Requirements Traceability

Een literatuurbeschouwing en praktijkvergelijking

Bachelorscriptie informatiekunde
Radboud Universiteit Nijmegen

Begeleider: Dr. ir. G.J. Tretmans

Thomas Dobbe (0115495)
Januari 2006

Inhoudsopgave

Samenvatting

Requirements Traceability (RT) kan toegepast worden bij het ontwikkelen of het aanpassen van software. Uit voorgaande onderzoeken blijkt dat bedrijven diverse problemen ondervinden met RT, terwijl er voldoende ondersteuningsmogelijkheden bestaan, in de vorm van bijvoorbeeld een requirements management tool. De ondersteuning blijkt voor een deel oplossingen te bieden voor de problemen, maar voor een deel zullen oplossingen gezocht moeten worden in maatregelen die samenhangen met organisatorische en menselijke factoren. Om literatuur over RT te vergelijken met toepassing ervan in de praktijk zijn drie interviews gehouden met bedrijven. Hieruit komen enkele overeenkomsten en verschillen naar voren. Van het grote aanbod aan tools wordt in de praktijk slechts een heel klein deel gebruikt, dat zich voornamelijk richt op het gedeelte van het ontwikkelingstraject na het opstellen van de requirementsspecificatie. Verder wordt het V-model gebruikt als raamwerk voor het ontwikkelingstraject. Daarnaast blijkt dat de in de literatuur genoemde problemen in de praktijk nauwelijks terugkeren. Tot slot komen, in tegenstelling tot de literatuur, die zich richt op problemen en ondersteuning, uit de interviews enkele overwegingen en doelen naar voren die belangrijk zijn bij het gebruik van RT. De rol van RT wordt bepaald door het uitgangspunt van de organisatie en het soort product. De doelen bepalen verder of RT wordt gebruikt als hulpmiddel om structuur aan te brengen in requirements en het ontwikkelingstraject of als ondersteuningen bij het waarborgen van kwaliteit.

1 Inleiding

Deze scriptie bevat een onderzoek naar requirements traceability (RT). Dit hoofdstuk geeft eerst een inleiding in de onderzoeksopzet (1.1) en de opbouw van de scriptie (1.2).

1.1 Het onderzoek

Bij ontwikkeling van software is het verzamelen en analyseren van requirements een zeer belangrijke activiteit. Op basis hiervan wordt het ontwerp gemaakt en volgt de implementatie. Later in het traject wordt getest of het product voldoet aan de requirements, waarbij RT een zeer belangrijke rol speelt. Er is veel literatuur over dit onderwerp te vinden, maar het is ook zeer relevant hoe men er in de praktijk mee omgaat. De literatuur beschrijft namelijk nogal wat problemen met betrekking tot dit onderwerp. Als verschil met de literatuur is het bovendien belangrijk welke factoren van invloed zijn op de rol van RT in de praktijk. De probleemstelling is dan ook als volgt.

Probleemstelling

- **Hoofdvraag**

Wat zijn de verschillen en overeenkomsten tussen de literatuur over requirements traceability en de praktijk ervan?

- **Deelvraag**

Welke factoren zijn van invloed op de rol van requirements traceability binnen softwareontwikkeling?

De deelvraag helpt de hoofdvraag te beantwoorden, omdat het antwoord erop weergeeft hoe de rol van RT afhangt van de organisatie en manier van werken. Overwegingen die gemaakt worden bij gebruik van RT worden hierbij in het kader van de hoofdvraag geplaatst.

Het onderzoek is kwalitatief van aard en is enerzijds gebaseerd op literatuur en anderzijds op drie praktijkinterviews. Het leek mij een zeer leerzame en interessante manier om met behulp van interviews de praktijkkant van het onderwerp te belichten en dit te vergelijken met literatuur. De interviews zijn gehouden aan de hand van een vragenlijst, waarna ik relevante punten heb geabstraheerd en in verband heb gebracht met de beschreven literatuur.
De onderzoeksvraag is een open vraag naar verschillen en overeenkomsten; het is dus een vergelijkend onderzoek. Het bereikte en bedoelde domein zijn gelijk en bestaan uit de geraadpleegde literatuur (zie Literatuur) en de drie geïnterviewde bedrijven.

1.2 Opbouw

Het onderzoek wordt eerst ingeleid door het begrip RT te definiëren en in zijn context te plaatsen. Dan wordt een hoofdstuk gewijd aan in de literatuur behandelde problemen met betrekking tot RT. Vervolgens wordt ingegaan op RT ondersteuning en wordt teruggeblikt op de problemen en in hoeverre de ondersteuning daarvoor oplossingen biedt.
Daarna wordt RT beschreven aan de hand van drie interviews, afgenomen bij verschillende bedrijven. In de conclusie worden tot slot de hoofd- en deelvraag beantwoord.

In verband met gevoelige bedrijfsinformatie worden in dit verslag de bedrijven aangeduid met X, Y en Z. De uitgewerkte interviews zijn om dezelfde reden niet bijgevoegd als bijlage, maar onder voorwaarden kunnen deze worden opgevraagd bij de auteur.

2 Begripsbepaling

In dit hoofdstuk wordt toegewerkt naar een definitie van het begrip RT. Eerst wordt het onderwerp ingeleid vanuit een breed kader van de software engineering en requirements engineering (2.1), waarna definities worden gegeven van de begrippen requirement (2.2) en RT (2.3). Vervolgens worden soorten RT besproken (2.4) en wordt het in zijn context geplaatst van ondersteunende activiteiten tijdens de ontwikkeling van software (2.5).

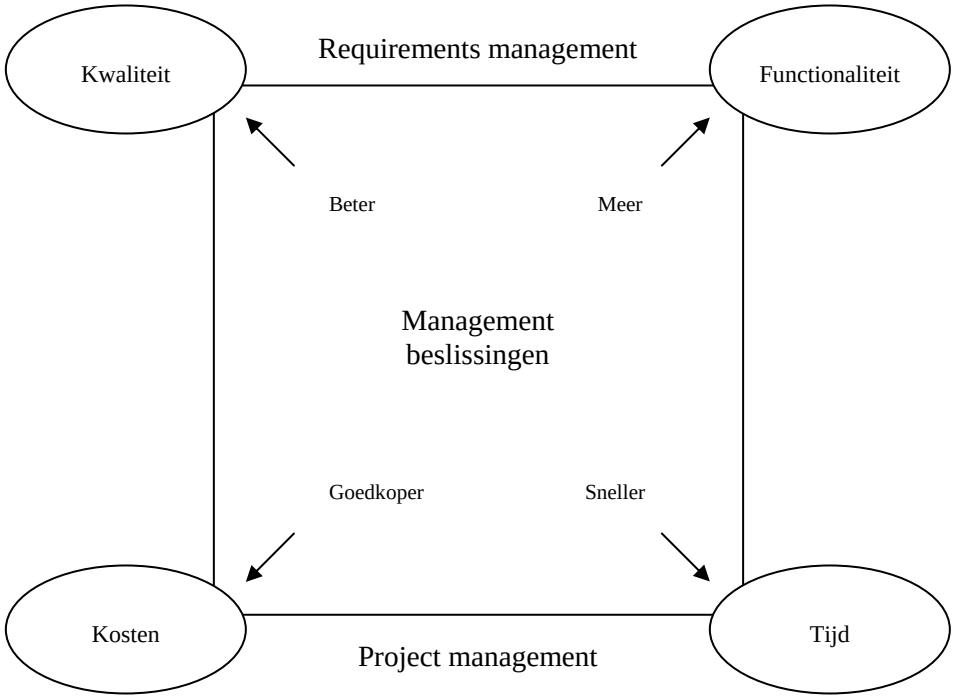
2.1 Software engineering - requirements engineering

