met het terugkoppelen van de SDD met bijbehorende kwesties of de klant wacht met het overdragen van de SRS′.

In het eerste geval verwerkt de ontwikkelaar de vernieuw-

de SRS′ in de SDD, resulterend in een nieuwe SDD′. In het tweede geval lost de klant eerst de kwesties van SDD op voor het vrijgeven van de nieuwe SRS′. Dit laatste geval is weergeven in de tweede iteratie van Figuur 1. Een hybride vorm van dit synchronisatieproces is ook mogelijk, maar in dat geval is het moeilijk alle systeemartefacten consistent te houden. Er zijn dus twee bronnen voor de evolutie van het systeem: voortschrijdend inzicht aan de kant van de klant en voortschrijdend inzicht aan de kant van de ontwikkelaar. Beide inzichten moeten gestroomlijnd worden.

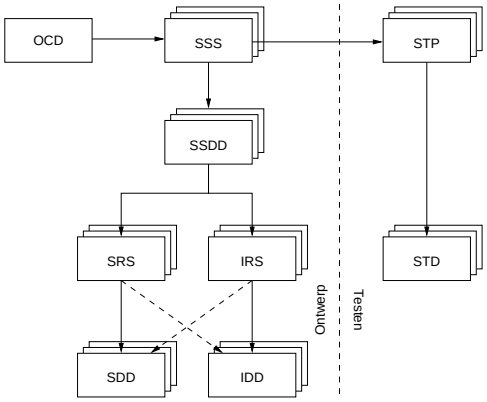
3.2. Gevolgen voor het requirements-managementsysteem

De twee bronnen van requirements-evolutie (van de zijde van de klant en van de ontwikkelaar) in een outsourcing context stellen speciale eisen aan het te gebruiken requirements-managementsysteem (RMS):

- Verandermanagement; projectmedewerkers moeten op de hoogte zijn van de bijgewerkte requirementsverzameling, de evolutie (verwerkte veranderingen) en verwachte aanpassingen.
- Kwaliteitsgarantie; de coherentie en toegepaste terminologie van de requirements moeten geverifieerd zijn voordat een ontwerp wordt opgeleverd.
- Kwestiemonitoring; tijdens het project worden veranderingsaanvragen door middel van kwesties gecommuniceerd (mogelijk over meedere kanalen). De status en geschiedenis van deze kwesties moeten toegankelijk zijn en geregistreerd worden.
- Testrapportage; de ontwikkelaar moet via testrapporten aantonen dat de requirements afgedekt zijn.
- Statusrapportage; de ontwikkelaar moet via statusrapporten de vordering van het project terugkoppelen.
- Flexibele modularisatie; de mogelijkheid bestaat dat de systeemdecompositie van de klant niet overkomt met de wenselijke systeemdecompositie van de ontwikkelaar. Flexibele modularisatie geeft de ontwikkelaar de mogelijkheid intern een afwijkende decompositie te gebruiken.

4. De case

De eisen die we hierboven stellen aan het requirements management proces hebben gevolgen voor de implementatie bij LogicaCMG. Eerder genoemde concepten zoals de documentstructuur en de gereedschappen moeten geconfigureerd worden.



Figuur 2. Overzicht MIL-std 498

4.1. Documentstructuur

De eerste verantwoordelijkheid van de klant is het aanleveren van de requirements, de eerste verantwoordelijkheid van de ontwikkelaar is het implementeren van deze requirements. De requirements worden aangeleverd volgens de MIL-std 498 standaard. De gebruikte delen van de standaard zijn weergegeven in Figuur 2.

De klant levert de volgende documenten: de Operational Concept Description (OCD), System/Subsystem Specification (SSS), System/Subsystem Design Description (SSDD), de System Requirements Specification (SRS) en de Interface Requirements Specificatie (IRS). De klant levert ook de bijbehorende acceptatietests op systeemniveau; System Test Plan (STP). Alle documenten zijn Microsoft Word documenten. De documenten zijn input voor het ontwikkelteam van LogicaCMG. De SRS wordt omgezet naar een System Design Description (SDD), Interface Design Description (IDD) en System Test Descriptions (STD).

