UHASSELT

KNOWLEDGE IN ACTION

Faculteit Bedrijfseconomische Wetenschappen

master in de toegepaste economische wetenschappen: handelsingenieur in de beleidsinformatica

Masterthesis

De ontwikkeling van Business Process Intelligence Dashboards

Samina Breuls

Scriptie ingediend tot het behalen van de graad van master in de toegepaste economische wetenschappen: handelsingenieur in de beleidsinformatica

PROMOTOR :

Prof. dr. Benoit DEPAIRE

2016
2017

UHASSELT

KNOWLEDGE IN ACTION

www.uhasselt.be
Universiteit Hasselt
Campus Hasselt:
Martelarenlaan 42 | 3500 Hasselt
Campus Diepenbeek:
Agoralaan Gebouw D | 3590 Diepenbeek

UHASSELT

KNOWLEDGE IN ACTION

Faculteit Bedrijfseconomische Wetenschappen

master in de toegepaste economische wetenschappen: handelsingenieur in de beleidsinformatica

Masterthesis

De ontwikkeling van Business Process Intelligence Dashboards

Samina Breuls

Scriptie ingediend tot het behalen van de graad van master in de toegepaste economische wetenschappen: handelsingenieur in de beleidsinformatica

PROMOTOR :

Prof. dr. Benoit DEPAIRE

Woord vooraf

Deze masterproef vormt het sluitstuk van mijn opleiding Toegepaste Economische Wetenschappen – Handelsingenieur in de Beleidsinformatica die ik gevolgd heb aan de Universiteit Hasselt. Het onderwerp van deze thesis is "het ontwikkelen van business process intelligence dashboards". Specifieker draait het in dit eindwerk om de definiëring van de procesperformantie-indicatoren waar deze dashboards uit zijn opgebouwd.

Ik heb dit onderwerp gekozen aangezien ik erg geïnteresseerd ben in business intelligence en dit onderwerp ook dicht aanleunt bij de bedrijfswereld. Het was echter geen eenvoudige opdracht, maar met de hulp van verschillende personen ben ik erin geslaagd om deze te voltooien. Om deze reden wil ik hen dan ook enorm bedanken.

De belangrijkste inhoudelijke ondersteuning kreeg ik van mij promotor prof. dr. Benoit Depaire. Dit resultaat zou onmogelijk geweest zijn zonder zijn constructieve feedback. Verder zou ik ook nog mijn oom Gerry en mijn mama willen bedanken voor hun taaladvies. De enorm belangrijke morele steun in deze periode heb ik te danken aan mijn ouders en enkele andere familieleden. Ten slotte zorgden mijn nicht en de studentenvereniging Hermes Diepenbeek voor de broodnodige ontspanning.

Samina Breuls

As, augustus 2017

Samenvatting

In de praktijk ligt er nog te weinig focus op het analyseren van procesperformantie binnen bedrijven. Toch is dit aspect enorm belangrijk voor de beslissingsnemers die gefundeerde beslissingen willen nemen. De processen vormen één van de hoofdaspecten van de knowhow die een organisatie bezit. Daarom is het documenteren en het analyseren van deze processen ook essentieel. De analyse ervan en dus ook het meten van de performantie gebeurt aan de hand van process performance measurement. Hier maken we gebruik van procesperformantie-indicatoren (PPI's), de kwantificeerbare metrieken van processen. Deze indicatoren kunnen later ook voorgesteld worden op een dashboard met behulp van business process intelligence.

Het probleem dat zich bij de analyse regelmatig stelt, is het feit dat men niet goed weet welke soort indicatoren van processen men moet meten en hoe men deze moet modelleren. Dit probleem trachten we op te lossen aan de hand van design science. Hierbij willen we dit praktijkprobleem aanpakken aan de hand van één of meerdere artefacten. In ons geval stellen we een typologie voor en ontwikkelen we een bijhorende modelleertaal in de vorm van een metamodel. Hierbij wordt er dus bepaald welke types PPI's er gedefinieerd zullen worden en hoe deze types specifiek gemodelleerd zullen worden.

Voor de artefacten, de typologie en het metamodel ontwikkeld worden, stellen we eerst vereisten op waaraan deze artefacten zouden moeten voldoen. Op basis hiervan kunnen we achteraf de kwaliteit van onze oplossing evalueren. Het artefact dat als eerste voorgesteld wordt, is de typologie. Hierbij bepalen we welke types indicatoren er later gemodelleerd zullen moeten worden.