Bij de ontwikkeling van software is het gebruikelijk om een traject te volgen met een aantal fasen of activiteiten, die in een bepaalde volgorde doorlopen worden. Zo'n softwareontwikkelingstraject kan bijvoorbeeld bestaan uit: requirements analyse - requirementsspecificatie - ontwerp - implementatie - testen - oplevering - onderhoud. Er zijn verschillende visies op ontwikkelmethoden, die elk een eigen idee hebben over hoe je de activiteiten doorloopt. Bij bijvoorbeeld de watervalmethode is er voor iedere activiteit een vaste, van tevoren afgesproken hoeveelheid tijd en wordt iedere activiteit eenmaal doorlopen. Maar bij een incrementele aanpak wordt de software beetje bij beetje opgebouwd, waarbij een nieuw deel steeds in het geheel geïntegreerd wordt. Iedere ontwikkelmethode heeft ook zijn eigen activiteiten en afbakening daarvan; zo zal de ene de oplevering niet tot het traject beschouwen en de andere wel. De zogenaamde requirements analyse, waar nu dieper op ingegaan zal worden, is een activiteit die bij ieder softwareproject (in mindere of meerdere mate) voorkomt en doorgaans voorafgaat aan het ontwerp en de implementatie. Het kan hier gaan om het bouwen van een volledig nieuw stuk software of om het aanpassen of uitbreiden van bestaande software.
Kort gezegd wordt bij de requirements analyse uitgezocht *wat* er gebouwd, veranderd of uitgebreid moet worden, waarna in het ontwerp wordt bepaald *hoe* dat gedaan wordt. Na het verzamelen worden de requirements vastgelegd in een zogenaamde requirementsspecificatie, die als basis dient voor het vervolg van het traject.

In de praktijk wordt het verzamelen van requirements en een grondige analyse ervan nog wel eens met een korrel zout genomen. En dat terwijl uit onderzoek blijkt dat fouten in deze activiteit relatief de meeste (financiële) schade opleveren. Bovendien blijven veel kant en klare (en overigens correct werkende) software systemen ongebruikt in de kast liggen, omdat ze in de praktijk niet aan de eisen blijken te voldoen of omdat de gebruikers er niet mee aan de slag willen of kunnen. Een ander gedeelte wordt pas moeizaam in gebruik genomen na ingrijpende aanpassingen. Dit geeft helder aan dat het kennelijk erg verleidelijk is om een product *goed* te bouwen, in plaats van heel bewust *het goede* product te maken. De requirements engineering draagt bij aan oplossingen voor deze problemen.

Een ander probleem is dat er over het algemeen zeer hoge eisen worden gesteld aan het budget en de deadline. Dit heeft als gevolg dat afwijkingen van de planning, door een tegenvaller of door nieuwe inzichten, ten koste kunnen gaan van het naleven van enkele requirements. Wanneer tijd (en / of geld) zeer belangrijk is kan een klant er in overleg bijvoorbeeld ook voor kiezen om een gedeelte van het systeem te laten vervallen of deze later toe te voegen of om een gedeelte van het systeem te leveren met een lagere kwaliteit. Dit wordt door figuur 1

geïllustreerd. Het veranderen van de waarde van een van de variabelen in de figuur heeft invloed op de andere variabelen. Het requirements management richt zich vooral op kwaliteit (worden requirements goed geïmplementeerd?) en functionaliteit (worden de goede requirements geïmplementeerd?), het projectmanagement richt zich daarentegen op eisen op het gebied van tijd en kosten. Deze tegengestelden dienen goed tegen elkaar afgewogen te worden om tot het juiste produkt te komen. Hierbij wordt *requirements management* gedefinieerd als “een set van activiteiten die het projectteam helpt om requirements en veranderingen aan requirements te identificeren, controleren en volgen op ieder moment van het project” [1]. Kiest de klant voor lagere kwaliteit of eerder voor minder functionaliteit, wanneer er minder tijd en / of geld beschikbaar is.



Figuur 1: Requirements management versus project management

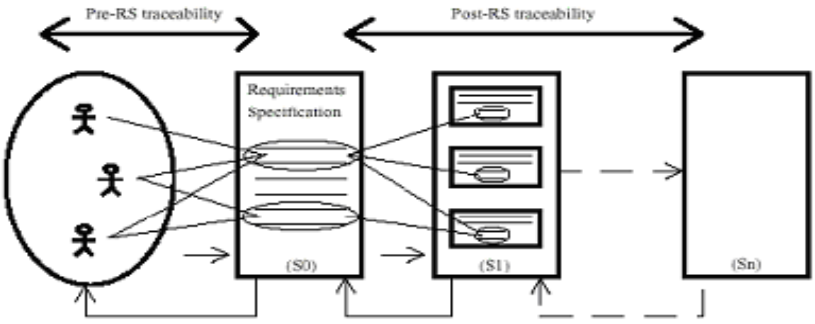
2.2 Soorten requirements

Hoewel er al over gesproken is in voorgaande tekst, is het toch belangrijk hier te definiëren wat precies bedoeld wordt met een requirement. Een requirement is iets dat een computerapplicatie moet doen voor zijn gebruikers. Het is een specifieke functie, feature, kwaliteit of principe die het systeem moet bieden om het bestaan waard te zijn. [2]
Een onderscheid in requirements kan als volgt worden gemaakt [3]. Allereerst zijn er op hoog niveau de *system requirements*: de vereisten voor een volledig systeem, dat zowel hardware als software omvat. Deze requirements zijn vervolgens onder te verdelen in *software requirements* en *hardware requirements*, waarop de system requirements van invloed zijn. De software requirements, die in dit onderzoek essentieel zijn, zijn weer onder te verdelen in *functionele requirements* en *nonfunctionele requirements*.

Functionele requirements zijn wat de gebruikers nodig hebben voor het systeem om te werken; functies en features [2]. Bijvoorbeeld ‘De interface moet een klok bevatten, waarop de tijd digitaal in uren en minuten wordt weergegeven’. Nonfunctionele requirements zijn de verborgen gebieden van het systeem die belangrijk zijn voor de gebruikers, ook al realiseren zij dit wellicht niet; de globale, vage factoren of principes die relateren aan het algemeen functioneren van het systeem [2]. Bijvoorbeeld gebruiksvriendelijkheid of robuustheid. Indien er behoefte aan is kunnen nog andere soorten requirements onderscheiden worden. Bijvoorbeeld user-requirements, die aangeven wat gebruikers kunnen en mogen en technische requirements die betrekking hebben op de technische kant van het product, zoals de hardware-eisen.
Wanneer het in dit onderzoek gaat over requirements worden software requirements bedoeld, tenzij dit expliciet staat aangegeven.

2.3 Requirements traceability

De requirementsspecificatie geldt over het algemeen als een overeenkomst, op basis waarvan de ontwikkeling plaatsvindt. Vooral als het om grote projecten gaat, waar veel geld mee gemoeid is, is het van groot belang dat er van tevoren zo concreet mogelijk wordt vastgelegd wat er opgeleverd moet worden en aan welke eisen dit moet voldoen. Hiernaast kunnen de wensen van de klant veranderen, door bijvoorbeeld wijzigingen in beleid, organisatie en markt. RT is een goed hulpmiddel om hier correct mee om te gaan, daarnaast kan het zogenaamde *change management* uitkomst bieden, waar verderop kort op in wordt gegaan. Voor een goed verloop van en een goede controle op het project is het van groot belang dat het eindproduct herleid kan worden tot requirements en dat requirements op hun beurt herleid kunnen worden tot hun oorsprong (zoals stakeholders en documenten). Dit wordt aangeduid met het begrip *requirements traceability (RT)*. Gotel en Finkelstein definiëren het als volgt. RT verwijst naar het vermogen om de levensloop van een requirement te kunnen beschrijven en volgen in zowel voorwaartse als achterwaartse richting (van zijn bronnen, via zijn ontwikkeling en specificatie, tot de oplevering)[4]. Zie figuur 2.



Figuur 2[4]: pre-RST en post-RST

2.4 Soorten RT

RT kan van toepassing zijn op twee niveaus: tussen de bronnen en de requirements en tussen de requirements en het eindproduct. De volgende vier vormen werden in 1990 door Davis onderscheiden [3].
1. Backward-from-requirements traceability

De herkomst van iedere requirement is te herleiden tot documenten, stakeholders, system requirements, etc.

