Wat kan het open source initiatief Eclipse Proces Framework (EPF) betekenen voor organisaties die bezig zijn hun software-ontwikkelprocessen te verbeteren en met de selectie van een ontwikkelproces? Dit artikel gaat in op de processen en tooling die binnen het EPF-platform beschikbaar zijn. Andere vergelijkbare software die beschikbaar is in de markt laat ik veelal buiten beschouwing.

EPF en de wereld van ontwikkelprocessen

EPF is begin 2006 ontstaan vanuit een donatie van IBM. Zij hebben delen van RUP (Rational Unified Process) en de broncode voor de software waarmee deze wordt onderhouden geschonken aan de Eclipse Foundation. Dit luidde de geboorte van EPF in. OpenUP en de proces composer vormen de twee primaire elementen welke door EPF worden geleverd. OpenUP definieert een volledig ontwikkelproces geënt op Unified Proces maar met daarin veel 'agile' aspecten toegevoegd.

In OpenUP zijn kernelementen van RUP zoals use cases, architectuur gedreven ontwerp en de iteratieve patronen van RUP in de basis verankerd. Binnen de iteratieve fases zijn echter meer agile technieken gedefinieerd om tot het gewenste resultaat te komen. Door deze opzet is OpenUP volgens mij niet altijd zomaar te implementeren binnen organisaties die de overstap willen maken naar een iteratief proces. Deze organisaties zullen waarschijnlijk een bepaalde vorm van coaching en/of verscherping nodig hebben in het toepassen van het proces. Toch is OpenUP een volledig proces wat een goede mix geeft tussen een formeel iteratief proces met daarin agile technieken die op een pragmatische en effectieve manier zijn verwerkt.

Waarom een tool voor proces definitie?

Het definiëren van processen kun je simpel gezegd in elke willekeurige tekstverwerker of HTML-tool doen. Zolang je maar in staat bent om teksten aan elkaar te koppelen en/of met elkaar in verband te brengen. Echter wanneer je gaat kijken naar elementen als onderhoud en definitie van varianten op standaard processen, dan kom je al snel tot de conclusie dat standaard tekstbestanden op de lange duur niet handig zullen zijn.

Wanneer we kijken naar organisaties die bezig zijn met procesverbetering, dan zien we uitdagingen zoals:

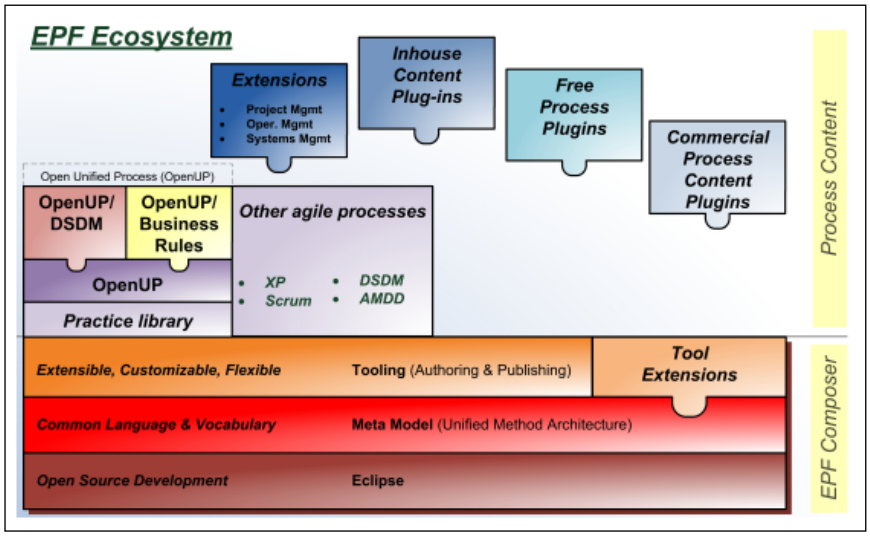
- Hoe creëren we een centrale plek voor richtlijnen?
- Waarin onderhouden we best practices die we kunnen toepassen binnen nieuwe processen?
- Hoe voorkomen het 'not invented here' syndroom?

Bij het tackelen van deze uitdagingen kunnen gespecialiseerde tools je helpen. Een ander bijkomend voordeel is dat dit soort tools de auteur forceren en helpen om in vaste structuren te denken tijdens de definitie van taken, rollen en andere proceselementen. Denk hierbij aan de structuur van een taak, deze heeft bijvoorbeeld altijd een doel, omschrijving, een set van rollen en een input en output.