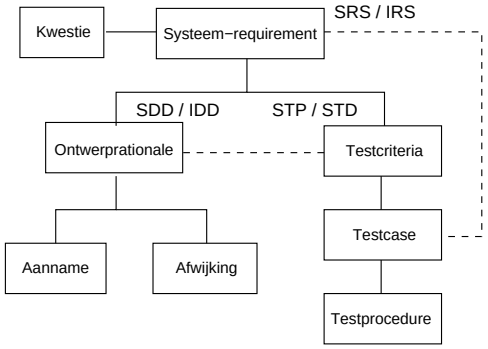
4.2. Requirements Traceerbaarheidsmodel

Vooraf is een requirements traceerbaarheidsmodel gedefinieerd (Figuur 3). Het model definieert een aantal typen requirements en typen van mogelijke traceerbaarheidsrelaties.

Een *systeem-requirement* is het traditionele requirementtype dat beschrijft *wat* het systeem moet doen. Dit type bevat functionele en niet-functionele eisen, zoals ontwerpbeperkingen.

Een *kwestie* registreert een aandachtspunt dat nader onderzoek behoeft. Kwesties zijn tevens het middel om met de klant te communiceren.

Ontwerpbeslissingen, *aannames* en *afwijkingen* hebben allen betrekking op het ontwerp van het systeem. De eerste registreert *hoe* het systeem geïmplementeerd wordt, de tweede registeert eventuele aannames die hierbij gedaan zijn, de laatste registeert bewuste afwijkingen waar men zich niet



Figuur 3. Het traceerbaarheidsmodel

houdt aan eerder genomen beslissingen. Deze drie typen maken in beginsle impliciete ontwerpinformatie expliciet.

De *ontwerprationale* registreert de argumentatie achter een ontwerpbeslissing, aanname of afwijking. Het registreren van deze ontwerprationales helpt eindeloze discussies te voorkomen. Ontwerprationales zijn gedefinieerd als attribuut van een requirement en zijn zelf dus geen requirementtype.

Vergelijkbaar met de requirement-typen voor ontwerp heeft LogicaCMG ook requirement-typen voor testen gedefinieerd.

Ten behoeve van het test process zijn vergelijkbare requirement-typen en traceerbaarheidarelaties gedefinieerd. De *testcriteria* beschrijven de condities waaronder een requirement succesvol geïmplementeerd is. Een *testcase* beschrijft een autonome test, die één of meer testcriteria implementeerd. Een *testprocedure*, tot slot, beschrijft een opeenvolging van testcases.

Alle requirement-typen worden als unieke entiteiten opgeslagen in het requirements-managementsysteem met uitzondering van de ontwerprationale. De verbindingen tussen de verschillende entiteiten geven de traceerbaarheidsrelaties weer. Twee uitzonderingen zijn mogelijk; het is niet altijd eenvoudig testcriteria te beschrijven voor elke requirement en soms is het eenvoudiger testcriteria direct van de ontwerprationale af te leiden. De uitzonderingen zijn in Figuur 3 weergegeven door middel van de stippellijnen.

4.3. Het opzetten en gebruik van een requirements-managementsysteem

Het opzetten en gebruik van een requirements-managementsysteem vergt een aantal stappen, enkele daarvan moeten herhaaldelijk uitgevoerd worden.

In eerste instantie wordt Rational RequisitePro geconfigureerd, wat bestaat uit het opzetten van de documentstructuur en het traceerbaarheidsmodel.

Vervolgens moet het systeem gevuld worden met de aangeleverde requirements. RequisitePro geeft de mogelijkheid Microsoft Word documenten te importeren en paragrafen als entiteiten te markeren. De structuur van de Microsoft Word documenten is bepalend voor de hoeveelheid (vaak handmatig) werk.

Nadat de initiële requirements met hun attributen in het systeem zijn opgeslagen zullen eventuele aanpassingen verwerkt moeten worden. De gebruiker kan deze direct in het systeem verwerken. Het systeem markeert de aangepaste requirements automatisch, met behulp van het traceerbaarheidsmodel kan daarna de ernst van de invloed van deze verandering worden bepaald.

De klant is de eigenaar van de requirements en het heeft dus de voorkeur dat de klant aanpassingsverzoeken aanlevert (anders ontstaan inconsistenties). In het geval van VVS levert de klant echter volledig nieuwe versies van de requirementsdocumenten. Deze bewerkte Microsoft Word documenten worden door LogicaCMG verwerkt in het requirementsmanagementsysteem. Dit is een foutgevoelig proces met veel handmatig werk en slechts beperkte ondersteuning vanuit het requirements-managementsysteem.