2. Forward-from-requirements traceability
   Iedere requirement is tot (een deel van) een component of meer in het eindproduct te traceren.
3. Backward-to-requirements traceability
   Iedere component in het eindproduct is te herleiden tot een of meer requirements.
4. Forward-to-requirements traceability
   Ieder document of iedere bron heeft een of meerdere requirements tot gevolg.

Op het moment dat de requirementsspecificatie wordt goedgekeurd zijn vooral vorm 1 en 4 van belang. Dan kan er namelijk nog kritisch gekeken worden naar de herkomst van de requirements. Bij het testen is de beurt aan vorm 2 en 3, om de herleidbaarheid van het eindproduct en de requirements goed onder de loep te nemen.

Gotel en Finkelstein maakten in 1994 alleen een onderscheid in Pre-Requirements Specification Traceability (Pre-RST) en Post-Requirements Specification Traceability (Post-RST) [4]. Zie figuur 2.

Pre-RST
De activiteiten die voorafgaan aan het opstellen van de requirementsspecificatie bestaan voornamelijk uit het raadplegen van bronnen. Dit zijn vooral de sleutelfiguren in een organisatie, overige stakeholders en documenten. Vaak is de organisatiestructuur in de praktijk anders dan in theorie en bestaan er allerlei (belangrijke) informele netwerken. Het in kaart brengen van alle individuen, relaties en artefacten is hierdoor een lastig karwei, maar wel noodzakelijk om een grondige basis te hebben voor de requirementsspecificatie. Hierin moet vervolgens de verworven informatie worden geformaliseerd; een van de lastigste activiteiten van pre-RST. Dit formaliseren houdt in dat alle eisen die verzameld zijn zo eenduidig en helder mogelijk worden geformuleerd of anderszins worden vastgelegd, bijvoorbeeld door aantallen of afmetingen. Zo kan het hier gaan om de overzichtelijkheid van een systeem, om de exacte afmetingen van een venster in pixels of om het maximaal aantal opties in een menu.

Post-RST
Deze activiteiten omvatten de levensloop van requirements na de requirementsspecificatie, ofwel ‘requirement ontplooiing’. De concretisering van de requirements in de specificatie wordt als uitgangspunt gebruikt om tot het eindproduct te komen.

De verschillen in onderscheid tussen Davis en Gotel en Finkelstein komen als volgt overeen. Pre-RST omvat backward-from-requirements en forward-to-requirements traceability en post-RST omvat forward-from-requirements en backward-to-requirements traceability. Een onderscheid in pre-RST en post-RST is voor dit onderzoek voldoende en daarom zal deze terminologie in de rest van het verslag gebruikt worden.

2.5 Software Configuration Management

Om RT in zijn context te plaatsen is het ook van belang te kijken naar overige ondersteunende activiteiten tijdens het ontwikkelproces. Software Configuration Management (SCM), ook wel change management genoemd, is een verzameling activiteiten die toegepast kan worden tijdens softwareontwikkeling en die gericht zijn op veranderingen [1]. De activiteiten zijn ontwikkeld om een verandering te identificeren, te controleren, te implementeren en te rapporteren. De volgende voorbeelden zijn SCM activiteiten. Bij *versioncontrol* worden verschillende versies van systeemelementen beheerd en indien nodig kan worden teruggekeerd naar voormalige versies. Bij *dependency tracking and change management* worden allerlei soorten relaties tussen objecten opgeslagen, zoals tussen bedrijfsentiteiten en processen, tussen systeemelementen en tussen ontwerpcomponenten. De relaties verschillen van associaties tot afhankelijkheidsrelaties in UML. Bij een aanpassing kan zo makkelijk gedetecteerd worden hoe afhankelijkheden zich gedragen. *Configuration management* houdt verschillende configuraties bij die specifieke project mijlpalen of releases representeren. Een configuratie wordt hier gedefinieerd als de verzameling van programma's (code), documenten die de programma's beschrijven en data. *Requirements Tracing*, tot slot, is ook een SCM activiteit die de functionaliteit biedt om te kunnen volgen hoe deliverables resulteren uit de requirementsspecificatie en te kunnen identificeren welke requirements ten grondslag liggen aan deliverables. Deze activiteit, zoals beschreven in [1], richt zich op post-RST; uiteraard behoort een functionaliteit die pre-RST mogelijk maakt ook tot de SCM activiteiten, omdat het ook van belang is om te kunnen herleiden welke bronnen ten grondslag liggen aan requirements, wanneer deze wijzigen.

3 RT problemen

Dit hoofdstuk behandelt problemen met RT die naar voren zijn gekomen in eerdere onderzoeken (3.1). Daarna worden de oorzaken beproken die eraan ten grondslag liggen (3.2). In hoofdstuk 4 komt naar voren in hoeverre RT ondersteuning bijdraagt aan oplossingen.

3.1 Problemen

Gotel en Finkelstein vermelden dat er op het gebied van RT nog veel verbeterd kan worden[4]. Zij noemen de volgende problemen.

1. In de praktijk zijn requirements lang niet altijd te traceren, zowel voorwaarts naar het eindproduct als achterwaarts naar hun bronnen, door bijvoorbeeld missende informatie of verwijzingen.
2. Er blijken soms deels onjuiste eindproducten opgeleverd te worden door een karige probleemanalyse.

Egyed en Grünbacher onderkennen de volgende problemen [5].

1. Het verkrijgen van RT informatie is vaak een handmatig proces, wat resulteert in een hoge complexiteit en veel moeite.
2. RT is nuttig wanneer een compleet beeld bestaat over de loop van requirements. In de praktijk mist vaak informatie.
3. RT is steeds in beweging, omdat er kans is op wijziging van requirements door externe verandering of nieuwe inzichten en technologieën.
4. Het is vaak lastig om te anticiperen op kwesties die zich later in het ontwikkelingstraject misschien voordoen. De huidige RT informatie kan daarom onvoldoende zijn.
5. Het is lastig om betekenisvolle relaties te leggen in een complexe omgeving met veel objecten, documenten, modellen, code, et cetera.

Arkley en Riddle [6] denken dat het slecht beheren van RT informatie te wijten is aan het ontbreken van direct merkbaar voordeel in de hoofdlijnen van het ontwikkelproces. De consequentie hiervan is dat RT informatie incompleet, inaccuraat en verouderd is.
Uit een in 2002 uitgevoerd onderzoek komen de volgende probleemgebieden naar voren.

1. Data invoer en tool gebruik. Het invoeren van engineering data in een voorgeschreven formaat van een RT tool. De problemen betreffen het dubbel invoeren van data, de presentatie van gegevens en moeilijkheid om in een tool te navigeren.
2. Gebrek aan begrip en waarneembare bureaucratie. Medewerkers hebben geen inzicht in hoe RT informatie toegepast gaat worden. Ze ervaren dit meestal als een bureaucratische last opgelegd door het kwaliteitsteam. Dit inzicht komt tot uiting in het algemeen gebruik van afkortingen bij de invoer van data; bijvoorbeeld TBA (To Be Announced), TBN (To Be Notified) of NK (Not Known).
3. Gebrek aan direct voordeel voor de hoofdlijnen in het ontwikkelproces. RT wordt gezien als iets dat geen direct voordeel oplevert en het ontwikkelproces tot last is.