De definiëring van de PPI's zelf, zal gebeuren aan de hand van ons metamodel. Het metamodel dat we bouwen is gebaseerd op drie metamodellen uit de literatuur om een zo volledig mogelijke definiëring van de PPI's te bekomen. Deze metamodellen bevatten ieder een ander aspect. Het eerste metamodel zal KPI's definiëren in tegenstelling tot het tweede metamodel dat zich echt focust op PPI's. Verder vult het derde metamodel de twee voorgaande aan met het business activity monitoring (BAM) aspect. Bij BAM zullen de PPI's in real-time gemeten en voorgesteld kunnen worden. De abstracte syntax van het uiteindelijke metamodel zal gebouwd worden op basis van een UML (unified modeling language) klasse diagram. Met behulp van deze abstracte syntax zal ook ieder type PPI uit de typologie gedefinieerd worden.

Nadat het metamodel ontwikkeld is, zal het ook gedemonstreerd worden aan de hand van een voorbeeldproces. Op basis van dit proces worden er voorbeeld PPI's gekozen uit iedere categorie van de typologie. Alsook wordt er aangetoond dat ieder scenario dat we opgesomd hebben op basis van de drie modellen ook door ons model gedefinieerd kan worden. Hierdoor tonen we het nut aan van ons eigen metamodel.

Inhoudsopgave

1. Inleiding

Het management van een organisatie neemt beslissingen om op deze manier het bedrijf in een bepaalde richting te sturen naar de toekomst toe. Daarom is het noodzakelijk om de performantie te meten om hier belangrijke beslissingen op te kunnen baseren. Performantie draait deels om de prestaties uit het verleden, maar de focus ligt grotendeels op de toekomst. Een bedrijf is dus zeker niet performant wanneer enkel de doelstellingen uit het verleden behaald zijn. Het meten van de performantie is noodzakelijk om niet stuurloos te varen. Dit gebeurt dan ook al verschillende decennia in het domein performance measurement (PM) [1]. Performance measurement wordt door Bourne et al. [2] gedefinieerd als "het gebruik van een multi-dimensionele set van performantie-indicatoren voor de planning en het management van een bedrijf".

Om succes te bereiken in de toekomst op basis van huidige keuzes, moet men de prestaties van het bedrijf kunnen meten aan de hand van indicatoren [3]. Deze indicatoren kunnen na de definiëring voorgesteld worden in een dashboard of een andere business intelligence toepassing. Volgens Chauduri et al. [4] is business intelligence "een verzameling van beslissingsondersteunende technologieën voor een bedrijf dat als doel heeft uitvoerend personeel, managers en analisten in de gelegenheid te stellen betere en snellere beslissingen te nemen".

Deze metingen zijn echter geen evidente zaak aangezien de complexe realiteit vertaald moet worden naar een groep meetindicatoren die slechts een beperkte kijk hierop weergeven. De indicatoren die gemeten worden, zijn gekozen op basis van een bepaald doel. Om deze reden is er geen consensus op het gebied van indicatoren die bijdragen tot het meten van de performantie. Ieder bedrijf zal andere indicatoren opnemen in zijn performantiemeetsysteem op basis van de strategische keuzes die er gemaakt worden [1].

Om te zorgen dat de performantie-indicatoren de bedrijfsperformantie zo volledig mogelijk weerspiegelen, is er door Kaplan en Norton de 'balanced scorecard' ontwikkeld [3],[5]. Dit is een framework dat bedrijven helpt om een evenwichtige set van indicatoren te implementeren in hun performantiemeetsysteem. Hierbij zijn er vier verschillende dimensies aanwezig die bijdragen tot een zo volledig mogelijke meting van de performantie. Deze worden aangegeven aan de hand van perspectieven, namelijk het financiële perspectief, het interne bedrijfsperspectief, het klantenperspectief en het innovatie- en leerperspectief. Er moeten indicatoren bepaald worden uit al deze perspectieven om een complete kijk te krijgen op de performantie [5].

