Automatiseer het automatiseringsbedrijf

Het optimaliseren van de ontwikkelstraat

Bachelor afstudeeronderzoek

Stef Roskam

December 2010 – Augustus 2011

Bedrijfsbegeleider: R. van der Sanden

Afstudeerbegeleider: dr. M.E. Iacob

Meelezer: L.O. Meertens MSc.

Faculteit Management en Bestuur

Universiteit Twente

Management samenvatting

Kader

Dit onderzoek is uitgevoerd bij Empirion B.V. in het kader van mijn bacheloropdracht Technische Bedrijfskunde. Empirion B.V. is onderdeel van de Empirion Groep en levert “Software as a Service” (SaaS) op het gebied van verzuimregistratie. Hierbij zijn de huidige bedrijfsprocessen geanalyseerd en zijn geoptimaliseerde bedrijfsprocessen voorgesteld, waarbij deze tevens voldoen aan de eisen voor een certificering volgens ISO 27001 (informatiebeveiliging).

Bevindingen

Op basis van een analyse van de huidige situatie is geconcludeerd dat de huidige bedrijfsprocessen niet voldoen aan de ISO 27001 eisen. Hierbij ontbreken formele processen voor het afhandelen van wijzigingsverzoeken & incidenten. Daarnaast is er weinig aandacht voor de informatiebeveiliging gedurende het ontwikkelproces en vindt kwaliteitsbewaking alleen aan het eind van het ontwikkeltraject plaats.

Resultaten

Voor het ontbreken van de formele processen voor het afhandelen van wijzigingsverzoeken & incidenten zijn drie verschillende bedrijfsprocessen opgesteld. Met elk van deze processen wordt voldaan aan de eisen van de ISO 27001 norm. Bij de medewerkers is er een duidelijke voorkeur om een van de drie processen niet in te voeren, voor een van de overige twee alternatieven is geen duidelijke voorkeur.

Voor het ontwikkelproces zelf zijn twee verschillende alternatieven opgesteld, waarbij de eerste gebruik maakt van de software-ontwikkelmethodiek Scrum. Scrum is een methodiek met een vaste releasecyclus en dagelijkse voortgangsoverleg. Het tweede alternatief is gebaseerd op het Rational Unified Process, hierbij ligt de releasecyclus niet vast en is er geen dagelijks voortgangsoverleg. Beide alternatieven maken gebruik van de Secure Development Lifecycle om informatiebeveiliging te integreren in het ontwikkelproces. Daarnaast maken beide alternatieven gebruik van geautomatiseerde testen, zodat kwaliteitsbewaking in het hele traject plaats vindt. Bij de medewerkers is een duidelijke voorkeur aanwezig voor het gebruik van het alternatief op basis van Scrum.

Aanbevelingen

Voor het invoeren van de voorgestelde verbeteringen wordt aanbevolen om een aparte werknemer vrij te maken of aan te nemen. Dit in verband met de werkdruk bij bestaande medewerkers.

Omdat tijdens de interviews de alternatieven alleen in hoofdlijnen behandeld zijn, is het ook aan te bevelen om voor de daadwerkelijke implementatie een implementatieplan uit te werken en door te spreken met de betrokken medewerkers, zodat hierop eventueel nog aanpassingen doorgevoerd kunnen worden.

Indien voor het alternatief op basis van Scrum gekozen wordt is het noodzakelijk dat er een goede leider van de dagelijkse update-meetings is, gezien de resultaten met deze meetings tot op heden is het in mijn ogen aan te bevelen om een medewerker hiervoor goed op te leiden.

Inhoudsopgave

Voorwoord

Voor u ligt het verslag van mijn bacheloropdracht bij Empirion B.V. Dit verslag is tevens het einde van mijn studieperiode aan de Universiteit Twente, een studieperiode die op z'n minst niet helemaal standaard verlopen is.

Mijn studietijd in Enschede is begonnen in 2002, maar werd na een jaar onderbroken voor een bestuursjaar bij de Drienerlose Roei-Vereniging Euros. Na dit bestuursjaar is mijn studie echter nooit meer op de eerste plaats komen te staan, het herinrichten van een roeiloods, coachen en werken beviel mij duidelijk beter. Uiteindelijk besloot ik in 2006 samen met een oud-bestuursgenoot een eigen transportbedrijf op te starten, dit betekende, dacht ik toen, definitief het einde van mijn studie. Toch heb ik in 2010 de draad weer opgepakt en heeft dit geresulteerd in onderstaand verslag van mijn bacheloropdracht.

Net als mijn studieverloop is ook de bacheloropdracht niet op de gebruikelijke wijze uitgevoerd. In plaats van in een kwartiel, is de opdracht uitgevoerd tussen december 2010 en juni 2011 naast het afronden van de resterende vakken van de bachelor. Hiervoor zijn een aantal redenen aan te wijzen:

De praktijkwerkzaamheden bleken wederom een stuk aantrekkelijker dan de vakken op de UT, waardoor de voorbereiding van tentamens vaak aankwam op het laatste moment. Daarnaast is mijn vader op 10 mei plotseling overleden, helaas maakt hij de afsluiting van mijn bachelor niet meer mee...

Bij deze wil ik ook Roel van der Sanden bedanken voor de flexibiliteit m.b.t. de uitvoer van de opdracht binnen Empirion, zonder deze flexibiliteit was de afronding van mijn bachelor zeker niet gelukt. Verder gaat mijn dank uit naar mevrouw Iacob voor de kritiek op mijn verslag en de hulp bij het zoeken van literatuur en het vinden van de juiste richting van mijn onderzoek en naar de heer Meertens om mijn opdracht als meelezer te beoordelen. Ook wil ik bij deze Xander van der Voort bedanken voor zijn advies en feedback m.b.t. de processen rond ISO 27001. Tenslotte bedank ik alle collega's binnen Empirion voor de input gedurende de interviews, de feedback en de prettige werksfeer.

Verder wens ik u veel plezier bij het lezen van het verslag.

's-Hertogenbosch, augustus 2011

Stef Roskam

1 Probleemomschrijving en onderzoeksaanpak

Dit hoofdstuk beschrijft de onderzoeksopzet. Om een goed beeld te krijgen van de onderzoeksomgeving geeft de tweede paragraaf een korte beschrijving van Empirion B.V., de organisatie waarbinnen deze bacheloropdracht uitgevoerd is, de opdracht en de gebruikte methodiek. In de derde en vierde paragraaf vindt de probleemidentificatie plaats en wordt de probleemaanpak beschreven.

1.1 Opdrachtomschrijving

1.1.1 De organisatie

Empirion B.V. is onderdeel van de Empirion Groep en levert “Software as a Service” (SaaS) op het gebied van verzuimregistratie. Hierbij draait de applicatie in een door Empirion beheerde omgeving en is de applicatie door de gebruiker te benaderen door middel van een webbrowser. Empirion is gevestigd in 's-Hertogenbosch en er zijn 25 mensen werkzaam binnen het bedrijf. Het bedrijf is een van de top-3 IT-leveranciers voor verzuimmanagement binnen Nederland en streeft er naar om binnen 3 jaar marktleider te zijn. Daarnaast is in 2010 het verzuimsysteem van Empirion door BG Magazine, het vaktijdschrift voor bedrijfsgezondheid, uitgeroepen tot een van de beste systemen op het gebied van verzuimmanagement met een minimaal verschil ten opzichte van de winnaar.

Binnen Empirion BV wordt de software in een eigen opgezette “software ontwikkelstraat” ontwikkeld door een groep van **7** softwareontwikkelaars. Deze ontwikkelstraat maakt voor ondersteuning van het ontwikkelproces gebruik van Team Foundation Server, een product van Microsoft ter verbetering van de samenwerking bij software-ontwikkeling.

1.1.2 De opdracht

Zoals aangegeven in de vorige paragraaf maakt Empirion gebruik van een zelf opgezette “software ontwikkelstraat” voor het ontwikkelen van de applicaties. Hierbij streeft Empirion er naar om betrouwbare en stabiele applicaties snel op te leveren, zonder verstoringen bij nieuwe releases.

Een groot deel van het ontwikkel- en opleverproces is momenteel niet geformaliseerd. Dit heeft in de ogen van Empirion een aantal negatieve effecten, zoals onverwachte verstoringen tijdens de acceptatie van een nieuwe release. Om deze negatieve effecten te vermijden dient het proce geoptimaliseerd te worden.

Daarnaast heeft de ontwikkelstraat als doel om tijdig stabiele en betrouwbare applicaties op te leveren. In de ogen van Empirion is tevens een optimalisatieslag van de daadwerkelijke werkprocessen van de ontwikkelstraat nodig. Het volgende hoofdstuk gaat dieper in op de probleemidentificatie van de ontwikkelstraat.

1.1.3 De methodiek

Binnen dit verslag wordt de The Managerial Problem Solving Method, ook wel Algemene Bedrijfskundige Probleemaanpak (ABP) genoemd [Heerkens, 2005], gebruikt als methodiek om het bedrijfskundig probleem binnen Empirion aan te pakken. Er is gekozen voor de ABP omdat deze op een eenvoudige wijze dwingt om het probleem op een gestructureerde manier aan te pakken. De ABP is onder verdeeld in de volgende fasen:

1. Probleemidentificatie

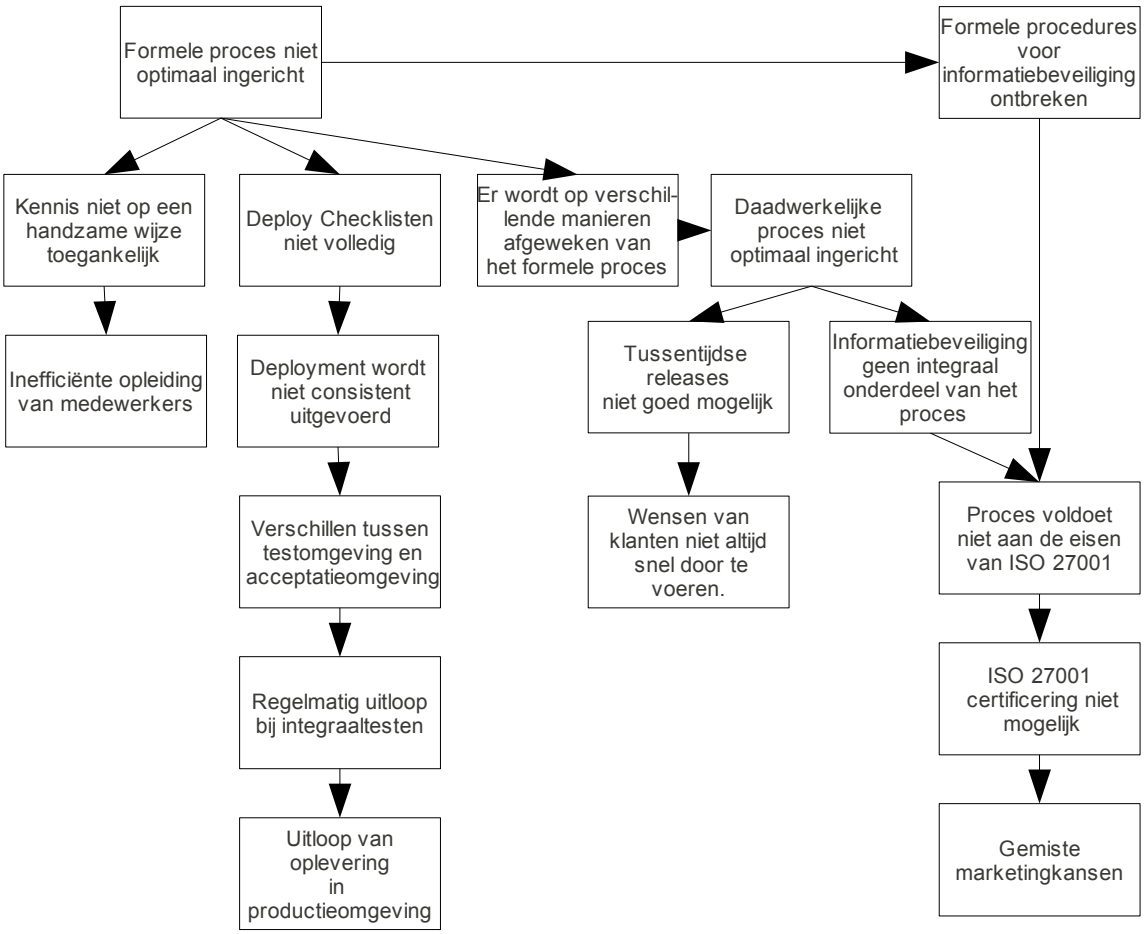
2. Het formuleren van de probleemaanpak
3. De probleemanalyse
4. De formulering van alternatieve oplossingen
5. Het kiezen van de oplossing
6. De implementatie van de oplossing
7. De evaluatie van de oplossing

Vanwege de beschikbare tijd doorloopt dit onderzoek alleen de eerste 4 fases van de ABP, waarbij als eindresultaat een geoptimaliseerd en geformaliseerd proces voor de ontwikkelstraat van Empirion aangedragen wordt. Een besluit over de invoering van de processen wordt binnen dit onderzoek niet genomen.

1.2 Probleemidentificatie

Op basis van de opdracht van Empirion B.V. en een gesprek met de manager R&D en de manager Infrastructuur van Empirion, beschrijft deze paragraaf de probleemidentificatie. Hierbij wordt gebruik gemaakt van probleemkluwen om het kernprobleem te identificeren.

1.2.1 Probleemkluwen



1.2.2 Kernprobleem

Op basis van de probleemkluwen is duidelijk dat hier twee kandidaat kernproblemen zijn:

- Het formele proces is niet optimaal ingericht.
- Het daadwerkelijke proces is niet optimaal ingericht.

Dit verslag pakt de niet optimale inrichting van het formele proces aan, waarbij het onderzoek ook de de tekortkomingen van het daadwerkelijke proces mee neemt in het vastleggen van het geoptimaliseerde formele proces.

Zoals aangegeven in paragraaf 1.2.3 wordt de daadwerkelijke implementatie van het geoptimaliseerde formele proces niet uitgevoerd.

1.3 Probleemaanpak

Deze paragraaf presenteert op basis van de in paragraaf 1.3 geschetste probleemidentificatie een plan van aanpak om de genoemde kernproblemen op te lossen.

1.3.1 Doel van de opdracht

Op basis van de opdrachtomschrijving en de kernproblemen is het volgende doel opgesteld:

- Het opstellen van een optimaal formeel ontwikkelproces voor de ontwikkelstraat van Empirion.

Hierbij zijn door Empirion de volgende randvoorwaarden gesteld aan het nieuwe formele proces:

- Het proces voldoet aan de eisen van ISO 27001.
- Het proces is bruikbaar in combinatie met Microsoft Team Foundation Server.

Hierbij is het uitgangspunt om de opdracht binnen 420 uur uit te voeren, waarbij 98 uur extra beschikbaar is voor het afronden van het onderzoeksvoorstel.

1.3.2 Plan van aanpak

Om het doel van de opdracht te bereiken worden achtereenvolgens de volgende stappen doorlopen, een gedetailleerde aanpak wordt in de genoemde paragraaf gegeven:

1. Het leggen van een theoretische basis, §1.4.4
2. Probleemanalyse, §1.4.5
    - Analyse huidige situatie
    - Modelleren huidige situatie
3. Genereren alternatieve oplossingen, §1.4.6
    - Opstellen alternatieven
    - Beoordeling alternatieven

1.3.3 Theoretische basis

Als eerste beschrijft dit verslag een theoretische basis, op basis waarvan de analyse van het probleem en de generatie van alternatieven plaatsvindt.

In de theoretische basis komen de volgende aspecten aan de orde:

- Softwareontwikkelmethodieken, op basis van een literatuurstudie
- Best-practices m.b.t. softwareontwikkeling, op basis van een literatuurstudie

Daarnaast beschrijft de verslag kort de ISO 27001 norm en de basis van Business Process Modeling Notation.

1.3.4 Probleemanalyse

De verkregen theoretische kennis vormt de basis voor de analyse van het huidige bedrijfsproces. Vervolgens wordt op basis van deze analyse de huidige situatie gemodelleerd. Hierbij is gekozen voor de modelleertaal Business Process Modeling Notation (BPMN) [White, 2004][OMG, 2011], omdat deze van de twee meest populaire grafische modelleertalen voor bedrijfsprocessen het meest gericht is op bedrijfsanalyse en het mogelijk is om diverse rollen te modelleren [Ryan et al., 2009]. Met het programma BizAgi [Bizagi, 2011] worden deze modellen opgesteld.

Tijdens de analysefase komen de volgende onderzoeksvragen aan bod om het huidige proces vast te leggen en om de randvoorwaarde van de ISO 27001 norm uit te werken:

- Hoe ziet het huidige bedrijfsproces van de ontwikkelafdeling van Empirion er uit?
- Welke eisen stelt de ISO 27001 norm aan het bedrijfsproces van de ontwikkelafdeling?

Op basis van deze onderzoeksdoelen en de vertaling naar de elementen van de modelleertaal BPMN zijn de volgende twee probleemstellingen en onderzoeksvragen opgesteld:

- Hoe kunnen de huidige bedrijfsprocessen binnen de ontwikkelafdeling van Empirion beschreven worden in verantwoordelijken, activiteiten, gebeurtenissen, beslissingen, verbindingen en gegevens.
    - Welke externe gebeurtenissen hebben invloed op de bedrijfsprocessen binnen de ontwikkelafdeling van Empirion?
    - Welke activiteiten vinden plaats binnen de ontwikkelafdeling van Empirion?
    - Welke gegevensstromen zijn er binnen de bedrijfsprocessen van de ontwikkelafdeling?
    - Welke beslismomenten zijn er binnen de bedrijfsprocessen binnen de ontwikkelafdeling van Empirion?
    - Op basis van welke gegevens worden beslissingen genomen bij deze beslismomenten?
    - Wie is/zijn er verantwoordelijk voor de verschillende activiteiten binnen de ontwikkelafdeling?
    - Hoe zijn de onderlinge verbintenissen tussen de diverse activiteiten, gebeurtenissen en beslismomenten?

    Hierbij worden de volgende uitgangspunten aangenomen: De startende gebeurtenis van de bedrijfsprocessen is een binnengekomen Request for Change (RFC) of ontdekte/gemelde bug. De laatste gebeurtenis van de bedrijfsprocessen is de oplevering van de aangepaste software in de productieomgeving.

Met gebruik van de beschikbare documentatie van de huidige bedrijfsprocessen en door middel van participerende observatie wordt een globaal beeld van de bedrijfsprocessen ontwikkeld, op basis van het globale beeld worden semi-gestructureerde interviews gehouden om de benodigde details van de bedrijfsprocedures te verkrijgen. Voor het verkrijgen van de bedrijfsdocumentatie is

medewerking noodzakelijk van het management van Empirion, daarnaast is voor het houden van de observatie en de interviews medewerking nodig van de medewerkers van de ontwikkelafdeling. Na beantwoording van deze onderzoeksvraag wordt een procesmodel opgesteld.

- Welke eisen stelt de ISO 27001 aan de activiteiten, gebeurtenissen, beslissingen en gegevens binnen de ontwikkelafdeling van Empirion?
    - Wat zijn de eisen die de ISO27001 stelt aan de activiteiten rond software ontwikkeling?
    - Hoe dient volgens de ISO27001 gereageerd te worden op bepaalde externe gebeurtenissen?
    - Dienen bepaalde beslismomenten ingebouwd te worden volgens de ISO 27001?
    - Op basis van welke gegevens worden in dat geval de beslissingen gemaakt?
    - Welke eisen zijn er m.b.t. de omgang met gegevens voor de ontwikkeling van software volgens de ISO 27001?

Reeds aanwezige documentatie binnen Empirion wordt gebruikt om deze onderzoeksvragen te beantwoorden.

Op basis van het opgestelde model en deze kennis wordt een overzicht gemaakt van de discrepanties tussen het huidige proces en de eisen die daar aan gesteld worden door de ISO 27001. Tevens wordt op basis van het procesmodel de oorzaken van de tekortkomingen in het daadwerkelijke proces bepaald.

Omdat het bestuderen van de ISO 27001 norm waarschijnlijk veel aandachtspunten naar voren brengt, welke in de interviews gebruikt kunnen worden om een duidelijk overzicht van het proces te verkrijgen, wordt het tweede kennisprobleem eerst opgelost, voordat de interviews m.b.t. het eerste kennisprobleem gehouden worden.

1.3.5 Genereren alternatieve oplossingen

Op basis van de opgedane kennis tijdens de literatuurstudie en de geïdentificeerde problemen tijdens de probleemanalyse worden alternatieve bedrijfsprocesmodellen opgesteld. Dit onderzoeksdoel leidt tot de volgende probleemstelling en onderzoeksvragen:

- Welke inrichting van de bedrijfsprocessen is mogelijk voor de ontwikkelomgeving van Empirion om de gevonden knelpunten op te lossen op basis van de bestudeerde literatuur, waarbij rekening gehouden wordt met de randvoorwaarden vanuit Empirion, waaronder de randvoorwaarde dat het proces dient te voldoen aan de ISO 27001 norm?
    - Hoe kan op basis van onder andere de literatuur het proces aangepast worden om de gevonden knelpunten op te lossen?
    - Voldoen de gevonden oplossingen aan de door Empirion gestelde randvoorwaarden?
    - Voldoen de gevonden oplossingen aan de eisen gesteld in de ISO 27001 norm?
    - Wat voor aanpassingen van het bedrijfsproces zijn nodig om de gevonden oplossingen op te nemen in het bedrijfsproces?

Op basis van de gevonden oplossingen worden alternatieve procesmodellen gemaakt d.m.v. BPMN, waarbij deze vergeleken worden met bij de probleemanalyse opgestelde model. De gevonden alternatieven worden vervolgens beoordeeld aan de hand van interviews met de betrokken medewerkers. Hierbij wordt aandacht besteedt aan onder andere onderstaande criteria:

- Kwaliteit van het product
    - De mate waarin de oplossing voldoet aan de ISO 27001 norm
    - De mate waarin het testen van de software volledig is
- Betrouwbaarheid van het product
    - De mate waarin de geplande deadlines gehaald worden
    - De mate waarin het proces tot een voorspelbaar resultaat leiden
- Werkbaarheid van het proces
    - Beheersbaarheid van het proces
    - Duidelijkheid van het proces
- Efficiënt verloop van het proces
    - De mate waarin tussentijdse releases mogelijk zijn
    - De mate waarin kennis op een handzame wijze voorhanden is
- Impact van de veranderingen

2 Theoretische basis

In dit hoofdstuk wordt in eerste instantie een theoretische basis gelegd op het gebied van software-ontwikkelmethodieken op basis waarvan de huidige bedrijfsprocessen worden geanalyseerd en gepositioneerd in hoofdstuk 3. Tevens geeft dit hoofdstuk in de literatuur gevonden best-practices weer, welke gebruikt worden in hoofdstuk 4 bij de generatie van alternatieve procesmodellen. Daarnaast legt dit hoofdstuk ook BPMN, SRML en de ISO 27001 norm kort uit.

2.1 Software-ontwikkelmethodieken

Volgens [Ramsin, Paige 2008] zijn object georiënteerde software-ontwikkelmethodieken in te delen in drie groepen:

- Rudimentaire Methodologieën (eerste en tweede generatie)
- Geïntegreerde Methodologieën (derde generatie)
- Agile Methodologieën

Hierbij zijn de rudimentaire methodologieën hybride methodologieën, deels structureel en deels objectgeoriënteerd. De analyse fase was meestal op basis van gestructureerde analyse technieken, terwijl de ontwikkel fase de resultaten van de gestructureerde analyse omzette naar een objectgeoriënteerd resultaat. De geïntegreerde methodologieën worden gekarakteriseerd door de op processen gerichte houding ten opzichte van software ontwikkeling, deze hebben geleidt tot grote methodologieën welke moeilijk te managen zijn. Agile methodologieën zijn voornamelijk gebaseerd op iteratieve-incrementele processen. Deze methodologieën hebben als doel om de processen zo licht mogelijk te houden.

De geïntegreerde methodologieën zijn weer onder te verdelen in de volgende methodologieën:

- Object-process methodology (OPM)
- Catalysis
- Open
- Rational Unified Process (RUP)
- Enterprise Unified Process (EUP)
- Functional and object-oriented methodology (FOOM)

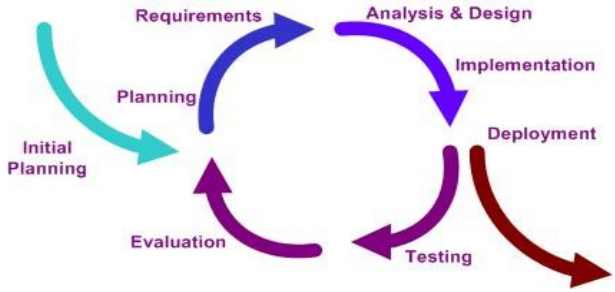
Binnen deze opdracht is gekozen om van deze methodologieën RUP te bestuderen en bij de probleemanalyse te gebruiken om discrepanties met het huidige proces te bepalen. Deze keuze is gemaakt omdat RUP een compleet, maar aanpasbaar, procesraamwerk biedt, zodat het proces beheersbaar kan blijven. Daarnaast biedt RUP het voordeel dat het gebruik maakt van de modelleertaal UML, de de facto standaard modelleertaal voor objectgeoriënteerd modelleren [Ramsin, Paige 2008].

Volgens bedrijfsdocumentatie volgen ontwikkeltrajecten van Empirion het volgende stappenplan:

1. Feasability studie
2. Requirements analyse
3. External design
4. Internal design
5. Realisatie
6. Functionele test

7. In productie name

Dit stappenplan is iteratief. Er kunnen tijdens latere stappen bijvoorbeeld gemakkelijk nieuwe requirements worden gevonden, die dan verwerkt moeten worden. Tevens is het mogelijk een project in meerdere subonderdelen te verwerken, waarbij elk subonderdeel dit stappenplan (ook nog iteratief) doorloopt [Empirion 2010]. Dit stappenplan komt in de basis ook overeen met het iteratieve en incrementele proces wat aan RUP ten grondslag ligt [Kruchten, P 1999], in figuur 1 is een visuele weergave gegeven van het iteratieve en incrementele proces. Door deze overeenkomst is een goede vergelijking tussen RUP en het huidige proces mogelijk.



Figuur 1: An iterative and incremental process

Daarnaast wordt binnen dit onderzoek het huidige proces ook vergeleken met een van de agile methodologieën, omdat dit eventueel ook oplossingen biedt voor o.a. de mogelijkheid om tussentijdse releases beter mogelijk te maken. Hierbij is een keuze gemaakt voor Scrum. De keuze is op Scrum gevallen vanwege de mogelijkheden om Scrum te integreren in het procesraamwerk van RUP [Krebs, J. 2005], zodat beide bestudeerde methodologieën gebruikt kunnen worden bij het opstellen van een geoptimaliseerd procesmodel.

2.1.1 Rational Unified Process

Het Rational Unified Process is een aanpasbaar procesraamwerk, waarbij een gedisciplineerde aanpak van taakverdeling en verantwoordelijkheden binnen een ontwikkelorganisatie wordt gegeven. Het proces ondersteund 6 best-practices [Kruchten, P 1999], namelijk:

1. Ontwikkel software iteratief
2. Beheer requirements
3. Gebruik op componenten gebaseerde architecturen
4. Modelleer software visueel (UML)
5. Controleer de kwaliteit van de software
6. Controleer veranderingen aan de software

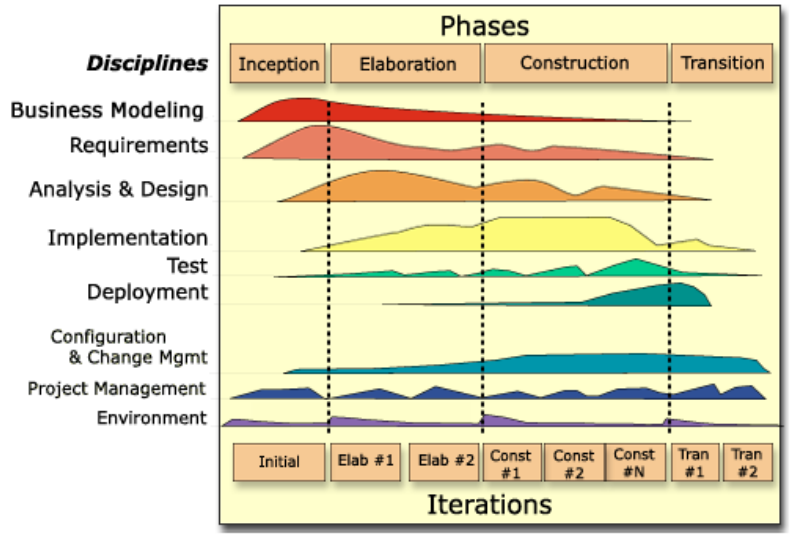
RUP bestaat uit vier fases [Ramsin, Paige 2008], te weten:

1. Inception: focus op het definiëren van de doelstellingen van het project, in het bijzonder de business case.
2. Elaboration: focus op het vastleggen van de cruciale requirements, ontwikkelen en valideren van de architectuur van het software systeem en plannen van de overige fases van het project.
3. Construction: focus op het implementeren van het systeem in een iteratieve en incrementele wijze, gebaseerd op de ontwikkelde architectuur in de voorgaande fase.
4. Transition: focus op het betatesten van het systeem en het klaarmaken voor het releasen van het systeem.

Deze fases kunnen verder opgebroken worden in iteraties, een iteratie is een complete ontwikkel-loop, waarbij het eindresultaat een uitvoerbare toevoeging aan het systeem is. De iteraties bestaan uit processen uit negen verschillende disciplines, te weten [Ramsin, Paige 2008]:

1. Business modeling: Houdt zich bezig met het beschrijven van bedrijfsprocessen en met de interne structuur van een bedrijf, met als doel om het bedrijf te begrijpen en de requirements van het software systeem te bepalen. Een business use case en een business object model worden ontwikkeld als resultaat van deze discipline.
2. Requirements management: Houdt zich bezig met het uitzoeken, organiseren en documenteren van requirements; Het use case model is een resultaat van deze discipline.
3. Analysis and design: Houdt zich bezig met het creëren van de architectuur en het ontwerp van het software systeem. Het resultaat van deze discipline is een design model en mogelijk een analysis model. Het design model bestaat uit design classes gestructureerd in design packages en design subsystems met goed gedefinieerde interfaces, het representeert toekomstige componenten tijdens de implementatie. Het model beschrijft tevens hoe de objecten samenwerken om de use cases te realiseren.
4. Implementation: Houdt zich bezig met het schrijven en debuggen van broncode, het testen van units en build management. Broncode bestanden, executables en ondersteunende bestanden worden geproduceerd.
5. Test: Houdt zich bezig met integratie-, systeem- en acceptatie-testen.
6. Deployment: Houdt zich bezig met het maken van packages, het creëren van installatie scripts, schrijven van documentatie voor eindgebruikers en overige taken om de software beschikbaar te stellen voor de eindgebruikers.
7. Project management: Houdt zich bezig met de projectplanning, scheduling en controle.
8. Configuration and change management: Houdt zich bezig met versie- en release-management en met changerequest management.
9. Environment: Houdt zich bezig met het aanpassen van de processen aan de verschillende benodigdheden van een project of en organisatie, en tevens met selecteren, introduceren en ondersteunen van ontwikkeltools.

De diverse disciplines komen niet in elke fase en iteratie even veel aan bod, figuur 2 geeft hiervan een overzicht.



Figuur 2: Rational Unified Process

Op basis van de gedetailleerde workflows en artefacten in [Kruchten, P 1999] geeft het derde hoofdstuk een vergelijking met het huidige proces van Empirion.

2.1.2 Scrum

Scrum is gebaseerd op een “empirical process control model” in tegen stelling tot de meeste traditionele software methodieken, welke gebaseerd zijn op het “defined process control model”. Het “defined process control model” is geschikt in situaties waarbij processen duidelijk zijn te formuleren, waarbij bij herhaaldelijk uitvoeren van het proces de uitkomsten identiek zijn. In de ogen van Ken Schwaber en Mike Beedle is het “defined process control model” echter niet geschikt voor het ontwikkelen van software, omdat bijna geen systeemontwikkel project zo eenvoudig is met zo weinig ruis dat bij herhaling hetzelfde resultaat behaald wordt [Schaber, K, Beedle, M 2002]. “Empirical process control models” gebruiken frequente inspecties en adaptieve respons om het proces te controleren, waardoor het model beter geschikt is voor omgevingen met veel ruis en verschillende uitkomsten bij herhaling van het proces.