3.2 Oorzaken

Veel problemen zijn te wijten aan slecht onderscheid tussen de twee vormen van RT, terwijl iedere component zijn eigen problemen, bronnen en informatie omvat. Ondersteuning blijkt zich het gemakkelijkst te richten op post-RST. Er bestaan namelijk voldoende tools die een ondersteuning bieden bij het opslaan, beheren en koppelen van requirements. Zodoende kan er beter gevolgd worden hoe requirements zich vertalen naar (delen van) het eindproduct. De pre-RST daarentegen is veel lastiger te ondersteunen, omdat het vertalen van wensen uit de organisatie naar een requirementsspecificatie een proces is waar veel gemakkelijker fouten gemaakt kunnen worden door verschillen in interpretatie en communicatie. Het gaat hierbij om vragen als '*Wie is verantwoordelijk voor het toevoegen van deze requirement?*', '*Naar wie moet ik verwijzen voor meer informatie?*' en '*Was deze requirement het gevolg van een bijeenkomst van meerdere stakeholders of een individu?*'. Een tekort aan pre-RST levert volgens Gotel en Finkelstein het vaakst problemen op. [4]

Hiernaast ontbreekt een universele definitie van RT, waardoor het begrip vanuit verschillende richtingen bekeken kan worden. Dit vergroot het gebied waarin verschillende mensen over hetzelfde begrip pogen te spreken en zorgt voor verschil in nadruk. Is een definitie bijvoorbeeld gericht op het documenteren van relaties tussen en informatie over artefacten en de wijzigingen daarvan, of gaat het meer om de mogelijkheid te kunnen controleren dat requirements zijn geïmplementeerd. Verder is er geen gemeenschappelijke mening over het bestaan van de problemen en lopen de mogelijke oplossingen dus uiteen. [4]
In de literatuur wordt over het algemeen gebruik gemaakt van de eerder genoemde definitie van Gotel en Finkelstein, toch is het nog geen gemeengoed. In [7] wordt bijvoorbeeld vermeld dat "het doel van RT is om te valideren dat alle requirements voor het product zijn geïmplementeerd en getest". Dit wordt geïllustreerd met een traceability matrix die requirements koppelt aan secties in het ontwerp en tests. Deze definitie richt zich dus volledig op post-RST en laat pre-RST buiten beschouwing. Ook de SCM funcionaliteit Requirements Tracing[1], zoals beschreven in 2.5, richt zich puur op post-RST. Dit betreft een functionaliteit die RT mogelijk maakt en gaat uit van een definitie van RT zonder pre-RST.

Een van de oorzaken is volgens Arkley en Riddle [6] het scheiden van het ontwikkelproces van het traceability proces. In het betreffende survey werd namelijk duidelijk dat een apart kwaliteitsteam vaak alle RT werkzaamheden voor hun rekening nam en alleen het team overweg kon met een tool. Hoewel de ontwikkelaars hierdoor meer tijd hadden verlaagde de kwaliteit van de RT informatie. Zo waren er bijvoorbeeld meer foute RT relaties gelegd. De conclusie is dat alleen mensen die bij het ontwikkelproces betrokken zijn goed in staat zijn om RT informatie goed te registreren. Nu zijn zij juist degenen die daar geen direct voordeel aan ondervinden.

4 RT ondersteuning

In dit hoofdstuk wordt ingegaan op ondersteuning van RT. Er worden eerst technieken (4.1) en tools (4.2) behandeld, waarna enkele voorbeelden besproken worden om hiervan een beeld te krijgen (4.3). Dan wordt een vergelijking gemaakt tussen de besproken voorbeelden (4.4). Tot slot wordt teruggeblikt op hoofdstuk 3: “RT problemen”, waarbij wordt aangegeven in hoeverre RT ondersteuning oplossingen biedt voor de genoemde problemen (4.5).

4.1 Technieken

Om RT te ondersteunen bestaan er allerlei technieken om relaties aan te geven, verwijzingen te maken en statussen toe te kennen[4]. ‘Cross referencing schemes’ is er een van, waarbij met behulp van een schema verwijzingen gemaakt kunnen worden tussen requirements en bijvoorbeeld systeemelementen. Dit kan ook in de vorm van ‘hypertext’, waarbij binnen één of meerdere tekstdocumenten met hyperlinks verwijzingen gemaakt kunnen worden. Andere voorbeelden zijn ‘matrices’ en ‘templates’. In een matrix kun je een overzicht geven van requirements en een verwijzing plaatsen. Het is een tabel waarin je per requirement zaken kunt aangeven als het soort requirement, tot welk subsysteem of element het behoort, in welk document de test is terug te vinden en samenhangende of afgeleide requirements. Hoe zo’n matrix eruit ziet hangt af van het project en welke gegevens relevant zijn. Bij bedrijven of in projecten kunnen ook templates gehanteerd worden, zodat alle betrokken individuen op gelijke wijze gegevens opslaan met betrekking tot requirements.

Hiernaast kunnen talen, modellen en methoden van pas komen om RT te ondersteunen. Hiermee wordt op een formelere wijze getracht om informatie omtrent requirements op te slaan. Denk hierbij aan modellen die relaties leggen tussen entiteiten, verschillende processen onderscheiden tijdens de ontwikkeling en requirements beheren en koppelen. Formele talen en methoden kunnen gebruikt worden om requirements op een eenduidige manier op te slaan en beheren.
De kwaliteit van de resulterende RT hangt af van het naleven van de procedures die in de betreffende ontwikkeling gelden. Ook moet opgemerkt worden dat per geval overwogen dient te worden welke rol RT speelt en welke RT technieken daarbij passen.

4.2 Tools

Veel technieken die RT ondersteunen zijn geautomatiseerd en deze zijn onder te verdelen in onderstaande tools[4].

- *Algemene tools*
  Onder anderen tekstverwerkers, spreadsheets en databases. Deze kunnen handmatig geconfigureerd worden om bepaalde technieken toe te passen, bijvoorbeeld cross references en matrices.
- *Speciale tools*
  Dit zijn tools die zaken kunnen weergeven omtrent de requirements engineering. Bijvoorbeeld een programma dat relaties legt tussen ideeën en requirements of tussen requirements en testcases.

- *Workbenches*

Een workbench bevat een verzameling van enkele algemene of speciale tools die een gelijksoortige set van activiteiten ondersteunt. Zulke systemen zijn vaak gebaseerd op database management systemen en bevatten allerlei tools om requirements te verzamelen, indelen, bewerken, koppelen en dergelijke. Systemen waarin RT en requirements management centraal staan zijn vaak workbenches en worden aangeduid met requirements management tools.

- *Omgevingen*
  Een omgeving integreert allerlei tools die nodig zijn voor de ontwikkeling en kan RT van dienst zijn gedurende het gehele traject. De integratie vindt plaats op basis van een algemene taal, structuur of methode; er kunnen dus ook tools van verschillende leveranciers geïntegreerd worden. Een omgeving wordt afgestemd op het project en RT heeft meestal geen hoofdprioriteit.

4.3 Voorbeelden

Op dit moment zijn er op internet tientallen en misschien wel honderden tools te vinden op het gebied van requirements. De meeste zijn workbenches die zich richten op requirements management. Ter illustratie worden twee requirements management tools behandeld, waarbij enkele technieken naar voren komen. De tools zijn DOORS en Analyst Pro, die in respectievelijk 4.3.1 en 4.3.2 worden besproken.

Gotel en Finkelstein ondervonden dat de meeste problemen met RT te wijten zijn aan pre-RST. Hoewel de meeste ondersteuning zich richt op post-RST zijn er twee belangrijke methoden die zich op het gedeelte van de requirements analyse richten dat vooraf gaat aan het opstellen van de requirementsspecificatie. Deze zijn Contribution Structures (CS) [8] en PRO-ART (Process and Repository Based Approach to Requirements Traceability) [9], die in respectievelijk 4.3.3 en 4.3.4 worden besproken.

4.3.1 DOORS

Producent: Telelogic, http://www.telelogic.com

DOORS (Dynamic Object Oriented Requirements System) is een requirements management tool dat momenteel zeer populair is en veel wordt gebruikt. Requirements worden in DOORS beschouwd als objecten waaraan attributen kunnen worden toegevoegd, zoals bijvoorbeeld ‘approved’, waarmee kan worden aangegeven of een requirement is goedgekeurd, en ‘id’ voor requirementidentificatie. Iedere requirement kan bovendien gekoppeld worden aan één of meerdere andere requirements en kan verwijzen naar tests. Daarnaast wordt een historie bijgehouden van alle wijzigingen die hebben plaatsgevonden met bijbehorend tijdstip en betreffende gebruiker. Requirements kunnen worden weergegeven middels een gestructureerde interface, waarbij gekozen kan worden uit diverse views, zoals een view met attributen, met slechts de titels, met budgetvermelding, etcetera.
Het bevat verder de volgende functionaliteiten.
-Een ‘change proposal and review system’ dat gebruikers de mogelijkheid biedt om (beargumenteerde) voorstellen te doen tot verandering.
-Een uitgebreide gebruikersfunctionaliteit, waarmee verschillende gebruikers kunnen worden aangemaakt met verschillende gebruiks- en toegangsrechten.