Architectuur van EPF

Voordat we verder ingaan op de functionaliteit en mogelijkheden van EPF wil ik eerst inzoomen op hoe de architectuur van EPF is opgebouwd. De architectuur kan in twee lagen worden opgedeeld. Het eerste deel is de composer die functionaliteit geeft om processen te kunnen onderhouden en beheren. Het tweede deel zijn de processen die binnen het EPF-platform beschikbaar zijn en worden meegeleverd. De structuur van het EPF-platform is in bovenstaande illustratie weergegeven.

Ronald van Aken
is werkzaam als managing consultant bij Sirius ICT Solutions BV te Amsterdam.



Figuur 1: Architectuur van het EPF platform, afbeelding is gebaseerd op EPF documentatie

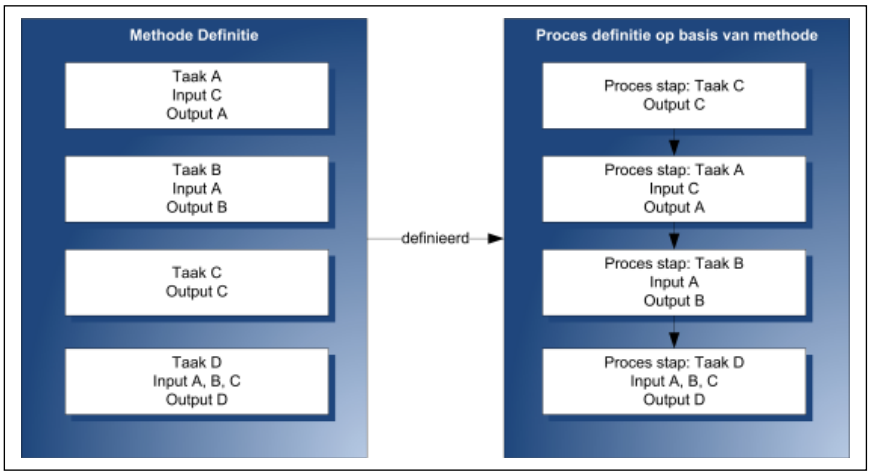
EPF composer is de tooling waarmee we processen definiëren en onderhouden. Het is gebouwd op basis van het bekende Eclipse Applicatie Platform dat de basis vormt voor veel open source en commerciële producten. Voor de opslag van procesinformatie wordt binnen EPF-composer gebruik gemaakt van UMA (Unified Metamodel Architecture). UMA is een XML-definitie waarin de processen worden vastgelegd en komt voort uit het metamodel wat voor RUP is gebruikt.

Het tweede deel van de EPF architectuur beslaat de processen die momenteel zijn opgenomen in het EPF project en de andere extensies die samenwerken en of gebruik maken van de EPF processen.

De extensies die in figuur 1 los zijn getoond illustreren commerciële en open source oplossingen die op een bepaalde wijze gebruik maken van de EPF processen en tooling.

Methode en proces scheiding

Als we kijken naar software die ons helpt met het beschrijven van processen volgens de SPEM-specificatie, dan merken we al snel dat er een scheiding wordt gemaakt tussen methode en proces.



Figuur 2: Scheiding van methode en proces

De methode beschrijft hoe een bepaald resultaat (product) door middel van een bepaalde set van gedefinieerde stappen en rollen wordt gerealiseerd. Het proces heeft als scope dat het een beschrijving geeft van de levenscyclus van het ontwikkelproces en tot stand komt door gebruik te maken van methode elementen.

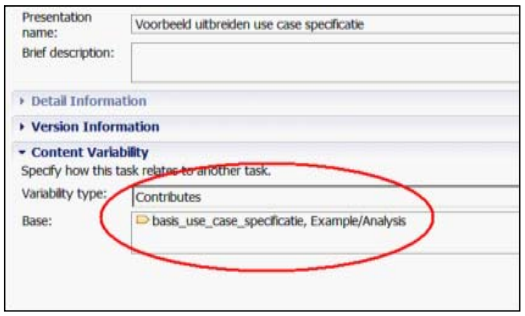
Door een knip te leggen tussen methode en proces worden we in staat gesteld om binnen de definitie van processen gemakkelijk te refereren naar de methode-elementen en dus maar eenmalig hoeven te definiëren. Dit is een groot pluspunt in het gebruik van EPF-composer in plaats van de traditionele tekstverwerker.