Tevens sluit software-ontwikkeling beter aan bij processen voor het ontwikkelen van nieuwe producten, dan bij productiemodellen. Dit omdat softwareontwikkeling niets anders is dan het creëren van nieuwe producten en creativiteit hierbij daarom ook een belangrijke rol speelt. Scrum is daarom gebaseerd op de 6 karakteristieken van succesvolle bedrijven uit het onderzoek van Takeuchi en Nonaka bij de top-10 meest competitieve en innovatieve bedrijven [Takeuchi H., Nonaka, I. 1986]:

- Built-in instability
- Self-organizing project teams
- Overlapping development phases
- Multilearning
- Subtle controle
- Organizational transfer of learning

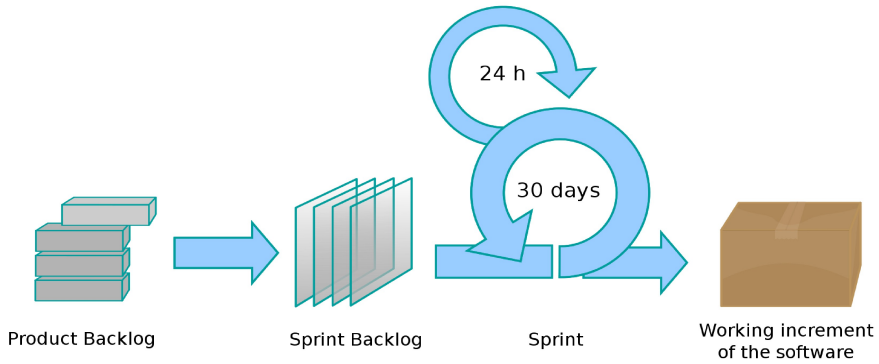
Bij ontwikkeling van software volgens de Scrum methodiek worden Scrum teams samengesteld van ongeveer 7 personen met verschillende disciplines, elk team wordt geleidt door een Scrum Master. De Scrum Master is verantwoordelijk voor het succes van het Scrum team. Het ontwikkelproces wordt hierbij opgebroken in sprints van 30 dagen, na elke sprint wordt een werkend systeem opgeleverd.

Aan het begin van een sprint worden de doelstellingen van de sprint vastgesteld in de sprint planning. Hierbij worden door de Scrum Master, Product Owner en het Scrum team op basis van het product backlog een sprint backlog opgesteld. Het sprint backlog bestaat uit de werkzaamheden die in de sprint gerealiseerd moeten worden. Het product backlog wordt enkel bijgehouden door de Product Owner, een persoon die de klant representeert. Het product backlog is een lijst, waarop alle te ontwikkelen functionaliteiten staan op volgorde van prioriteit. Deze lijst kan continu aangevuld worden op basis van nieuwe bevindingen, waarbij de volgorde en prioriteit van de lijst kan wijzigen. Als een sprint backlog eenmaal vastgesteld is kunnen geen extra zaken toegevoegd worden aan het sprint backlog, zodat het team in alle rust kan werken aan de vastgestelde activiteiten. Na afronding van de sprint vindt een sprint review plaats, waarbij de resultaten van de afgelopen sprint worden gepresenteerd aan het management, klanten, gebruikers en de Product Owner.

Gedurende de sprint wordt dagelijks, op een vaste tijd, een Scrum meeting gehouden. Een scrum meeting duurt ongeveer 15 minuten en behandelt enkel de volgende punten:

- Werkzaamheden sinds de vorige meeting
- Werkzaamheden tot de volgende meeting
- Obstakels bij het uitvoeren van de werkzaamheden

Deze meetings hebben als doel om de communicatie te verbeteren, overige meetings overbodig te maken, ontdekken en oplossen van obstakels, het snel maken van beslissingen te bevorderen en het kennisniveau m.b.t. het project bij iedereen te verhogen [Schaber, K, Beedle, M 2002]. Bij deze meetings hebben enkel de team leden spreekrecht, andere aanwezigen mogen op geen enkele wijze interfereren bij de meetings. De Scrum Master is verantwoordelijk voor het verloop van deze meetings, tevens dient hij er voor te zorgen dat de gevonden obstakels worden opgelost.



Figuur 3: Scrum

In figuur 3 wordt een visuele weergave gegeven van het proces. Om Scrum effectief in te kunnen voeren is het noodzakelijk dat minstens 50% van de medewerkers het expert-niveau heeft. Daarbij heeft Scrum als voordeel dat Scrum snel leren promoot en kennis snel en effectief overgedragen wordt op de teamleden. [Schaber, K, Beedle, M 2002]. Op basis van [Schaber, K, Beedle, M 2002] is een BPMN model opgesteld van Scrum, dit model is bij dit verslag gevoegd als bijlage A.

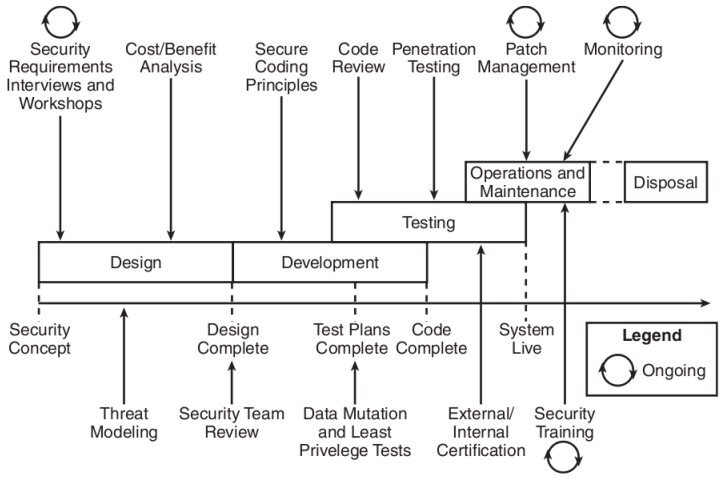
2.2 Best practices – Secure Coding

De software van Empirion wordt gebruikt als “Software-as-a-Service (SaaS) en de applicatie is voor iedereen met een internettoegang toegankelijk. Hierdoor is de beveiliging van de applicatie essentieel, zeker in combinatie met de opslag van o.a. medische gegevens in het systeem.

In [Flutcher, L. , Solms, R. von 2008] worden de volgende richtlijnen, verdeeld over drie deelgebieden, genoemd:

- Manage het software ontwikkel proces
    - Integreer beveiliging in de “Software Development Live Cycle” (SDLC)
    - Definieer beveiligingsrollen
    - Stel educatie en training m.b.t. beveiliging beschikbaar
    - Voer risicomanagement uit
- Elementen in het software ontwikkel proces
    - Stel beveiligingseisen vast
    - Modeleer bedreigingen
    - Pas code- en teststandaarden toe
        - Gebruik testmiddelen voor beveiliging en code
        - Voer code reviews uit.
- Beveiligingsfuncties om in te bouwen in de applicaties
    - Bepaal relevante beveiligings-services
    - Implementeer geschikte beveiligingscontroles en mechanismes

Ook in [Jones, R.L., Rastogi, A. 2004] wordt het belang van het integreren van beveiliging in de SDLC aangegeven, deze integratie is in figuur 4 weergegeven.



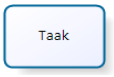
Figuur 4: Secure SDLC

Bovengenoemde best-practices worden gebruikt bij het modelleren van nieuwe modellen in hoofdstuk 4.

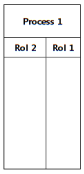
2.3 BPMN

Zoals in paragraaf 1.4.5 beschreven is, is BPMN een modelleertaal gericht op het modelleren van bedrijfsprocessen. De activiteiten van een bedrijfsproces worden hierbij weergegeven als een netwerk van grafische objecten. Hierbij worden de volgende objecten gebruikt [White, 2004]:

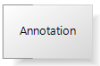
- Flow Objects:
  - 

Event: Representeert een gebeurtenis en beïnvloedt de loop van het proces. Gewoonlijk hebben ze een oorzaak (trigger) of impact (result). Er zijn drie types: start, intermediate en end. Deze worden weergegeven met een circel.
  - 

Activity: Een activiteit wordt afgebeeld door een afgeronde rechthoek en representeert werkzaamheden die door het bedrijf worden uitgevoerd. Er zijn twee types: Taak en subproces. Een subproces wordt onderscheiden door een + teken in de rechthoek.
  - 

Gateway: Een gateway wordt weergegeven door een ruit en wordt gebruikt om een splitsing of samenkomst van diverse stromen weer te geven. Het o.a. beslismomenten, splitsingen en samenvoegingen aan.
- Connecting Objects:
  - 
Sequence Flow: Een vaste lijn met een pijl, geeft de volgorde van de activiteiten aan.
  -
Associaton: Een gestippelde lijn, associeert data of aantekeningen met flow objects.
- Swimlanes
  - 

Lane: Worden gebruikt om activiteiten geassocieerd met specifieke functies of rollen te onderscheiden.
- Artifacts
  - 

Data Object: Geeft aan welke data benodigd is voor de activiteit of welke data een activiteit oplevert.
  - 

Annotation: Geeft de mogelijkheid om extra aantekeningen bij het diagram toe te voegen.

2.4 ISO 27001

Het nieuwe bedrijfsproces moet voldoen aan de ISO 27001 norm, hierbij gaat het volledigheidshalve om de ISO/IEC 27001:2005 norm.

Volgens de website van de International Organization for Standardization [ISO 2010] specificeert de ISO/IEC 27001:2005 norm de eisen aan het vaststellen, implementeren, gebruiken, monitoren, evalueren, onderhouden en verbeteren van een gedocumenteerd Information Security Management System (ISMS), binnen de context van de algemene bedrijfsrisico's. Het ISMS stelt eisen aan de implementatie van “security controls” aangepast aan de behoeften van individuele organisaties of delen daarvan.

ISO/IEC 27001:2005 is ontwikkeld om te waarborgen dat adequate en geproportioneerde beveiligingsmaatregel genomen worden om informatiemiddelen te beschermen en vertrouwen te geven aan betrokken partijen. Bij het beschermen van de informatiemiddelen gaat het om de vertrouwelijkheid, beschikbaarheid en integriteit van de informatiemiddelen.

ISO/IEC 27001:2005 kan gebruikt worden voor verschillende doelen, waaronder:

- Gebruik binnen organisaties om beveiligingseisen en doelen te formuleren.
- Gebruik binnen organisaties om naleving van wet en regelgeving te waarborgen.
- Gebruik binnen een organisatie als een procesraamwerk voor het implementeren en managen van controles om te waarborgen dat de specifieke beveiligingsdoelen behaald worden.
- Gebruik door organisaties om relevante informatie over informatiebeveiliging te verstrekken aan klanten.

Voor Empirion is certificering conform de ISO 27001 norm van belang, omdat de software van het bedrijf medische gegevens opslaat. En door middel van deze certificering kan Empirion aantonen dat het bedrijf wet en regelgeving m.b.t. informatiebeveiliging van deze gegevens na leeft. Empirion heeft als doelstelling om de certificering in 2011 af te ronden.

Bij deze opdracht wordt voor het vaststellen van de eisen uit de ISO 27001 norm gebruik gemaakt van een door Empirion beschikbaar gesteld excelbestand.

2.5 Conclusie

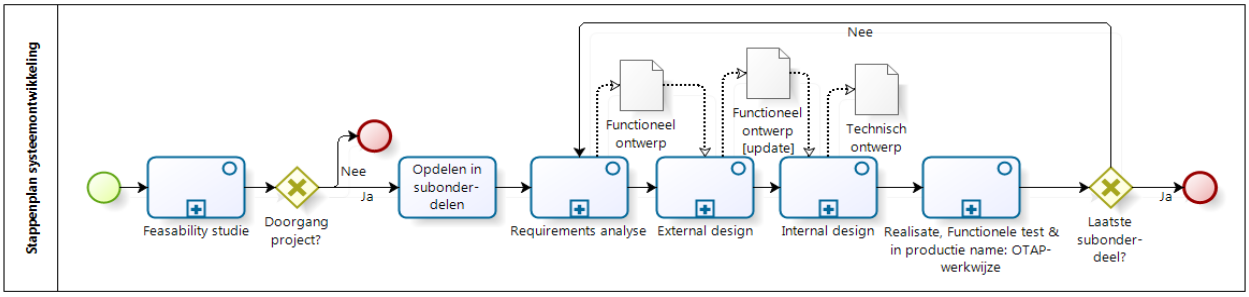
In dit hoofdstuk zijn twee softwaremethodieken beschreven, RUP en Scrum. Deze methodieken worden in hoofdstuk drie gebruikt om een positionering te maken ten opzichte van de ontwikkelwijze bij Empirion. Daarnaast wordt de hierbij opgedane kennis gebruikt voor het opstellen van de alternatieven in het vierde hoofdstuk. Ook de kennis van secure coding wordt bij het opstellen van de alternatieven gebruikt om waarborging van informatiebeveiliging te integreren in het gehele ontwikkelproces. De paragraaf over BPMN is toegevoegd om zonder extra achtergrondkennis de bijgevoegde modellen te kunnen lezen, ook is kort het belang en de basis van de ISO 27001 certificering aangegeven.

3 Probleemanalyse

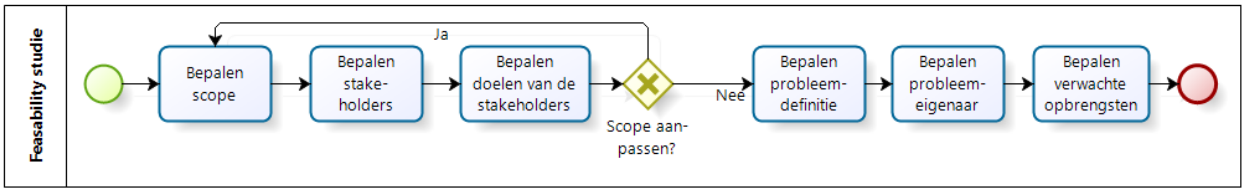
Het derde hoofdstuk beschrijft de huidige bedrijfsprocessen Hierbij wordt enerzijds gebruik gemaakt van de aanwezige procesbeschrijvingen, anderzijds worden interviews gebruikt om verschillen tussen de gedocumenteerde processen en de praktijk boven water te krijgen. Op basis van het huidige proces wordt een positionering gemaakt ten opzichte van de ontwikkelmethodieken uit hoofdstuk twee. De derde paragraaf zet tevens uiteen wat nodig is om te voldoen aan de ISO 27001 norm. De laatste paragraaf van dit hoofdstuk geeft discrepanties tussen het huidige en het gewenste proces aan.

3.1 Formeel bedrijfsproces

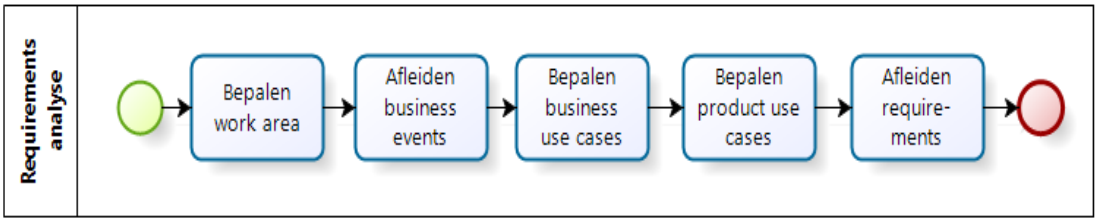
Het formele bedrijfsproces van Empirion is beschreven in een drietal documenten op de wiki van Empirion. Het eerste document is ook in hoofdstuk twee benoemd en bevat het stappenplan van de systeemontwikkeling, wat op hoofdlijnen het ontwikkelproces beschrijft. Het tweede document beschrijft de procedures voor releasemanagement van de ontwikkelde software [Empirion 2011a]. Dit document gaat dieper in op de procedure van ontwikkelen, testen, accepteren en in productie nemen van de software en wordt daarom de OTAP-werkwijze genoemd. Deze twee formele procesbeschrijvingen zijn omgezet in een BPMN-model en hieronder in figuur zes tot en met vijftien weergegeven.



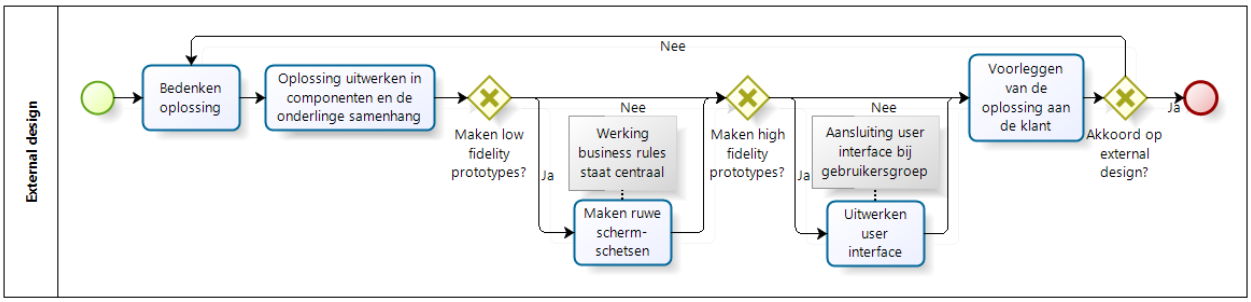
Figuur 6: Overzicht formeel bedrijfsproces



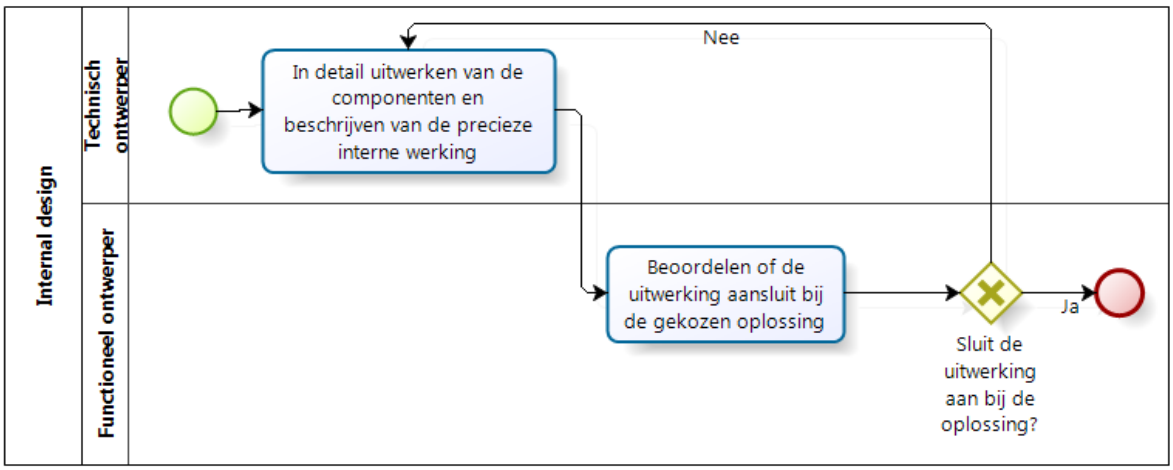
Figuur 7: Feasability studie



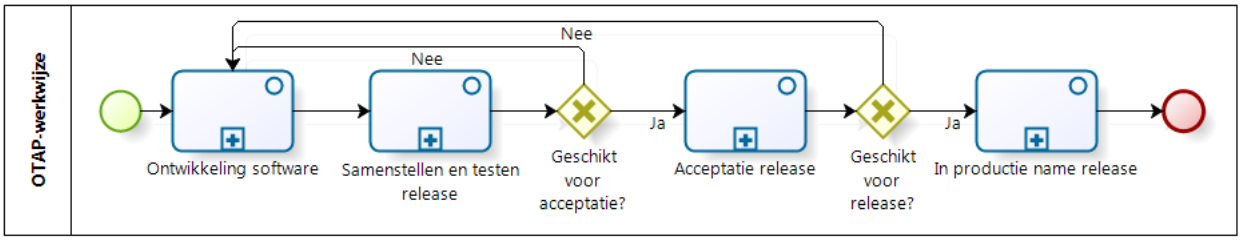
Figuur 8: Requirements analyse



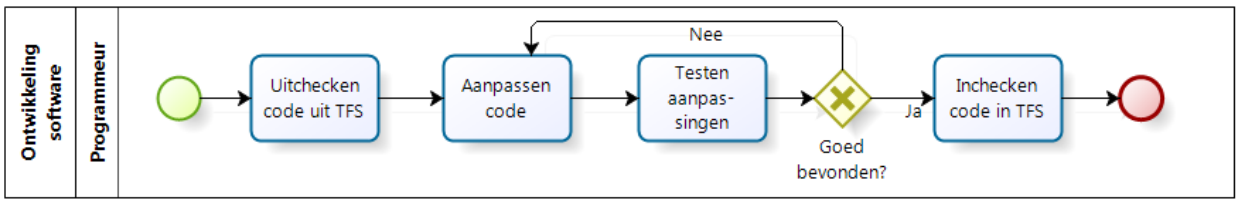
Figuur 9: External design



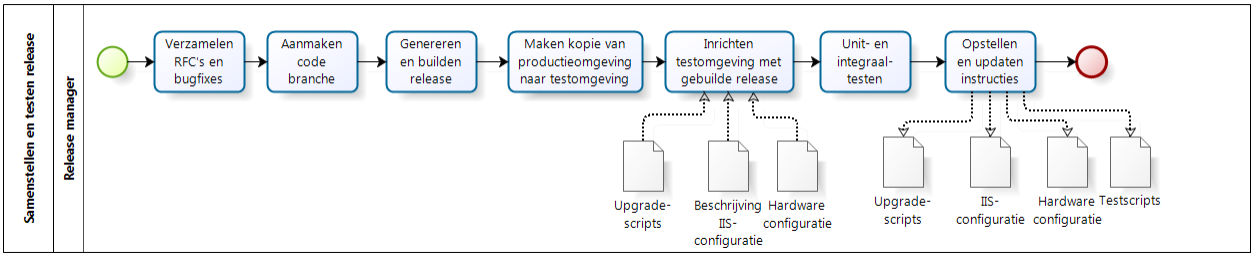
Figuur 10: Internal design



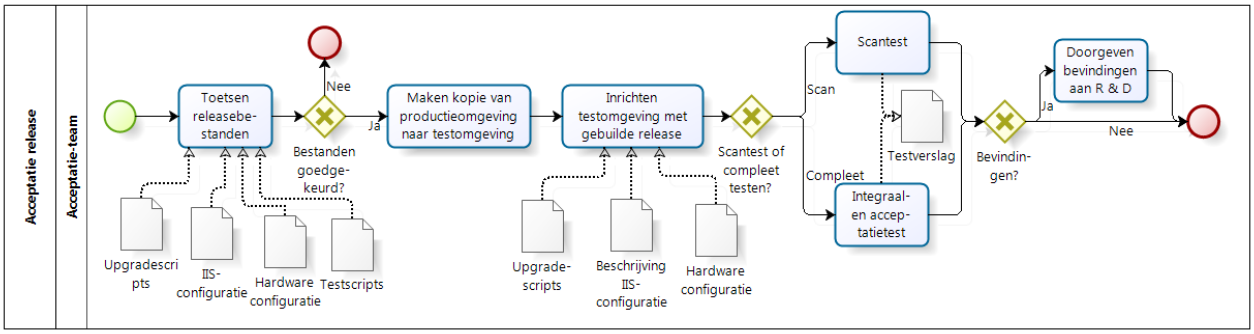
Figuur 11: OTAP-werkwijze



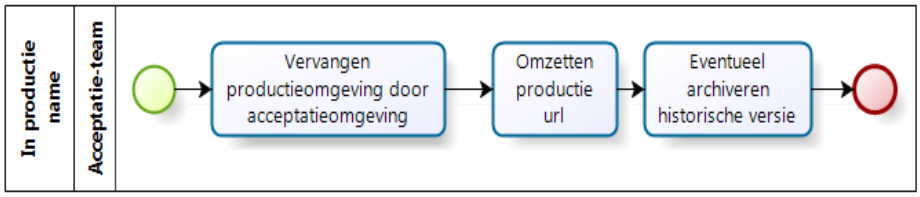
Figuur 12: Ontwikkeling software



Figuur 13: Samenstellen en testen release



Figuur 14: Acceptatie release



Figuur 15: In productie name

Naast deze twee documenten staat op de wiki nog een derde document, wat de deploymentprocedure beschrijft van de applicatie [Empirion 2011b], echter wordt in het betreffende document nog uitgegaan van programmatuur welke momenteel niet meer in gebruik is, deze procedure is dan ook achterhaald. Tevens staan op de wiki een groot aantal richtlijnen voor onder andere het grafische ontwerp van de applicatie en eisen met betrekking tot documentatie bij de systeemontwikkeling, niet alle richtlijnen zijn echter actueel.

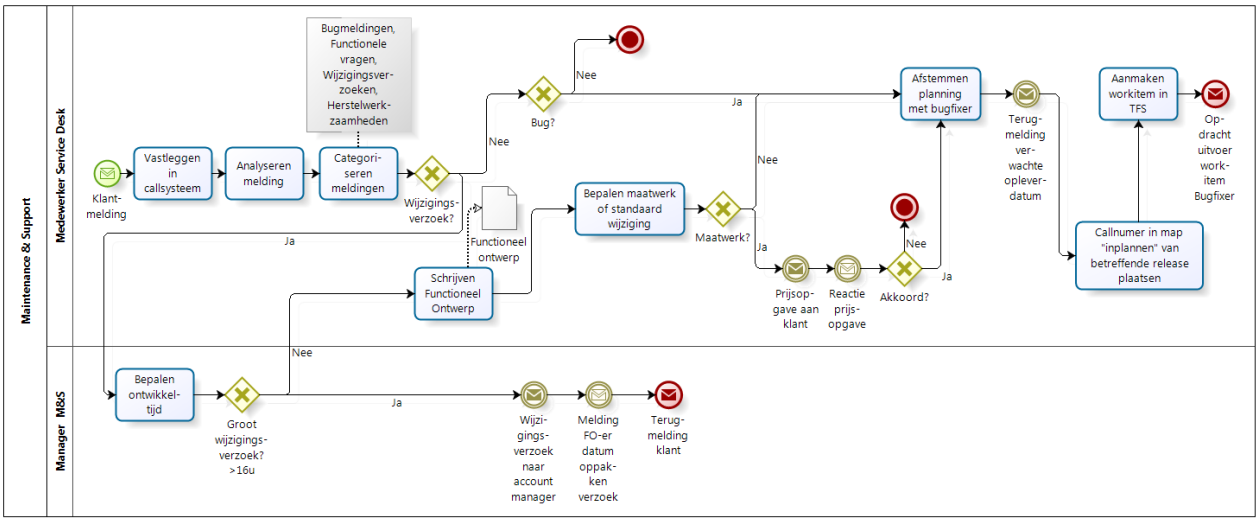
3.2 Daadwerkelijk bedrijfsproces

Om het daadwerkelijke bedrijfsproces in kaart te brengen zijn semi-gestructureerde interviews gehouden met de volgende personen:

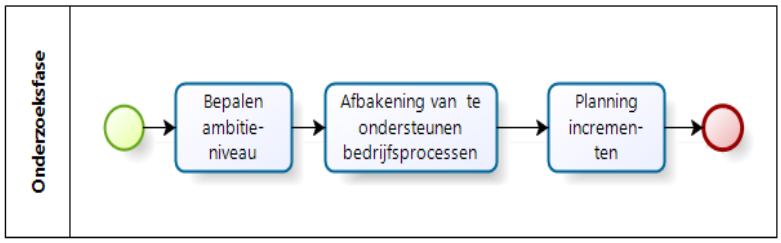
- Manager Maintenance & Support
- Management Research & Development
- Integratie tester
- Programmeur
- Functioneel ontwerper (2x)
- Software architect

Tijdens de interviews is naar voren gekomen dat in het traject voor de daadwerkelijke realisatie drie

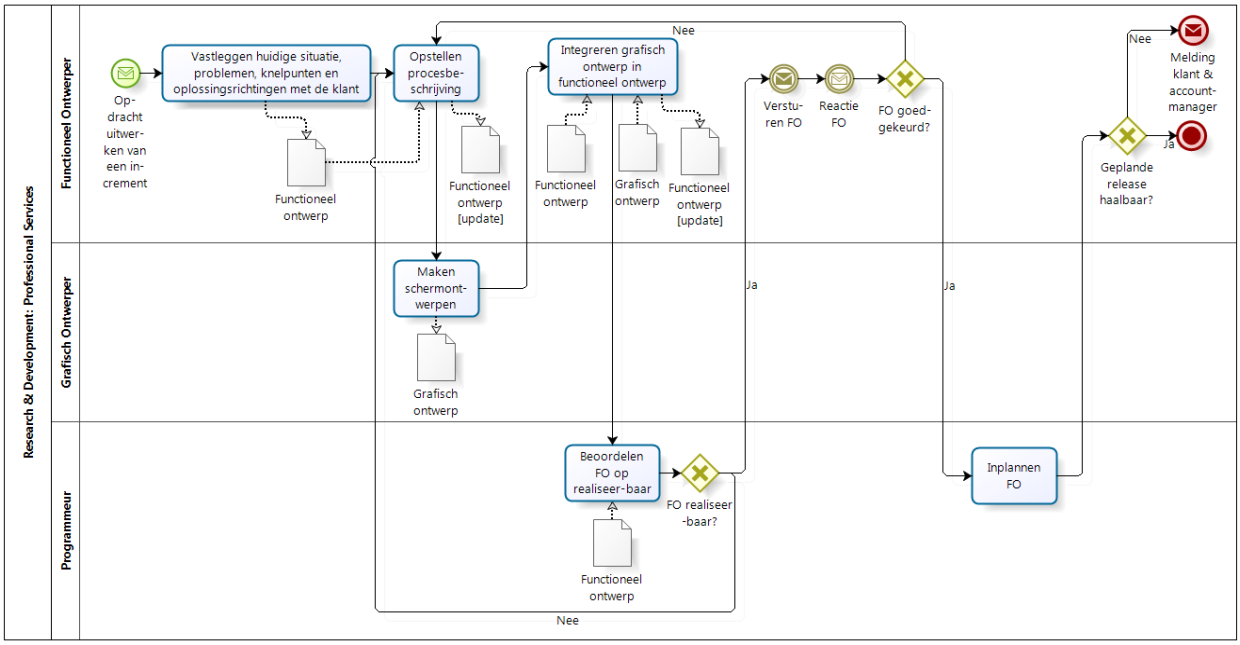
verschillende stromen aan te wijzen zijn. Het betreft de strategische systeemontwikkeling, grote wijzigingsverzoeken van klanten en de kleine wijzigingsverzoeken & bugs. Hieronder zijn de bedrijfsprocessen van zowel de grote wijzigingsverzoeken als de kleine wijzigingsverzoeken en bugs uitgewerkt. De strategische systeemontwikkeling is buiten beschouwing gelaten, omdat deze qua werkwijze lijkt op de grote wijzigingsverzoeken, waarbij het een interne klant betreft. Het bedrijfsproces van de kleine wijzigingsverzoeken en bugs is opgenomen in figuur zestien, de grote wijzigingsverzoeken volgen het proces van figuur zeventien en achttien.



Figuur 16: Kleine wijzigingsverzoeken & bugs

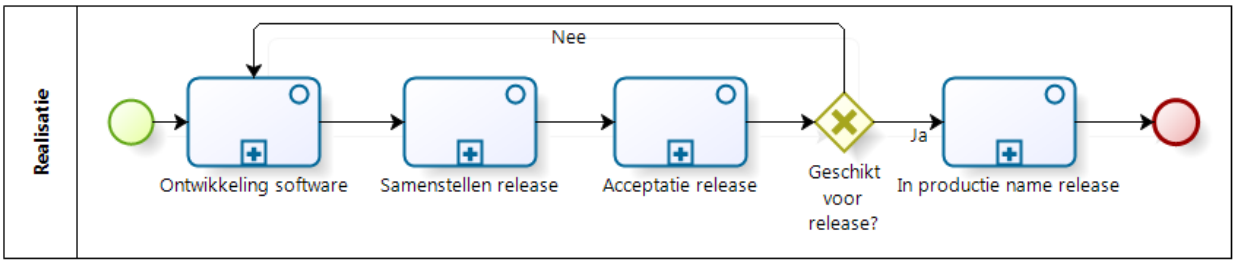


Figuur 17: Onderzoeksfase

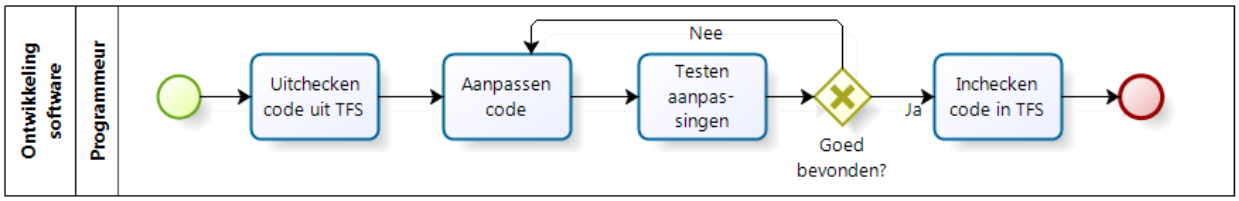


Figuur 18: Uitwerken functioneel ontwerp

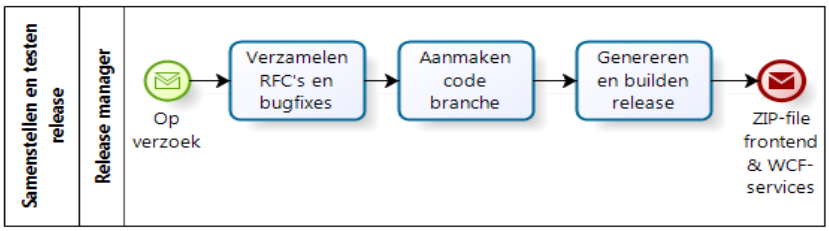
Na het uitwerken van het FO wordt overgegaan tot de realisatie van het FO. Het overzicht van de realisatie is weergegeven in figuur negentien, de details zijn weergegeven in figuur twintig tot en met 24.