-Een ‘traceability analysis’ geeft (tot een van tevoren opgegeven diepte) weer welke requirement(s) naar het huidige verwijst of verwijzen.
-Een ‘impact analysis’ geeft (tot een van tevoren opgegeven diepte) weer naar welke requirement(s) het huidige verwijst.
-De zogenaamde ‘traceability explorer’ biedt de mogelijkheid op overzichtelijke wijze requirements met bijbehorende links (van en naar requirements) weer te geven.
-Gegevens kunnen op allerlei manieren worden weergegeven en worden ge-exporteerd naar andere bestandsformaten zoals Rich Text Format.
-Telelogic biedt verder nog andere tools aan uit de DOORS familie, zoals een speciale versie voor wereldwijde ondernemingen en een uitbreiding om met DOORS te kunnen werken vanaf een externe internettoegang.

4.3.2 Analyst pro

Producent: Goda Software, Inc., http://www.analysttool.com

Analyst pro is bedoeld voor requirements management, traceability en analyse en dekt het software ontwikkelingstraject van requirements specificeren tot ontwerp en implementatie. Requirements worden gekoppeld aan een bron, gebruiker en deel van het eindproduct. Analyst pro bevat onder anderen de volgende functionaliteiten:
Importing Requirements- Bestaande documenten met requirements kunnen in diverse formaten worden geïmporteerd.
Requirements Sharing - Gebruikers kunnen requirements toevoegen en delen met andere gebruikers van het systeem.
Requirements Configuration Management - Doordat requirements, gebruikers en artefacten aan elkaar gekoppeld worden door het leggen van relaties, kan het systeem bepalen wat er verandert, wanneer een requirement verandert.
Requirements Change Management - Analyst Pro houdt automatisch een logboek bij van veranderingen aan het project, waarbij de verandering zelf en de gebruiker worden opgeslagen.
Requirements Assignment Gebruikers kunnen requirements toewijzen aan teamleden en hun status bijhouden.
Requirements Graphs - Gebruikers kunnen allerlei documenten en visuele weergaven genereren van de requirements, attributen en hun statistieken.

4.3.3 Contribution Structures

Een Contribution Stucture is een dynamisch model van alle partijen (contributors) die een aandeel hebben in het genereren van requirements artifacten (contributions). In het model kunnen links worden gelegd tussen de contributors, tussen de contributions en tussen beide. De sociale structuur kan worden weergegeven door het aangeven van de soort rol van de contributor, zoals degene die de oorspronkelijke bron is, de auteur die verantwoordelijk is voor syntax en semantiek en de persoon die verantwoordelijk is voor het beheer. Verder kan het soort relatie worden aangegeven tussen contributions en de zekerheid waarmee informatie wordt vastgelegd. CS biedt talloze mogelijkheden om de gegevens in het model te representeren, zoals allerlei soorten tabellen en modellen waarin bijvoorbeeld soorten rollen, statussen en attributen kunnen worden weergegeven.

4.3.4 PRO-ART

PRO-ART biedt een RT framework met drie dimensies, te weten representatie, specificatiediepte en overeenstemming, waarin het proces van requirements engineering gezien kan worden als een lijn die zich ontwikkelt langs de drie dimensies. Representatie kan variëren van informeel tot formeel, specificatie van onvolledig tot compleet en overeenstemming (over de specificatie) van verschillende, persoonlijke inzichten tot algemene overeenstemming. Verder biedt PRO-ART een zogenaamde trace-repository om structuur aan te brengen in RT informatie en selectief requirements te kunnen herleiden. Tot slot is er een functionaliteit die het mogelijk maakt automatisch RT informatie vast te leggen. Dit is mogelijk doordat er bij een toevoeging, wijziging of verwijdering van een element uit de trace-repository de bijbehorende afhankelijkheden worden bijgewerkt, indien nodig.

4.4 Vergelijking

Nu er enkele manieren besproken zijn om RT informatie te representeren en enkele voorbeelden behandeld zijn wordt het tijd om een vergelijking te maken tussen deze voorbeelden en na te gaan in hoeverre deze van nut zijn voor de genoemde problemen in hoofdstuk 3.

DOORS en Analyst Pro zijn uitgebreide requirements management tools en allebei beslaan ze meerdere fasen van het ontwikkelingstraject en niet alleen software requirements, maar ook system requirements. CS en PRO-ART zijn beide complexe modellen, die het mogelijk maken de meer sociale kanten van RT informatie te beheren in een systeem.
DOORS en Analyst Pro zijn beide duidelijk alleen gericht op post-RST, omdat er geen aandacht wordt geschonken aan het proces dat vooraf gaat aan het invoeren van requirements in het systeem. Er wordt wel bijgehouden wie requirements toevoegen en wijzigen in het systeem, maar bijvoorbeeld niet welke partijen betrokken zijn geweest bij de totstandkoming ervan en mate van overeenstemming. Verder bevatten ze beide functionaliteiten om een traceability analysis en een impact analysis uit te voeren, om te zien welke requirements naar het huidige requirement verwijzen en hoe het systeem verandert, wanneer een requirement verandert. DOORS is meer dan Analyst Pro bedoeld voor grote, internationale organisaties met veel gebruikers en biedt daarvoor enkele uitbreidingsmogelijkheden.

CS en PRO-ART richten zich daarentegen duidelijk wel op pre-RST en presenteren zichzelf als een tool dat als uitbreiding kan fungeren op bestaande software. Ze bieden allebei de mogelijkheid om, in een aantal van tevoren vastgelegde klassen, informatie op te slaan; PRO-ART geeft hierin de meeste vrijheid. Beide tools zijn in staat de sociale structuren weer te geven op diverse manieren. Gebruikers kunnen RT informatie raadplegen en eenvoudig requirements herleiden naar hun bron. De veranderingen kunnen in een logboek worden bijgehouden met bijbehorende afwegingen. Over het algemeen is PRO-ART een meer veelomvattende benadering die robuust is en uitbreidbaar [10].

Over een keuze van RT ondersteuning en de vraag in hoeverre gebruik ervan iets oplevert is nog het volgende op te merken [11]. Requirements kunnen het best getraceerd worden met verschillende technieken, die ieder hun eigen voor- en nadelen hebben. Functionele requirements zijn het best te traceren met de functionaliteiten van een requirements

management tool zoals DOORS en Analyst Pro. Nonfunctionele requirements die bijvoorbeeld kwaliteitsaspecten belichten, zijn hier minder geschikt voor, omdat deze minder concreet zijn, en vragen andere manieren. Wil je dus meerdere soorten requirements optimaal kunnen traceren, dan zou je dus verscheidene technieken naast elkaar moeten gebruiken. Deze benadering leidt in de praktijk tot een onhoudbare situatie met een overdaad aan links. Daarom moeten beslissingen over RT genomen worden gezien vanaf projectniveau met gewenste RT behoeften.

4.5 Oplossingen voor RT problemen?

Wanneer je de uit eerder onderzoek naar voren gekomen RT problemen en oorzaken in het kader plaatst van de in dit hoofdstuk besproken technieken, tools en voorbeelden, kunnen daar de volgende opmerkingen over gemaakt worden.

De oorzaken van de genoemde RT problemen zijn samen te vatten in drie kernpunten: een slecht onderscheid tussen pre-RST en post-RST, het ontbreken van een universele definitie van RT en het scheiden van het ontwikkelproces en het RT-proces. De besproken technieken en tools lijken niet terug te kunnen grijpen op de oorzaken van de problemen, omdat technieken en tools juist worden afgestemd op een organisatie of project uitgaande van een bepaalde definitie van RT, in plaats van andersom. Eerst moet men het eens worden over het begrip RT en over de wijze waarop het waarop het toegepast zou moeten worden, pas dan kunnen technieken en tools dienen als ondersteuning.