LogicaCMG is contractueel verplicht informatie terug te koppelen naar de klant in de vorm van verschillende rapportages. Deze rapporten worden gegenereerd vanuit het requirements-managementsysteem en dienen als ondersteuning voor reviews, projectmanagement en testen. Enkele voorbeelden zijn: een overzicht van alle kwesties inclusief hun status of een overzicht van de requirements die afgedekt zijn door een testcase. Rational RequisitePro biedt slechts beperkte ondersteuning voor het genereren van deze rapporten.

4.4. Statistieken

Tabel 1 bevat enkele kengetallen over de omvang van de in eerste instantie aangeleverde requirements, en over het aantal wijzigingen dat hierop is aangebracht in de tweede versie. Op het hoogste niveau zijn een dertigtal documenten te onderscheiden, die weer onderverdeeld zijn in ongeveer 900 individuele SSS-, SRS-, of IRS-specificaties. Het implementeren hiervan leidde tot meer dan 500 testgevallen, 600 gedocumenteerde ontwerpbeslissingen, en 160 kwesties die nader onderzoek behoefden.

Deze kwesties zijn teruggekoppeld naar de klant. Dit resulteerde in 240 aangepaste requirements, en 80 nieuwe requirements voor de tweede release van VVS. Aan de zijde van LogicaCMG leidde dit tot 700 gewijzigde en nieuwe ontwerp- en testdocumenten.

5. Discussie

We hebben laten zien hoe het requirements-managementproces is ingevuld binnen het VVS-systeem, zullen we de belangrijkste lessen bespreken. We doen dit aan de hand van een aantal stellingen uitgesplitst naar

Tabel 1. Kentallen van één iteratie

Documenttype	# Aange-leverd	# Gewij-zigd	# Nieuw
SSS	1	0	0
SRS	3	3	0
IRS	9	4	0
IDD	3	1	0
SDD	9	9	0
IDD	6	3	0
Totaal	31	20	0
# Requirements	# Aange-leverd	# Gewij-zigd	# Nieuw
SSS	142	32	20
SRS	448	120	32
IRS	308	88	28
Totaal	898	240	80
Item	# Ont-wikkeld	# Gewij-zigd	# Nieuw
Testcase	564	220	160
Ontwerpbeslissing	638	260	60
Kwestie	160	-	-
Totaal	1362	480	220

de gevonden aandachtsgebieden van een requirements-managementsysteem: requirements-perspectieven, database-systeem, traceerbaarheid en de interactie met de klant.

5.1. Lessen

Het feit dat we een requirements-managementsysteem gebruiken in een outsourcing-project heeft consequenties voor het gebruik.

Stelling 1. *Vanwege het gedistribueerde requirements-engineering-proces in een outsourcing-project moet de synchronisatieactiviteit expliciet gemaakt worden.*

Interactie met de klant. Het importeren en bijwerken van requirements heeft vooral betrekking op verandermanagement. Het importeren van requirements in het requirements-managementsysteem is een lastige taak. Een automatisch proces heeft de voorkeur, maar in de praktijk moet er veel handmatig werk worden verricht om de gewenste kwaliteit de realiseren. Modellen bijvoorbeeld spelen een belangrijke rol bij het toelichten van de requirements. Echter wegens gebrek aan formaliteit en ondersteuning worden deze veelal buiten het requirements-managementsysteem gelaten.

Het bijwerken van requirements is volledig handwerk. Eerst wordt het verschil tussen de documenten bepaald en vervolgens worden de wijzigingen handmatig verwerkt in het requirements-managementsysteem inclusief het aanbrengen van traceerbaarheidsrelaties. Traceerbaarheidsrelaties kunnen verdwijnen of kunnen ongeldig worden. Het corrigeren van een typefout en het volledig herschrijven van een requirement hebben hetzelfde effect op de traceerbaarheidsrelaties.

Kwestiemonitoring en kwaliteitsgarantie worden grotendeels afgedekt door de kwesties als communicatiemiddel. Dit werkt goed voor beide partijen. Het probleem van het verwerken van aanpassingen in het requirements-managementsysteem blijft echter bestaan.