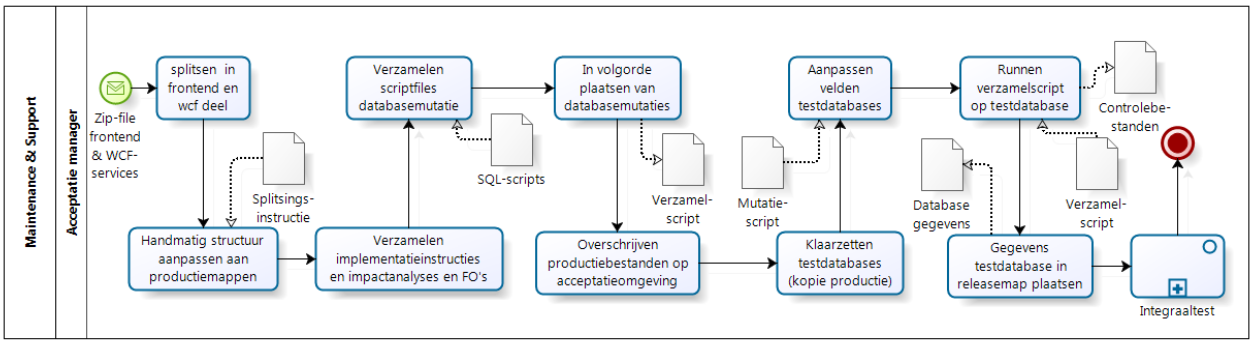
Figuur 19: Realisatie



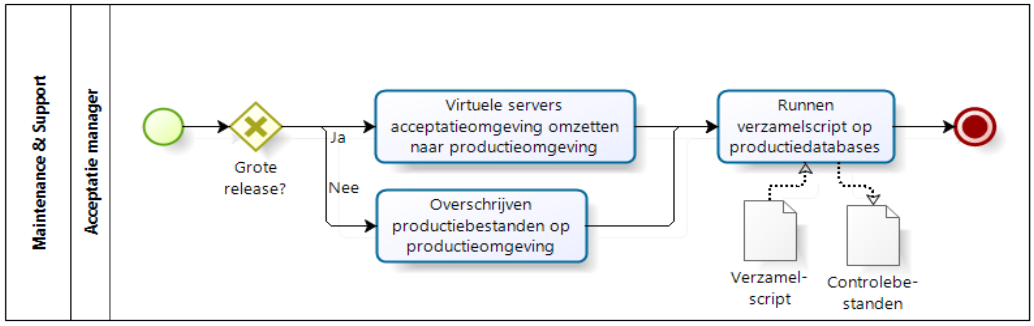
Figuur 20: Ontwikkeling



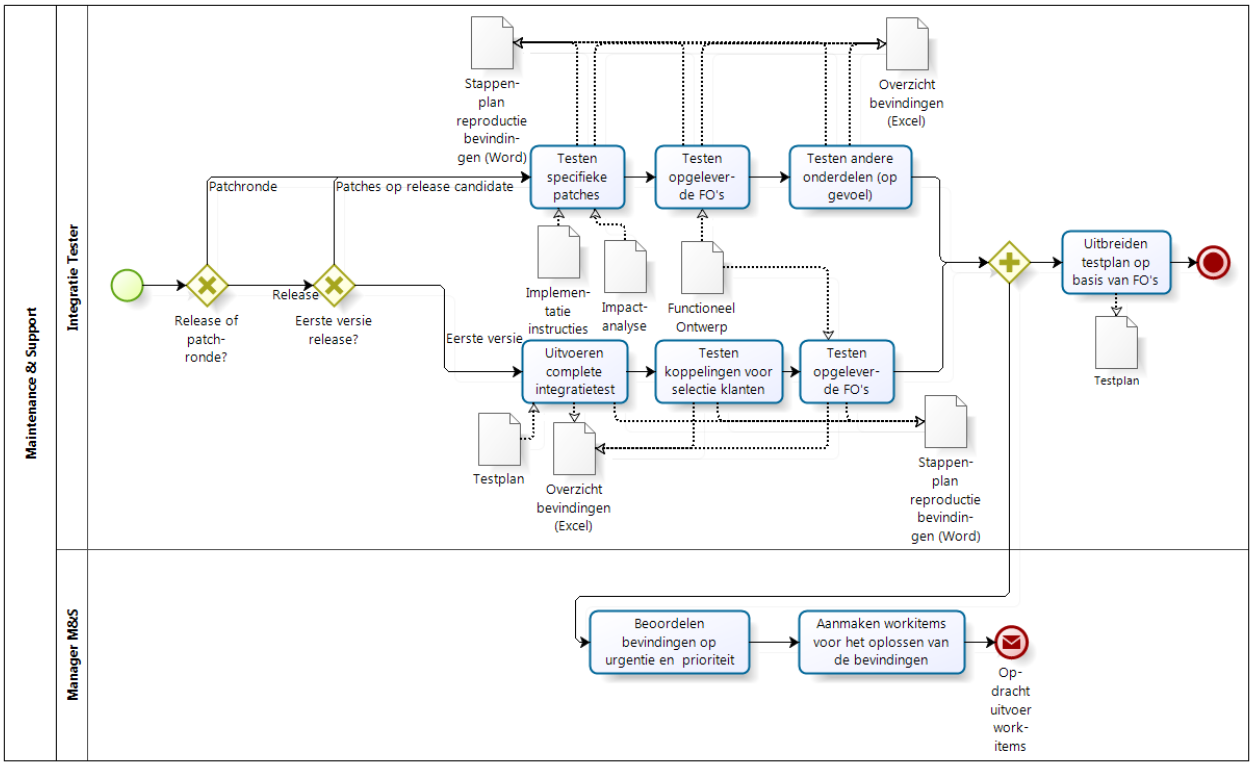
Figuur 21: Samenstellen release



Figuur 22: Acceptatie release



Figuur 23: In productie name release



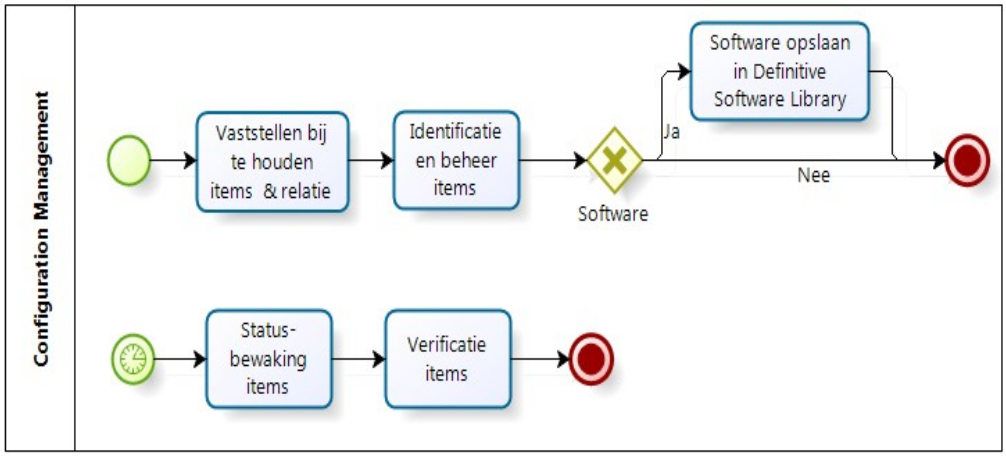
Figuur 24: Integraaltest

3.3 Gewenste proces in verband met ISO 27001

Zoals in paragraaf 2.5 beschreven, stelt de ISO 27001 norm eisen aan de implementatie van "security controls", deze "control objectives" beperken zich echter niet tot de ontwikkelomgeving van Empirion, maar raken de gehele organisatie. In deze paragraaf is beschreven welke "control objectives" meegenomen worden binnen deze opdracht. Deze keuze is gemaakt op basis van de in paragraaf 2.5 beschreven excelfile en gesprekken met de manager R&D en een externe consultant. Hierbij is er voor gekozen om de "control objectives" aan de bedrijfsprocessen uit de IT Infrastructure Library (ITIL) te koppelen [Bon et al. 2006], deze processen zijn de basis voor de alternatieve modellen in het vierde hoofdstuk.

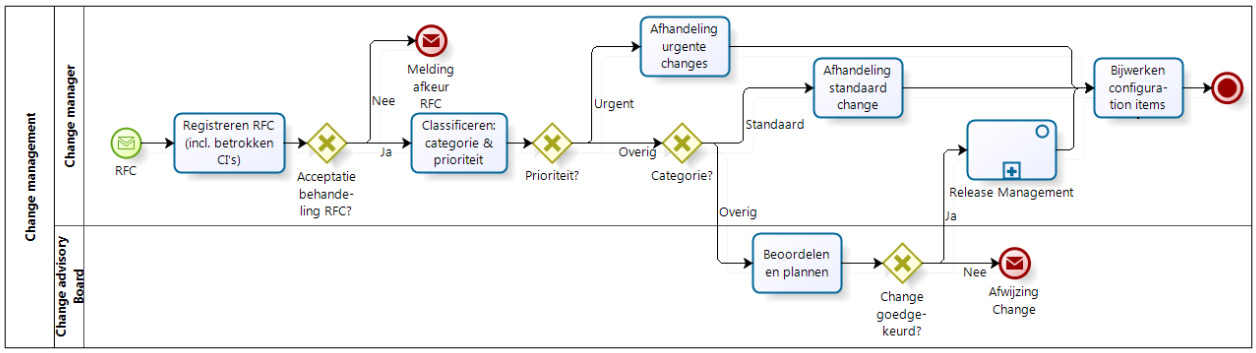
Voor het gewenste proces is het in de eerste plaats van belang om de processen gedocumenteerd te hebben, door een goede documentatie wordt voldaan aan het control objective "Documented operating procedures" (sectie 10.1.1), de opgestelde BPMN-modellen van dit verslag kunnen daarvoor gebruikt worden.

De control objectives "Inventory of assets" (sectie 7.1.1), "Ownership of assets" (sectie 7.1.2) en "Information classification" (sectie 7.2) vereisen een goed overzicht van de aanwezige assets en de status van deze assets. Binnen ITIL wordt hiervoor het proces configuration management gebruikt, waarmee ook een koppeling gelegd wordt tussen incidenten, problemen en wijzigingen en de betrokken items. Wijziging van details van het configuration item (CI) vindt ook plaats bij de processen change management, incident management en problem management. Het proces-model van configuration management is toegevoegd als figuur 25.



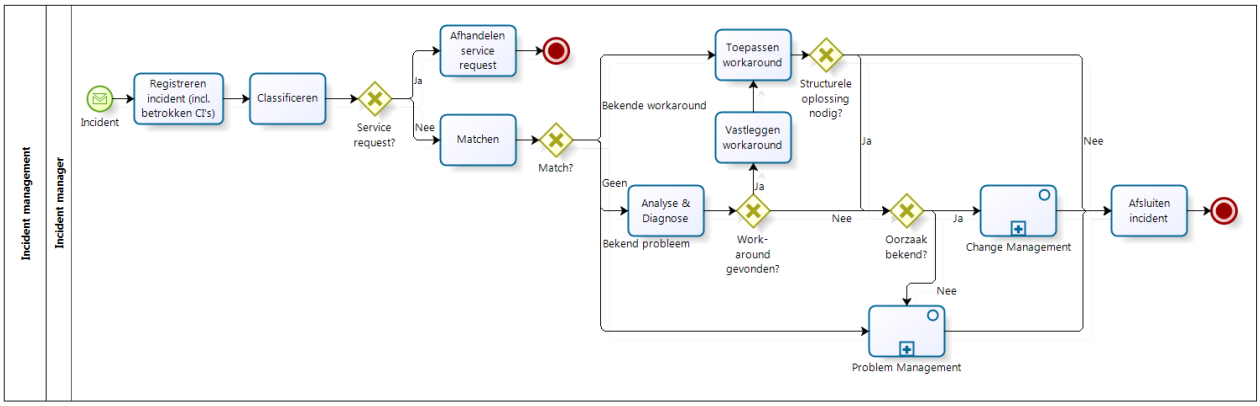
Figuur 25: ITIL-configuration management

Voor de control objectives "Change management" (sectie 10.1.2), "Control of operational software" (sectie 12.4.1) en "Change control procedures" (sectie 12.5.1) is een formele change management procedure noodzakelijk. In figuur 26 is het gewenste proces van change management opgenomen op basis van ITIL. Het proces heeft een connectie met releasemanagement, wat verder op in dit hoofdstuk besproken wordt, daarnaast maakt het gebruik van gegevens van configuration management.

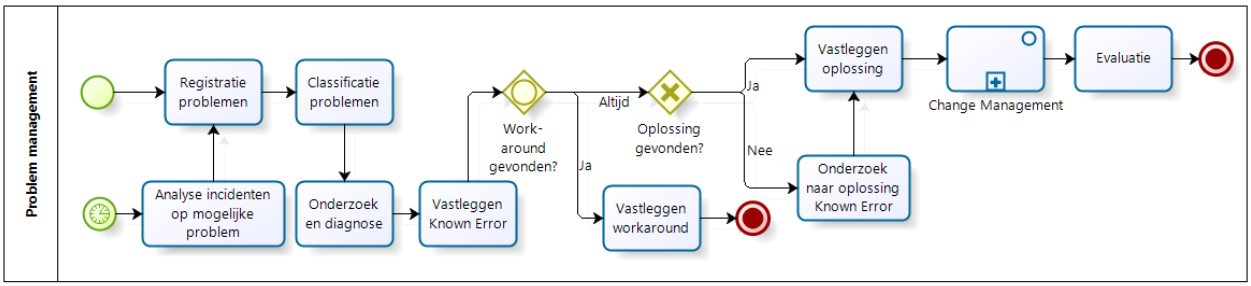


Figuur 26: ITIL-change management

Voor de control objectives “Reporting information security events and weaknesses” (sectie 13.1), “Control of technical vulnerabilities” (sectie 12.6.1) en “Management of information security incidents and improvements” (sectie 13.2) is een goede registratie en afhandeling van incidenten noodzakelijk. Hiervoor worden de ITIL processen incident management en problem management gebruikt, deze zijn weergegeven in figuur 27 en 28. Deze processen maken gebruik van de gegevens van configuration management en kunnen leiden tot wijzigingsverzoeken, welke behandeld worden door change management.

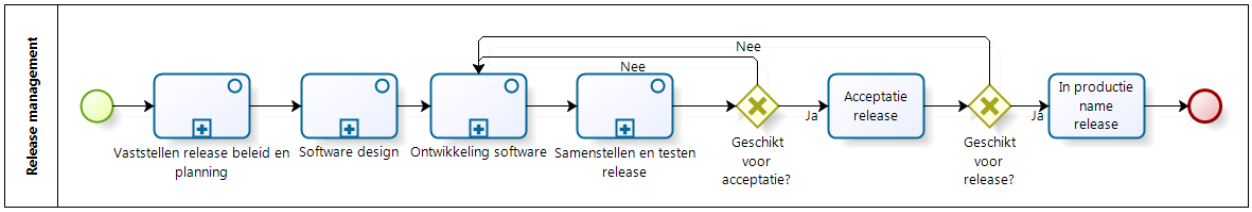


Figuur 27: ITIL-incident management

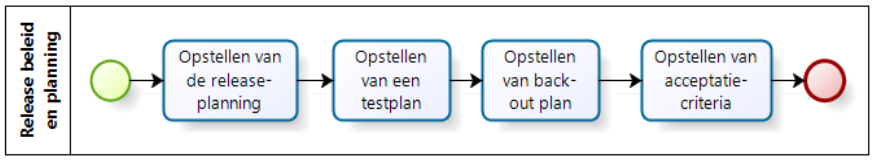


Figuur 28: ITIL-problem management

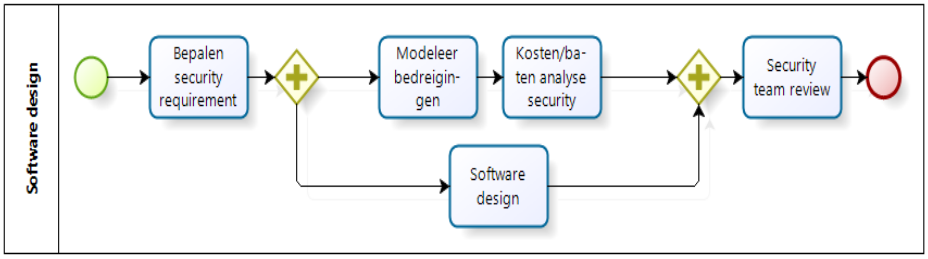
Voor de ontwikkeling van software zijn een aantal control objectives van belang, het betreft “System acceptance” (10.3.2), “Security requirement analysis and specifications” (12.1.1), “Correct processing in applications” (12.2) en “Protection of system test data” (12.4.2). Hiervoor wordt het ITIL-proces release management gebruikt in combinatie met onderdelen uit de SDLC, zoals deze in paragraaf 2.3 is weergegeven. Dit proces is beschreven in figuur 29 tot en met 33.



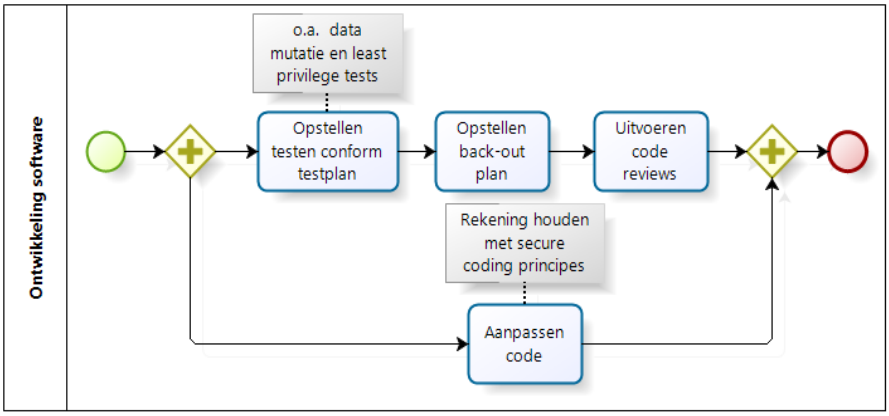
Figuur 29: ITIL-releasemanagement i.c.m. SDLC



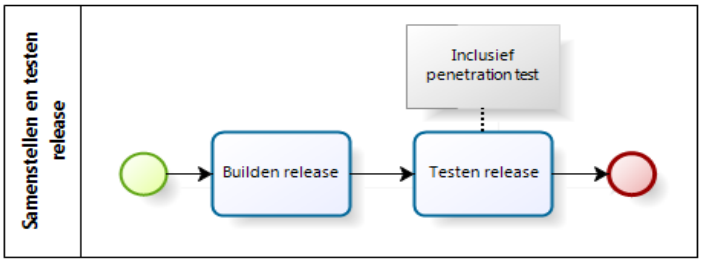
Figuur 30: Release beleid en planning



Figuur 31: Software design



Figuur 32: Ontwikkeling software



Figuur 33: Samenstellen en testen release

3.4 Discrepanties

Op basis van de opgestelde procesmodellen van het huidige, gewenste en formele proces zijn een aantal discrepanties tussen deze processen aan te wijzen. Ook zijn er een aantal discrepanties tussen het werkelijke proces en RUP en Scrum aan te geven. Dit hoofdstuk geeft de belangrijkste discrepanties weer.

1. Huidig proces ↔ ITIL-processen (m.u.v. release management)

- Configuration management

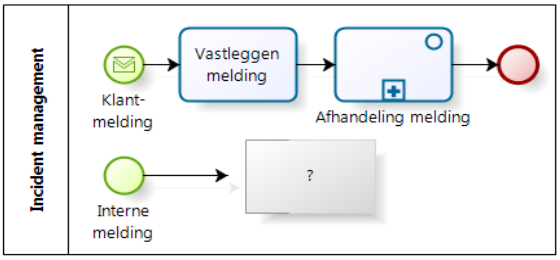
  Er is nog geen bedrijfsproces voor configuration management.

- Change management

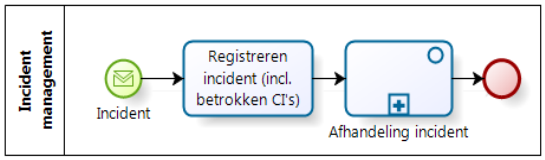
  De doelstelling van change management is een gestandaardiseerde procedure van wijzigingsverzoeken. Voor kleine wijzigingsverzoeken & bugs m.b.t. de software, welke binnenkomen bij M&S wordt een standaard procedure gebruikt (figuur zestien), voor overige wijzigingen is geen standaard procedure.

- Incident management

  Meldingen van klanten worden geregistreerd in een helpdesk-systeem, intern geconstateerde incidenten worden echter niet geregistreerd. Daarnaast is er geen koppeling met configuration items. Op basis van figuur zestien is een vereenvoudigde weergave van de registratie van incidenten bijgevoegd als figuur 34, de gewenste situatie is beschreven in figuur 35.



Figuur 34: Huidig incident management



Figuur 35: Gewenst incident management

- Problem management

  Er is nog geen bedrijfsproces voor problem management.

2. Daadwerkelijk release management ↔ Formeel release management

- Bij het formele proces is het software design onderverdeeld in drie fases, requirements analyse, external design en internal design (figuur zes), resultaat van deze stappen is een functioneel en technisch ontwerp. Bij het daadwerkelijke proces wordt enkel het functionele ontwerp opgesteld. (figuur achttien).

- Bij het formele proces wordt de software na de ontwikkeling samengesteld, getest en worden instructies bijgewerkt in de testomgeving (figuur dertien), bij het daadwerkelijke proces wordt de software enkel samengesteld en doorgegeven aan het acceptatieteam (figuur 21).

3. Formeel release management ↔ ITIL-release management i.c.m. SDLC

- Bij het formele bedrijfsproces wordt een planning van de subonderdelen opgesteld (figuur zes), een release beleid en planning met een duidelijk testplan, back-out plan en

acceptatiecriteria ontbreken (figuur 30).

- Bij het formele proces wordt bij de requirements analyse, het external design en het internal design geen rekening gehouden met security requirements (figuur acht,negen,tien), bij het gewenste proces wordt hier tijdens het hele design-traject rekening mee gehouden (figuur 31).
- In het formele proces is geen sprake van code reviews (figuur twaalf), deze zijn wel opgenomen in het gewenste proces op basis van de SDLC (figuur 32). Daarnaast is er in het huidige proces weinig aandacht voor secure coding principles en wordt tijdens de ontwikkeling niet getest op basis van een testplan.

4. Formeel proces ↔ RUP

Zoals ook in hoofdstuk twee verwacht werd, lijkt het bedrijfsproces van Empirion in grote lijnen op het Rational Unified Process, echter zijn er een aantal grote verschillen tussen RUP en het formele bedrijfsproces:

- RUP kent een uitgebreide taakverdeling, zo worden in [Kruchten, P 1999] meer dan twintig verschillende rollen genoemd. Dit in tegenstelling tot de formele procesbeschrijving van Empirion, waar maar een paar verschillende rollen expliciet met naam genoemd worden. Impliciet zijn er in het bedrijfsproces ongeveer zes rollen vast te stellen.
- RUP kent veel op te leveren artefacten, terwijl in het bedrijfsproces van Empirion maar een paar artefacten beschreven zijn.
- Een van de uitgangspunten van RUP is het visueel modelleren van software, het visueel modelleren vindt bij Empirion echter niet structureel plaats.
- Zoals ook bij de vergelijking met ITIL-change management beschreven is, wordt geen formeel proces van change request management beschreven.

5. Formeel proces ↔ Scrum

- In de procesbeschrijving van Empirion is niet opgenomen wat de duur van een iteratie is, tevens loopt de duur van een iteratie regelmatig uit. Dit in tegenstelling tot een vaste iteratie duur van 30 dagen van de Scrum-sprint.
- Binnen het formele bedrijfsproces vindt geen gestructureerde opdeling van taken plaats. Taken hebben daarmee in de praktijk een omvang tussen enkele uren tot meer dan een week. Dit in tegenstelling tot de opdeling van taken bij Scrum, waarbij tijdens de “Sprint Planning Meeting” de werkzaamheden opgedeeld worden in taken tussen vier en zestien uur werk.
- Voortgangsbewaking van de werkzaamheden is niet beschreven in het formele bedrijfsproces van Empirion, dit in tegenstelling tot de dagelijkse scrum meeting bij de Scrum-methodiek.
- De planning voor de komende iteratie is niet bekend bij de medewerkers van de afdeling R&D. Deze werkzaamheden wijzigen ook binnen de iteratie. Dit terwijl bij de Scrum-methodiek tijdens de “Sprint Planning Meeting” de werkzaamheden voor de komende sprint van 30 dagen bepaalt worden.

3.5 Conclusie

In dit hoofdstuk is een beschrijving gegeven van het huidige formele en daadwerkelijke bedrijfsproces van Empirion. Daarnaast is een beschrijving gegeven van de gewenste bedrijfsprocessen op basis van ITIL in combinatie met de SDLC om aan de ISO 27001 norm te

voldoen. Ten slotte zijn discrepanties tussen deze processen en de ontwikkel-methodieken RUP en Scrum aangegeven. Hieronder wordt een overzicht gegeven van deze discrepanties.

ITIL i.c.m. SDLC	**RUP**	**Scrum**
- Geen configuration management - Zeer beperkte procedures voor change management - Zeer beperkte procedures voor incident management - Geen problem management - Beperkt release beleid en planning - Geen aandacht voor security requirements tijdens het design-traject - Geen aandacht voor secure coding principles - Geen code reviews	- RUP kent een uitgebreide taakverdeling met veel op te leveren artefacten in tegenstelling tot het bedrijfsproces van Empirion - Visueel modelleren vindt niet structureel plaats - Change request management ontbreekt grotendeels	- Geen vaste iteratie duur (sprint van 30 dagen) - Geen opdeling van wijzigingen in taken van vier tot zestien uur - Geen dagelijkse scrum meeting - Geen duidelijk overzicht van te verrichten werk in de huidige iteratie (sprint backlog)

Tabel 1: Overzicht discrepanties t.o.v. het formele proces

4 Alternatieve oplossingen

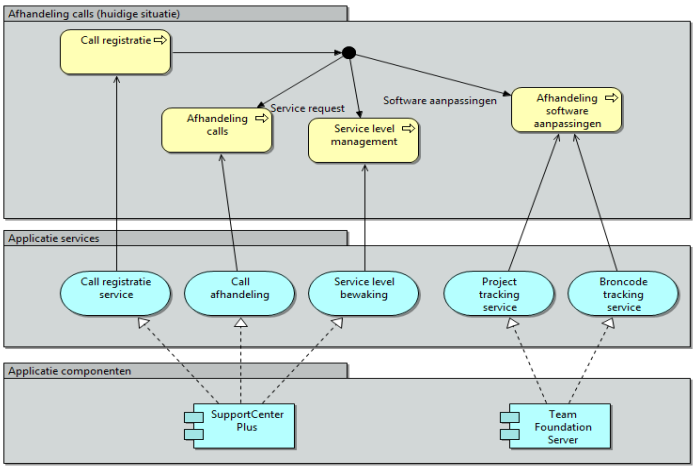
Het vierde hoofdstuk stelt op basis van de gewenste processen een aantal alternatieve oplossingen voor om de discrepanties uit hoofdstuk drie aan te pakken en de gewenste processen in te voeren. Hierbij zijn de alternatieven verdeeld in twee categorieën, de controlegerichte processen change, incident, problem en configuration management en het uitvoerings-gerichte proces release management. Deze onderverdeling wordt gehanteerd, omdat tussen de controlegerichte processen veel onderlinge relaties aanwezig zijn, terwijl de relatie tussen de controle-gerichte processen en het uitvoerende proces release management minder aanwezig is. Door middel van interviews worden de alternatieven beoordeeld.

4.1 Controlegerichte processen

In hoofdstuk drie is geconcludeerd dat er nog geen huidige bedrijfsprocessen zijn ingericht conform de gewenste ITIL-processen configuration management, change management, incident management en problem management. Deze paragraf geeft een aantal alternatieven voor de inrichting van deze processen.

Huidige ondersteuning bedrijfsprocessen

Bij het opstellen van de alternatieven m.b.t. de controle-gerichte processen is in eerste instantie bekeken in hoeverre de gewenste ITIL-processen ondersteund kunnen worden met de door Empirion gebruikte applicaties. Hierbij is gebruik gemaakt van Archi [Archi, 2011], een tool om in de taal ArchiMate [ArchiMate, 2011] de applicatie architectuur in combinatie met de bedrijfsprocessen weer te geven. In figuur 36 zijn deze applicaties in combinatie met de ondersteunde bedrijfsprocessen weergegeven.



Figuur 36: Huidige applicatie architectuur

De applicatie SupportCenter Plus biedt geen ondersteuning voor het bijhouden van configuration items, daarnaast biedt de applicatie ook geen goede ondersteuning voor het bijhouden van incidenten en wijzigingen m.b.t. de infrastructuur.

Binnen Team Foundation Server worden wijzigingen met betrekking tot de broncode bijgehouden (versiebeheer), voor overige configuration items biedt dat systeem ook geen oplossing. De ondersteuning van Team Foundation Server met betrekking tot de overige ITIL-processen is afhankelijk van de inrichting van het systeem op basis van projecttemplates. Het gebruikte projecttemplate binnen Empirion is momenteel het agile template. Dit template biedt ondersteuning voor de volgende workitems: Scenario, Quality of service, Tasks, Bugs, Risks. Naast dit projecttemplate biedt Team Foundation Server echter ook de mogelijkheid om gebruik te maken van

andere templates, zodat ook het registreren en behandelen van RFC's met betrekking tot de software mogelijk is binnen Team Foundation Server.

Pakketselectie

Omdat de huidige applicaties de ITIL-processen niet volledig ondersteunen is bekeken welke applicatie deze ondersteuning zou kunnen bieden. Hiervoor zijn de volgende functionele criteria opgesteld:

Noodzakelijke functionaliteit:

- Ondersteuning van configuration management
  - Aanpasbaar datamodel
  - Relaties tussen configuration items
- Ondersteuning van change management
  - Koppeling met configuration items
  - Ondersteuning van change-approval
- Ondersteuning van incident management
  - Koppeling met configuration items
- Ondersteuning van problem management
  - Koppeling met configuration items
- Ondersteuning voor acces-control
- Bijhouden van wijzigingshistorie t.b.v. audits

Gewenste functionaliteit:

- Integratiemogelijkheden met andere applicaties

Overig

- Kosten van de applicatie
- Licentie van de applicatie
- Benodigde implementatiewerkzaamheden
- Beschikbare documentatie
- Gebruiksvriendelijkheid van de applicatie

Naast deze criteria had het management van Empirion de voorkeur om SupportCenter Plus te behouden, omdat de inrichting en ingebruikname van deze applicatie net afgerond was.

Op basis van bovenstaande criteria is in eerste instantie bekeken of de leverancier van de huidige helpdesk applicatie SupportCenter Plus ook een applicatie aanbiedt met dezelfde functionaliteit als SupportCenter Plus i.c.m. ondersteuning voor de ITIL-processen, zodat de inrichting van SupportCenter Plus overgenomen kon worden in deze applicatie. Hierbij is een inventarisatie gemaakt van de functionaliteit van SupportDesk Plus op basis van de online demo en de beschikbare documentatie. Uit deze inventarisatie kwam naar voren dat dit pakket niet voldoet aan de criteria voor configuration management en dat de applicaties dermate verschillen dat de werkzaamheden ter inrichting van SupportCenter Plus voor SupportDesk Plus opnieuw uitgevoerd zouden moeten worden.

Vervolgens zijn onderstaande applicaties beoordeeld op basis van beschikbare documentatie en online demo's:

- Beetil
- SysAid IT Enterprise Edition
- Aegis Service Desk
- Minitor 24-7 incidentMonitor
- EasyCMDB
- Quism
- iTOP

De bekende ITIL-applicaties van de bedrijven HP, IBM, CA en BMC zijn niet beoordeeld i.v.m. de licentiekosten van deze applicaties.

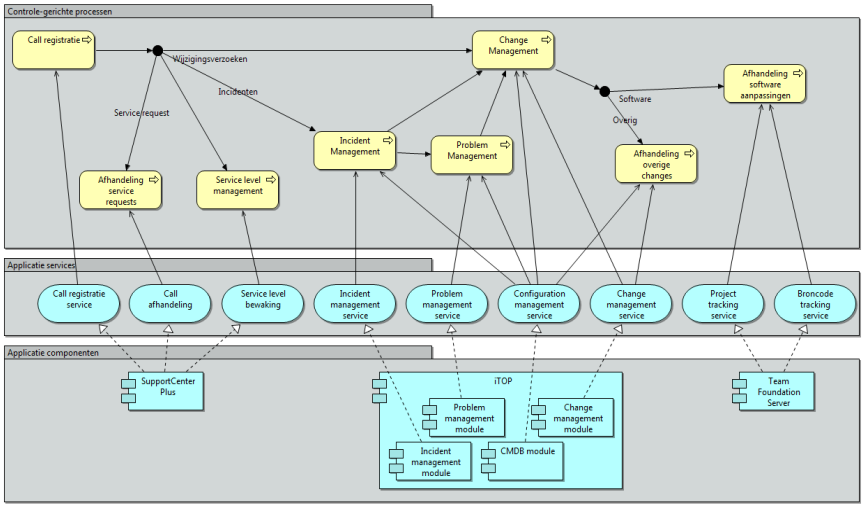
Uit deze beoordeling kwam naar voren dat iTOP voldeed aan alle functionele criteria, daarnaast had iTOP het voordeel boven de andere pakketten dat deze beschikbaar is onder een open source licentie, waardoor er geen kosten verbonden zijn aan het gebruik van het pakket en er getest kon worden of de applicatie inderdaad bruikbaar is voor Empirion. Daarnaast is er online documentatie beschikbaar voor de installatie, configuratie en het gebruik van de applicatie. Omdat geen van de overige applicaties op basis van gebruiksvriendelijkheid ver boven iTOP uit stak is geen nadere informatie opgevraagd m.b.t. de prijsstelling van deze pakketten.