Daarnaast worden nu de problemen besproken.
Gotel en Finkelstein geven aan dat requirements niet altijd te traceren zijn en dat onjuiste eindproducten worden opgeleverd door missende informatie of verwijzingen en een karige probleemanalyse. Dit zijn problemen die door een nauwkeurige manier van werken zouden kunnen worden voorkomen, maar niet zozeer door technieken en tools. Wanneer namelijk informatie of verwijzingen missen, ligt dit over het algemeen aan de analysten die deze achterhalen of beheren en een karige probleemanalyse is ook te wijten aan menselijke inspanningen.

Dan de door Egyed en Grünbacher genoemde problemen. Het eerste probleem betreft een hoge complexiteit en veel moeite, doordat het verkrijgen van RT informatie vaak een handmatig proces is. In de besproken ondersteuning is duidelijk naar voren gekomen dat er functionaliteiten bestaan die automatisch RT informatie vastleggen en beheren. Met zowel DOORS als Analyst Pro kun je ook een duidelijk overzicht genereren van relaties. Dit vormt dus een oplossing voor dit probleem.
Probleem twee en vier betreffen missende en onvoldoende RT informatie. Oplossingen voor deze problemen zijn niet te vinden bij tools en technieken, omdat deze afhankelijk zijn van hun gebruikers. Hiervoor zullen requirements analysten meer relevante informatie moeten achterhalen, die wel zorgt voor een voldoende compleet beeld over de loop van de requirements.
Tot slot is het omgaan met het leggen van betekenisvolle relaties in een complexe omgeving en met veranderende requirements (problemen drie en vijf) in principe mogelijk met een geschikte tool en adequate manier van werken. De ondersteuning voor deze problemen biedt oplossingen, maar deze moeten wel volledig gebruikt worden.

De door Arkley en Riddle aangestipte problemen hangen samen met incomplete, inaccurate en verouderde RT informatie. Het eerste probleemgebied betreft het data invoer en tool gebruik, waaronder presentatie van gegevens en navigatie, wat uiteraard wel een punt is dat concreet verholpen kan worden in een tool. Het gebrek aan begrip en waarneembare bureaucratie (probleem twee) is een probleem dat niet te verhelpen is met een techniek of tool, maar met de manier van werken en betrokkenheid. De medewerkers zouden meer betrokken kunnen worden en gestimuleerd bij hun inbreng in het geheel. Een bureaucratische manier van werken helpt hier niet bij. Voor het gebrek aan direct voordeel voor de hoofdlijnen in het ontwikkelproces (probleem drie) geldt ook dat een bureaucratische manier niet bevorderlijk werkt. Wanneer het veel moeite kost om relevante RT informatie te verwerken, tijdens de hoofdwerkzaamheden, zal een medewerker hierin eerder tekort schieten. Meer inzicht in het gehele ontwikkelingstraject zal wellicht ook helpen om te doen inzien dat het van groot belang is om RT informatie te vergaren.

Voor een deel van de genoemde problemen, maar niet voor de oorzaken, worden oplossingen geboden door de besproken ondersteuning. Daarnaast blijkt dat de oorzaken en het overige deel van de problemen niet te verhelpen zijn met het gebruik van een techniek of tool, maar dat oplossingen ook gezocht moeten worden in maatregelen die te maken hebben met organisatorische en menselijke factoren.

5 Praktijk

Om een vergelijking te kunnen maken met de tot nu toe behandelde literatuur heb ik drie interviews (zie bijlage) gehouden bij bedrijven die met RT werken, te weten X, Y en Z (5.1). Ter vergelijking zal ik enkele overeenkomsten en verschillen bespreken wat betreft de werkwijzen van deze bedrijven met betrekking tot RT (5.2 – 5.7). Hierbij worden de relevante punten ook in de literatuur geplaatst.

5.1 Drie bedrijven

Allereerst vermeld ik hier de bedrijven aan de hand van een korte samenvatting van de interviews. In verband met gevoelige bedrijfsinformatie worden de bedrijven gerepresenteerd met een afkorting: X, Y of Z.

5.1.1 X

X ontwikkelt zeer omvangrijke en complexe systemen aan de hand van het V-model (zie figuur 3). Per niveau (systeem, subsysteem, cluster, element) worden specificaties beheerd in documenten, die centraal worden opgeslagen. RT heeft geen hoofdprioriteit, maar de werkwijze heeft een sterk time-to-market uitgangspunt. De klant hecht grote waarde aan het halen van de deadline, waar de ontwikkeling op wordt afgestemd.

5.1.2 Y

Y werkt ook met het V-model, maar hier is de ontwikkeling afgestemd op massaproductie. Verder is time-to-market hier van groot belang, omdat het op te leveren product voor de klant een schakel is in het productieproces van het uiteindelijke product. Wanneer Y de deadline dus niet haalt zou dit voor de klant een (dure) vertraging opleveren.
Een niveau in het V-model kan soms voor meerdere klanten gelijk zijn (bijvoorbeeld het platform), terwijl een onderliggend niveau verschilt (een functionaliteit). RT wordt ondersteund met DOORS.

5.1.3 Z

Z ontwikkelt systemen met hoge kwaliteitseisen. De klant hecht grote waarde aan het correct functioneren van het systeem; de manier van ontwikkeling wordt hier dan ook op afgestemd. Dat betekent dat de klant veel meekijkt in het ontwikkelingsproces en er steeds mijlpalen afgesloten worden met essentiële tests. RT wordt hier ook ondersteund met DOORS.

5.2 Time-to-market versus kwaliteit

Een van de duidelijkste verschillen ligt op het uitgangspunt in de werkwijzen van de bedrijven. De rol die een bedrijf inneemt ten opzichte van een klant is essentieel voor de plaats die RT inneemt in het ontwikkelingstraject. Bij X en Y is time-to-market zeer belangrijk gedurende de ontwikkeling, terwijl bij Z de kwaliteit zeer hoog in het vaandel staat. Bij alle bedrijven wordt RT belangrijk gevonden, maar doordat Z producten ontwikkelt met hoge kwaliteitseisen, waarbij een klein foutje desastreus kan zijn is kwaliteit een belangrijker uitgangspunt dan time-to-market. Bij de andere bedrijven is kwaliteit ook belangrijk, maar in mindere mate.

Deze verschillen vertalen zich in de rol van RT en welke RT informatie wordt gedocumenteerd. X heeft de minst strikte omgang met RT in de zin dat veel processen nog handmatig gebeuren en dat RT ook geen deliverable vormt. Bij Y krijgt de klant de resultaten van essentiële tests te zien en bij Z moet nog meer (aan)getoond worden aan de klant en vormt RT weldegelijk een deliverable. Het is duidelijk dat het uitgangspunt van de werkwijze van het bedrijf en daarmee samenhangend, de rol die de klant speelt, van invloed zijn op RT.

5.3 Soort product

Wat óók van invloed is op RT is het soort product dat geproduceerd wordt. X is gericht op het ontwikkelen van een zeer groot en complex product en Y op massaproductie. Z daarentegen ontwikkelt producten met hoge kwaliteitseisen. Allen maken een onderscheid in bestaande en innovatieve producten.
Bij reeds bestaande producten, dus producten die het bedrijf al eens geleverd heeft, wordt RT gebruikt om te traceren welke elementen eventueel aangepast moeten worden aan de omgeving van de klant. Bij innovatieve producten echter, speelt RT een grotere rol, omdat dan nog niet van tevoren bekend is of requirements haalbaar zijn. Veranderingen van requirements komen in dat geval dus vaker voor en RT is dan belangrijk om wijzigingen op systeemniveau goed te vertalen naar de lagere niveaus.