In PM is één van de grootste tekortkomingen dat de indicatoren niet gelinkt kunnen worden aan de uitvoering van processen. De focus ligt er meestal op operationele resultaten van het bedrijf. Om de algemene performantie van een bedrijf te verbeteren is het echter noodzakelijk om ook te kijken naar de procesperformantie. Het procesaspect is enorm essentieel wanneer men gegronde beslissingen wil nemen. Wanneer bedrijven focussen op processen zal dit betere resultaten en een

hogere efficiëntie met zich meebrengen. Volgens de definitie van Kolar [6] is een bedrijfsproces "een volgorde van stappen en regels die het gedrag van een bedrijf in bepaalde situaties definieert". Deze volgordes en de bijhorende regels dragen bij aan de knowhow die een bedrijf bezit en moeten dan ook nauwkeurig geanalyseerd worden [7],[4].

Om dit mogelijk te maken, werd er process performance measurement (PPM) geïntroduceerd. Dit kan gedefinieerd worden als "het formeel, gepland monitoren van de procesuitvoering en het nagaan van de resultaten hiervan om de efficiëntie en effectiviteit van de processen te bepalen" in CBOK [8]. PPM is een kritiek onderdeel van 'business process management' (BPM). BPM is "een gedisciplineerde aanpak om zowel geautomatiseerde als niet-geautomatiseerde processen te identificeren, ontwerpen, uitvoeren, documenteren, meten, monitoren en controleren." Deze aspecten streven allemaal naar het behalen van de strategische doelstellingen. Met dit doel voor ogen is het noodzakelijk om processen te meten en te monitoren [8].

Om binnen PPM de procesperformantie te kunnen meten, maakt men gebruik van procesperformantie-indicatoren (PPI's). Dit zijn specifieke gevallen van performantie-indicatoren (KPI's) die gemeten worden op basis van de data die gegenereerd zijn binnen de proces flow. Procesperformantie-indicatoren (PPI's) zijn volgens Del Rio Ortega [9] "kwantificeerbare metrieken die ervoor zorgen dat de efficiëntie en effectiviteit van processen geëvalueerd kunnen worden". Metrieken zijn dan weer de meetbare aspecten van in ons geval een proces. Hierbij zal de focus duidelijk liggen op het procesaspect [9].

De eerste stap bij het analyseren van de procesperformantie is het definiëren en modelleren van deze PPI's [5]. Omdat de PPI's ervoor zorgen dat de performantie van de processen geëvalueerd kan worden, is het enorm belangrijk om het definiëren van deze indicatoren op een zo correct mogelijke manier te doen [9]. Toch worden deze PPI's vaak niet gedefinieerd om verschillende redenen: men weet niet wat men zou moeten meten van het proces, er zijn gewoonweg geen data van het proces voorhanden of er zijn problemen bij het bepalen welke specifieke data er per indicator gemeten moeten worden. De focus ligt in dit onderzoek vooral op dit laatste. We gaan bepalen welk type indicatoren van de processen en ook welke aspecten van deze indicatoren gemeten moeten worden. Dit gebeurt aan de hand van de definiëring van de PPI's.

Na de definiëring kan men deze PPI's monitoren en nagaan of de strategische doelstellingen die vooropgesteld zijn ook daadwerkelijk bereikt worden. PPI's zullen op ieder moment weergeven hoe hun bijhorend proces verloopt en op basis van deze informatie krijgen de bedrijfsleiders nieuwe inzichten. Om dit overzichtelijker voor te stellen, kan men gebruik maken van een business process intelligence dashboard. Bij BPI worden bestaande business intelligence technieken toegepast op de geanalyseerde bedrijfsprocessen [10],[4],[11].

Een ander onderzoeksdomein dat zijn opmars maakt is 'business activity monitoring' (BAM). Volgens Friedenstab et al. [12] is BAM "de observatie, analyse en presentatie van real-time informatie over bedrijfsactiviteiten over de verschillende systemen van bedrijven heen". BAM kan er dus voor zorgen dat de PPI's real-time gemeten worden. Dit is een aspect waar PPM vaak niet

over beschikt. BAM kan dus perfect ter aanvulling gebruikt worden op PPM. BAM is tevens ook een onderdeel van BPM, waardoor PPM en BAM mogelijk samen geïmplementeerd kunnen worden binnen een BPM systeem [13].