Figuur 2 illustreert hoe we de scheiding tussen methode en proces toepassen bij de totstandkoming van een proces. Bij de specificatie van het proces richten we ons eerst op de methode-elementen. Vervolgens kunnen één of meerdere processen naar de elementen refereren om een proces definitie op te stellen.

Adopteren en delen van best practices

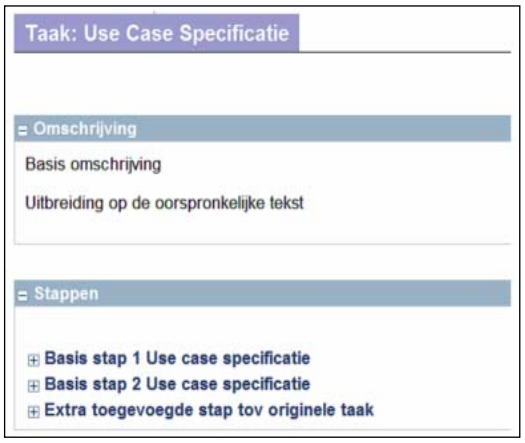
Het voordeel om gebruik te maken van de basis-elementen van processen als OpenUP geeft ons onder andere de voordelen van het niet weer opnieuw specificeren van kennis en kunde en het refereren naar technieken die zijn toegepast in andere processen en welke al breed worden ondersteund in de industrie (best practices). Dit heeft ertoe geleid dat de best practices van OpenUP beschikbaar zijn gesteld in een aparte bibliotheek genaamd EPF Best Practices. Hiermee kun je zonder al direct voor een proces te kiezen gemakkelijk beproefde technieken adopteren in een nieuw proces of dit combineren met bestaande zaken. Door deze opsplitsing hoopt het team achter EPF de technieken toegankelijker en gemakkelijker hergebruikbaar te maken.

Je kunt als organisatie kiezen om de processen te implementeren zoals zij beschikbaar zijn of deze aan te passen. Wanneer je voor de optie kiest om ze aan te passen, dan kun je gebruik maken van de eigenschap dat je een bestaand element uitbreidt of vervangt zonder het originele element te wijzigen. Dit realiseer je door een nieuw element te definiëren dat een bestaand element in dezelfde of een andere plugin vervangt, overerft of een aanvulling is van het al bestaande element. De illustratie in figuur 3 geeft aan hoe we een nieuw element op basis van een bestaande definiëren.



Figuur 3: Definitie van een nieuw element op basis van bestaande methode elementen

Het resultaat wanneer je dit publiceert is te zien in figuur 4. Je ziet dat wanneer we de taak publiceren de basisgegevens van het originele element worden overgenomen en worden aangevuld met de gegevens van de nieuwe taak.