Om de aanpasbaarheid van de applicatie te testen zijn de configuratiebestanden van de configuration-management module van iTOP aangepast aan de wensen van Empirion. Ook is de applicatie geïnstalleerd op een laptop om deze te kunnen testen en te demonstreren aan het management en een externe consultant m.b.t. de ISO 27001 certificering. Hieruit kwamen geen problemen m.b.t. het gebruik van iTOP naar voren.

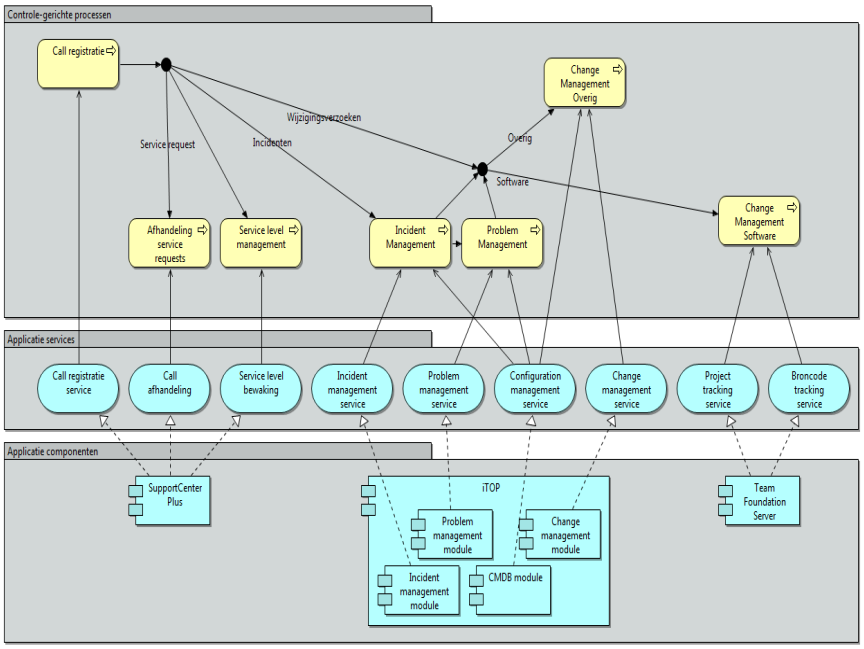
Mogelijke ondersteuning bedrijfsprocessen

Op basis van het gebruik van de drie applicaties SupportCenter Plus, iTOP en Team Foundation Server zijn een drietal mogelijkheden om de gewenste bedrijfsprocessen te ondersteunen uitgewerkt in figuur 37 t/m 39.

Bij combinatie I vormt iTOP de centrale spil, waarin alle incidenten en wijzigingsverzoeken afgehandeld worden. Hierbij wordt SupportCenter Plus gebruikt voor binnenkomende calls van klanten en Team Foundation Server voor het afhandelen van goedgekeurde software aanpassingen.

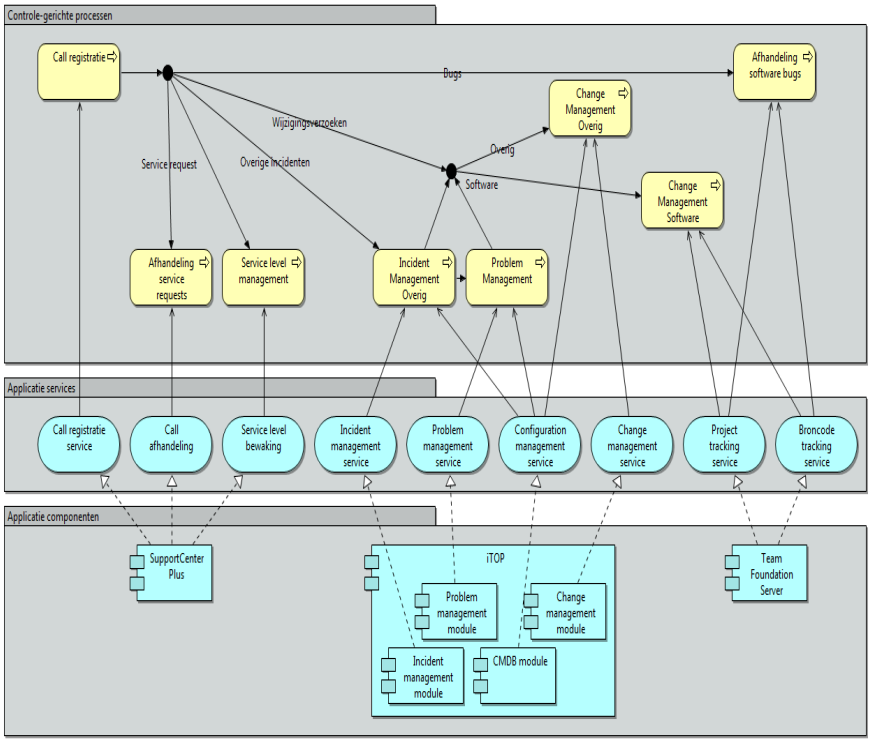


Figuur 37: Combinatie I



Figuur 38: Combinatie II

In tegenstelling tot de eerste combinatie worden in combinatie II de wijzigingsverzoeken m.b.t. de software in Team Foundation Server afgehandeld. Wel worden alle incidenten nog steeds in iTOP afgehandeld.



Figuur 39: Combinatie III

Bij de derde combinatie worden naast de wijzigingsverzoeken ook de duidelijke software bugs direct in Team Foundation Server afgehandeld. Alleen overige incidenten worden afgehandeld in iTOP.

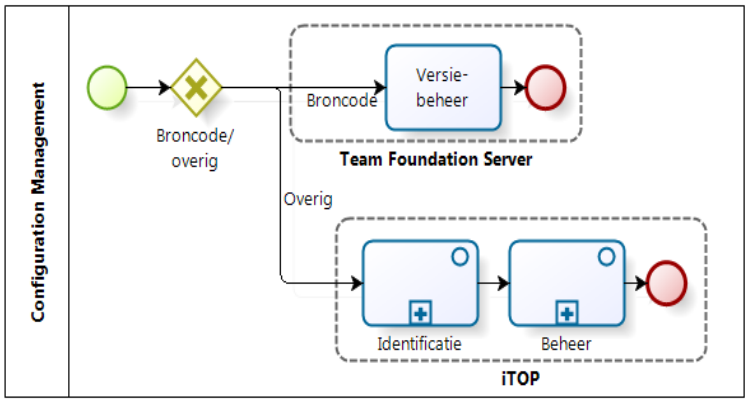
Samengevat verschillen de drie combinatie in het moment dat er een ontkoppeling plaats vindt tussen de calls m.b.t. de software en calls m.b.t. overige items. Dit is weergegeven in tabel twee.

Combinatie	Incident management	Change management	Ontkoppeling software/overig
I	Compleet via iTOP	Compleet via iTOP	Ontkoppeling na goedkeuring change.
II	Compleet via iTOP	Software via TFS, overig via iTOP	Ontkoppeling bij aanvang change management.
III	Bugs via TFS, overig via iTOP	Software via TFS, overig via iTOP	Ontkoppeling bij aanvang incident management en aanvang change management.

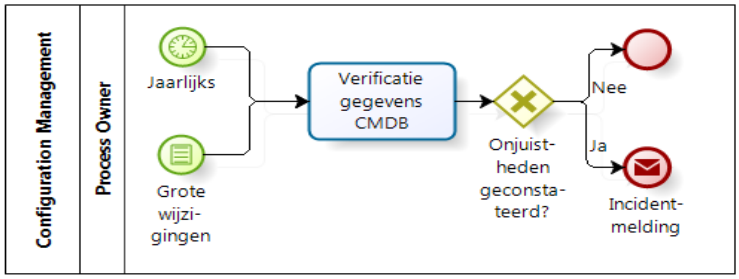
Tabel 2: Mogelijke combinaties Incident & Change management

Configuration management

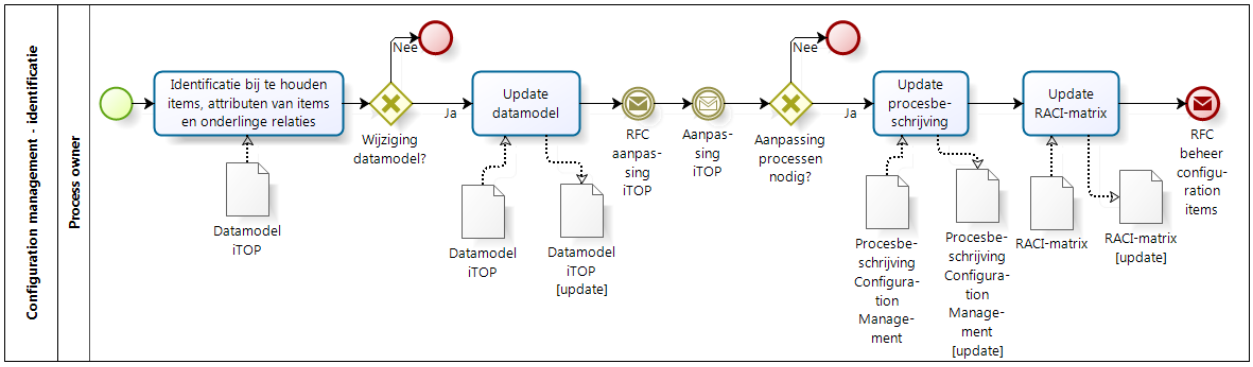
Configuration management draagt zorg voor alle configuration items en dient ter ondersteuning van change management, incident management en problem management. Binnen configuration management zijn een aantal verschillende processen te onderscheiden. De identificatie en het beheer is weergegeven in figuur 40 en dit is verder uitgewerkt in figuur 42 t/m 49. Naast de identificatie en het beheer is het van belang dat de configuration items in de configuration management database ook geverifieerd worden. Dit is weergegeven in figuur 41. D.m.v. deze processen wordt voldaan aan de ISO 27001 eisen m.b.t. asset-management, zoals beschreven in paragraaf 3.3.



Figuur 40: Configuration management



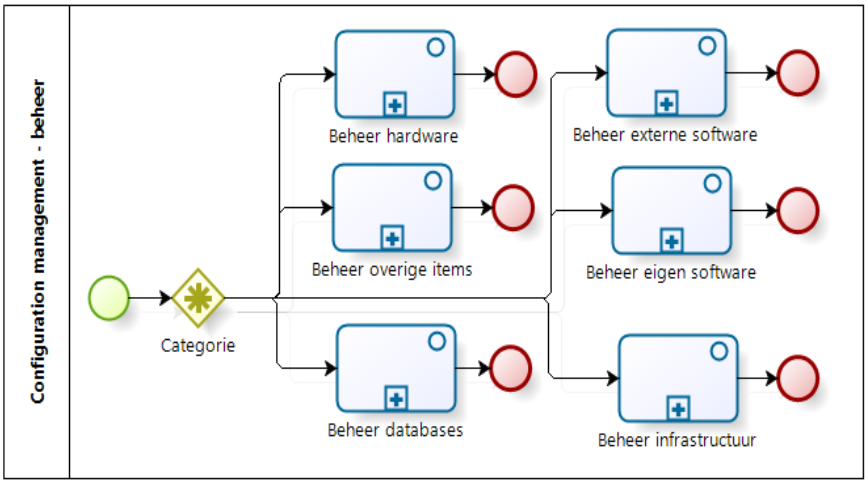
Figuur 41: Configuration management - verificatie



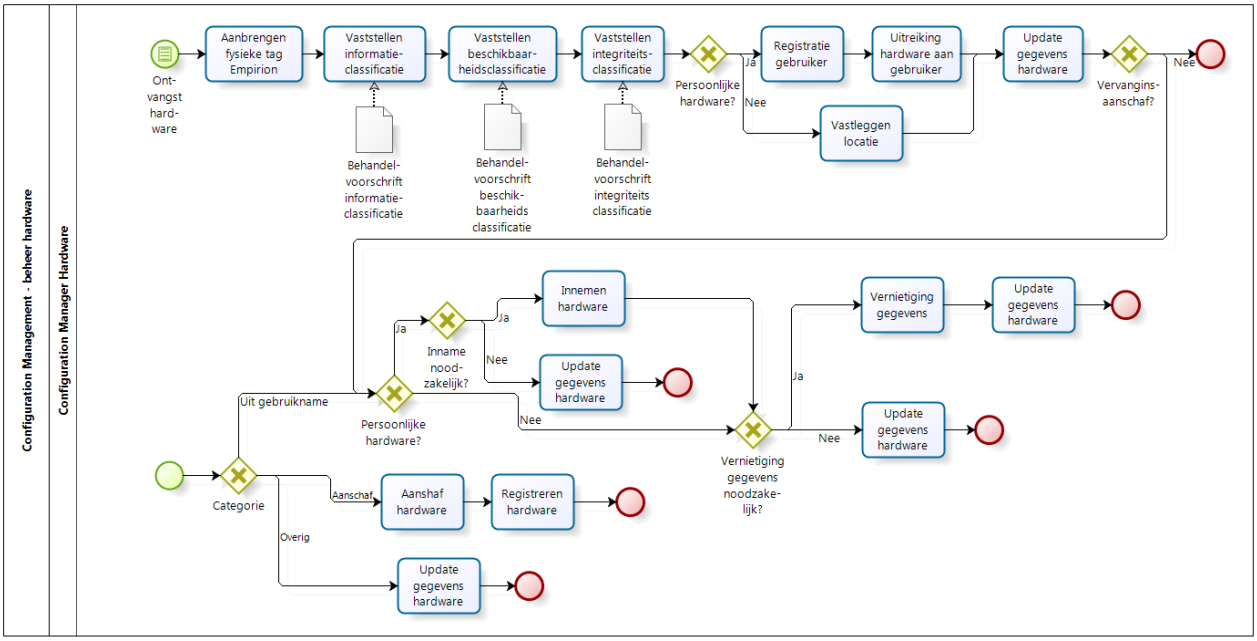
Figuur 42: Configuration management - identificatie

Door aanpassing in het beleid m.b.t. informatiebeveiliging kan er een wijziging in de bij te houden configuration items plaatsvinden. Daarnaast is het de bedoeling om voor asset management alle assets vast te leggen waarop afgeschreven wordt, dit kan ook leiden tot benodigde aanpassingen in het configuration management proces en het datamodel van de applicatie.

Het beheer van de configuration items is opgedeeld in een aantal verschillende activiteiten, omdat de behandeling van CI's verschilt tussen de verschillende fysieke items, externe software en eigen software. Daarnaast zijn er verschillende personen verantwoordelijk voor de verschillende activiteiten.

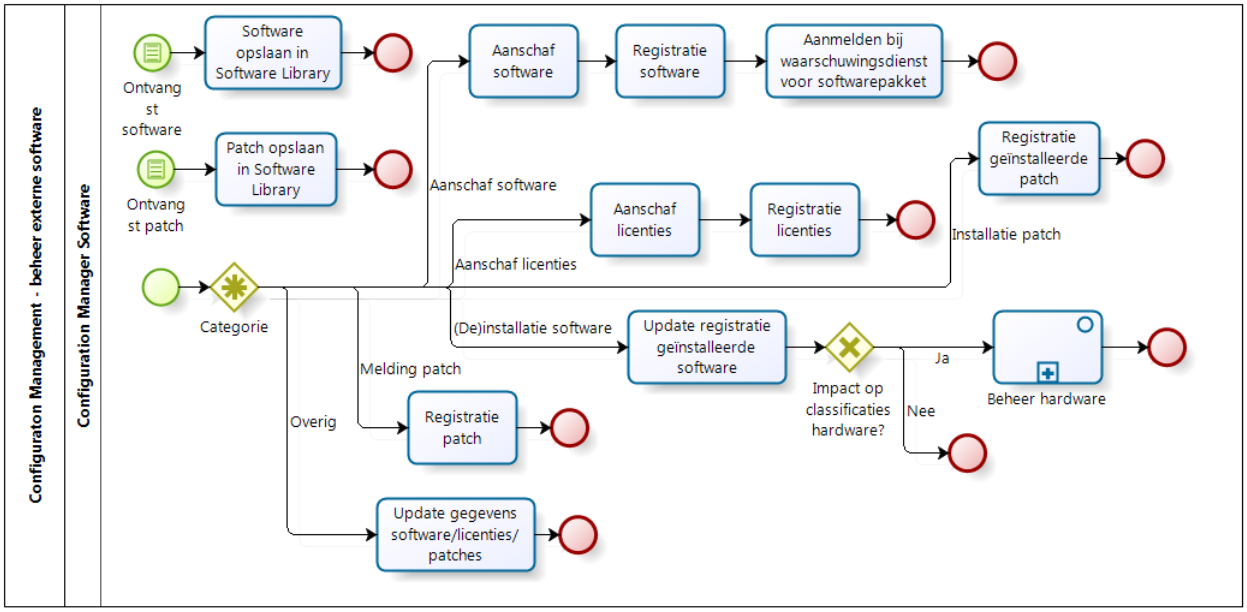


Figuur 43: Configuration management - beheer



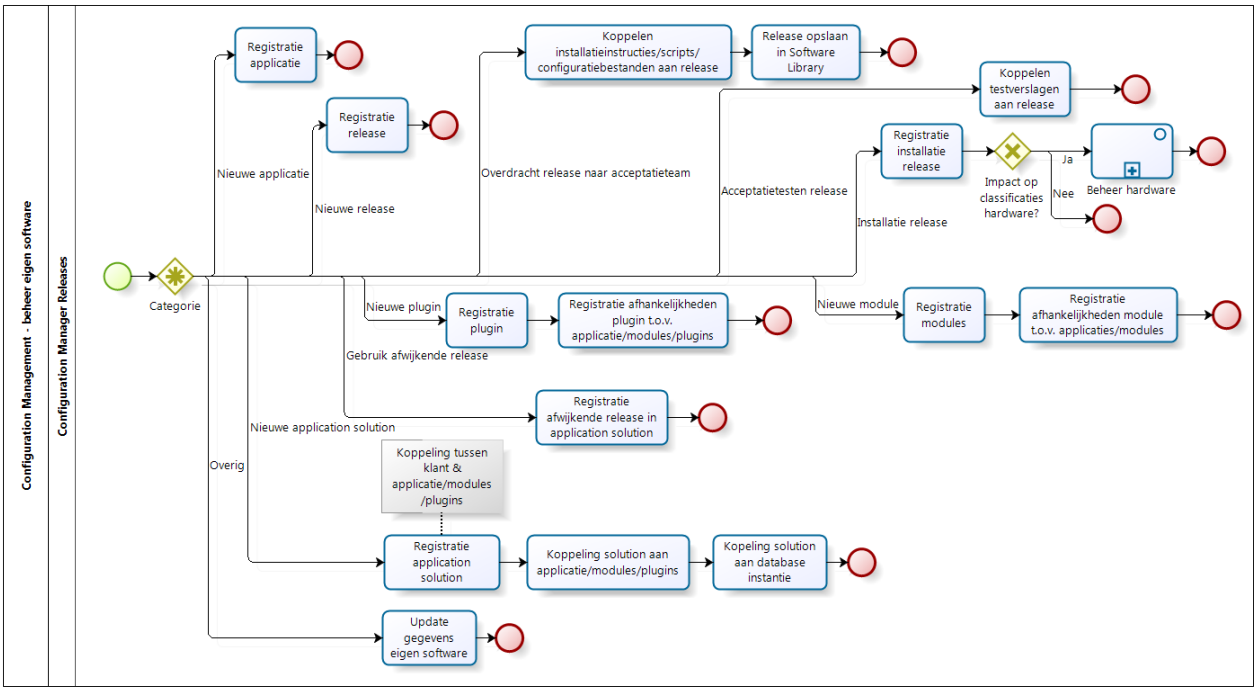
Figuur 44: Configuration management - beheer hardware

In figuur 44 is het beheer van de hardware weergegeven. Hierbij wordt onderscheidt gemaakt tussen een aantal categorieën: aanschaf, ontvangst, uit gebruik name van hardware en het updaten van gegevens. Bij de items worden drie classificaties vastgelegd op basis van de drie aandachtspunten bij informatiebeveiliging: vertrouwelijkheid, beschikbaarheid en integriteit. Deze zijn afzonderlijk opgenomen, omdat dit van belang is voor de behandeling van de items. Een harddisk met vertrouwelijke informatie dient bijvoorbeeld vernietigt te worden bij uit gebruik name, terwijl een router een hoge beschikbaarheidsclassificatie heeft, maar bij uit gebruik name niet vernietigt hoeft te worden. Ook is het van belang om bij te houden wie de gebruiker is van hardware, zodat dit bij bijvoorbeeld uitdiensttreding goed na te gaan is.



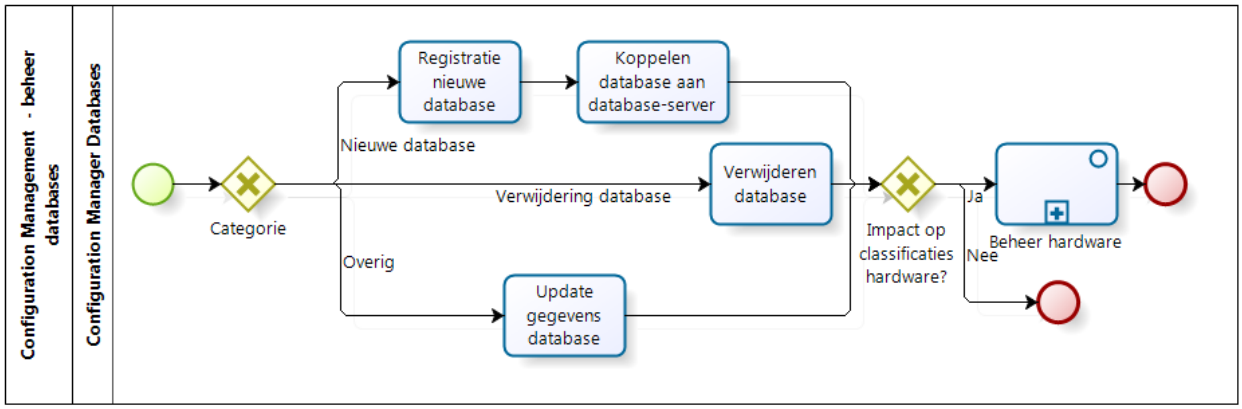
Figuur 45: Configuration management - beheer externe software

Bij het beheer van externe software is het van belang dat bekend is welke software geïnstalleerd is op met name productieservers, zodat deze bij uitval snel hersteld kunnen worden. Hiervoor dient de software ook opgeslagen te worden op een bekende locatie, de software library, zodat er geen tijd kostbare tijd verloren gaat. Daarnaast is het voor de beveiliging van applicaties van belang dat beveiligingsrisico's snel bekend zijn, zodat de hiermee samenhangende risico's beter afgedekt worden. Om dit af te dekken is aanmelding bij een waarschuwingsdienst, zoals secunia, belangrijk. Door het registreren van patches kan snel geïnventariseerd worden of alle applicaties up-to-date zijn, om te voorkomen dat applicaties onbewust aan beveiligingsrisico's bloot gesteld worden. Door het licentiebeheer kunnen risico's van ontbrekende licenties afgedekt worden, ook kan dit leiden tot besparingen door inzicht in de verschillen tussen gebruikte en aangeschafte licenties. Deze activiteiten zijn weergegeven in figuur 45.



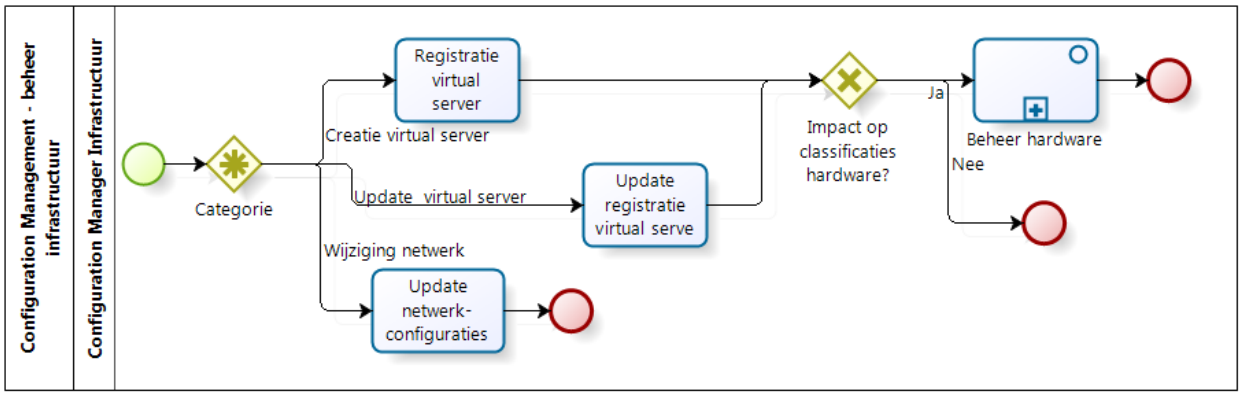
Figuur 46: Configuration management - beheer eigen software

In figuur 46 is het configuration management proces weergegeven voor eigen software. Hierbij wordt voor elke applicatie vastgelegd welke releases, modules en plugins er zijn en waarop deze geïnstalleerd zijn. Door deze te koppelen aan de klanten is bij een probleem direct na te gaan welke klanten getroffen worden door het probleem, zodat snel een inschatting gemaakt kan worden van de urgentie en prioriteit om het probleem op te lossen.



Figuur 47: Configuration management - beheer databases

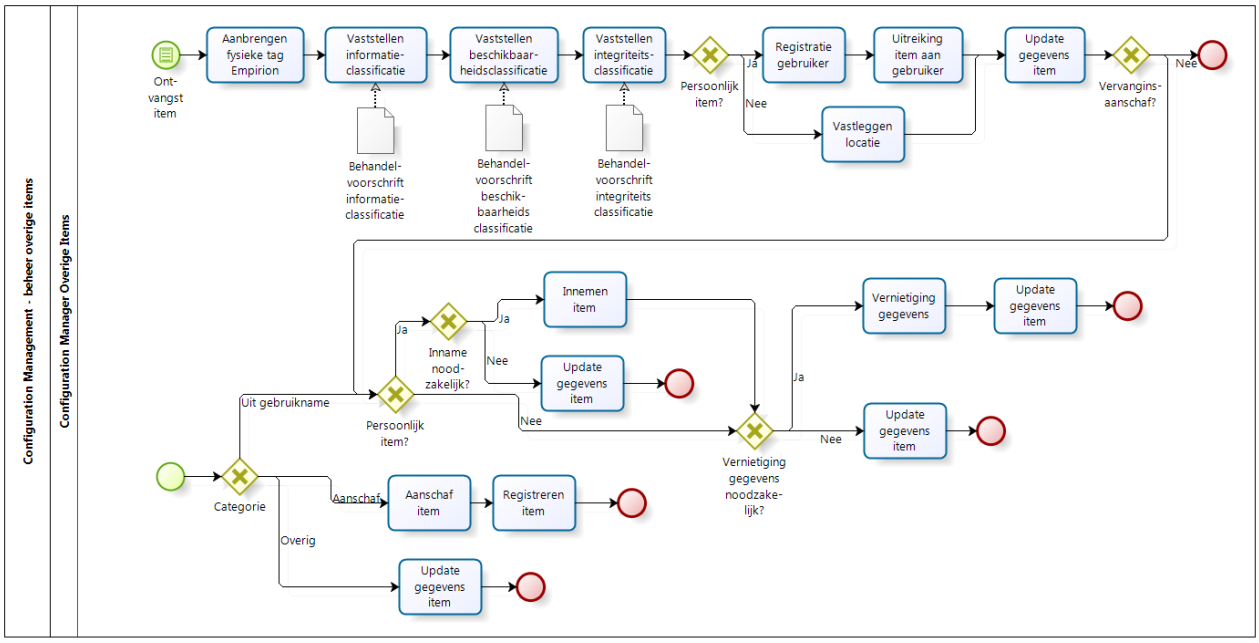
Om bij problemen snel te kunnen bepalen welke databases getroffen worden is het proces in figuur 47 opgesteld. Door het vastleggen van de databases is bij storingen aan de hardware en infrastructuur waar de database-server van afhankelijk is direct inzichtelijk welke databases getroffen worden. Ook ondersteund dit de mogelijkheid om d.m.v. change management wijzigingen aan de database buiten de applicaties om vast te leggen t.o.v. de database, zodat deze acties inzichtelijk zijn.



Figuur 48: Configuration management - beheer infrastructuur

Voor het inzichtelijk maken van de impact van storingen is het van belang dat vastligt welke virtuele server op welke hardware draait. Dit inzicht is ook van belang voor netwerkconfiguraties, zodat bij storingen aan netwerkapparatuur ook direct inzichtelijk is op welke servers en applicaties de storing impact heeft.

Naast de processen voor specifieke items is een algemeen proces opgenomen voor assets, welke niet direct gelinkt zijn aan de applicatiearchitectuur, maar waarbij het wel van belang is dat deze geregistreerd worden.



Figuur 49: Configuration management - overige items

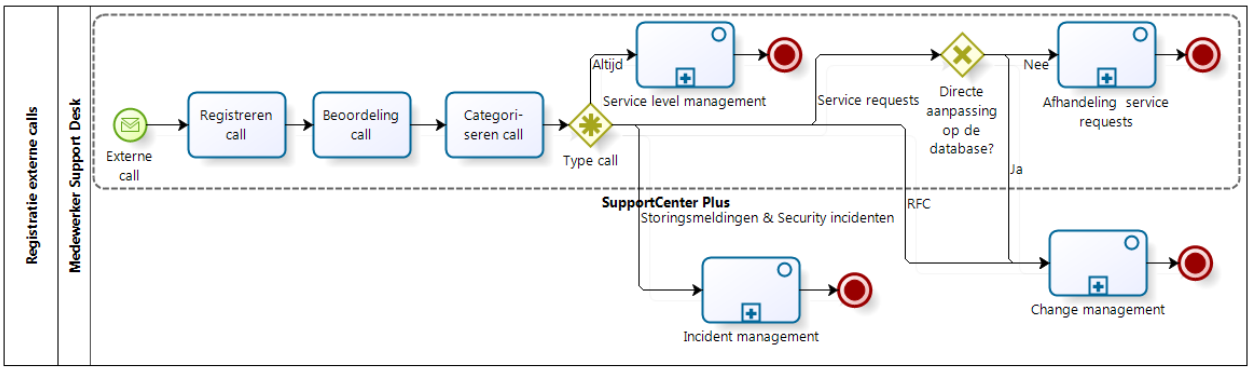
Overige controlegerichte processen

De controlegerichte processen dienen ter verbetering van de informatiebeveiliging. Zoals in het begin van deze paragraaf aangegeven is, zijn er een drietal verschillende combinaties mogelijk om de drie applicaties in te zetten. Deze combinaties worden hieronder behandeld.

Combinatie I

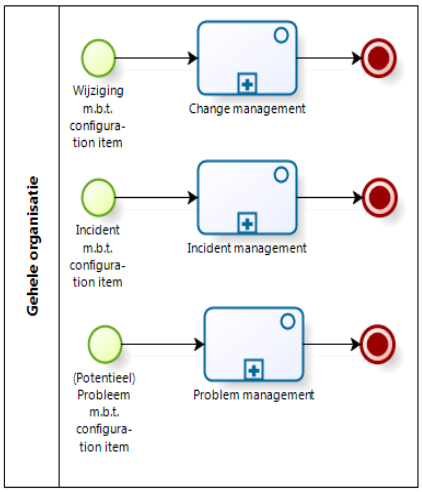
De eerste combinatie is uitgewerkt in figuur 46 t/m 50. Kenmerken van deze combinatie zijn:

- Registratie en afhandeling externe service requests in SupportCenter Plus (m.u.v. directe wijziging op de database)
- Registratie overige externe calls en bewaking SLA-termijnen in SupportCenter Plus
- Registratie van interne calls in iTOP
- Afhandeling van alle incidenten in iTOP
- Afhandeling van alle wijzigingsverzoeken in iTOP



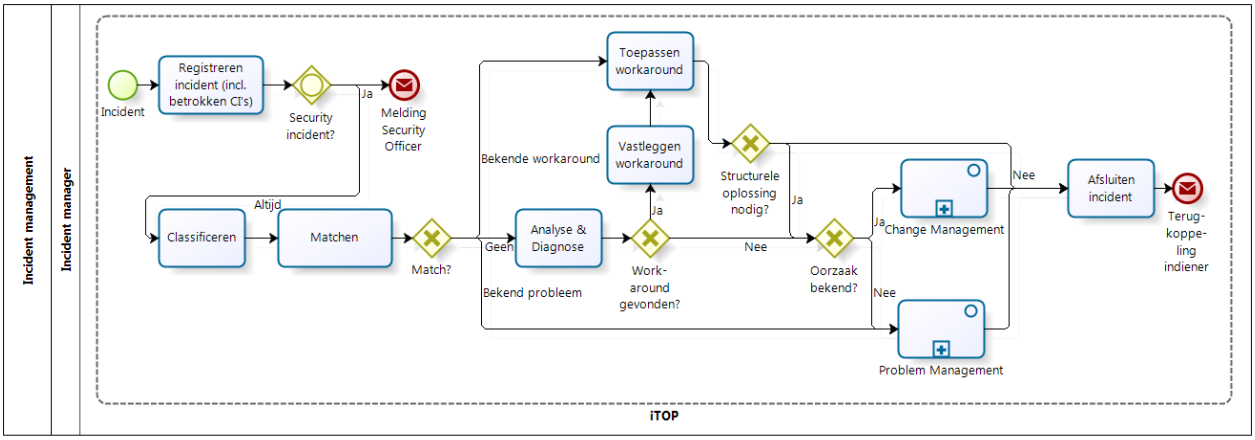
Figuur 50: Combinatie I - externe calls

In figuur 50 is de afhandeling van externe calls weergegeven. In de huidige situatie worden alle calls afgehandeld in SupportCenter Plus, waarbij geen relatie gelegd kan worden tussen de betrokken CI's en de call. Omdat door een goede koppeling het inzichtelijker is welke CI's verantwoordelijk zijn voor storingen, kan hier ook beter op gestuurd worden. Omdat service requests, met uitzondering van directe database aanpassingen, geen impact hebben op configuration items worden deze nog steeds in SupportCenter Plus afgehandeld.