Het modelleren en definiëren van PPI's zullen we onderzoeken op basis van drie modelleringstalen uit verschillende onderzoeksdomeinen. De drie modelleringstalen waarop wij ons baseren afkomstig uit de wetenschappelijke literatuur zijn MetricML [14], PPINOT [15] en de BAM extensie voor BPMN [12]. Het eerste model MetricML definieert KPI's en bevindt zich dus in het performance measurement domein. Dit hebben we weerhouden omdat er in de literatuur vaak specifieke definiëringen van indicatoren aanwezig zijn. Hier is er een definiëring opgesteld voor indicatoren die in verschillende organisaties kan toegepast worden. Het tweede model PPINOT definieert specifiek PPI's en kan men dus plaatsen binnen process performance measurement. Dit model is vernieuwend omdat er nauwelijks gekende werken zijn waarin procesperformantie-indicatoren gedefinieerd worden. Het laatste model, de BAM extensie voor BPMN, zorgt ervoor dat processen real-time gemonitord kunnen worden. Door dit model te gebruiken, kunnen we dit unieke aspect verwerken in ons eigen model. Dit behoort tot het 'business activity monitoring' domein waar we net zoals op de andere twee domeinen, in het volgende hoofdstuk verder op zullen ingaan.

We combineren de drie modellen die hierboven vermeld zijn omdat ieder model zijn tekortkomingen en positieve kanten heeft. Deze worden in het derde hoofdstuk besproken, maar hier halen we reeds één tekortkoming per model aan. Het nut van de combinatie van deze modellen wordt hier dan ook duidelijker. Voor MetricML is de grootste tekortkoming dat dit metamodel KPI's modelleert en er dus geen koppeling aanwezig is met bedrijfsprocessen of de bijhorende elementen. Voor PPINOT moeten de elementen uit deze bedrijfsprocessen duidelijker gemodelleerd worden. Aangezien enkele elementen in dit model ontbreken, is de koppeling met bedrijfsprocessen niet helder genoeg. Bij BAM zijn er vervolgens bepaalde types indicatoren die niet gemodelleerd worden die in PPINOT wel aanwezig zijn.

1.1 Onderzoeksdoelstelling

In deze thesis zullen we de procesperformantie bekijken aan de hand van PPI's. De focus in dit onderzoek ligt op het definiëren van de PPI's. We gaan bepalen welke types PPI's wij zullen definiëren en hoe we deze types zullen gaan modelleren. Dit doen we door een typologie voor te stellen die de types PPI's bepaalt en een bijhorende modelleertaal waarop we de definiëring van de PPI's zullen baseren. De modelleertaal die we ontwikkelen zal concepten uit de modellen van de hierboven vermelde wetenschappelijke bronnen MetricM, PPINOT en BAM bevatten. Om ons eigen model te bekomen, bekijken we hier de verschillen tussen deze modellen en gebruiken we concepten uit ieder van deze modellen.

De onderzoeksvraag kan dus als volgt geformuleerd worden:

Wat is het verschil tussen de drie modelleertalen uit MetricM, PPINOT en BAM en wat kunnen ze bijdragen aan elkaar?

1.2 Onderzoeksaanpak

Het probleem dat zich hier stelt is het feit dat men niet weet welk soort indicatoren van processen men moet meten en hoe men deze moet definiëren. Zolang men de PPI's niet kan definiëren, zal men niet voldoende informatie over de procesperformantie hebben. Dit probleem zullen we proberen op te lossen aan de hand van design science.

Design science is een wetenschappelijke studie waarbij men een artefact creëert om een relevant bedrijfsprobleem op te lossen. Hierbij is het de bedoeling dat er een nieuw en innovatief artefact ontwikkeld wordt dat algemeen toepasbaar is en dus niet enkel van toepassing is op een specifiek geval. Bij design science wordt het mogelijk gemaakt dat onderzoek toegepast kan worden op de praktijk en niet louter theoretisch blijft. Het artefact dat ontworpen wordt, is dan ook zowel gebaseerd op theorie uit eerder onderzoek als op andere relevante artefacten. Het resultaat van design science gaat dan ook niet enkel het artefact zijn, maar ook de informatie over dit artefact [16], [17], [18].

Bij design science zijn er zes grote stappen die in de meeste gevallen doorlopen worden.