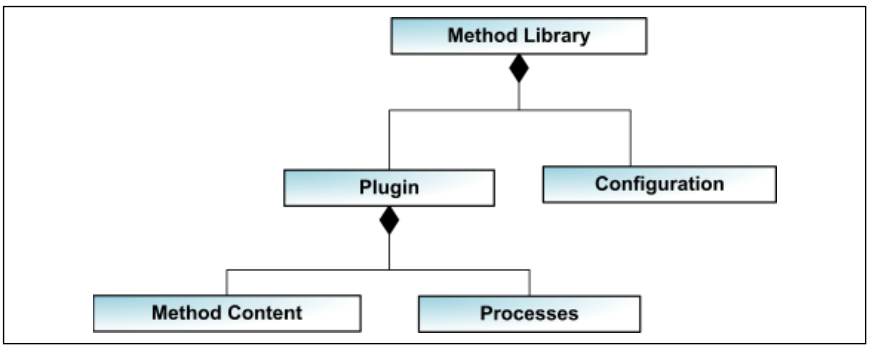


Figuur 4: HTML publicatie resultaat wanneer methode elementen worden hergebruikt

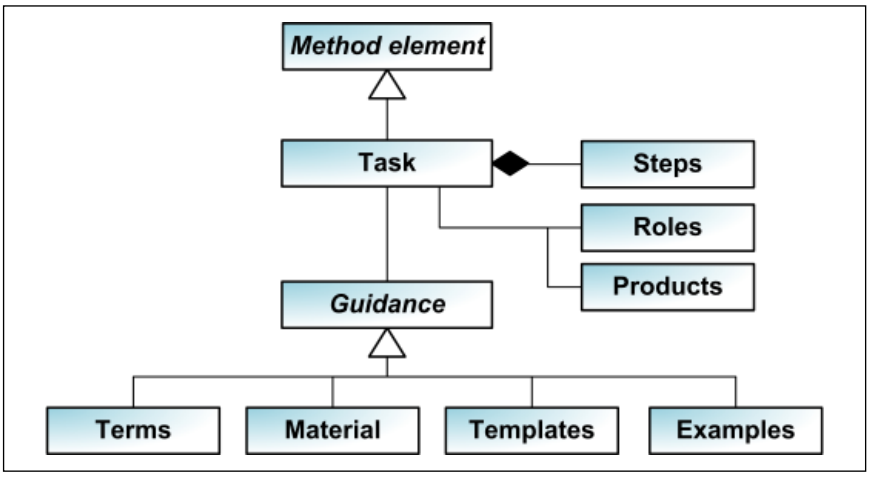
Als voorbeeld kun je denken aan de beschrijving van het opstellen van een use case. Deze kun je direct uit OpenUP halen en eventueel alleen aanvullen met richtlijnen die gelden voor hoe use cases worden toegepast in de organisatie. Dit in tegenstelling tot weer het hele verhaal zelf op te stellen en te schrijven met al het onderhoud, gevaren en risico's van dien.

Definiëren van processen en methoden in EPF

In bepaalde situaties kan het besluit vallen om het proces helemaal zelf te definiëren. Wanneer dit zich voordoet kan EPF zeker ook helpen. Het biedt een vast model qua definities en structuur. Mijn ervaring is dat de structurering je sneller op weg helpt en helpt te concentreren op de specificatie van het proces. Het basiselement waarin je verschillende processen en methodes kunt definiëren, is de method library. Binnen deze library kunnen we plugins specificeren. Een plugin is een belangrijk element dat we als basis gebruiken voor het definiëren van een methode en of proces.



Figuur 5: Vereenvoudigde basis structuur van EPF Method Library



Figuur 6: Structuur van een EPF taak (vereenvoudigt)

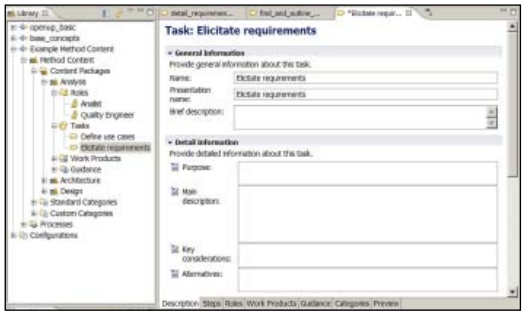
De illustratie in figuur 5 toont hoe een method library is opgebouwd uit plugins en configurations.

Wanneer je begint met definiëren van een proces in EPF, dan is het startpunt een plugin. De plugin stelt ons in staat om de inhoud van de methode te beschrijven. Binnen de plugin hebben we de mogelijkheid om deze op de delen in zogenaamde 'content packages'. Een package kun je gebruiken voor het groeperen van een set gerelateerde elementen. Je kunt ook binnen een package weer een nieuwe package definiëren. De subelementen die je in een package kunt toevoegen zijn als volgt:

- Rollen. Hier beschrijf je de relevante rollen binnen de context van een package. Bij de rol wordt een beschrijving, benodigde disciplines en andere eigenschappen beschreven.
- Taken. Een lijst en beschrijving van de taken die kunnen worden uitgevoerd
- Werkproducten. Producten die worden opgeleverd in de context van deze package. Bij de producten wordt een onderverdeling in drie soorten gemaakt afhankelijk van het resultaat van een taak.
- Ondersteunende materialen. Hier definiëren we ondersteunend materiaal zoals onder andere best practices, white papers, richtlijnen, termen & definities en roadmaps.
- Andere packages.

Het bespreken van al deze elementen zou teveel zijn maar om een beeld te geven gaan we iets dieper in op het beschrijven van een taak. De structuur die wordt gebruikt bij het beschrijven van een taak zien we in figuur 6.

Het UML-diagram in figuur 6 toont ons dat de taak is opgebouwd uit een aantal elementen. Bij de taak kun je ook ondersteunend materiaal toevoegen in verschillende varianten. De reden om dit materiaal separaat te definiëren en te koppelen aan de taak heeft als reden dat de ondersteunende materialen dan later ook aan andere methode-elementen kunnen worden gekoppeld. De illustratie in figuur 7 toont ons hoe we in EPF composer de eigenschappen van een taak kunnen invullen.