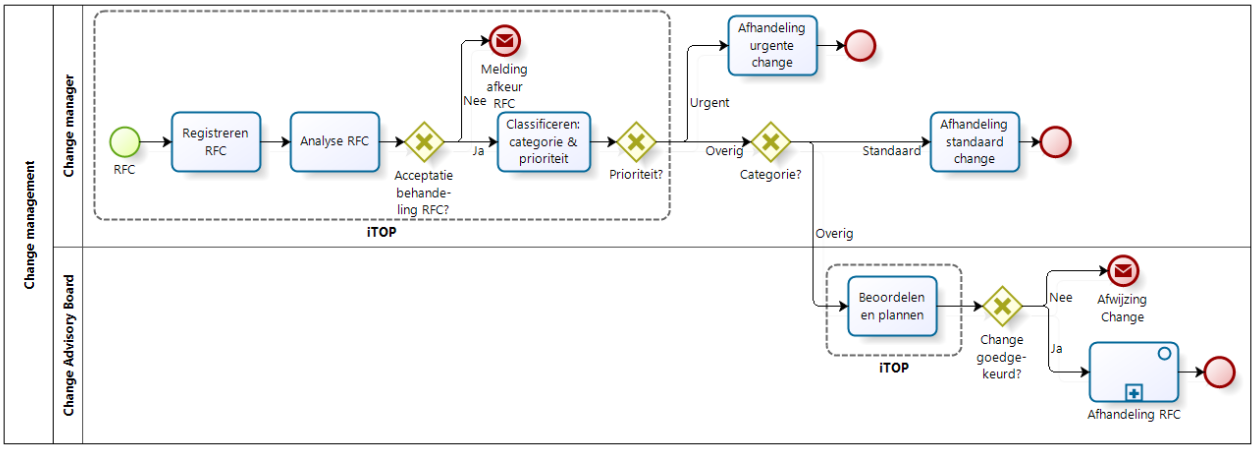
Figuur 51: Combinatie I - interne calls

Momenteel worden interne calls niet vastgelegd en worden wijzigingen vaak mondeling afgestemd. Voor de ISO 27001 certificering is het van belang dat interne incidenten, wijzigingsverzoeken en problemen ook goed geregistreerd worden.

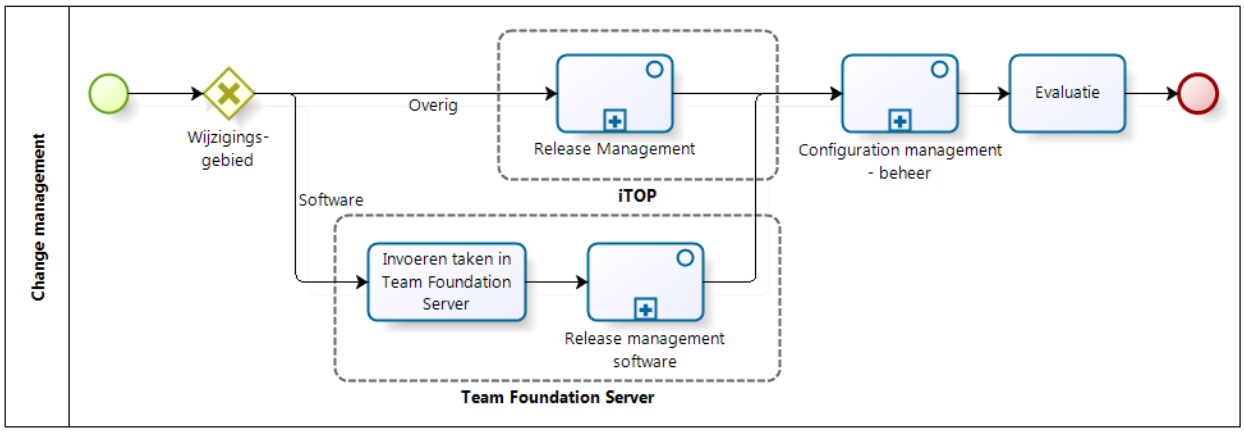


Figuur 52: Combinatie I - incident management

In figuur 52 is het proces van incident management beschreven. Dit proces is gelijk aan het beschreven ITIL-proces in hoofdstuk 3, met de toevoeging dat in het geval van security incidenten ook direct de Security Officer wordt ingelicht. Dit omdat een goede afhandeling van security incidenten essentieel is.



Figuur 53: Combinatie I - change management



Figuur 54: Combinatie I - afhandeling change

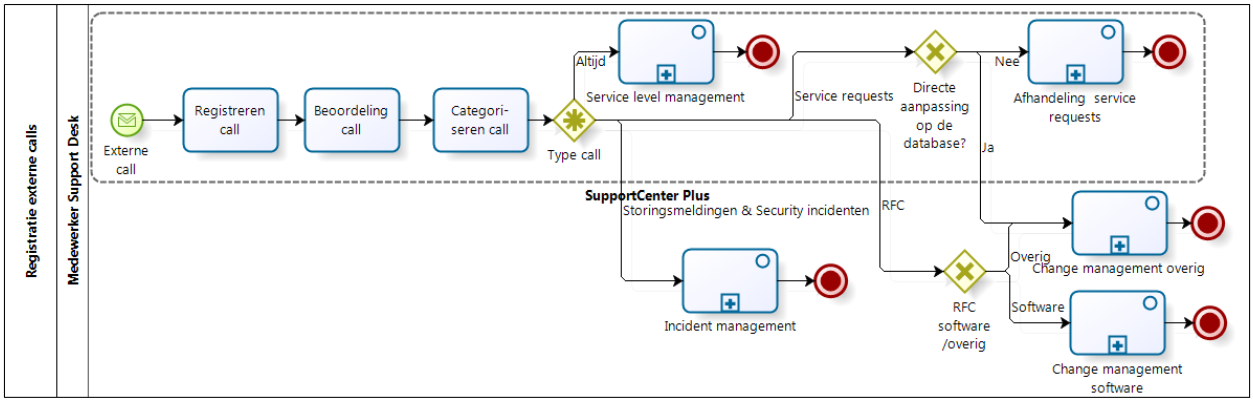
In figuur 53 en 54 is het change management proces weergegeven. Na goedkeuring van het wijzigingsverzoek wordt een onderscheid gemaakt tussen software aanpassingen en overige aanpassingen. In het geval van software aanpassingen vindt de afhandeling plaats in Team Foundation Server, vandaar dat voor de uitvoer van de aanpassing eerst taken aangemaakt moeten worden in Team Foundation Server.

Combinatie II

De tweede combinatie is uitgewerkt in figuur 55 t/m 59. Kenmerken van deze combinatie zijn:

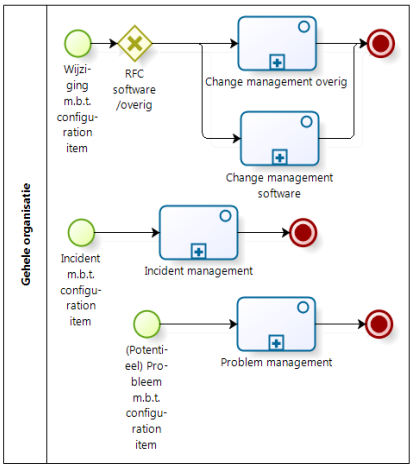
- Registratie en afhandeling externe service requests in SupportCenter Plus (m.u.v. directe wijziging van de database)
- Registratie overige externe calls en bewaking SLA-termijnen in SupportCenter Plus
- Registratie van interne incidenten en problemen in iTOP
- Registratie van interne wijzigingsverzoeken m.b.t. eigen software in Team Foundation Server
- Registratie van overige wijzigingsverzoeken in iTOP
- Afhandeling van alle incidenten en problemen in iTOP
- Afhandeling van wijzigingsverzoeken m.b.t. eigen software in Team Foundation Server

- Afhandeling van overige wijzigingsverzoeken in iTOP

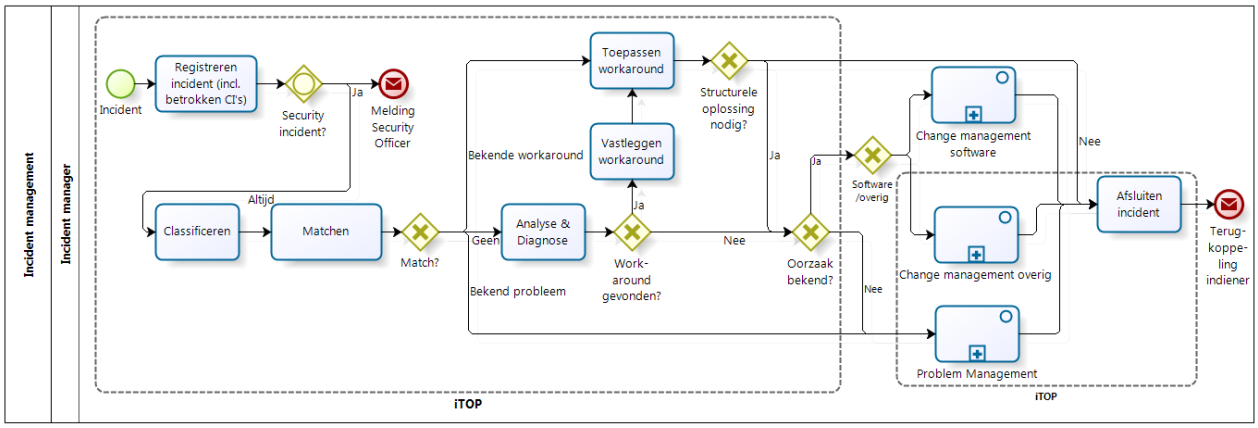


Figuur 55: Combinatie II - externe calls

De afhandeling van externe calls in de tweede combinatie verschilt ten opzichte van de eerste combinatie in het onderscheid wat gemaakt wordt tussen wijzigingen m.b.t. de software en overige wijzigingen. Dit verschil wordt ook gemaakt bij interne gemelde calls. Dit is weergegeven in figuur 55 en 56. Indien uit een incident een wijzigingsverzoek voorkomt, wordt hierin ook onderscheid gemaakt tussen software wijzigingen en overige wijzigingen (figuur 57). Dit onderscheid vindt plaats omdat deze wijzigingsverzoeken in twee verschillende applicaties afgehandeld worden.

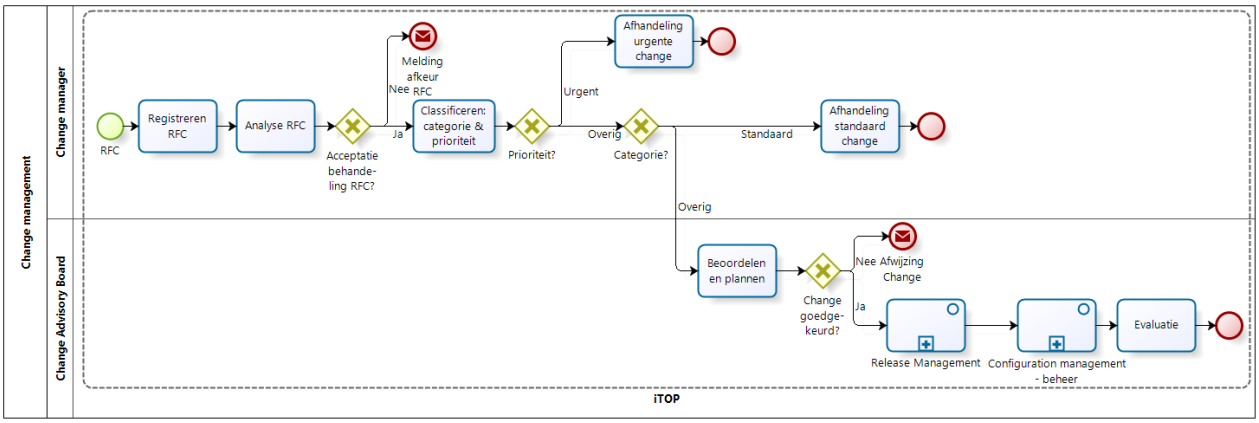


Figuur 56: Combinatie II - interne calls

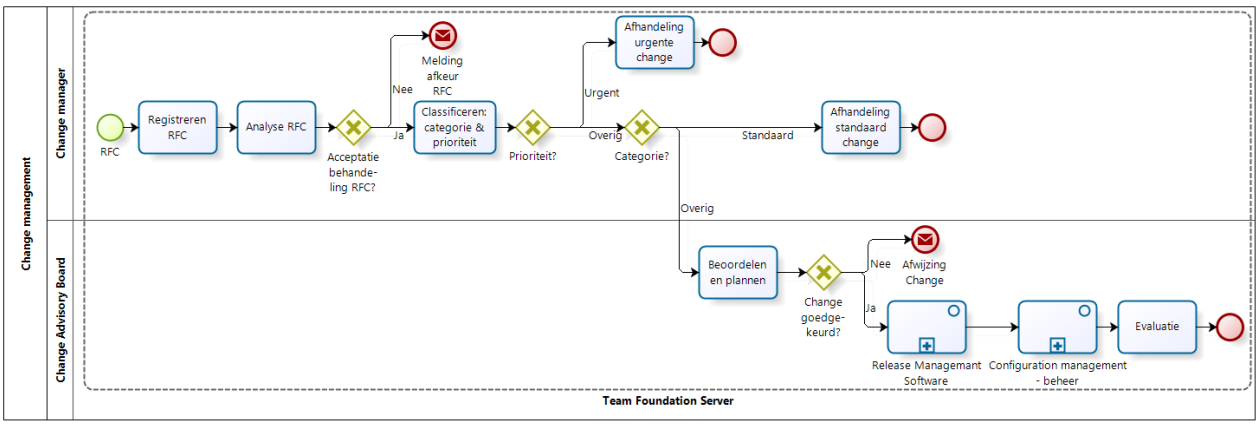


Figuur 57: Combinatie II - incident management

Zoals in figuur 58 en 59 weergegeven is, verschilt het change management proces niet tussen de afhandeling in iTOP en Team Foundation Server, deze is ook gelijk aan het proces bij de eerste combinatie. Wel is het niet meer nodig om voor een software wijzigingsverzoek taken aan te maken in Team Foundation Server na het beoordelen van het wijzigingsverzoek, omdat deze aangemaakt worden bij registratie van het verzoek.



Figuur 58: Combinatie II - change management overig



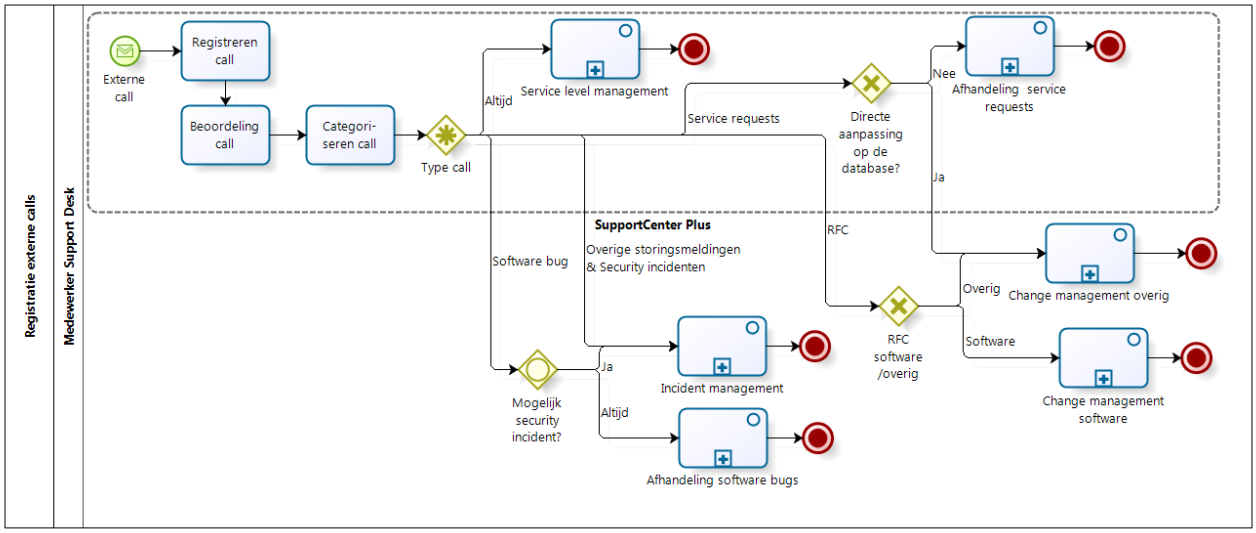
Figuur 59: Combinatie II - change management software

Combinatie III

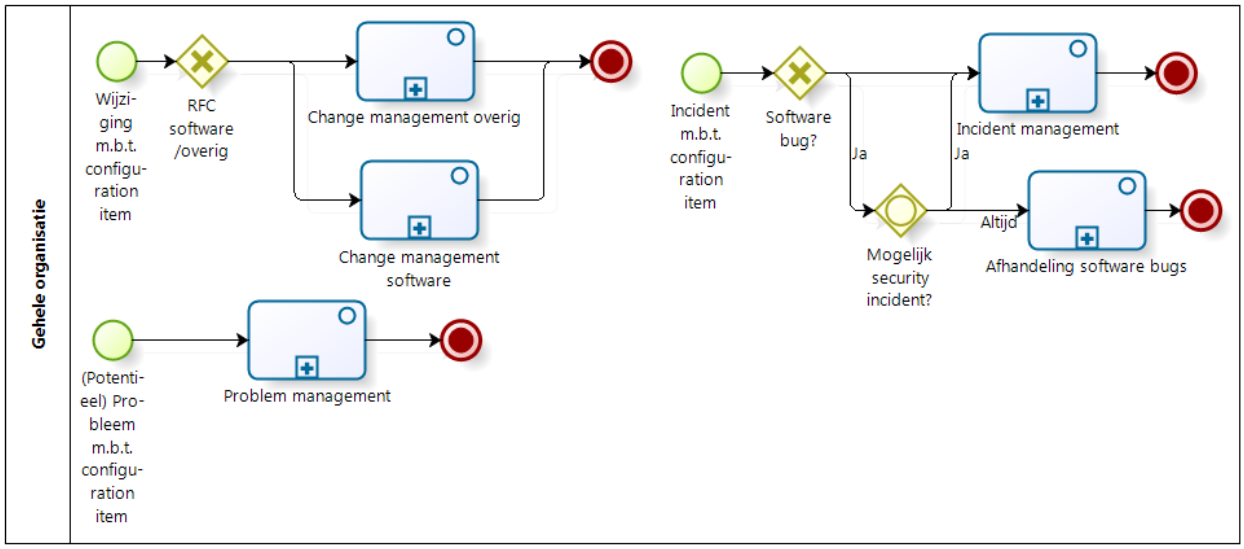
De derde combinatie is uitgewerkt in figuur 60 t/m 65. Kenmerken van deze combinatie zijn:

- Registratie en afhandeling externe service requests in SupportCenter Plus (m.u.v. directe wijziging van de database)
- Registratie overige externe calls en bewaking SLA-termijnen in SupportCenter Plus
- Registratie van intern gemelde bugs direct in Team Foundation Server
- Registratie van overige interne incidenten in iTOP
- Registratie van interne wijzigingsverzoeken m.b.t. eigen software in Team Foundation Server
- Registratie van overige wijzigingsverzoeken in iTOP
- Afhandeling van bugs in Team Foundation Server
- Afhandeling van overige incidenten en problemen in iTOP

- Afhandeling van wijzigingsverzoeken m.b.t. eigen software in Team Foundation Server
- Afhandeling van overige wijzigingsverzoeken in iTOP

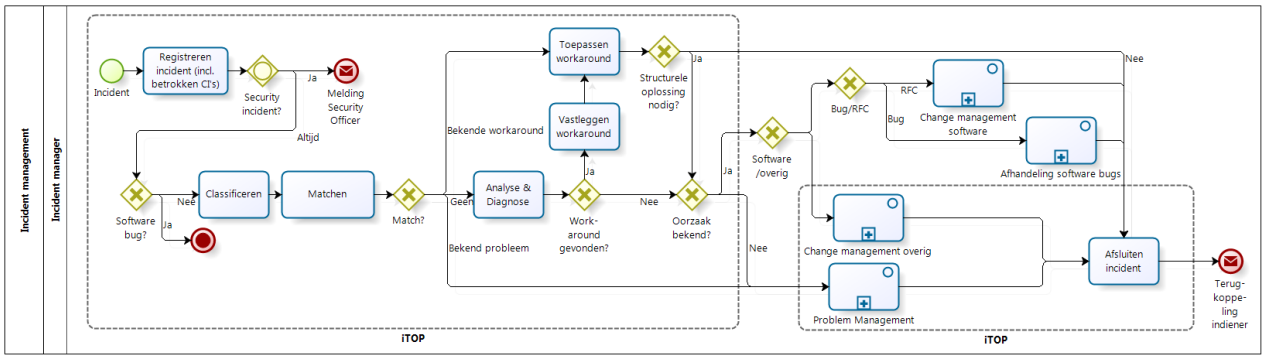


Figuur 60: Combinatie III - externe calls

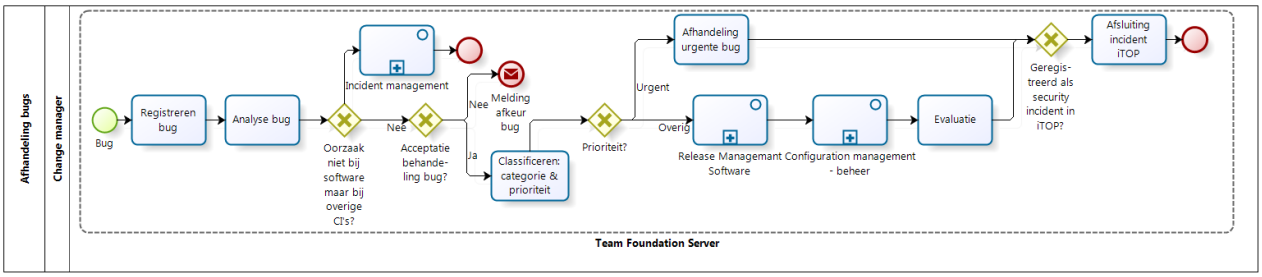


Figuur 61: Combinatie III - interne calls

Bij de derde combinatie worden naast de wijzigingsverzoeken ook de software bugs direct in Team Foundation Server behandeld in plaats van in iTOP. Indien het van een incident niet duidelijk is of het een software bug betreft of dat er een andere oorzaak is, wordt deze in iTOP geregistreerd. Hierdoor kan het voorkomen dat na analyse en diagnose van het incident blijkt dat het een software bug betreft, vandaar dat in figuur 62 onderscheid gemaakt wordt tussen de twee activiteiten voor change management. Daarnaast worden security incidenten altijd ook in iTOP geregistreerd, ook al zijn het bugs. Dit om een compleet overzicht van de security incidenten te hebben.

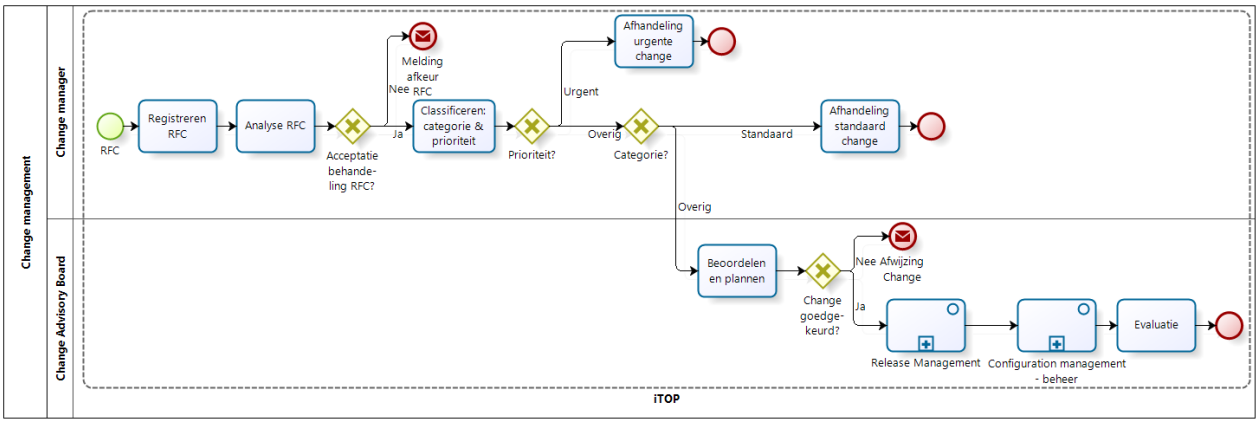


Figuur 62: Combinatie III - incident management

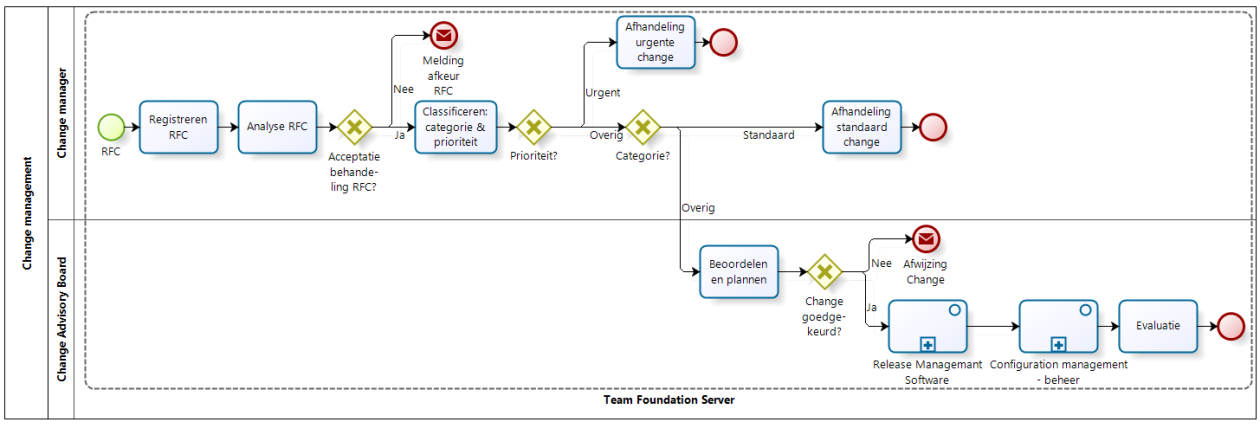


Figuur 63: Combinatie III - afhandeling bugs

Omdat bij de derde combinatie bugs niet als incident in iTOP geregistreerd worden, maar als bug in Team Foundation Server, is hiervoor een apart proces opgesteld. Dit proces is weergegeven in figuur 63. Omdat alleen duidelijke bugs geregistreerd worden is geen proces voor problem management opgenomen. Daarnaast kan bij nadere analyse blijken dat het incident toch niet veroorzaakt is door een softwarebug, deze wordt dan in iTOP geregistreerd. De change management procedures zijn bij combinatie III hetzelfde als bij combinatie II. Daarnaast is het tweede deel van het proces gelijk aan het change management proces. Dit omdat het oplossen van een bug naast het oplossen van een incident ook het uitvoeren van een wijzigingsverzoek is.



Figuur 64: Combinatie III - change management overig



Figuur 65: Combinatie III - change management software

Voordelen en nadelen

Hieronder worden de voor- en nadelen van de drie combinaties aangegeven.

Combinatie I:

Voordelen:

Alle incidenten en wijzigingsverzoeken worden in een centraal systeem geregistreerd. Dit maakt analyses en audits makkelijker. Ook worden alle wijzigingsverzoeken in een applicatie beoordeeld.

Bij invoering van combinatie I is het niet nodig om aan het projecttemplate van Team Foundation Server een wijziging door te voeren. Omdat de inrichting van iTOP niet verschilt tussen de drie combinaties is bij deze combinatie minder tijd nodig voor de inrichting van applicaties.

Nadelen:

Er is geen directe koppeling tussen een wijzigingsverzoek in iTOP en de taken en wijzigingen aan de code in Team Foundation Server.

Dubbele invoer van software bugs, deze worden eerst in iTOP geregistreerd en daarna in Team Foundation Server.

Voor de analyse en beoordeling van wijzigingsverzoeken worden deze vaak opgesplitst in deeltaken. Bij invoer in iTOP moeten deze na goedkeuring nogmaals ingevoerd worden in Team Foundation Server.

Combinatie II:

Voordelen:

Alle incidenten worden in een systeem geregistreerd, zodat een goede audit en analyse van incidenten mogelijk is.

Er is een directe koppeling tussen een wijzigingsverzoek, taken en code wijzigingen.

Bij opsplitsing van wijzigingsverzoeken in deeltaken zijn deze taken direct aangemaakt in Team Foundation Server, waardoor dubbel werk voorkomen wordt.

Nadeel:

Beoordeling van wijzigingsverzoeken vindt plaats in twee verschillende systemen.

Aanpassing van het projecttemplate van Team Foundation server is noodzakelijk voor het gebruik van RFC's

Dubbele invoer van software bugs, deze worden eerst in iTOP geregistreerd en daarna in Team Foundation Server.

Combinatie III:

Voordeel:

Er is een directe koppeling tussen een wijzigingsverzoek, taken en code wijzigingen.

Bij opsplitsing van wijzigingsverzoeken in deeltaken zijn deze taken direct aangemaakt in Team Foundation Server, waardoor dubbel werk voorkomen wordt.

Er is geen dubbele invoer van bugs.

Nadeel:

Incidenten zijn vastgelegd in twee systemen, waardoor een analyse van alle incidenten en audits lastiger zijn.

Bij onduidelijke incidenten kan het incident tussen iTOP en Team Foundation Server heen en weer kaatsen.

Beoordeling van wijzigingsverzoeken vindt plaats in twee verschillende systemen.

Aanpassing van het projecttemplate van Team Foundation server is noodzakelijk voor het gebruik van RFC's

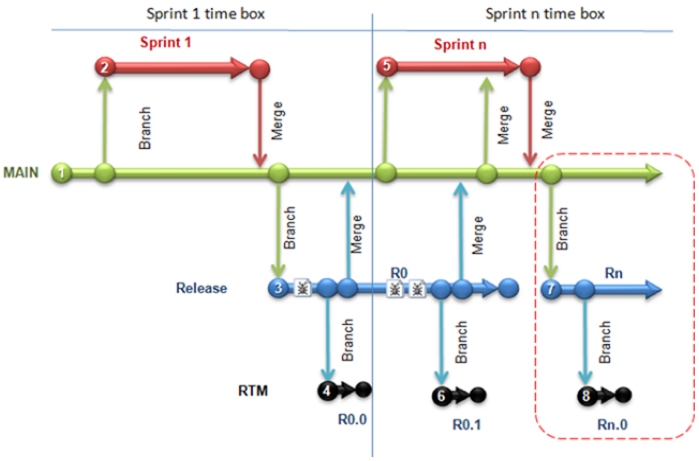
Conclusie

Voor het oplossen van de discrepanties m.b.t. de controlegerichte processen maakt het niet uit welke van de drie alternatieven gekozen wordt, in alle gevallen vindt de gewenste formalisering plaats. Op basis van de inschatting dat bij binnenkomst van incidenten in de meeste gevallen een duidelijk onderscheid te maken is tussen software gerelateerde incidenten en overige incidenten lijkt mij de derde combinatie echter het meest werkbare voor Empirion, voornamelijk door het voorkomen van dubbel invoerwerk. Zeker omdat binnen Empirion een vrij strikte scheiding aanwezig is tussen de verantwoordelijke medewerkers voor de software en en de infrastructuur. De directe koppeling tussen wijzigingsverzoeken en aanpassingen aan de code weegt in mijn ogen ook zwaarder dan het nadeel dat beoordeling in twee verschillende systemen dient plaats te vinden.