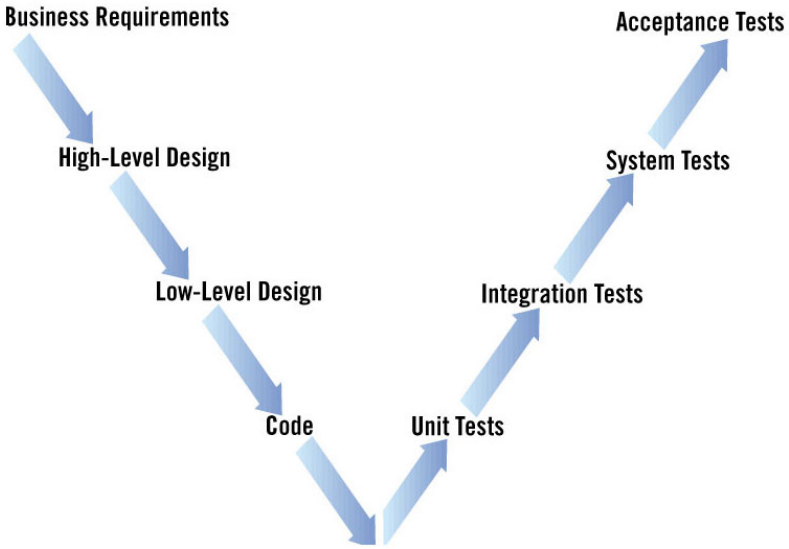
5.4 Ondersteuning

De RT ondersteuning die de bedrijven gebruiken varieert nogal. X werkt met een document management systeem dat simpelweg losse documenten beheert waarin de requirements per niveau worden opgeslagen. Y en Z werken beide met het requirements management systeem DOORS. Y werkt daarnaast ook nog met Rational Rose, waarin requirements worden toegewezen aan subsystemen, en met TEMPPO, dat testcases beheert en koppelt aan requirements.
Alle bedrijven vinden ook ondersteuning door gebruik van het V-model, waarop wordt teruggekomen in de volgende paragraaf.

5.5 Pre-RST en post-RST

Net als in de literatuur is er bij alle bedrijven een onderscheid tussen pre-RST en post-RST, al wordt de terminologie niet gebruikt. Een aparte afdeling of speciale groep mensen onderhoudt het contact met de klant en zorgt ervoor dat er een requirementsspecificatie wordt opgesteld, die volgt uit de juiste bronnen. De klant speelt hierin een grote rol en beschikt over het algemeen over voldoende kennis om requirements zelf op te kunnen stellen.

Bij de bedrijven wordt post-RST toegepast in termen van het V-model (zie figuur 3). X en Y werken met dit model als basis. Z werkt in termen van horizontale en verticale RT. Dit komt in principe op hetzelfde neer, omdat verticale RT het vertalen is van requirements op systeemniveau naar element- en functieniveau en terug en horizontale RT is het op elk niveau valideren en verifiëren van requirements met behulp van tests.
Hoewel het V-model niets voorschrijft over planning ervaren de bedrijven het als een overzichtelijke manier om mee te werken. Bij producten met veel subsystemen en disciplines die apart van elkaar ontwikkeld worden is het een manier om de requirements per niveau te kunnen bekijken en deze steeds verder te decomponeren.



Figuur 3: V-model [12]

Het V-model is een veelgebruikte manier in organisaties en projecten om een productontwikkeling mee te structureren. Het is een algemeen raamwerk dat per organisatie of project moet worden geconcretiseerd.

De linker tak van de V representeert de verdeling van de specificatie in niveaus, waarbij het hoogste niveau het systeemniveau voorstelt en bij ieder lager niveau steeds requirements opgedeeld worden en uitgewerkt worden in meer detail. De rechter tak van de V representeert vervolgens de tests waaraan ieder corresponderend niveau op de linker tak moet voldoen. Hoewel het V-model een volgorde lijkt voor te schrijven, kan de uitvoering van de activiteiten deels parallel plaatsvinden. Zo kan het opstellen van de acceptance tests al plaatsvinden wanneer de requirementsspecificatie gereed is; dit geldt voor ieder niveau. Bovendien kunnen eenheden die geïmplementeerd zijn al getest worden en zo, volgens de rechter tak van het model, per unit, functie of subsysteem getest worden, waarbij als laatste de acceptance test aangeeft of het systeem als geheel voldoet aan de verwachting van de klant.

5.6 Doelen

Voor X geldt dat het doel van RT is om requirements per niveau (discipline, subsysteem, element) terug te kunnen vinden, zodat medewerkers op ieder niveau een afbakening hebben en weten hoe deze zich verhoudt tot het geheel.
Voor Z geldt dat RT essentieel is om aan te tonen aan de klant dat (een deel van) het product voldoet aan de requirements. De klant heeft hier veel inzicht in het proces en krijgt uitslagen van tests te zien, die vaak als contractuele mijlpalen in het project gelden. Bij Y helpt RT ook om de kwaliteit te waarborgen, maar kijkt de klant minder mee. Relevante tests worden wel aan de klant getoond. Voor Y en Z geldt overigens ook dat RT helpt om overzicht te hebben over de niveaus, omdat door het leggen van relaties zichtbaar is hoe requirements verdeeld zijn over verschillende componenenten.
Bij de bedrijven komen dus twee verschillende doelen naar voren van RT. Enerzijds is het een hulpmiddel om structuur aan te brengen in de requirements en het ontwikkelingstraject. Het V-model helpt hierbij door als raamwerk te fungeren. Anderzijds ondersteunt RT het waarborgen van kwaliteit. Een klant kan zo controleren dat requirements terugkomen in het eindproduct.

5.7 Problemen

In de interviews is nauwelijks naar voren gekomen dat er problemen zijn met het werken met RT. Bij X speelt RT een kleine rol en wordt het meer als handvat en ondersteuning gebruikt. Er wordt in nauwe samenspraak met de klant gedurende het ontwikkelingstraject afgewogen wat er in de resterende tijd nog kan gebeuren en dit levert altijd een juist resultaat op. Ook bij Y en bij Z wordt met de klant steeds doorgesproken hoe de ontwikkeling verloopt, waarbij RT

als belangrijk hulpmiddel wordt gebruikt om het overzicht te bewaren en de kwaliteit te waarborgen. Bij alle bedrijven zijn er enkele personen die zich speciaal met RT bezighouden. Het in hoofdstuk 3 genoemde ‘probleem’ dat wel bestaat is dat het verkrijgen van RT informatie een handmatig proces is, dat complex is en veel moeite kost. Het leggen van links zal deels handmatig moeten gebeuren, bijvoorbeeld bij het decomponeren van requirements over disciplines of subsystemen. Je zou wel kunnen zeggen dat het bij X weliswaar een meer handmatig proces is dan bij Y en Z, die ondersteuning hebben van DOORS. Daarbij worden relaties op een meer geautomatiseerde manier gelegd en aangepast bij het toevoegen of wijzigen van requirements; bovendien kan hierdoor gemakkelijk een overzicht gegenereerd worden. Bij X moet dit geheel handmatig gebeuren.
Een ander probleem is dat bij Z de verschillende systemen niet gekoppeld kunnen worden, waardoor het meer moeite kost om RT informatie te beheren en te verwerken. Requirements worden beheerd in DOORS, toegewezen aan subsystemen in Rational Rose en de testcases worden in TEMPPO beheerd. Wanneer dit geïntegreerd zou worden zou dit waarschijnlijk voor de medewerkers gemakkelijker zijn en meer inzicht in het geheel opleveren, omdat alle handelingen dan in één tool kunnen plaatsvinden en alle informatie aanwezig is.

6 Conclusie

Eerst wordt antwoord gegeven op de deelvraag (6.1), daarna op de hoofdvraag (6.2).

6.1 Deelvraag

Welke factoren zijn van invloed op de rol van requirements traceability binnen softwareontwikkeling?