Stelling 2. *Kwesties zijn een zeer effectief communicatiemiddel. Kwesties moeten daarom net als requirements identificeerbare entiteiten zijn binnen het requirements-managementproces en ook als zodanig worden behandeld.*

Requirements-perspectieven. Statusrapportage wordt voornamelijk gebruikt voor projectmanagement. De rapporten geven informatie over de status van requirements en openstaande kwesties. De rapporten worden gebruikt voor het terugkoppelen van de status naar de klant en voor interne sturing. Het genereren van rapporten is problematisch. Rational SoDa heeft veel beperkingen; zoals het gebrek aan kwaliteit, gebrek aan flexibiliteit en de tijd die nodig is om een document te genereren.

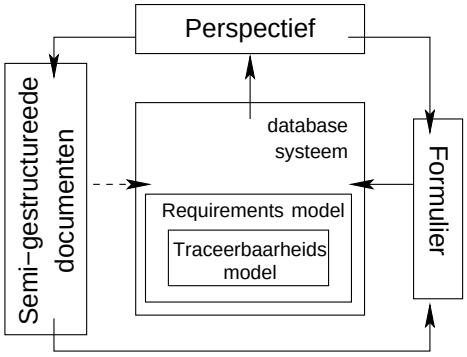
Als gevolg hiervan gebruiken de ontwikkelaars de navigatiemogelijkheden van het requirments-managementsysteem in plaats van de rapporten voor het vergaren van informatie over de requirements. Echter niet alle gewenste perspectieven op de requirements kunnen worden gecreëerd. Een sprekend voorbeeld is de modularisatie van de requirements. De managementstructuur wordt nu bepaald door de aangeleverde documentstructuur, in ons geval die van MIL-std 498. Voor het ontwikkelproces is vaak behoefte een andere structuur.

Stelling 3. *Flexibele modularisatie en generatie van requirements-perspectieven wordt door de huidige tools in onvoldoende mate ondersteund.*

Traceerbaarheid. Traceerbaarheid is een belangrijke toegevoegde waarde voor een requirements-managementsysteem. Een traceerbaarheidsmodel helpt een 'spaghetti' van relaties te voorkomen. De consistentie van relaties is echter noodzaak, maar omdat deze handmatig worden aangebracht loert het gevaar van inconsistentie. Traceerbaarheidsrelaties moeten dus opzichzelfstaande entiteiten vormen binnen het requirements-managementsysteem met een eigen rationale. Voor het overige bevat het traceerbaarheidsmodel voldoende detail om alle benodigde informatie vast te leggen.

Analyse van requirements wordt onvoldoende ondersteund door het huidige requirements-managementsysteem. Alleen impactanalyse wordt ondersteund vanwege het traceerbaarheidsmodel. Andere vormen van analyse, zoals 'conformance checking' worden niet ondersteund.

Stelling 4. *Traceerbaarheid kan effectief geïmplementeerd worden door middel van een traceerbaarheidsmodel. De traceerbaarjheidsrelaties moeten identificeerbare entiteiten zijn*



Figuur 4. Requirements-engineering-systeem

binnen het requirements-managementsysteem.

Database-systeem. Het gekozen database-systeem ondersteunt de zoekvragen in voldoende mate. Dit geldt waarschijnlijk voor elk modern database-systeem.

5.2. Een Conceptueel Raamwerk

We beëindigen de discussie met een voorstel voor een conceptueel raamwerk van een requirements-engineering-systeem (RES) [3]. Ons systeem structureert de aandachtsgebieden van een requirements-managementssysteem in de context van outsourcing en bevat de lessen uit de bovenstaande sectie. RES geeft sturing aan de keuze en het gebruik van gereedschappen voor de inrichting van een requirements-managementsysteem. Vooralsnog zijn er geen commerciële gereedschappen beschikbaar te zijn die aan alle eisen voldoen [1].

Database-systeem. De kern van het raamwerk, afgebeeld in Figuur 4, wordt gevormd door het requirements-model inclusief het traceerbaarheidsmodel. Normaal gesproken is dit model opgebouwd volgens een sjabloon, zoals MIL-std 498 of IEEE-std 830-1998. Voor elke requirements-entiteit zijn een aantal attributen gedefinieerd en entiteiten komen overeen met het traceerbaarheidsmodel.