1) Probleem identificeren en motiveren (hoofdstuk 1, 2 en 3)
2) Vereisten van het artefact definiëren (hoofdstuk 4)
3) Ontwerpen en ontwikkelen van het artefact (hoofdstuk 4)
4) Het artefact in de realiteit demonstreren (hoofdstuk 4)
5) Het artefact evalueren (hoofdstuk 4)
6) De gevonden oplossing naar de buitenwereld communiceren

[16], [17]

In stap één 'probleem identificeren en motiveren' wordt er uitgelegd wat het probleem is en waarom het ook echt relevant is om het in de praktijk te onderzoeken. In de volgende stap 'vereisten van het artefact definiëren' gaat men al aangeven welk artefact ontworpen zal worden om het voorgestelde probleem op te lossen en aan welke vereisten dit artefact zal moeten voldoen. Een artefact is een innovatieve oplossing voor een geobserveerd organisatorisch probleem in de vorm van een methode, een model of een andere ontwikkelde oplossing. Vervolgens zal in de derde stap het artefact ontworpen en ontwikkeld worden. Hierbij houdt men rekening met de vereisten die in de voorgaande stap opgesteld zijn. In de vierde stap wordt dan het artefact dat ontwikkeld is, toegepast op een realistische case. In de vijfde stap, de evaluatie fase, wordt er gekeken of het artefact dat ontwikkeld is ook echt aan de vereisten voldoet die vooropgesteld zijn. Aan de hand hiervan kan men dus de kwaliteit van het ontwikkelde artefact beoordelen. Hier kijkt men ook of het artefact daadwerkelijk het probleem dat zich stelt of een deel hiervan, kan oplossen. De evaluatie zorgt ervoor dat er indien nodig aanpassingen kunnen toegepast worden, alvorens de oplossing in de praktijk gebruikt wordt [16], [17], [18]. Ten slotte gaat men ook de oplossing, naar de nodige stakeholders communiceren die met deze resultaten mogelijk hun probleem of een deel hiervan in de praktijk kunnen oplossen [17].

Om het probleem dat hierboven beschreven staat op te lossen, zullen we de stappen uit design science toepassen in ons eigen onderzoek:

1. Probleem identificeren en motiveren

In de inleiding is het probleem dat zich voordoet reeds geïdentificeerd, net zoals in de rest van het onderzoek is het hier duidelijk dat de focus ligt op het definiëren van procesperformantie-indicatoren. Aangezien het uiteindelijke doel het ontwikkelen van een business process intelligence dashboard is, is deze definiëring enorm belangrijk. In hoofdstuk 2 worden de concepten die aangehaald zijn in de probleemstelling verder uitgewerkt.

2. Vereisten van het artefact definiëren

Voordat het artefact dat we hierna bespreken, ontwikkeld wordt, moeten er vereisten opgesteld worden. Aangezien onze modelleertaal een combinatie van functionaliteiten zal hebben van de drie modelleertalen waar we ons op baseren, zal dit ook zo zijn voor de vereisten. De vereisten zullen gebaseerd zijn op de verschillen die optreden tussen MetricM, PPINOT en BAM.

3. Ontwerpen en ontwikkelen van het artefact

Het artefact dat wij in ons onderzoek zullen ontwikkelen is een typologie die bepaalt welke types PPI's we definiëren en een bijhorende modelleertaal die deze types modelleert. Deze modelleertaal is gebaseerd op de eerder vermelde modelleertalen en combineert concepten van de verschillende domeinen in dit model.

Het eerste deel van het artefact is dus een typologie waar we de types die we beschouwen in vermelden. De typologie waar we ons op baseren, is deze van Del Rio Ortega [15]. Aangezien het hier ook draait om het definiëren van de PPI's, kunnen deze types ook gebruikt worden voor ons onderzoek. In deze typologie worden PPI's op basis van vooraf bepaalde eigenschappen verdeeld in categorieën die verder nog gegroepeerd kunnen worden tot klassen met bepaalde gemeenschappelijke eigenschappen.

Het tweede deel van het artefact, de modelleertaal, is verantwoordelijk voor de definiëring van de PPI's. De PPI's zullen dus eerst gemodelleerd moeten worden, waarna men ze kan definiëren op basis van deze modellen. Deze modelleertaal zal gebouwd worden aan de hand van een metamodel waarop deze PPI's gebaseerd worden. Een metamodel is een model waar de abstracte syntax (de gebruikte concepten en linken hiertussen) wordt weergegeven die gebruikt kan worden om de uiteindelijke definiëring op te baseren. Het metamodel dat wij ontwikkelen is een combinatie van drie bestaande modellen. Deze drie metamodellen zijn afkomstig uit de volgende artikels: 'MetricM: a modeling method in support of the reflective design and use of performance measurement systems' [14], 'On the definition and analysis of process performance indicators' [15] en 'Extending BPMN for business activity monitoring' [12]. Naar deze artikels wordt respectievelijk ook verwezen

als MetricM, PPINOT en BAM. Door deze modellen te combineren ontstaat er een model dat verschillende perspectieven bevat.