Uitgangspunt organisatie

Uit de interviews is naar voren gekomen dat het uitgangspunt waarmee een organisatie opereert van groot belang is voor de rol die RT vervult. Wanneer een bedrijf sterk gericht is op time-to-market, waarbij het van groot belang is dat de klant het product op tijd heeft, speelt RT een ondersteunende rol bij het structureren van requirements. De klant heeft meestal wel interesse in RT tijdens de productontwikkeling, maar ziet het niet als belangrijkste eis voor een correct werkend product. Bij een bedrijf, daarentegen, dat sterk gericht is op het leveren van producten met hoge kwaliteit, speelt RT een sterk kwalitatieve rol. De klant eist meestal inzicht in de processen en RT informatie, om de kwaliteit te kunnen controleren. Verder wordt RT gebruikt als middel om correctheid te kunnen aantonen. Dit is van levensbelang voor het slagen van een project.

Soort product

Het soort product is ook belangrijk voor de rol van RT. Als het om een routineproduct gaat, wordt RT gebruikt om te kunnen traceren welke componenten van het product moeten worden aangepast aan de (nieuwe) situatie van de klant. Als een klant bijvoorbeeld een subsysteem niet wenst, kan precies worden nagegaan welke requirements dan komen te vervallen en welke tests niet hoeven worden uitgevoerd. In het geval van de ontwikkeling van een innovatief product verloopt de ontwikkeling anders en speelt RT een belangrijke rol. Er treden namelijk erg vaak wijzigingen op, waarbij RT helpt bij het juist vertalen van een wijziging op hoog niveau naar de lagere niveaus.

6.2 Hoofdvraag

Wat zijn de verschillen en overeenkomsten tussen de literatuur over requirements traceability en de praktijk ervan?

RT ondersteuning

De literatuur bespreekt talloze mogelijkheden om RT te ondersteunen. Er bestaan zeer veel tools die op een of andere manier RT informatie beheren, met allerlei verschillende technieken, maar er worden er slechts enkelen echt gebruikt in de praktijk. DOORS is een van deze systemen, die in twee van de geïnterviewde bedrijven gebruikt wordt. Dit tool is vrij omvangrijk en RT is slechts een van de vele functionaliteiten. Tools die RT ondersteunen bieden over het algemeen talloze andere functionaliteiten, zoals in een requirements management tool.

Andere RT ondersteuning betreft het V-model als raamwerk voor het ontwikkelingstraject. Twee van de drie geïnterviewde bedrijven gebruiken het en één gebruikt een vergelijkbare

structuur; het vormt een goed kader voor de verschillende medewerkers tijdens de ontwikkeling. Medewerkers kunnen namelijk op ieder niveau goed zien hoe requirements worden gedecomponeerd over de verschillende niveaus en hoe requirements samenhangen met elkaar. Het is opvallend dat het V-model in de besproken literatuur niet naar voren komt als RT ondersteuning.

Vervolgens ga ik nog in op het onderscheid tussen pre-RST en post-RST. Pre-RST behoort bij de bedrijven niet tot het takenpakket van de ontwikkelaars, maar tot die van een aparte afdeling. Deze afdeling of speciale groep mensen stelt samen met de klant requirements op tot een specificatie. Hierbij is het belangrijk dat deze bedrijven over het algemeen klanten hebben die zelf ook voldoende kennis bezitten om requirements op te kunnen stellen.

RT problemen
In de literatuur zijn veel problemen beschreven en komen oplossingen naar voren. De oplossingen dekken de problemen voor een deel en voor een ander deel moeten oplossingen gezocht worden in maatregelen die te maken hebben met organisatorische en menselijke factoren.
Uit de interviews komen nauwelijks RT problemen naar voren en er wordt de nodige ondersteuning gebruikt voor RT.

RT overwegingen
In de literatatuur wordt voornamelijk ingegaan op problemen met RT en mogelijke ondersteuning en niet zozeer op overwegingen bij invoering of toepassing. In 4.4 wordt weliswaar genoemd dat keuzes over RT ondersteuning genomen moeten worden op projectniveau met gewenste RT behoeften en dat requirements het beste met meerdere technieken getraceerd kunnen worden, dit is echter nog vrij onduidelijk. De twee factoren die invloed hebben op de rol van RT uit de deelvraag samen met de twee doelen van RT uit de praktijk geven hier meer invulling aan. Bij de overweging van een bedrijf om RT in meer of mindere mate te gebruiken in het ontwikkelingstraject is het belangrijk in hoeverre het iets oplevert. Belangrijk om hierin mee te nemen is dat het uitgangspunt van de organisatie en het soort product de rol van RT bepalen. Daarnaast het doel waarmee RT gebruikt wordt van belang; wordt RT gebruikt als hulpmiddel om structuur aan te brengen in requirements en het ontwikkelingstraject, ondersteunt RT het waarborgen van kwaliteit, of zijn er nog andere doelen.

Bijlage

A. Vragenlijst

De interviews zijn afgenomen aan de hand van de volgende vragenlijst. De antwoorden zijn zoveel als mogelijk conform de kernpunten van deze lijst uitgewerkt.

Definitie

- Wat verstaat u onder RT?
    - Maakt u een onderscheid in soorten RT, bv. pre-RST en post-RST?

Doel

- Wat wilt u bereiken met RT?
- Worden de doelen ook bereikt?

Context

- Wat voor rol speelt RT bij uw bedrijf?
- Wat voor rol speelt RT binnen het ontwikkelingstraject?
- In welke fasen van een softwareontwikkelingstraject speelt RT een rol?
    - Hoe wordt in de verschillende fasen met RT omgegaan?
- Kan RT toegepast worden op alle soorten requirements?
    - Non-functionele, functionele, system requirements?
    - Worden requirements voor het geheel opgesteld (voor meerdere disciplines) of alleen voor software?

Klant

- Wat voor rol speelt RT in de afspraken met de klant (contractueel / juridisch)?
    - Is RT een garantie voor het naleven van requirements?
- In hoeverre heeft de klant te maken met RT?
    - Wordt de klant betrokken of wordt RT alleen intern gebruikt?

Hulpmiddelen

- Hoe wordt RT gewaarborgd?
- Worden speciale tools / hulpmiddelen gebruikt?
    - Waarom wel/niet? Alternatieven?
    - Gaat het om kleine tools of volledige omgevingen?

Overig

- Hoe wordt met RT omgegaan bij deelsystemen?
- Hoe wordt omgegaan met veranderende requirements?

Literatuur

[1] Pressman, Roger S., *Software engineering, A practitioner's approach*, sixth edition, McGraw-Hill, 2006.

[2] Kulak, D., Guiney, E., *Use Cases, Requirements in context*, second edition, Pearson Education Inc., 2004.

[3] Davis, Alan M., *Software Requirements Analysis and Specification*, Prentice-Hall, New Jersey, 1990.

[4] Gotel, O., Finkelstein, A., "An analysis of the requirements traceability problem", *Software Change Impact Analysis*, R. Arnold and S. Bohner, eds.: IEEE Computer Society Press, 1996.

[5] Egyed, A., Grünbacher, P., *Automating Requirements Traceability: Beyond the Record & Replay Paradigm*, Proceedings of the 17th IEEE International Conference on Automated Software Engineering, 2002.

[6] Arkley, P., Riddle, S., *Overcoming traceability benefit problem*, School of Computing Science, University of Newcastle upon Tyne, technical report CS-TR: 906, april 2005.

[7] O'Regan, G., *A Practical Approach to Software Quality*, Springer-Verlag, 2002.

[8] Gotel, O., Finkelstein, A., *Contribution Structures*, Proceedings of the 2nd International Symposium on Requirements Engineering, p. 100-107, York, U.K., maart 1995.

[9] Pohl, K., *PRO-ART: Enabling Requirements Pre-Traceability*, Proceedings of the Second IEEE International Conference on Requirements Engineering, p. 76-84, Colorado Springs, 1996.

[10] Stepanian, L., *Solving the Requirements Traceability Problem: A Comparison of Two Pre-Requirements Specification Traceability Enablers*, technical report CSC2106, 2004.

[11] Cleland-Huang, J., Zemont, G., Lukasik, W., *A Heterogeneous Solution for Improving the Return on Investment of Requirements Traceability*, Proceedings of the 12th IEEE International Requirements Engineering Conference, p. 230-239, 2004.

[12] *Software Development Magazine*, http://www.sdmagazine.com.