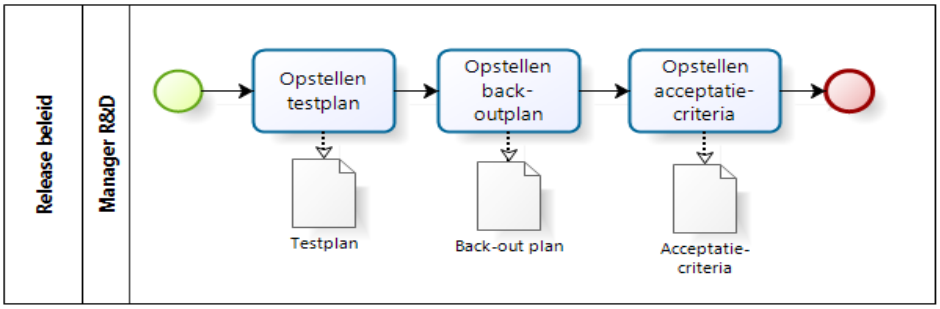
4.1.1 Release management

Alternatief 1

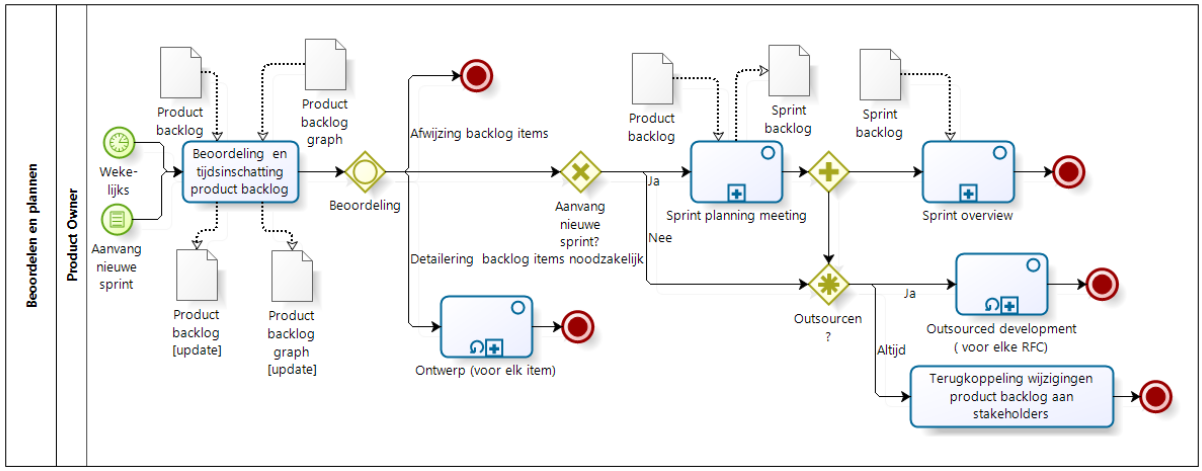
Het eerste alternatief voor release management is voornamelijk gebaseerd op de ontwikkel-methodiek Scrum in combinatie met onderdelen uit RUP en de gewenste processen voor release management op basis van ITIL en de SDLC, zoals deze in het derde hoofdstuk is weergegeven. Bij dit alternatief is voor branching gebruik gemaakt van de strategie volgens figuur 66, conform [Hinshelwood, M. 2011]. Naast onderstaand schema wordt bij het uitbesteden van ontwikkeling een branch per feature aangemaakt. Gezien de verminderde controle op uitbesteedt werk en het daarmee samenhangende grotere risico's is het niet verstandig deze code direct in de sprintbranch op te nemen.



Figuur 66: Scrum branching

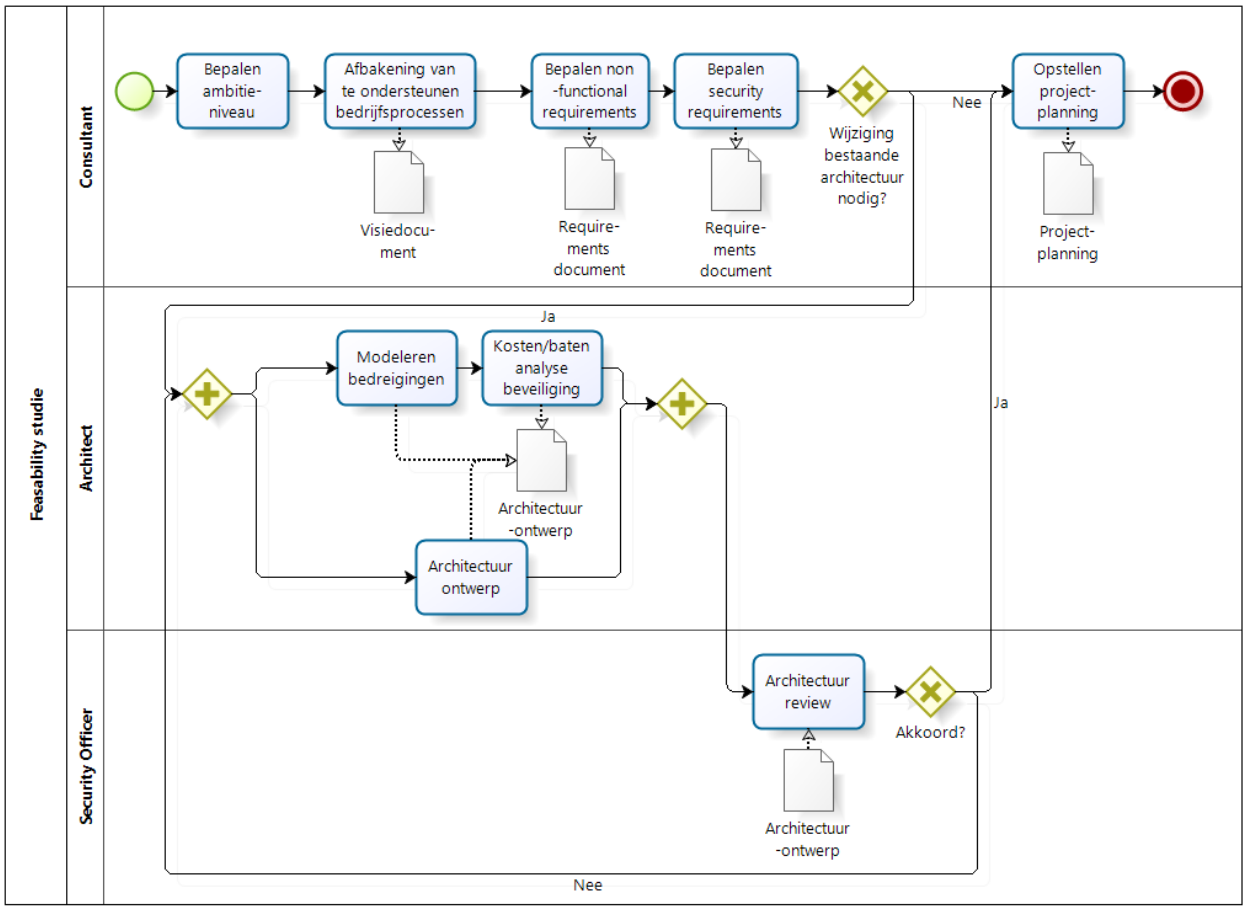


Figuur 67: Release beleid



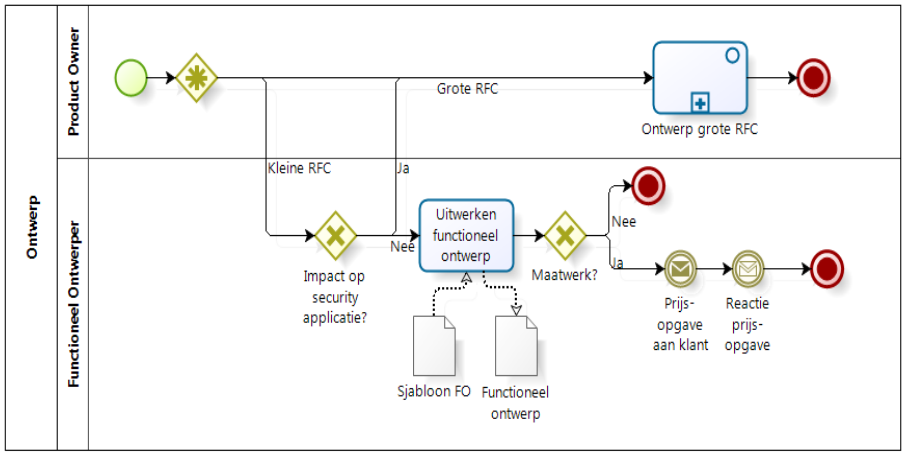
Figuur 68: Beoordelen en plannen

Het proces beoordelen en plannen (figuur 68) is een activiteit van change management, zoals het in het eerste deel van hoofdstuk vier is beschreven. In het geval van Scrum wordt hierbij de productbacklog geprioriteerd en hierbij worden ook nieuwe change requests toegevoegd of afgewezen. Of uitwerking van het ontwerp voor de sprint noodzakelijk is, wordt ook bepaald binnen dit proces.



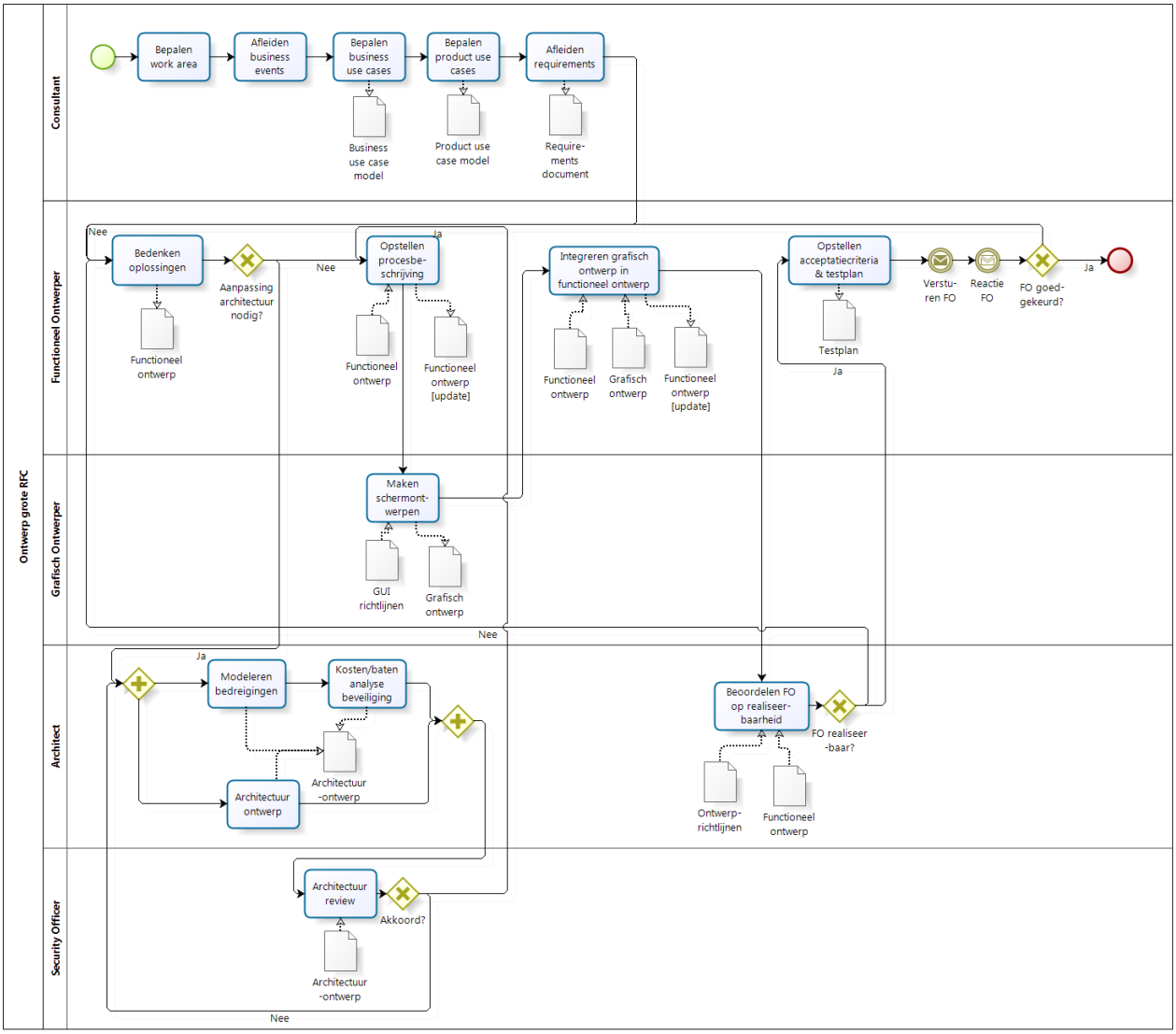
Figuur 69: Feasability studie

In figuur 69 is de feasability studie opgenomen. Naast de bestaande processen is hierbij toegevoegd dat de non-functional requirements en vooral ook de specifieke security requirements bepaald worden bij nieuwe projecten en indien dit wijzigingen van de architectuur inhoudt, dat dan ook gekeken wordt naar de security aspecten.

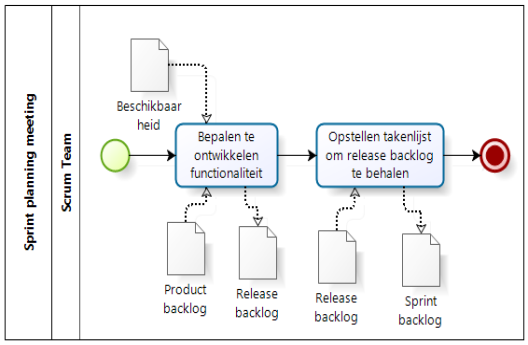


Figuur 70: Ontwerp

In figuur 70 en 71 is het ontwerp proces opgenomen. In tegenstelling tot de filosofie van Scrum is het bij Empirion vaak noodzakelijk dat ontwerpen voor de sprint volledig uitgewerkt worden, omdat klanten hier vooraf akkoord op willen geven. Om deze reden is het opstellen van het functionele ontwerp buiten de sprint gehouden. Indien de oplossing niet past binnen de kaders van de huidige architectuur, wordt dit tevens getoetst op security aspecten. Het opstellen van een technisch ontwerp is niet apart opgenomen in het proces, omdat hierop geen akkoord gegeven hoeft te worden door de klant. Binnen de sprint wordt bepaalt hoe de functionaliteit technisch gerealiseerd wordt, o.a. tijdens de sprint planning meeting bij het opstellen van de takenlijst. (figuur 72)



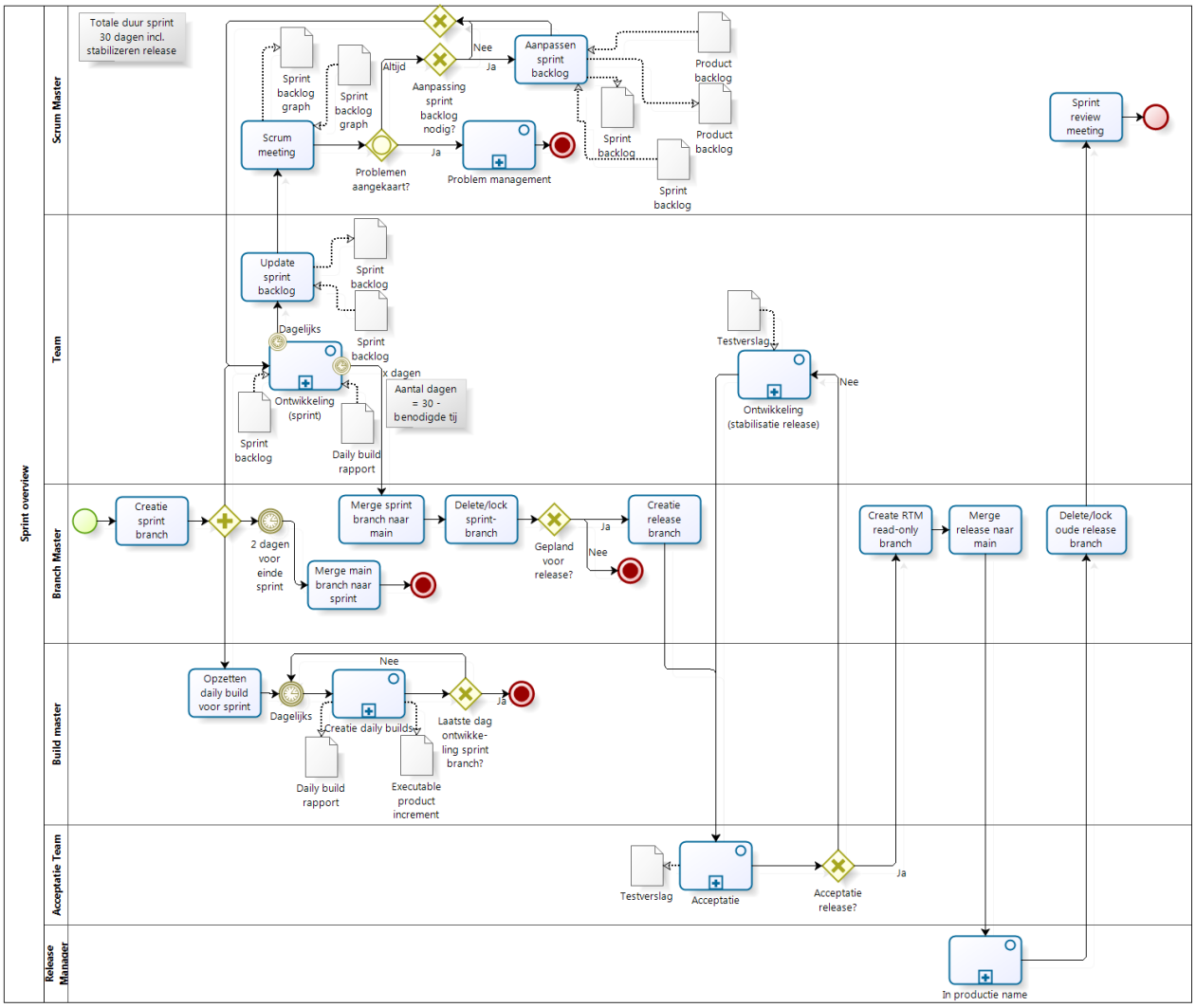
Figuur 71: Ontwerp grote RFC



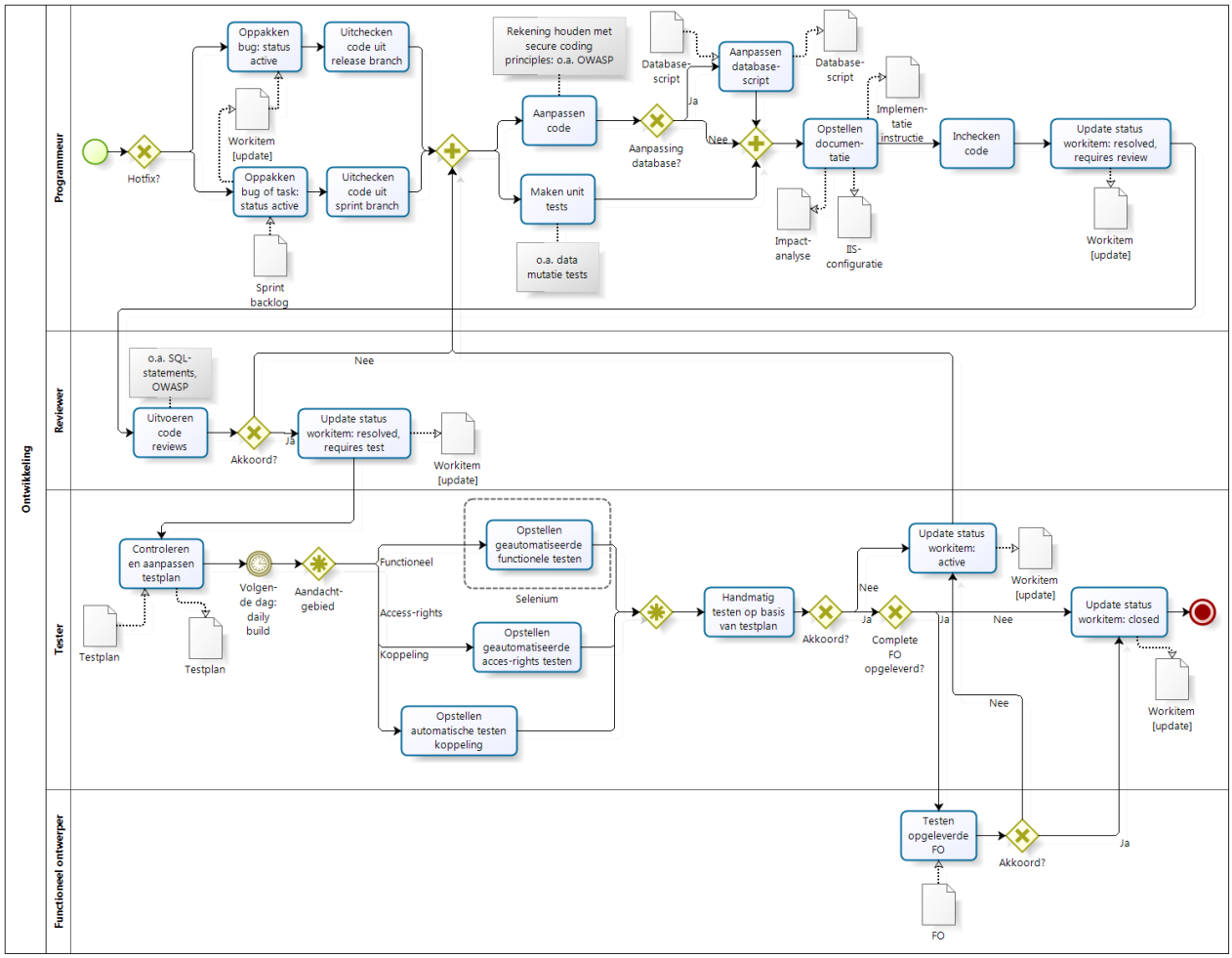
Figuur 72: Sprint planning meeting

Tijdens het eerste deel van de sprint planning meeting wordt de te ontwikkelen functionaliteit voor de komende sprint van 30 dagen vastgesteld, hierbij zijn de Product Owner, het Scrum Team en de Scrum Master aanwezig. Gedurende het tweede deel wordt de functionaliteit opgedeeld in deeltaken met een duur tussen vier en zestien uur, deze deeltaken vormen de sprint backlog. Eventueel wordt een deel van de werkzaamheden later opgesplitst, omdat hier tijdens de sprint planning meeting nog onvoldoende informatie over is. Bij het tweede deel zijn de Scrum Master en het Scrum Team aanwezig.

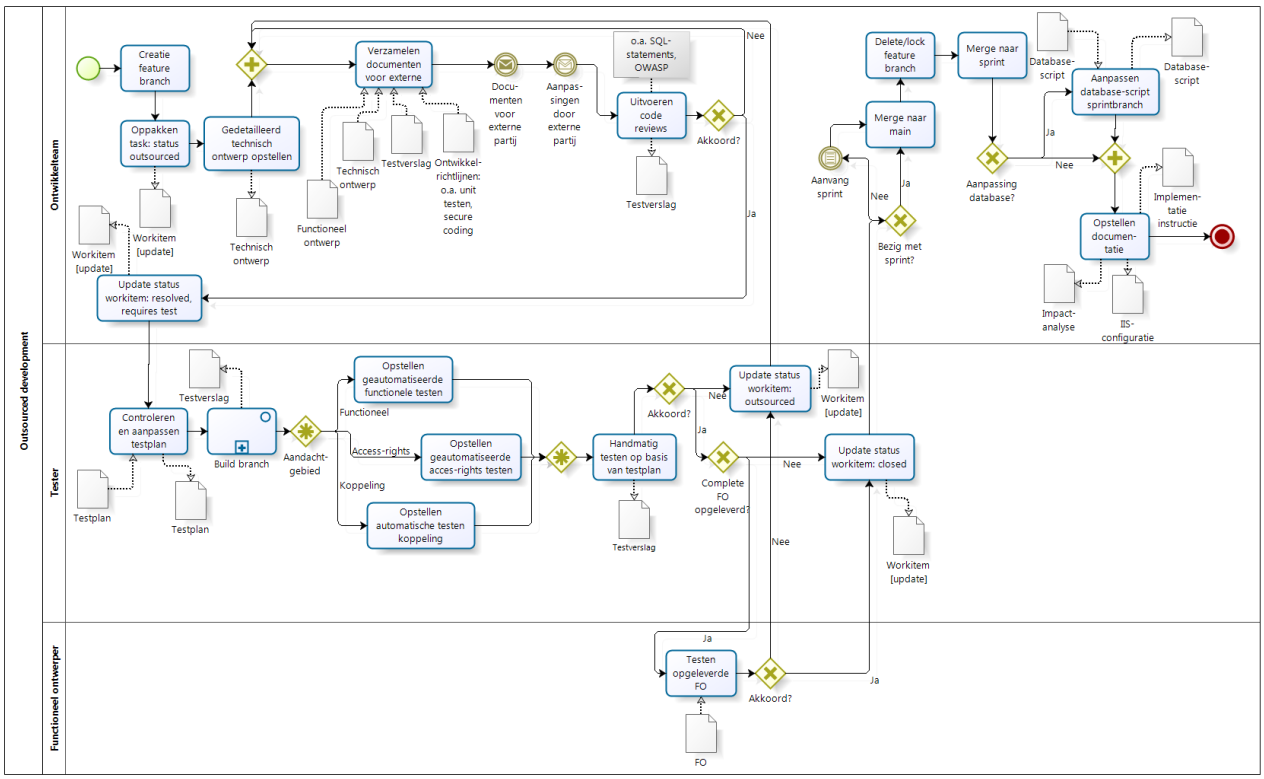
Binnen de sprint overview (figuur 73) wordt een onderscheid gemaakt tussen een ontwikkelperiode en een stabilisatieperiode, zoals ook in [Hinshelwood, M. 2011] is aangegeven. Daarnaast wordt er gebruik gemaakt van “daily builds” om directe feedback te geven aan de ontwikkelaars.



Figuur 73: Sprint overview

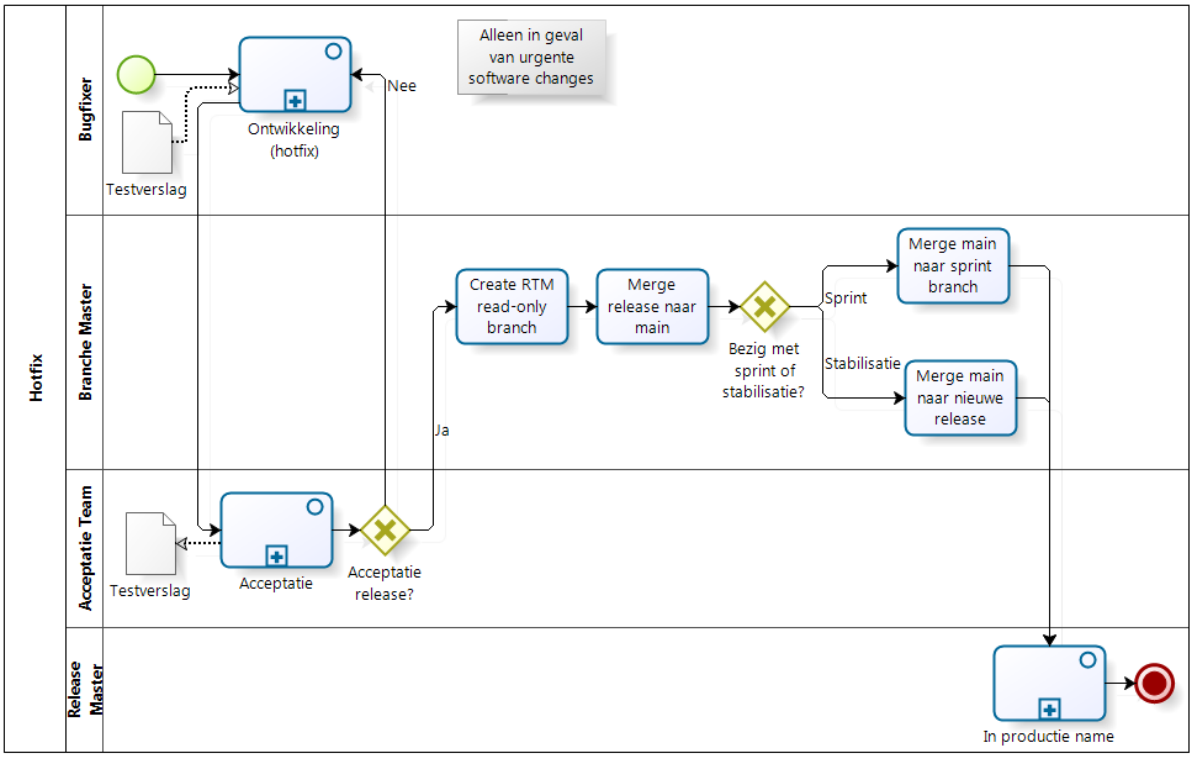


Figuur 74: Ontwikkeling

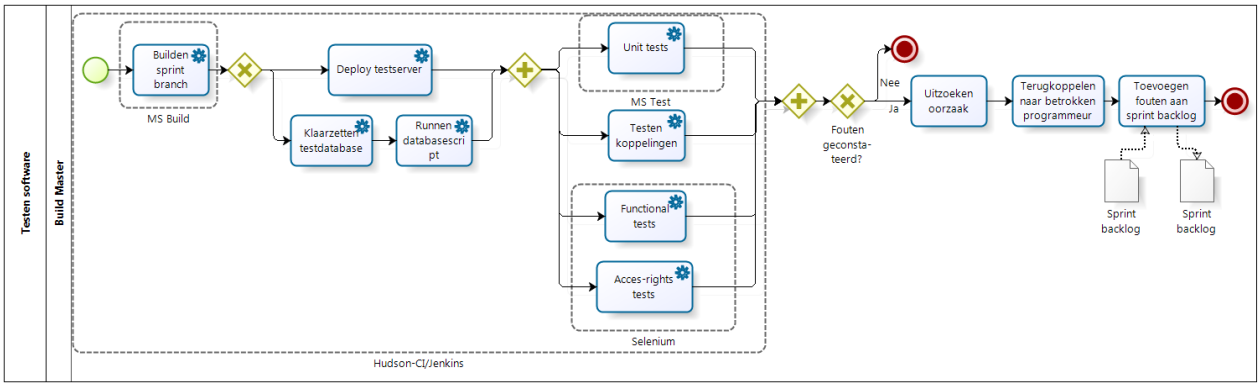


Figuur 75: Outsourced development

Bij het outsourcen van ontwikkeling (figuur 75) wordt in tegenstelling tot bij de eigen ontwikkeling wel een volledig technisch ontwerp uitgeschreven, daarnaast worden deze wijzigingen in aparte branches bijgehouden. Pas na volledige oplevering wordt dit omgezet naar de dan lopende sprint. Tijdens de stabilisatieperiode wordt een wijziging niet opgenomen in een release, zodat de deadline van de sprint niet in gevaar komt. In dat geval wordt de wijziging pas bij de volgende sprint meegenomen.

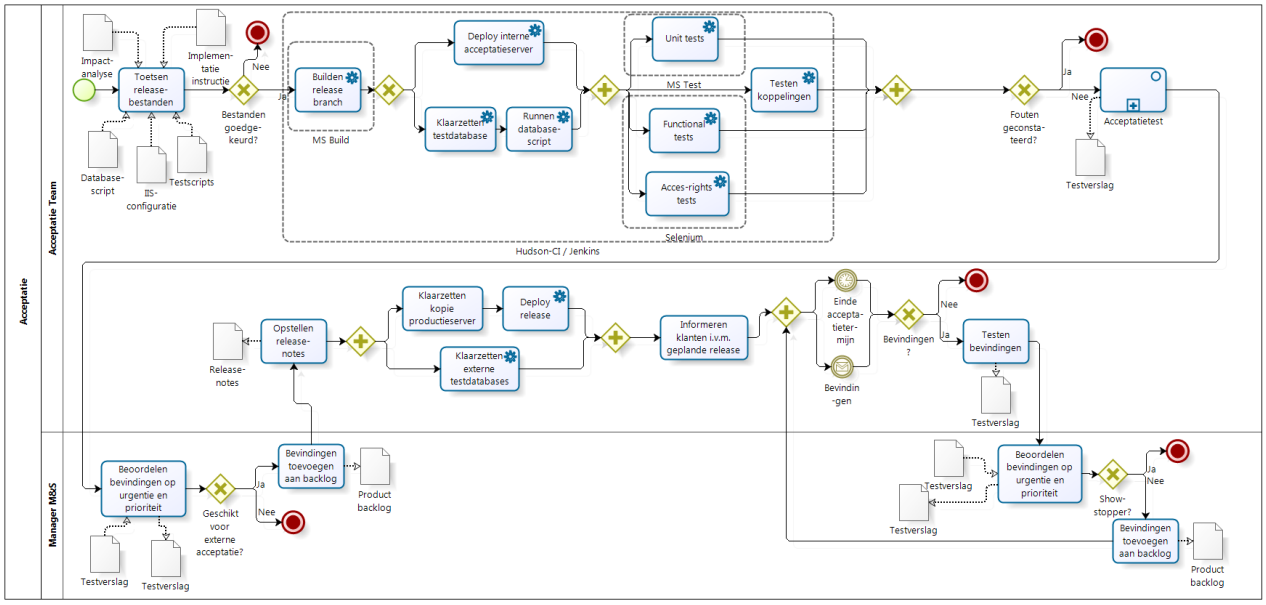


Figuur 76: Hotfix

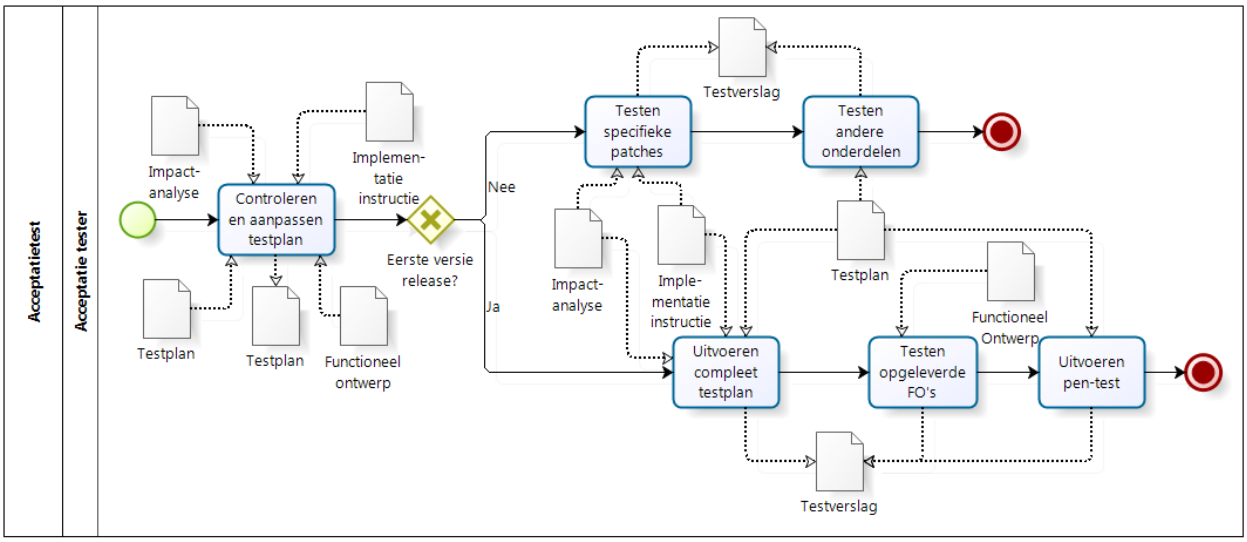


Figuur 77: Daily builds

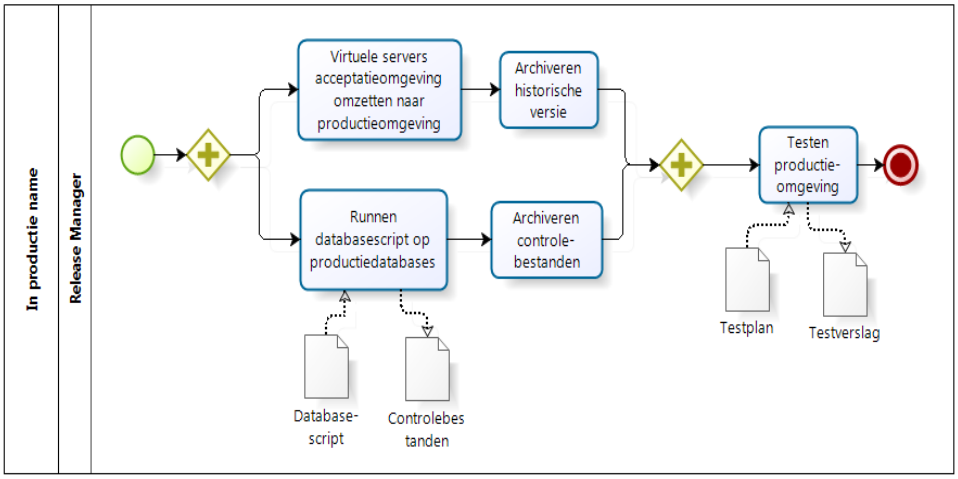
In figuur 77 zijn de “daily builds” weergegeven. Hierbij wordt elke nacht een nieuwe versie van de software gebuild, zodat er directe feedback plaats vindt naar de ontwikkelaars. Hierbij is het van belang dat de software automatisch getest wordt, om zinvolle feedback te geven. Door deze automatische tests wordt gecontroleerd of bestaande functionaliteit niet omvalt door uitgevoerde wijzigingen. Het opstellen van deze tests vindt plaats tijdens het ontwikkelen, zoals ook is weergegeven in figuur 70. Voor het opstellen van functionele testen kan Selenium gebruikt worden (http://seleniumhq.org), dit is een geautomatiseerd web-app testing framework wat op diverse platformen onder diverse browsers draait.