Figuur 7: EPF Composer methode structuur

Goed overzicht en handig voor het toevoegen van beslismomenten

Wanneer je de basiselementen hebt ingevuld, kun je de stappen die nodig zijn voor het behalen van het resultaat beschrijven. De rollen en werkproducten koppelen die nodig zijn bij het uitvoeren van de taak. Daarnaast koppel je de ondersteunende elementen die van belang zijn. Doordat de onderliggende structuur van de teksten gebaseerd is op HTML, kun je gemakkelijk in de teksten hyperlinks opnemen naar andere relevante methode-elementen en of externe resources.
Wanneer we alle benodigde methodetaken hebben gedefinieerd, kunnen we beginnen aan de daadwerkelijke procesdefinitie. De definitie komt tot stand door een nieuw proces te definiëren

Presentation Name	In...	Predecessors	Model Info	Type	Pla
Simple Open UP	0			Delivery Process	
Define Vision	1			Task Descriptor	
Manage Requirements	2	1		Capability Pattern	
Find and Outline Require	3			Task Descriptor	
Detail Requirements	4			Task Descriptor	
Create Test Cases	5			Task Descriptor	
Analyze Architectural Requir	6	2		Task Descriptor	
Design the Solution	7			Task Descriptor	
Request Change	8			Task Descriptor	

Figuur 8: Samenstellen van een nieuw proces

waarin je vervolgens een sequentie van methodestappen opneemt.
De bovenstaande illustratie in figuur 8 toont hoe we een simpel proces hebben samengesteld op basis van methodestappen zoals ze in OpenUP beschikbaar zijn.

Vervolgens kunnen we verschillende diagrammen op procesniveau genereren die in de publicatie worden meegenomen. De uiteindelijke diagrammen zijn contextgevoelig en gebaseerd op UML activiteiten diagrammen. Deze diagrammen kunnen uitstekend worden gebruikt voor het geven van een overzicht en het toevoegen van beslismomenten. Je kunt er ook voor kiezen om andere diagrammen te gebruiken wanneer de standaard meegeleverde opties niet voldoen. Bij het gebruik van diagrammen met een afwijkend formaat kun je de diagramelementen koppelen aan methode-elementen om ze contextgevoelig te maken. De illustratie in figuur 9 is een voorbeeld van hoe een proces diagram standaard wordt gevisualiseerd.

Beschikbare processen voor EPF

Wanneer je met EPF aan de slag gaat, krijg je standaard OpenUP meegeleverd. Dit is een basisuitvoering van het Unified Process dat is gedoneerd door IBM als open source. Daarnaast zijn ook andere processen beschikbaar voor op het platform. Door de beschikbaarheid van de processen zijn we gemakkelijk in staat om best practices en andere methodestukken uit deze processen over te nemen in nieuwe processen

of een afgeleid proces op basis daarvan te definiëren. Je kunt kiezen om aan elementen uit andere processen te refereren, ze te vervangen of aan te vullen zonder het oorspronkelijke element te wijzigen. Je kunt dit vergelijken met het concept overerving zoals we dat kennen vanuit objectoriëntatie. Momenteel zijn de volgende processen onder andere beschikbaar binnen het EPF platform:

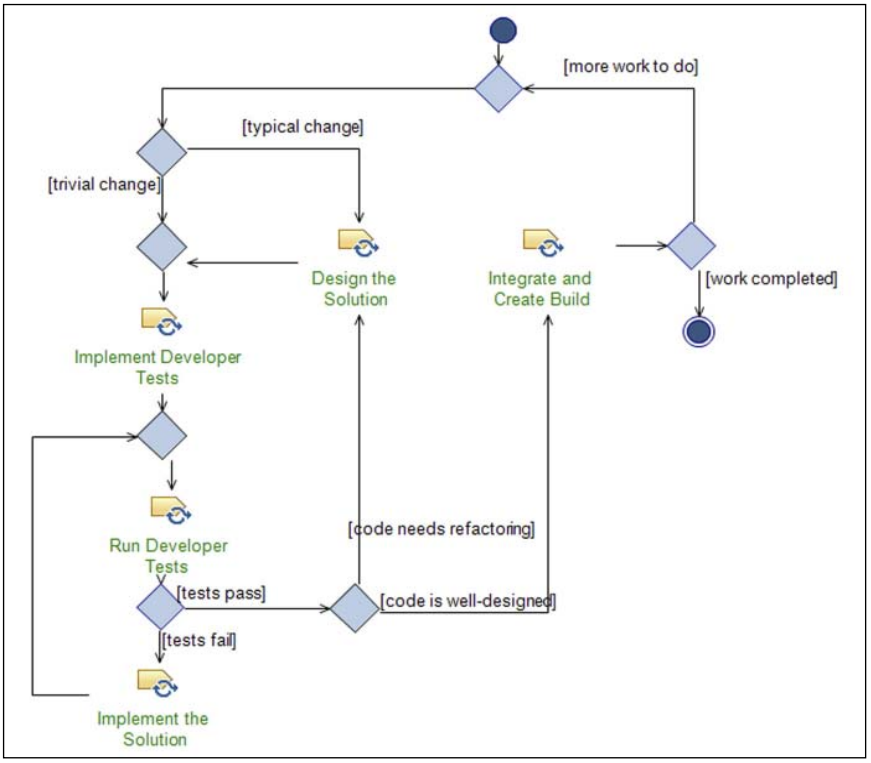
1. OpenUp/DSDM
2. OpenUP/Business Rules
3. XP
4. Scrum
5. Agile Enterprise Architecture