4. Het artefact in de realiteit demonstreren en evalueren

Design evaluatie methodes	
1. Observationeel	Case studie: Bestudeer het artefact grondig in de bedrijfsomgeving
	Veldstudie: Monitor het gebruik van het artefact in verschillende projecten
2. Analytisch	Statistische analyse: Onderzoek de structuur van het artefact op het gebied van statistische kwaliteiten (bv. complexiteit)
	Architectuur analyse: Bestudeer de fit van het artefact binnen de technische IS architectuur
	Optimalisatie: Demonstreer de optimale eigenschappen van het artefact of voorzie optimaliteitsgrenzen op het gedrag van het artefact
	Dynamische analyse: Bestudeer het artefact op het gebied van dynamische kwaliteiten (bv. performantie) wanneer het in gebruik is
3. Experimenteel	Gecontroleerd experiment: Bestudeer artefact in een gecontroleerde omgeving in verband met kwaliteiten (bv. nut)
	Simulatie: Voer artefact uit met artificiële data
4. Testen	Functioneel (black box) testen: Voer artefact interface uit om falingen en defecten te herkennen
	Structureel (white box) testen: Voer 'coverage' testen uit van een bepaalde metriek (bv. 'execution' paden) in de artefact implementatie
5. Descriptief	Geïnformeerd argument: Gebruik informatie van het kenniscentrum (bv. relevant onderzoek) om een overtuigend argument te bouwen in verband met het nut van het artefact
	Scenario's: Bouw gedetailleerde scenario's rond het artefact om het nut te demonstreren

Tabel 1 design evaluatie methodes: [18]

In deze stap combineren we zowel stap vier als vijf uit de design science. Om dit artefact later te kunnen evalueren maken we gebruik van één van de reeds bestaande methodes die we zien in tabel 1. In ons onderzoek kiezen we voor het gebruik van een descriptieve evaluatie. Dit doen we omdat ons artefact een innovatief werk is in de vorm van een conceptueel model. Hier is nog geen sprake van een operationeel informatiesysteem dat reeds geïmplementeerd is. Dit is wel het uiteindelijke doel waarvoor onze taal gebruikt kan worden. Bij deze evaluatiemethode maken we gebruik van bestaande werken om het nut van ons eigen artefact te staven [16], [17], [18]. Binnen de descriptieve evaluatie methode kunnen er ook scenario's worden beschreven om het nut

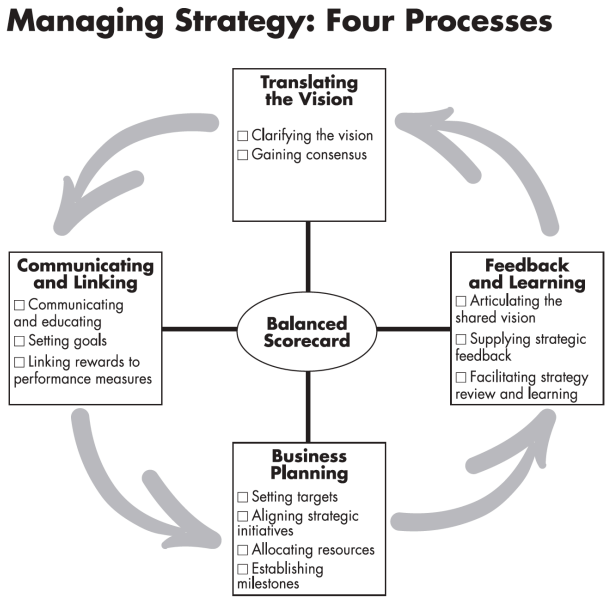
van het artefact aan te tonen. Dit zullen we doen aan de hand van een voorbeeldproces, waar we de verschillende types indicatoren in identificeren. Voor ieder van de types wordt dan de modelleertaal toegepast om de PPI te definiëren. Dit beschouwen we naast een evaluatiemethode daarom ook al als de demonstratie van het artefact [18].

5. De gevonden oplossing naar de buitenwereld communiceren

Aangezien het hier gaat om een thesis, zal dit onderzoek bekend gemaakt worden bij de verdediging ervan. Verder kan dit werk ook opgenomen worden in de masterproefbank van de UHasselt.