Figuur 78: Acceptatie



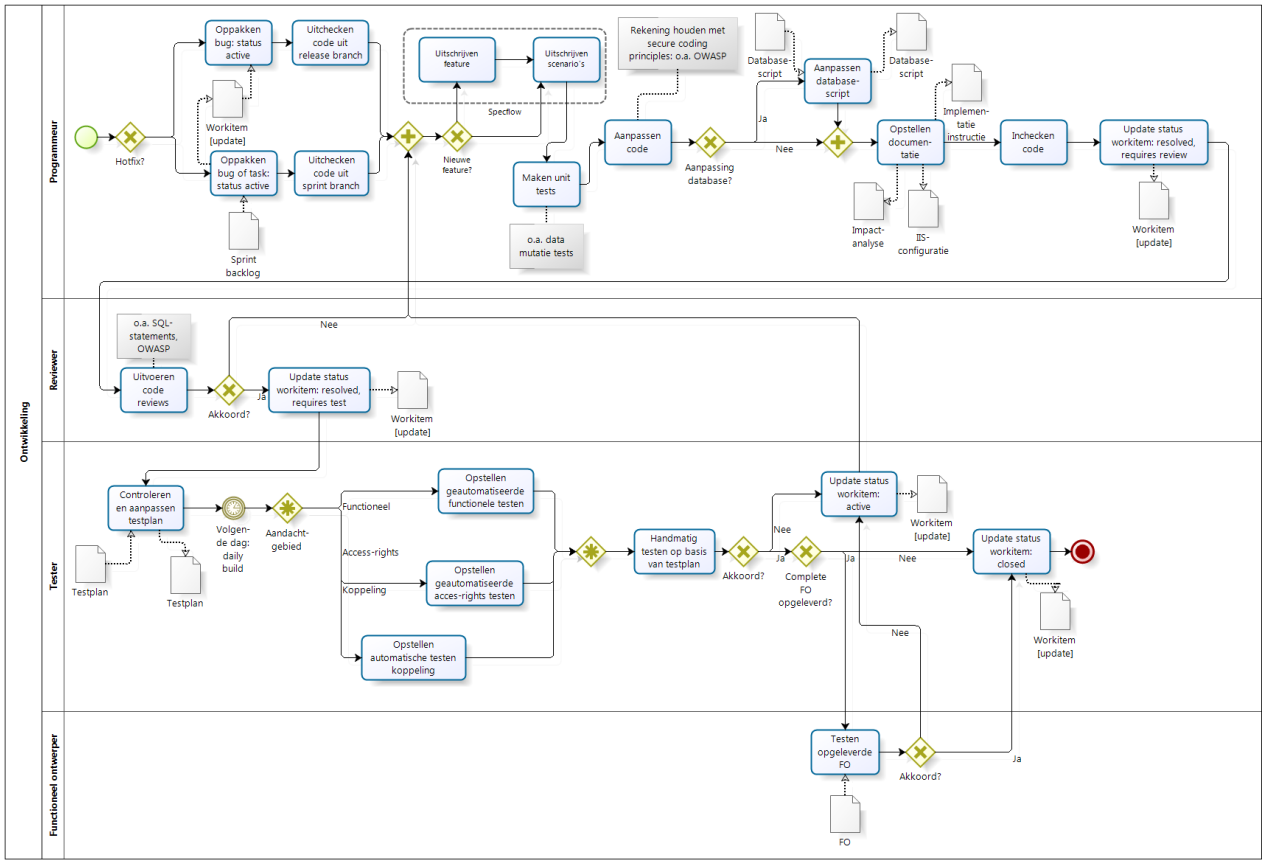
Figuur 79: Acceptatietest



Figuur 80: In productie name

Alternatief 1 – Behaviour Driven Development

Tijdens het ontwikkelen van de software is tijdens alternatief een de vrijheid bij de programmeur gelegd m.b.t. het moment waarop unit-testen worden ontwikkeld. In plaats hiervan is het ook mogelijk om structureel te werken volgens het principe van Behaviour Driven Development. Behaviour Driven Development heeft eenzelfde werkwijze als Test Driven Development. Echter worden de unit-tests niet direct uitgewerkt, maar worden deze gestructureerd door middel van features. Onder elke feature hangen een aantal scenario's op basis waarvan de daadwerkelijke test wordt samengesteld. Hierdoor wordt een directe vertaling van de functionaliteit naar de software gemaakt. Dit is weergegeven in figuur 81.



Figuur 81: Ontwikkeling - BDD

Studies hebben aangetoond dat het aantal geslaagde externe testcases bij ontwikkeling d.m.v. Test Driven Development tussen 18 en 50 procent hoger liggen dan bij ontwikkeling van code door controlegroepen zonder TDD [Crispin, L. (2006)]. Ook in een studie bij IBM is een verbetering van de defect rate met 50% behaald, met een lichte daling van de productiviteit [Maximilien, E.M., Williams, L. (2003)].

Alternatief 2

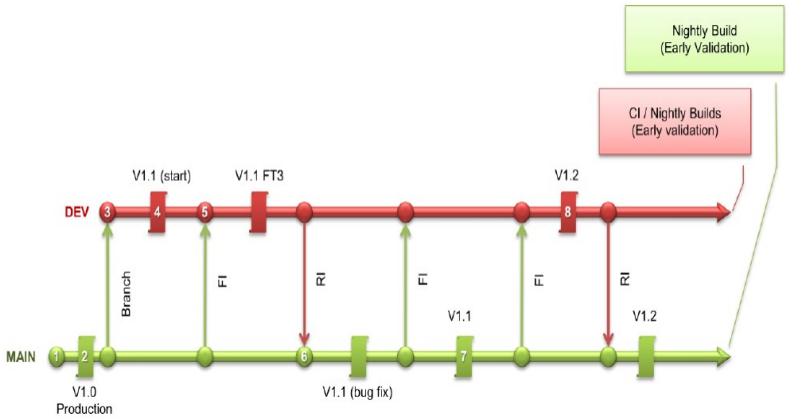
Het tweede alternatief voor release management maakt geen gebruik van Scrum en is gebaseerd op de ontwikkel-methodiek RUP in combinatie met de gewenste processen voor release management op basis van ITIL en de SDLC, zoals deze in het derde hoofdstuk zijn weergegeven. Dit alternatief sluit meer aan bij de nu gebruikte werkwijze. Wel is het dagelijks builden en automatisch testen van software aan de processen toegevoegd, zodat een betere kwaliteitsbewaking gedurende het proces plaats vindt.

Bij dit alternatief wordt voor de branching gebruik gemaakt van een development-branch, zoals

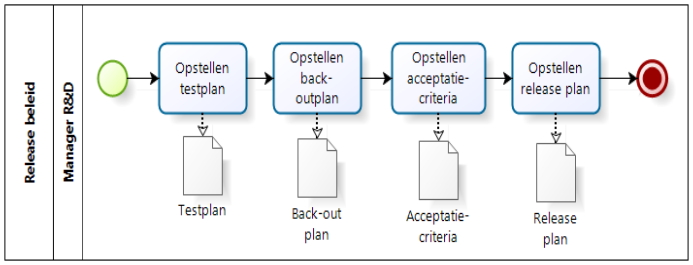
weergegeven in figuur 82. Uitbreiding met extra branches vindt plaats in geval van:

- Breaking changes
- Ontwikkeling voor toekomstige releases
- Uitbestede werkzaamheden

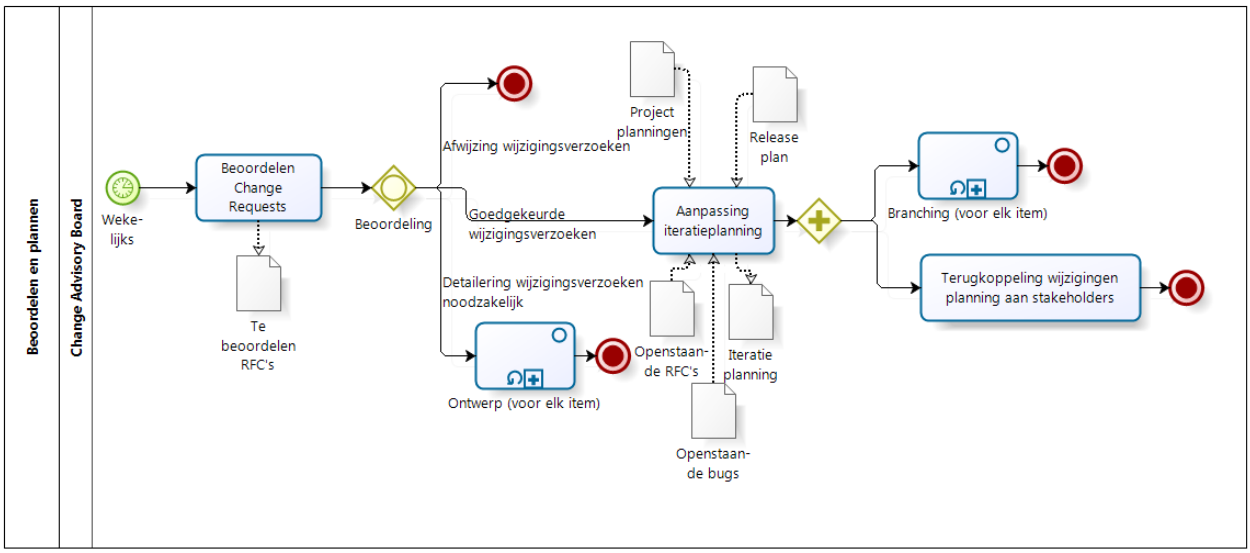
Daarnaast wordt dezelfde structuur van release-branches en RTM-branches gebruikt als bij alternatief een. Omdat het aantal benodigde builds, merge-acties en conflicten tussen verschillende branches toeneemt, als het aantal branches toeneemt is het van belang om alleen extra branches toe te voegen indien dit noodzakelijk is. Deze branching strategie is gebaseerd op [Javidi, B. et al. (2010)].



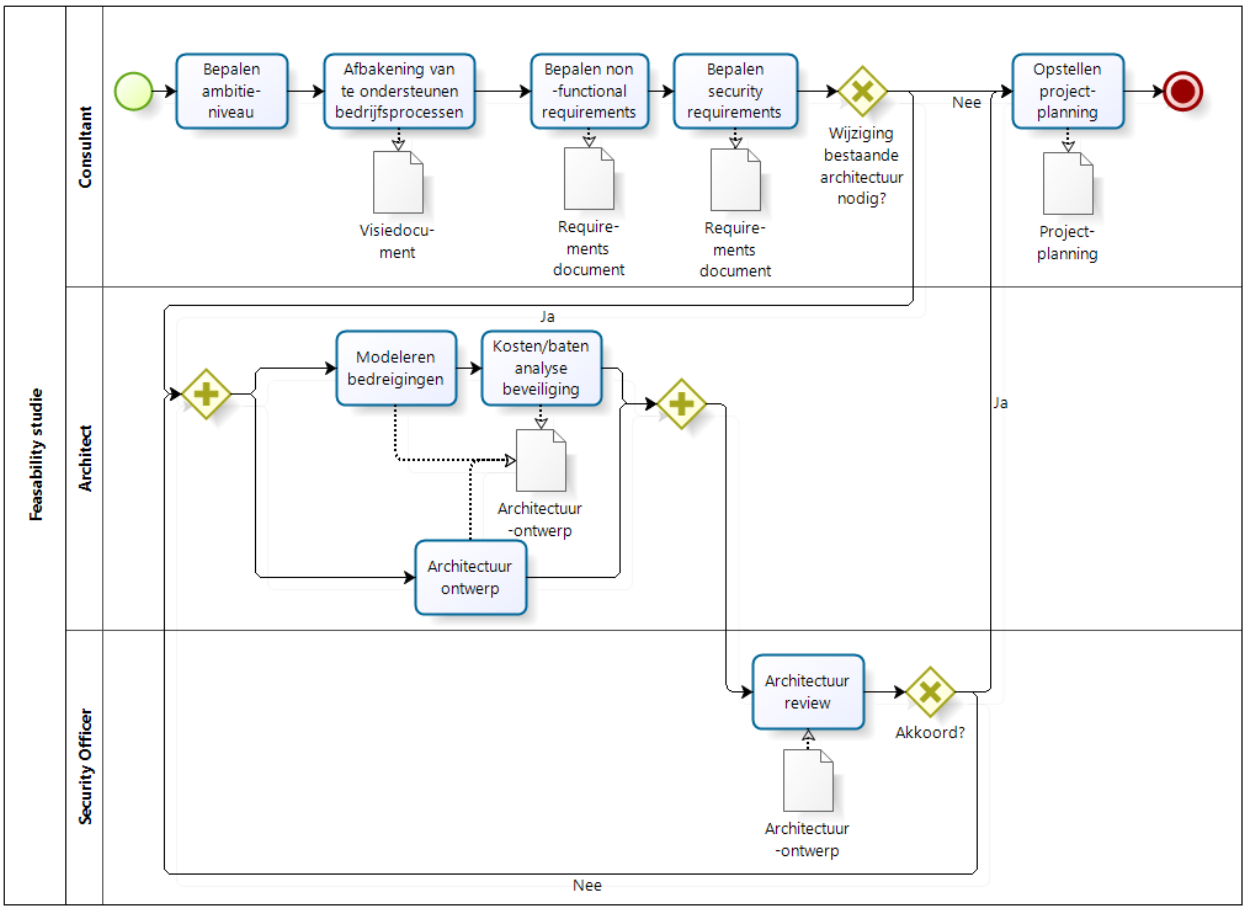
Figuur 82: Branching



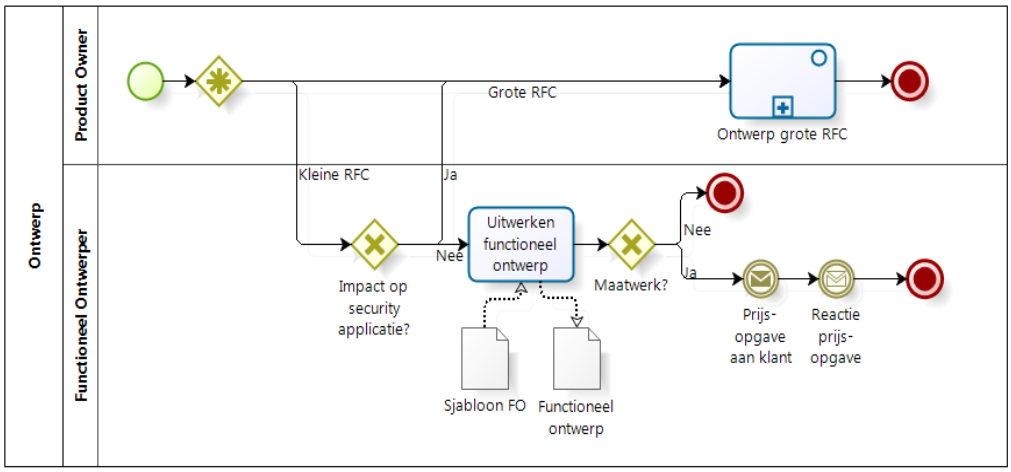
Figuur 83: Release beleid



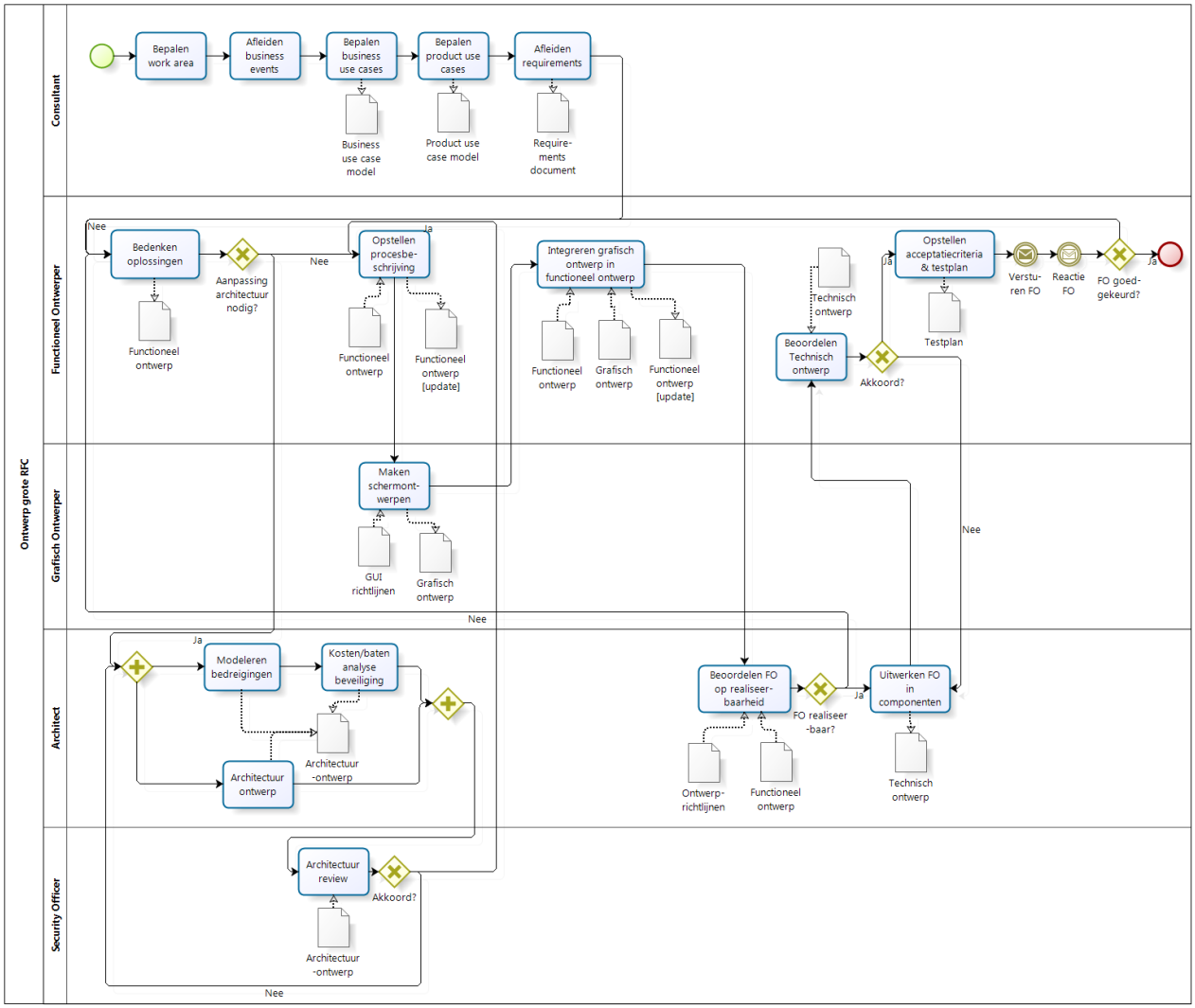
Figuur 84: Beoordelen en plannen



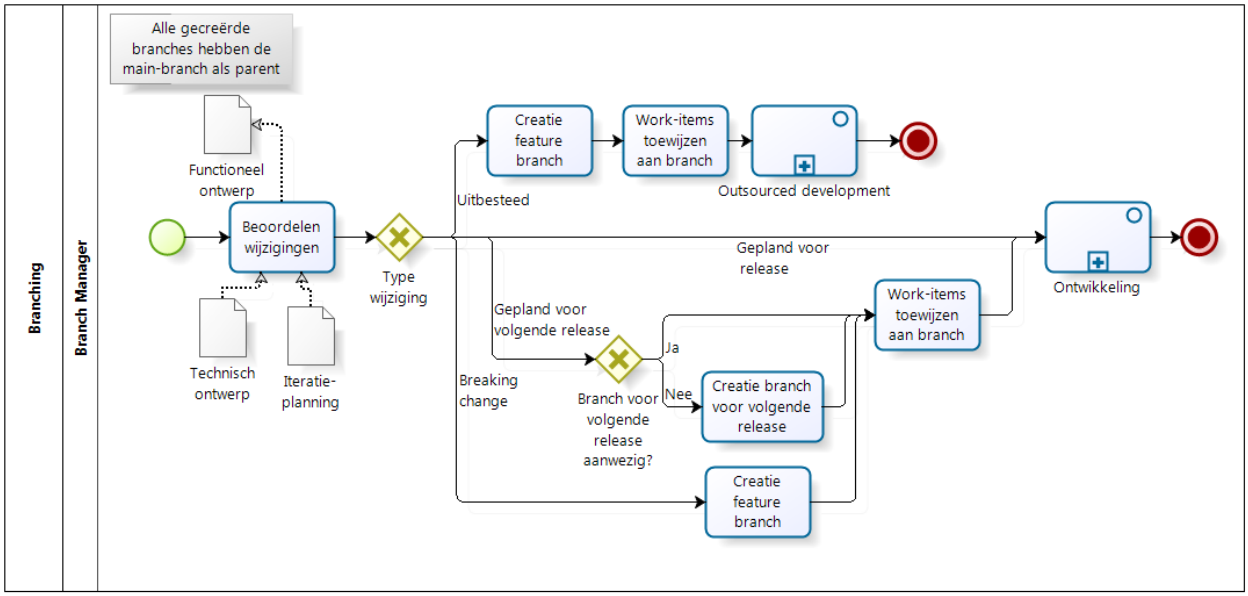
Figuur 85: Feasability studie



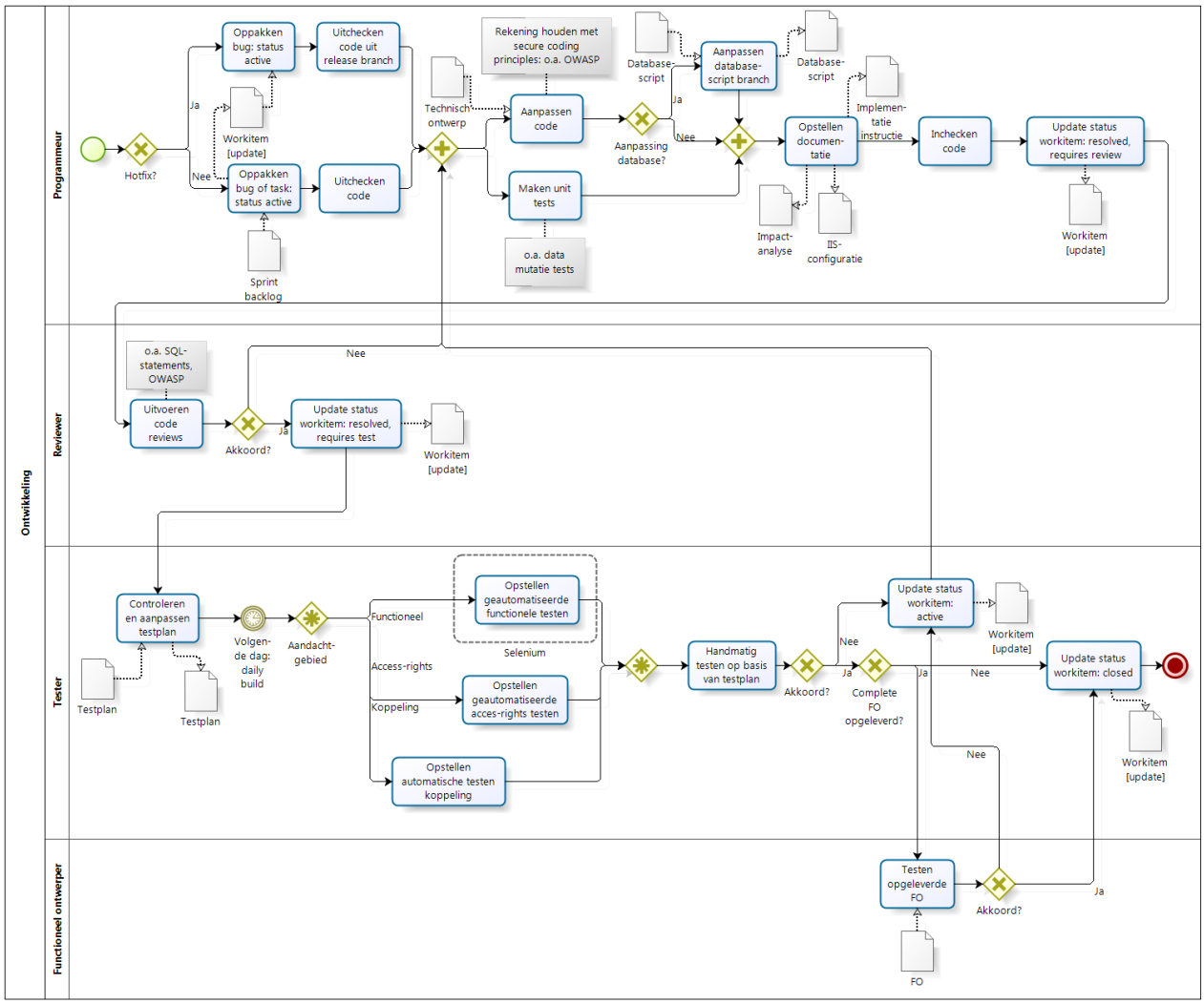
Figuur 86: Ontwerp



Figuur 87: Ontwerp grote RFC

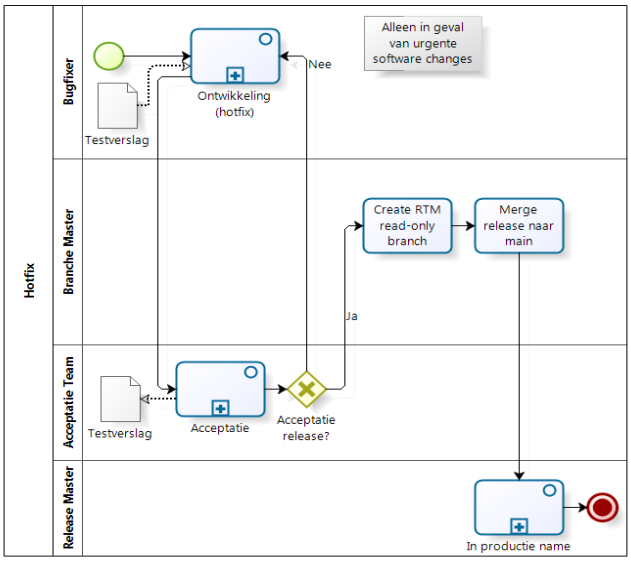


Figuur 88: Branching

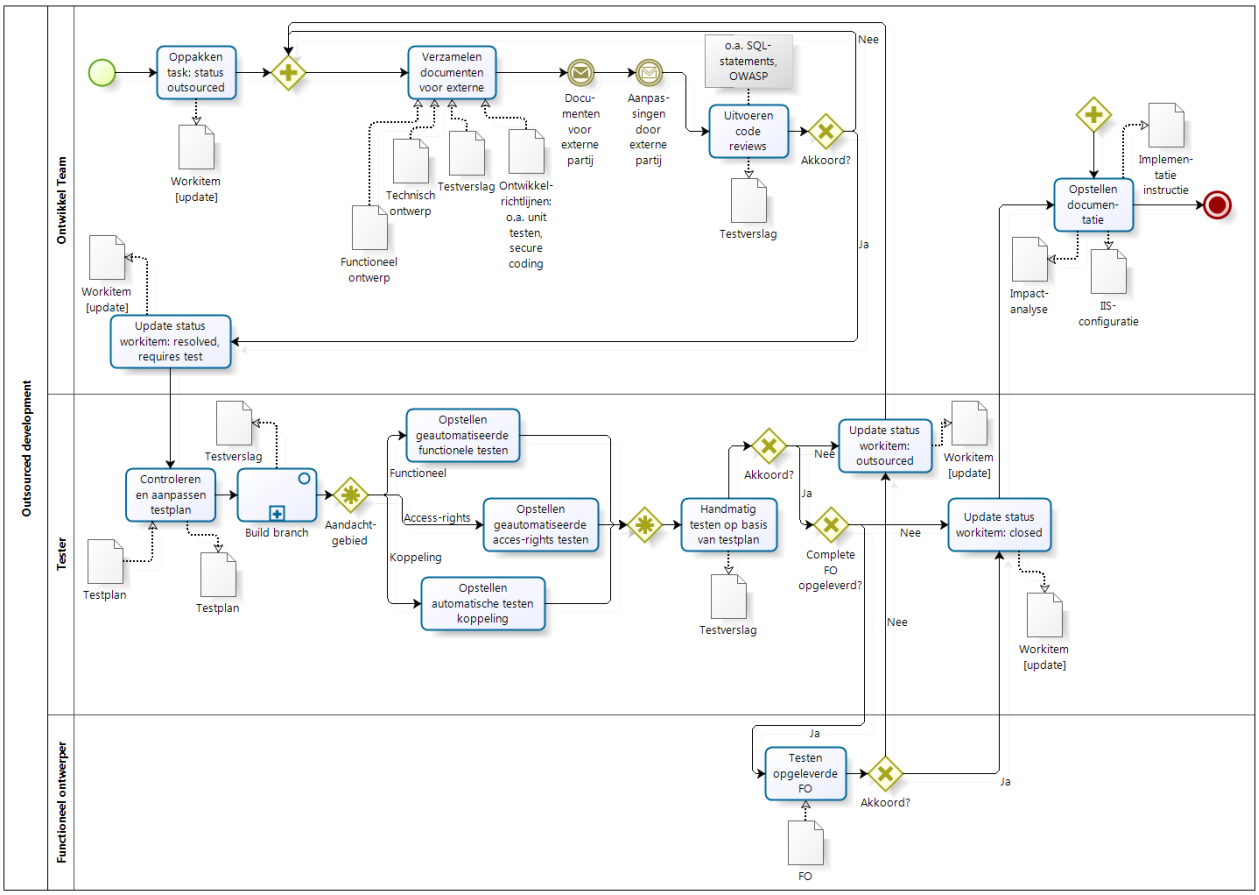


Figuur 89: Ontwikkeling

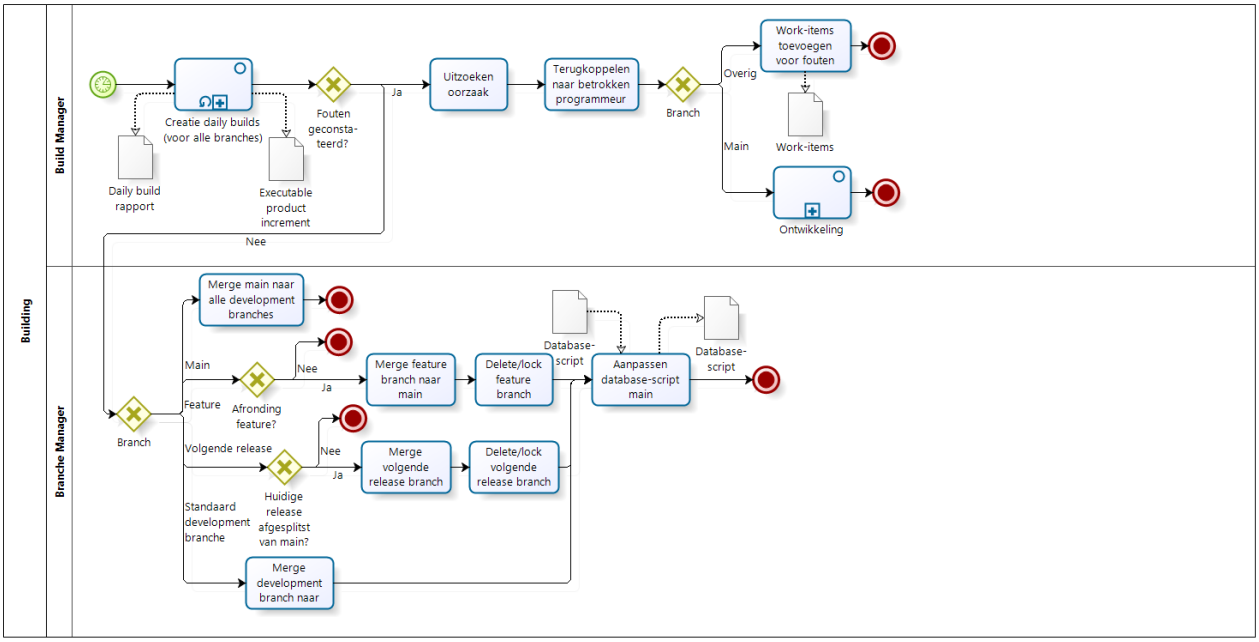
In figuur 85 is de ontwikkeling van de software weergegeven. In plaats van deze werkwijze kan ook het alternatief met BDD gebruikt worden, deze optie is niet afzonderlijk weergegeven.