Daarnaast geeft de bibliotheek EPF Best Practices ons bouwstenen om elementen uit OpenUP gemakkelijk te adopteren zonder referenties naar OpenUP, dat is gebaseerd op deze bibliotheek. De opsomming die ik hier geef is verre van compleet. Er zijn nog veel meer projecten voor het ontwikkelen van proces plugins voor EPF. Zoals je zag in het architectuurplaatje is er een scheiding tussen de OpenUP en XP/Scrum processen. De XP/Scrum processen worden onafhankelijk onderhouden van het EPF-project en worden niet standaard meegeleverd met de EPF-download. Ze leveren echter een goede implementatie en basis van de processen en geven ons een goed en toegankelijk platform om de kennis en procesinformatie te benaderen en te onderhouden via EPF. Er lopen ook voorstellen om andere softwareontwikkelprocessen toe te voegen aan het EPF platform. Door het scala aan processen en technieken krijgen we een brede set aan best practices en procespatronen tot onze beschikking die we out-of-the-box snel en doeltreffend kunnen gebruiken in op maat gemaakte ontwikkel processen.

Toekomst en marktpositie van EPF

De open aard van dit product met daarnaast de meegeleverde bronmaterialen maakt het een erg efficiënt platform voor het snel onderhouden en definiëren van methoden en processen. Het metamodel kun je niet uitbreiden of aanpassen maar dit beschouw ik dan meer als een voordeel dan nadeel. We conformeren ons hierdoor aan een standaard en de gegeven structuur.

Het is naar mijn mening een product dat zeker een sterke positie heeft verworven in zijn marktdeel en erg stabiel en volledig over komt. In de wereld van procesdefinitiesoftware zie je niet heel veel andere alternatieven die je ondersteunen wanneer je kritisch selecteert op standaarden. Wanneer we gelijksoortige producten zoeken dan kom je onder andere uit bij Osellus en IBM. Het bedrijf Osellus biedt een commerciële oplossing die veel overeenkomsten qua functionaliteit heeft met EPF. Het voordeel van EPF naast de commerciele varianten is dat de onderdelen van EPF door een breed publiek en verschillende organisaties worden ondersteund en onderhouden. IBM blijft daarnaast bezig met het onderhouden van de commerciële versie van Unified Process en levert naast meer gespecialiseerde varianten van dit proces ook ondersteuning en implementatie binnen organisaties voor de commerciële variant. IBM is echter sterk betrokken bij het EPF-project en je ziet ook ondersteuning voor EPF-processen terugkomen in nieuwe IBM-ontwikkelingen zoals Rational Team Concert. De keuze voor een commerciële variant zal erg organisatieafhankelijk zijn en ik voorzie dan ook dat deze elkaar niet uitsluiten. Zeker omdat EPF zich meer richt op de iets lichtere processen en de versie van RUP door IBM toch meer is gericht op de traditionele en formele ontwikkelmethode zoals we die kennen bij RUP.



Figuur 9: Opstellen en publiceren van proces diagrammen

Wanneer we kijken wat EPF in de breedte kan betekenen voor organisaties dan vinden we een aantal interessante aspecten. Het eerste aspect is het centraal beheren van methode, technieken, kennis en processen op een standaard manier. Een ander aspect is dat EPF een platform levert waarmee procesadviseurs, managers, ontwikkelaars, organisaties en projectleden hun kennis kunnen bundelen en elkaar op een standaard manier kunnen benaderen. Hierdoor krijgen ontwikkelteams processen en kennis aangereikt die aansluiten en exact dat wat nodig is leveren om tot het gewenste resultaat te komen. «