Traceerbaarheid. De traceerbaarheidsrelaties zijn opzichzelfstaande entiteiten binnen dit model. Dit betekent dat de relaties ook een unieke identificatie hebben, alsmede andere attributen zoals bijvoorbeeld een ontwerprationale of een eigen beschrijving.

Requirements-perspectieven. Het requirements-model biedt de mogelijkheid om analyse te plegen op de requirements-verzameling door middel van het genereren van diverse perspectieven op de verzameling. Aanpassingen aan de requirements worden automatisch verwerkt via zogenaamde formulieren. Requirements-perspectieven ondersteunen de mogelijkheid van dynamisch modulariseren, bijvoorbeeld volgens de documentstructuur of volgens de systeemdecompositie. Klanten kunnen desgewenst een requirements-perspectief krijgen die aansluit bij hun eigen, veelal semi-gestructureerde, documentstructuur.

Interactie met de klant. Het verwerken van de aanpassingen die uit de semi-gestructureerde omgeving komen is moeilijk. De klant en ontwikkelaar werken parallel en in verschillende ontwikkelomgevingen. RES biedt een oplossing voor de synchronisatie van beide omgevingen door eenmalig de conversie van semi-gestructureerde documenten naar het requirements-model te doen en de transformatie vast te leggen in een sjabloon. Bij een eventuele aanpassing van de externe documenten worden perspectieven en formulieren gegenereerd op basis van het sjabloon. De formulieren garanderen een consistente bijwerking van de requirements-verzameling.

RES-methode. Requirements-perspectieven en formulieren zijn de zichtbare onderdelen van het RES. Door de eigenschappen van de belangrijkste interne en externe (ten behoeve van de klant) perspectieven te identificeren kunnen we het requirements-model en traceerbaarheidsmodel opstellen. De modellen en de benodigde perspectieven bepalen de te stellen eisen aan de technologie voor een succesvolle implementatie. Het importeren van informatie en wijzigingen is in essentie een transformatie van ongestructureerde data via een perspectief naar een formulier. Wij werken aan de technologie voor deze stap.

6. Conclusie

We hebben laten zien hoe LogicaCMG een requirements-managementsysteem heeft geïmplementeerd voor het beheersen van requirements-evolutie. We beschrijven een project waar ze voor een externe klant een verkeersvolgsysteem hebben ontwikkeld. De belangrijkste implicaties van outsourcing in deze studie zijn dat er sprake is van aanneming met een vaste-prijs, dat de klant de requirements opstelt, dat de klant eigenaar blijft van deze documenten en dat de klant statusoverzichten wenst.

De lessen die we kunnen trekken uit deze studie zijn gevat in een raamwerk voor een requirements-managementsysteem. Ons RES bouwt op een requirements- en traceerbaarheidsmodel. Van het requirements-model kunnen meerdere requirements-perspectieven worden gegenereerd. Het requirements-perspectief dat openstaande kwesties uitlicht blijkt essentieel te zijn in de communicatie tussen de ontwikkelaar en de klant. RES gebruikt formulieren om op een consistente wijze de requirements aan te passen. Uit de studie blijkt dat met name de import van gewijzigde requirements problematisch is. Bovendien is gebleken dat de technologie om adequate statusrapporten te generen verbetering behoeft.

De uitwerking van het RES-raamwerk vormt de kern van ons huidige onderzoek.

Referenties

[1] Bas Graaf, Marco Lormans, and Hans Toetenel. Embedded software engineering: state of the practice. *IEEE Software*, 20(6):61–69, November–December 2003.

[2] Gerald Kontonya and Ian Sommerville. *Requirements Engineering: Processes and Techniques*. John Wiley and Sons, 1998.

[3] Marco Lormans, Hylke van Dijk, Arie van Deursen, Eric N¨ocker, and Aart de Zeeuw. Managing evolving requirements in an outsoucring context: An industrial experience report. In *Proceedings , International Workshop on Principles of Software Evolution*, pages 149–158. IWPSE04, 2004. accepted for publication.

[4] P¨aivi Parviainen, Maarit Tihinen, Marco Lormans, and Rini van Solingen. Requirements engineering: Dealing with the complexity of sociotechnical systems development. In Jos´e Luis Mat´e and Andr´es Silva, editors, *Requirements Engineering for Sociotechnical Systems*. IdeaGroup Inc, 2004. accepted for publication.