2. Literatuurstudie

Iedere organisatie heeft een bestaansreden en hecht belang aan bepaalde waarden. Dit wordt bepaald in hun missie. De missie van Google is bijvoorbeeld 'alle informatie wereldwijd toegankelijk en bruikbaar maken'. Deze wordt bepaald voor de lange termijn en zal niet snel veranderd worden. Op basis van deze missie baseren de bedrijfsleiders hun visie. De visie geeft een beeld over een algemene doelstelling die het bedrijf wil bereiken. Bij Google is deze visie 'het leveren van relevante zoekresultaten uit alle gegevensbronnen' [19], [20]. Nadat de visie bepaald is, is het ook noodzakelijk dat er een strategie bepaald wordt die definieert hoe men de visie kan bereiken [19]. Hier worden strategische doelstellingen gekozen, maar deze moeten vertaald worden naar korte-termijnacties. Dit zijn dan de operationele doelstellingen voor de verschillende teams en de manier waarop men deze wil behalen. Kaplan en Norton [21] worden vier managementprocessen geïntroduceerd die hieraan bijdragen [21]. Deze processen zijn weergegeven op figuur 1.



Figuur 1 management processen in verband met de strategie: [21]

Het eerste proces is het "vertalen van de visie". Hierbij wordt de visie verduidelijkt en zorgt men dat er consensus bereikt wordt in verband met de visie en de strategie. Men moet het eens zijn over de doelstellingen en indicatoren die gekozen zijn als lange-termijnsuccesfactoren.

Het tweede proces, "communiceren en linken", draait vooral om het communiceren van de strategie en het afstemmen van deze strategie op doelstellingen van de afdelingen en individuen binnen de organisatie. Samen met de balanced scorecard zorgt het management hiervoor.

Het derde proces "bedrijfsplanning" draait om het afstemmen van de bedrijfsplannen op de doelstellingen die bepaald zijn in de balanced scorecard. Men kan bijvoorbeeld resources toewijzen of andere acties ondernemen om de lange-termijn doelstellingen te behalen.

Bij het vierde proces "feedback en leren" wordt de strategie die op dit moment gekozen is, geëvalueerd. De strategie wordt beoordeeld op basis van recente performantie die weergegeven wordt door indicatoren die zich in de vier perspectieven bevinden [21].

Met behulp van deze vier processen wordt de strategie beheerd en performance measurement speelt hier een cruciale rol in. PM gebruikt men hier voor zowel de planning, het communiceren van de strategie als het evalueren van deze strategie.