Figuur 90: Hotfix



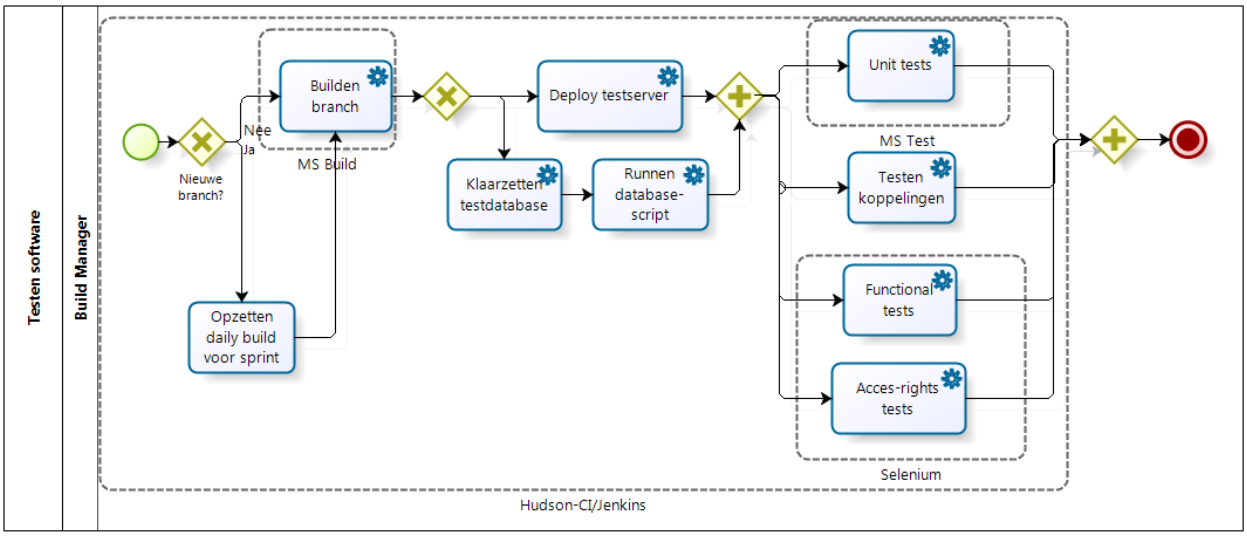
Figuur 91: Outsourced development



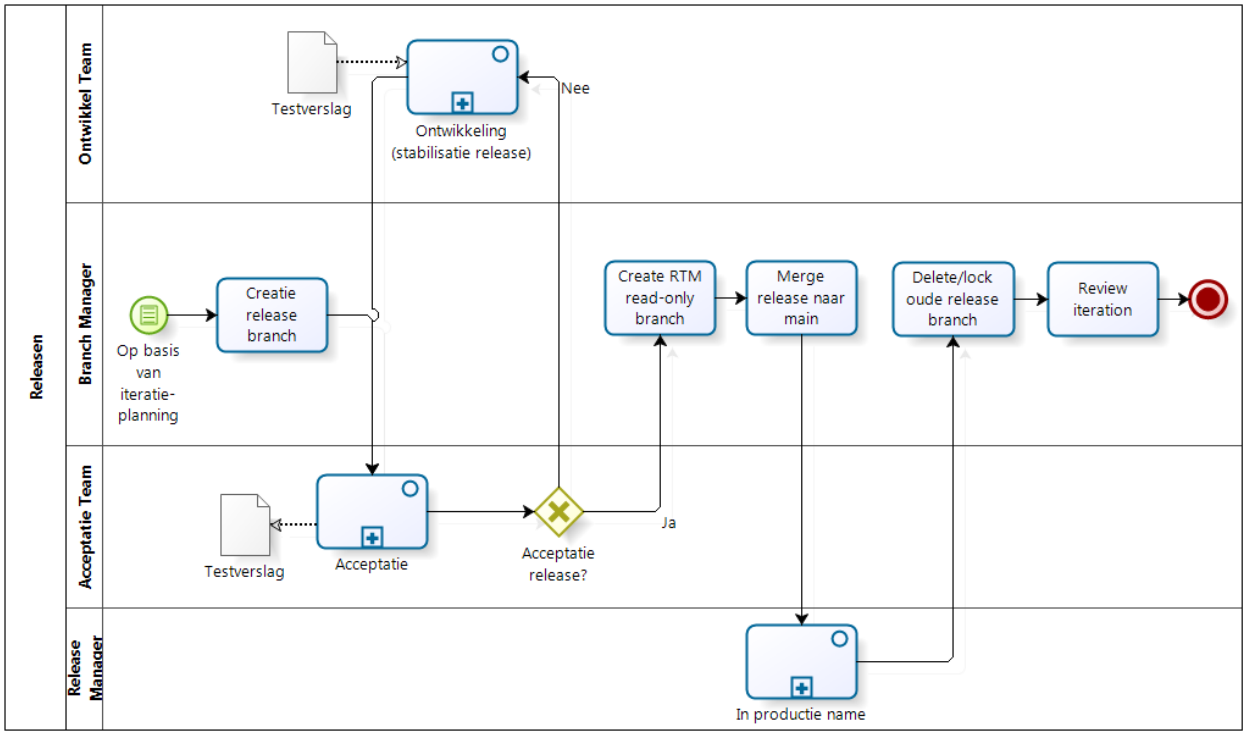
Figuur 92: Building

In figuur 88 is het build proces weergegeven. Om de kwaliteit van de main-branch te waarborgen is het van belang dat de verschillende branches gebuild en automatisch getest worden, pas na het succesvol doorlopen van alle automatische test mag een branch gemergd worden naar de mainbranch. Daarnaast wordt de mainbranch zeer regelmatig gemergd naar de development

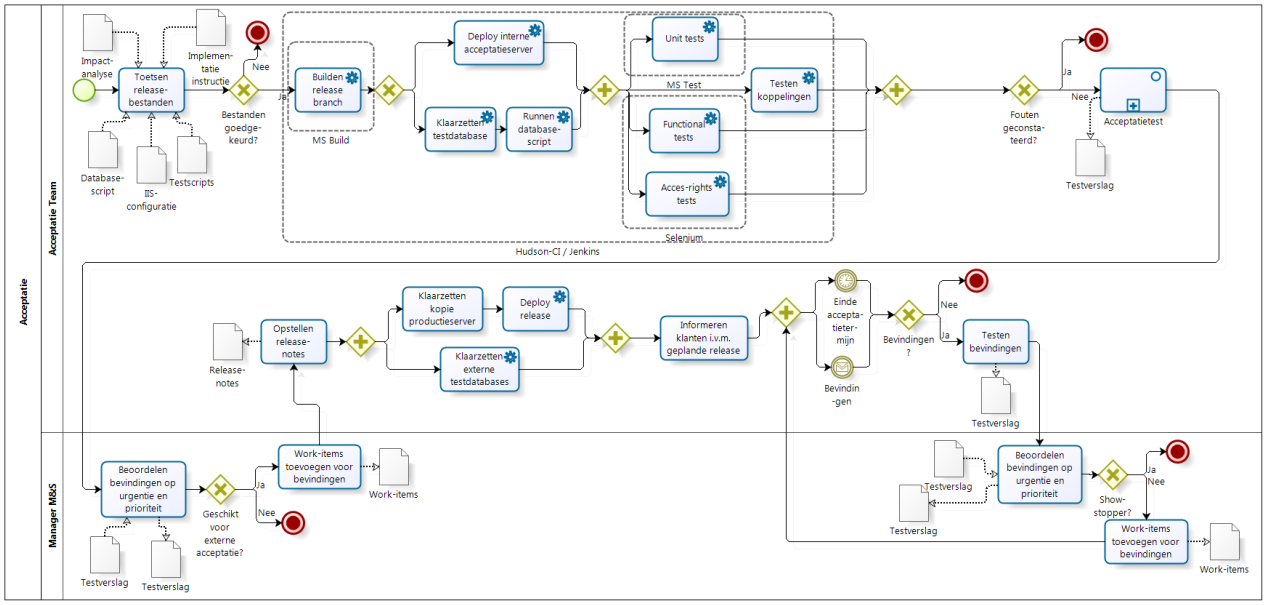
branches, dit om een “big bang” te voorkomen.



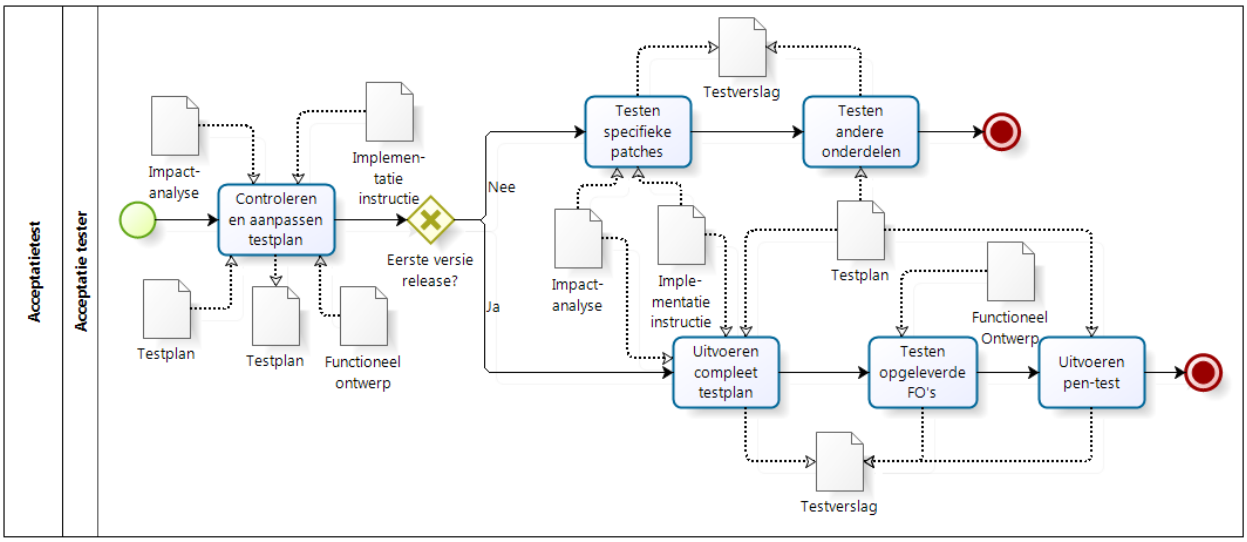
Figuur 93: Testen software



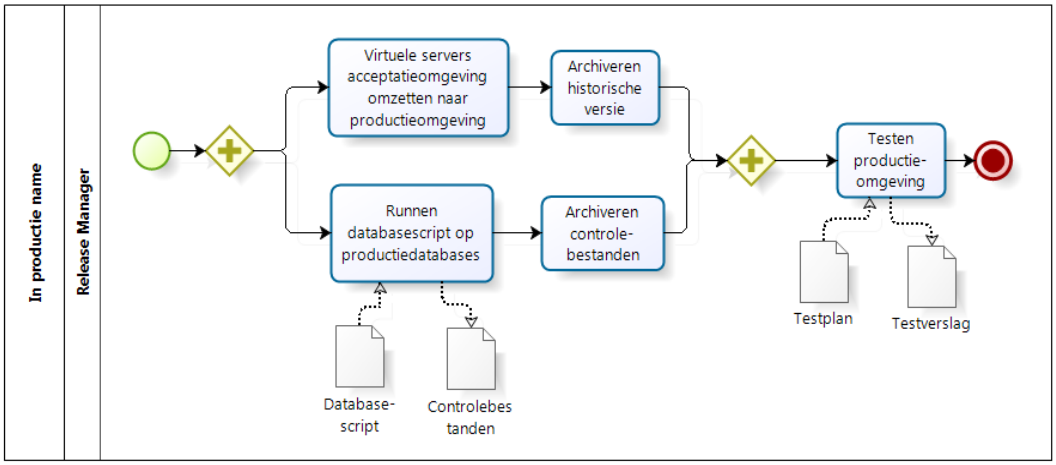
Figuur 94: Releasen



Figuur 95: Acceptatie



Figuur 96: Acceptatietest



Figuur 97: In productie name

Belangrijke verschillen tussen alternatief 1 en 2

Alternatief 1	Alternatief 2
Releaseplanning wordt bepaald duur Scrum sprint.	Releaseplanning afhankelijk van hiervoor opgesteld beleid.
Inhoud release wordt aangepast bij het niet behalen van de releaseplanning, tijdstip van release blijft gelijk.	Inhoud en/of release moment variabel, afhankelijk van beleid.
Inhoud release van te voren afgestemd in sprint planning meeting (figuur 72)	Inhoud release kan tussentijds wijzigen.
Focus van ontwikkeling enkel op komende release, daarmee minder flexibiliteit.	Ontwikkeling functionaliteit voor volgende release(s) mogelijk, hierdoor risico op verlies van focus op huidige release.
Wijze waarop gebouwd wordt, wordt bepaald tijdens de sprint planning meeting en gedurende de sprint.	Technisch ontwerp wordt opgesteld tijdens de ontwerpfase (figuur 87).
Een branch voor de ontwikkeling (m.u.v. uitbesteedde ontwikkeling).	Gebruik meerdere branches om risico's te minimaliseren.
Maar een build en weinig merge-acties.	Extra builds en merge-acties. De merge-acties kunnen echter problemen veroorzaken als de branches te ver uiteen lopen.
Dagelijks overleg d.m.v. scrum meeting.	Geen vaste overlegstructuur aanwezig.

Tabel 3: belangrijke verschillen tussen alternatief 1 en 2

Conclusie

Van de twee alternatieven is in mijn ogen alternatief I het meest geschikt om de discrepanties tussen de huidige en gewenste situatie op te lossen. Dit omdat de dagelijkse Scrum meeting een strakke voortgangsbewaking afdwingt, terwijl dit bij alternatief II niet expliciet opgenomen is. Deze voortgangsbewaking in combinatie met dagelijkse kwaliteitscontrole door middel van automatisch testen is in mijn ogen de beste methode om vertraging bij releases te voorkomen. Tevens zorgt alternatief I voor een stuk duidelijkheid en rust, omdat werkzaamheden gedurende de sprint planning meeting vastgelegd worden voor de komende periode.

Echter is het voor een korte stabilisatieperiode noodzakelijk dat de daily builds opgezet zijn en dat de basis functionaliteiten van de applicatie afgedekt worden door automatische testen, anders is voor de stabilisatieperiode niet te voorspellen welke bestaande functionaliteit geraakt is door de uitgevoerde aanpassingen en als gevolg daarvan kan de stabilisatieperiode uitlopen.

Met betrekking tot de ontwikkeling biedt in mijn ogen het alternatief van Behaviour Driven Development het meeste voordeel voor Empirion. Dit vanwege de gestructureerde opzet van de testen en de resultaten van Test Driven Development bij andere bedrijven.

4.2 Beoordeling van alternatieve oplossingen

Na het opstellen van de alternatieven is wederom een serie interviews afgenomen, hierbij zijn dezelfde personen geïnterviewd als aan het begin van het onderzoek. Tijdens deze interviews zijn voorkeuren aangegeven m.b.t. de diverse alternatieven en zijn diverse op- en aanmerkingen m.b.t. de keuzes besproken. Deze voorkeuren en de op- en aanmerkingen worden in deze paragraaf aangegeven. Ook zijn de alternatieven beoordeeld met betrekking tot de in hoofdstuk een genoemde criteria. Gezien de beschikbare tijd van de medewerkers zijn tijdens de interviews echter niet alle modellen in detail doorgesproken, maar zijn deze enkel op hoofdlijnen bekeken.

Controlegerichte processen

Bij het doorspreken van de controlegerichte processen was er een tweedeling zichtbaar tussen alternatief I en alternatief III. Alternatief I had de voorkeur van de systeem architect, een manager R&D en een functioneel ontwerper. Alternatief III had de voorkeur van de integraaltester, de manager M&S, een manager R&D en een functioneel ontwerper. Er was niemand van mening dat alternatief II de beste optie is.

De reden om voor alternatief I te kiezen was voor de voorstanders van deze optie de mogelijkheid om in een systeem een totaalplaatje te hebben van alle openstaande wijzigingsverzoeken en incidenten. Volgens de manager R&D was het nu niet goed mogelijk om een goed beeld te krijgen van alle wijzigingsverzoeken welke er nu aan zaten te komen. Door de systeem architect wordt aangegeven dat een automatische koppeling tussen iTOP en Team Foundation Server dan wel wenselijk is, zodat dubbele invoer wordt voorkomen.

De manager van M&S en de integraaltester hadden de voorkeur om voor alternatief III te kiezen, omdat dit dubbele registraties voorkomt en er volgens hem een goed onderscheid te maken is tussen software bugs en wijzigingsverzoeken en overige incidenten en wijzigingsverzoeken. De belangrijkste motivatie van de functioneel ontwerper was de directe koppeling tussen een RFC en de workitems in Team Foundation Server.

Binnen de interviews is nagegaan of door de verandering van de werkwijze het aantal te registreren incidenten en wijzigingsverzoeken een grote verhogen van de werklast zou veroorzaken voor de medewerkers van M&S. Hierbij zouden alleen het aantal database-wijzigingen wat zowel in SupportCenter Plus en in iTOP geregistreerd voor extra werk kunnen zorgen, dit zijn er +/- tien tot vijftien per week. Het aantal wijzigingsverzoeken en incidenten is maandelijks op een hand te tellen.

Voorafgaand aan de interviews zijn de drie alternatieven ook voorgelegd aan een externe consultant om deze te beoordelen met betrekking tot de ISO 27001 norm, hieruit kwam naar voren dat de drie oplossingen allemaal voldoen aan de eisen, mits deze goed geïmplementeerd worden.

Met betrekking tot de beoordelingscriteria worden in de interviews de volgende punten aangegeven:

De duidelijkheid van het proces neemt in de ogen van de geïnterviewde personen erg toe, omdat het voor iedereen inzichtelijk is welke wijzigingen en incidenten open staan. Dit overzicht ontbreekt nu bij een groot deel van de medewerkers. Hierbij geeft alternatief I een beter totaalplaatje, terwijl alternatief III een betere koppeling geeft tussen de RFC en de uitvoering er van.

Ook wordt aangegeven dat door het invoeren van de controlegerichte procedures de kennis makkelijker voorhanden is.

Release management

Binnen de interviews zijn de twee alternatieven m.b.t. release management doorgesproken. Hierbij had elke geïnterviewde de voorkeur voor alternatief I (Scrum) met uitzondering van een

programmeur. De motivatie om de voorkeur uit te spreken voor alternatief I was de controle, duidelijkheid en de rust binnen de organisatie die dit alternatief biedt boven het tweede alternatief. Daarnaast werd aangegeven dat de huidige, flexibele, situatie ongewenst is:

Manager M&S: Algehele rust en duidelijkheid, iedereen weet waar hij aan toe is, de huidige flexibiliteit zorgt alleen maar voor problemen.

Manager R&D: De huidige situatie is ongewenst, de tijdsplanning is incident gedreven, voortgang wordt in ieder geval in de gaten gehouden.

Ook werd de teamverantwoordelijkheid binnen Scrum als positief punt aangegeven tijdens de interviews.

Echter werd door de integraaltester ook de kanttekening geplaatst dat de invoering van alternatief I momenteel nog niet mogelijk is, omdat het behalen van de deadline in de stabilisatiefase van een release nog te onzeker is. Dit omdat momenteel de meeste aanpassingen aan het eind van het traject getest worden en er geen geautomatiseerde testen zijn.

De reden dat de geïnterviewde programmeur de voorkeur had voor alternatief II was de verwachting dat dit alternatief beter uitvoerbaar was binnen Empirion, waarbij wel aangegeven werd dat er een risico aanwezig is dat de planning dan niet strak genoeg gehanteerd wordt. Daarnaast is er in zijn ogen in de stabilisatiefase niet genoeg werk voor, waardoor het compleet na elkaar uitvoeren van de sprints productiviteitsverlies oplevert.

Met betrekking tot de dagelijkse scrum-meetings wordt aangegeven dat de tot op heden uitgevoerde scrum-meetings vaak ongestructureerd verliepen of overgeslagen werden en deze niet meer dan twee weken standgehouden hebben. Zonder goede scrum-master verloopt dit waarschijnlijk niet succesvol.

Naast deze op- en aanmerkingen zijn specifiek de volgende criteria beoordeeld tijdens de interviews:

De mate waarin geplande deadlines gehaald worden: In de interviews wordt over het algemeen aangegeven dat alternatief I hier een groot voordeel biedt. Alternatief II biedt te veel vrijheid om de deadline te verschuiven.

De mate waarin het proces tot een voorspelbaar resultaat leidt:

De mate waarin het proces beheersbaar is: Bij beide alternatieven is het proces beheersbaarder dan in de huidige situatie, omdat de software continu getest wordt. Echter biedt de dagelijkse Scrum-meeting meer controle, mits deze gedisciplineerd uitgevoerd wordt. Hierdoor wordt alternatief I beter beoordeeld op het gebied van beheersbaarheid.

De mate waarin het proces duidelijkheid biedt: Alternatief I is volgens iedereen duidelijker dan alternatief II, met name vanwege de vastgelegde werkzaamheden gedurende de sprint planning meeting.

De mate waarin tussentijdse releases mogelijk zijn: Hierop is alternatief II beter beoordeeld, omdat alternatief I hier totaal geen ruimte voor biedt i.v.m. de vaste sprint duur.

Impact van de veranderingen: Bij beide alternatieven wordt het huidige proces op veel punten gewijzigd, waarbij geen grote verschillen zijn tussen de twee alternatieven.

Ontwikkeling

Het alternatief met betrekking tot de ontwikkeling op basis van behaviour driven development is voorgelegd aan de systeem architect, programmeur, een management van research & development en aan een functioneel ontwerper.

Hierbij was bij deze personen een duidelijke voorkeur aanwezig voor ontwikkeling op basis van behaviour driven development, als motivatie kwamen onder andere de volgende punten naar voren:

- De structuur van de automatische testen is veel duidelijker en dat heeft zeker een toegevoegde waarde.
- Niet alleen eigen code is makkelijker aan te passen, maar vooral ook code van andere programmeurs.

Ook waren de geïnterviewde personen het er over eens dat de mate waarin van volledigheid m.b.t. het testen van de software bij de ontwikkelen met behulp van BDD beter gewaarborgd is dan bij het andere alternatief. Tevens vergroot dit daarmee de betrouwbaarheid van het eindproduct.

Daarnaast is aan de functioneel ontwerper voorgelegd of zijn functionele ontwerpen makkelijk om te zetten zijn in de features/scenario structuur welke bij BDD gebruikt wordt. Hij gaf aan dat dit geen enkel probleem was, omdat de structuur van zijn functionele ontwerpen hier al op aansluit.

Als kanttekeningen bij beide alternatieven werd aangegeven dat zonder strakke procedures het opstellen van unit-testen waarschijnlijk naar verloop van tijd weer gaat verwateren. Dit geldt echter voor beide alternatieven.

Algemeen

Naast de specifieke op- en aanmerkingen met betrekking tot de modellen werd in de interviews ook een aantal maal aangegeven dat medewerkers het nog niet zien gebeuren dat de voorgestelde veranderingen in dit onderzoek uitgevoerd worden. Enerzijds vanwege de drukte binnen het bedrijf, waardoor er geen tijd over is voor het doorvoeren van wijzigingen, anderzijds vanwege de managementcultuur binnen het bedrijf. Voor het succesvol doorvoeren van deze voorstellen is zowel een cultuurverandering als een mentaliteitsverandering van het gehele bedrijf noodzakelijk.

5 Conclusie en aanbevelingen

Conclusie

Aan het begin van de opdracht zijn de onderstaande twee kernproblemen geformuleerd op basis van de probleemkluwen:

- Het formele proces is niet optimaal ingericht.
- Het daadwerkelijke proces is niet optimaal ingericht.

Deze kernproblemen hebben in combinatie met de opdrachtomschrijving het volgende doel opgeleverd:

- Het opstellen van een optimaal formeel ontwikkelproces voor de ontwikkelstraat van Empirion.

Door Empirion zijn daarnaast de volgende randvoorwaarden gesteld aan het nieuwe formele proces:

- Het proces voldoet aan de eisen van ISO 27001.
- Het proces is bruikbaar in combinatie met Microsoft Team Foundation Server.

Tijdens het uitvoeren van het onderzoek is bevestigd dat de bestaande formele en daadwerkelijke bedrijfsprocessen niet optimaal ingericht zijn, zoals in hoofdstuk 3 uit een is gezet. Tevens is in het derde hoofdstuk geconcludeerd dat er een aantal discrepanties zijn tussen de huidige bedrijfsprocessen en de randvoorwaarde dat het proces moet voldoen aan de ISO 27001 norm.

In het vierde hoofdstuk zijn een drietal alternatieven opgesteld voor de controlegerichte processen en een tweetal alternatieven m.b.t. release management, tevens zijn er voor de daadwerkelijke ontwikkeling twee alternatieven opgesteld. Op basis van het raadplegen van een externe consultant kan geconcludeerd worden dat deze alternatieven voldoen aan de randvoorwaarden van de ISO 27001 norm. Tevens zijn de processen bruikbaar met Microsoft Team Foundation Server. In de interviews zijn deze alternatieven beoordeeld door een aantal medewerkers van Empirion. Uit deze interviews is gebleken dat m.b.t. de controlegerichte processen er geen eenduidige voorkeur bestaat voor een van de drie alternatieven, wel heeft alternatief II bij niemand de voorkeur. Bij de alternatieven voor release management blijkt duidelijk dat de medewerkers de voorkeur hebben voor alternatief I, op basis van Scrum. Bij de ontwikkeling zijn de ontwikkelaars van mening dat behaviour driven development de beste optie is.

Aanbevelingen

Tijdens deze opdracht zijn een aantal zaken naar voren gekomen, welke tijdens deze opdracht niet specifiek behandeld zijn. Op basis van deze punten ben ik gekomen tot onderstaande aanbevelingen:

1. Gezien de werkdruk binnen Empirion lijkt het mij noodzakelijk dat voor het oppakken en invoeren van verbeteringen een werknemer wordt vrijgemaakt of wordt aangenomen, zodat deze werknemer zich niet met klantspecifieke taken bezig hoeft te houden. Anders verwacht ik dat de druk om opdrachten uit te voeren voor klanten te groot blijft, waardoor er te weinig tijd over blijft om proceswijzigingen goed te implementeren.

2. Indien voor invoering alternatief I m.b.t. release management gekozen wordt, is een goede scrum master noodzakelijk voor een goed verloop van de daily scrum meetings. Gezien de resultaten tot op heden met deze meetings is het in mijn ogen aan te bevelen om een medewerker hiervoor goed op te leiden.

3. Tijdens de interviews zijn de alternatieven in hoofdlijnen behandeld, voor de daadwerkelijke

implementatie is het aan te bevelen om per onderdeel een implementatieplan uit te werken en door te spreken met de betrokken medewerkers, zodat hierop eventueel nog aanpassingen doorgevoerd kunnen worden.

Limitations & future work

Binnen dit verslag is geen rekening gehouden met de personele bezetting binnen Empirion en de benodigde resources met de daarbij behorende kosten voor de invoering van de voorgestelde alternatieven. Verder zijn er een aantal beperkingen m.b.t. de directe implementatie van de in dit verslag genoemde bedrijfsprocessen. Voor de invoering van de bedrijfsprocessen ontbreekt een implementatieplan, waarbij voor de controlegerichte processen minimaal de volgende onderdelen in meegenomen moeten worden:

- Uitwerking configuratiebestanden Team Foundation Server work-items
- Uitwerking configuratiebestanden iTOP datamodel
- Toewijzing van rollen aan personen

Voor het invoeren van het alternatief rond release management moeten eerst geautomatiseerde testen opgesteld worden, voordat de bedrijfsprocessen volledig ingevoerd kunnen worden. Hierbij moeten de voorgestelde testtools ook beproefd worden op hun geschiktheid.

Naast dit onderzoek naar de bedrijfsprocessen van de ontwikkelstraat lijkt het mij zinvol om ook een onderzoek naar de inhoudelijke kant van het ontwikkelproces plaats te laten uitvoeren. Met name naar de mogelijkheden om de applicaties duidelijker modulair op te bouwen.

Literatuur

Heerkens, J.M.G. (2005*), De Algemene Bedrijfskundige Probleemaanpak*, Enschede: TSM Business School
White, S.A. (2004), *Introduction to BPMN*
Ryan K.L. Ko, Stephen S.G. Lee, Eng Wah Lee (2009), *Business process management (BPM) standards: a survey*, Business Process Managemen Journal, Vol. 15, No. 5
Ramsin, R., Paige, R.F. (2008), *Process-centered review of object oriented software development methodologies*, ACM Comput. Surv., Vol. 40, No. 1
Kruchten, P (1999), *The Rational Unified Process: an introduction*, Reading, Massachusetts: Addison-Wesley Longman, Inc.
Krebs, J. (2005), *RUP in the dialogue with Scrum*, http://www.ibm.com/developerworks/rational/library/feb05/krebs/index.html
Schwaber, K, Beedle, M (2002), *Agile Software Development with Scrum*, Upper Saddle River: Prentice-Hall Inc.
Takeuchi H., Nonaka I. (1986), *The New New Product Development Game*, Harvard Business Review, Vol. 64, No. 1
Flutcher, L., Solms, R. von (2008), *Guidelines for secure software development*
Jones, R.L., Rastogi, A. (2004), *Secure coding: building security into the software development life cycle*, Information security journal, Vol. 13, No. 5
ISO (2010), *ISO/IEC 27001:2005*, http://www.iso.org/iso/catalogue_detail?csnumber=42103
Bon, J. van, Veen, A. van der (2006), *Foundations of IT Service Management, op basis van ITIL*, Zaltbommel: Van Haren Publishing
Hinshelwood, M. (2011), *Guidance: A Branching strategy for Scrum Teams*, http://geekswithblogs.net/hinshelm/archive/2010/04/14/guidance-a-branching-strategy-for-scrum-teams.aspx
Crispin, L. (2006), *Driving Software Quality: How Test-Driven Development Impacts Software Quality*, Software, IEEE, Vol. 23, No. 6
Maximilien, E.M., Williams, L. (2003), *Assessing Test-Driven Development at IBM*, ICSE `03 Proceedings of the 25th International Conference on Software Engineering
Javidi, B. et al. (2010), *Microsoft Visual Studio Team Foundation Server Branching Guidance 2010 Main*, Microsoft Corporation

Bedrijfsdocumentatie

Empirion 2010: Emprion, Stappenplan systeemontwikkeling, 2010, http://jupiter.empirion.local/wiki/index.php?title=Stappenplan_systeemontwikkeling
Empirion 2011a: Empirion, OTAP-werkwijze, 2011, http://jupiter.empirion.local/wiki/index.php?title=OTAP-werkwijze
Empirion 2011b: Empirion, Deployment .Net onderdelen, 2011, http://jupiter.empirion.local/wiki/index.php?title=Deployment_.Net_onderdelen

Gebruikte applicaties en modelleertalen

OMG, 2011: Object Management Group/Business Process Management Initiative, BPMN, www.bpmn.org
Bizagi, 2011: Bizagi, Bizagi Process Modeler, www.bizagi.com
Archi, 2011: Institute of Educational Cybernetics University of Bolton, Archi, archimate modelling, , archi.cetis.ac.uk
ArchiMate, 2011: The Open Group, Archimate Forum, , www.opengroup.org/archimate/index.htm

Bijlage A: BPMN Scrum