2.1 Performance measurement

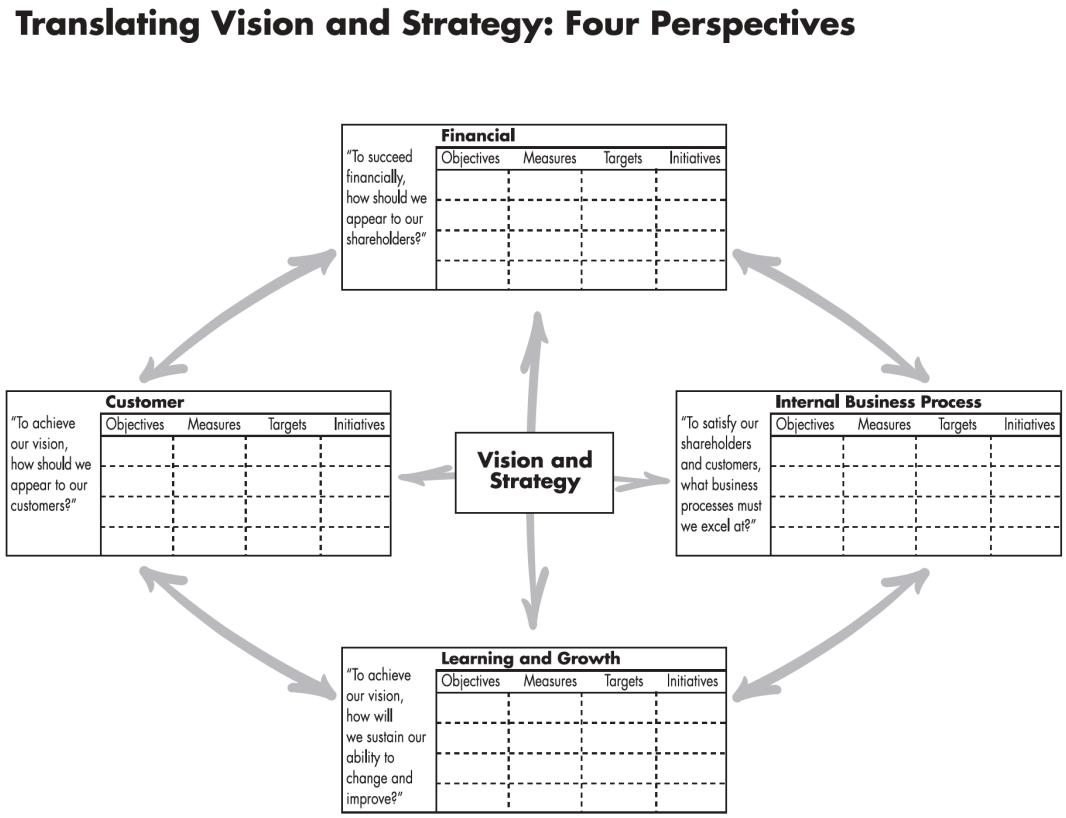
Performance measurement wordt volgens Bourne et al. [2] gedefinieerd als "het gebruik van een multi-dimensionele set van performantie-indicatoren voor de planning en het management van een bedrijf". Op basis van deze indicatoren wil men de prestaties van het bedrijf meten. Hierbij wil men focussen op informatie die aangeeft hoe succesvol het bedrijf zal zijn in de toekomst. Performantie is volgens Lebas [1] "het vermogen om bepaalde doelstellingen te behalen". Het gaat hier om het behalen van operationele doelstellingen, de tijdstippen waarop men deze wil behalen en de manier waarop men er wil geraken (bijvoorbeeld volgorde van doelstellingen) [1]. Performantie is dus zeker geen objectief aspect en zal verschillen naargelang de visie en bijhorende strategie. De indicatoren die men op basis hiervan bepaalt, zijn dus ook per bedrijf te bepalen. Ondanks het feit dat deze indicatoren gekozen worden naargelang de strategie, geven ze vaak een beperkt aantal aspecten van de performantie weer [3]. Om de strategie te kunnen evalueren, moet de performantie zo volledig mogelijk weergegeven worden aan de hand van performantie-indicatoren. Hiervoor ontwikkelden Kaplan en Norton de 'balanced scorecard' [3], [5].

2.1.1 balanced scorecard

De balanced scorecard is een framework dat bedrijven ondersteunt bij het implementeren van een evenwichtige set van indicatoren in hun performantiemeetsysteem die overeenstemmen met hun strategie. Om een algemene kijk te krijgen op de performantie zijn er vier verschillende dimensies die ieder bijdragen tot de volledigheid van het meetsysteem. Deze dimensies worden aangegeven aan de hand van perspectieven, namelijk het financieel perspectief, het intern bedrijfsperspectief, het klantenperspectief en het innovatie- en leerperspectief [5]. In ieder van deze perspectieven zullen strategische doelstellingen vertaald worden in operationele doelstellingen en bijhorende meetbare indicatoren. Wanneer men dus een balanced scorecard wil implementeren, moet men eerst met het management komen tot een gedeelde missie en visie. Op basis van de visie zal men een strategie uitwerken en hieraan worden dan weer strategische doelstellingen gekoppeld. Aan deze doelstellingen worden operationele doelstellingen en de performantie-indicatoren gekoppeld [22]. Om een complete kijk te krijgen op de performantie moeten er indicatoren bepaald worden uit al deze perspectieven. In ieder van deze perspectieven maakt men een selectie van een beperkt aantal kritische indicatoren om de focus op de visie te behouden. Hier gaat het zowel om interne als externe indicatoren. Zo zal vanuit verschillende standpunten gekeken kunnen worden of men daadwerkelijk de bepaalde strategie volgt. Er wordt dus echter ook vermeden dat er overbodige indicatoren gemeten worden die niets weergeven in verband met de strategische doelstellingen.

Naargelang de strategie en bijhorende indicatoren die men bepaalt, zal de balanced scorecard voor ieder bedrijf verschillen [5], [22].

De indicatoren die zich in de perspectieven bevinden, moeten volgens Neely et al. [5] de volgende vragen zo goed mogelijk beantwoorden:
"Hoe komen we over ten opzichte van onze aandeelhouders?" (financieel perspectief)
"Waar moeten we in uitblinken?" (intern bedrijfsperspectief)
"Hoe zien onze klanten ons?" (klantenperspectief)
"Hoe kunnen we steeds verbeteren en waarde creëren?" (innovatie en leerperspectief)



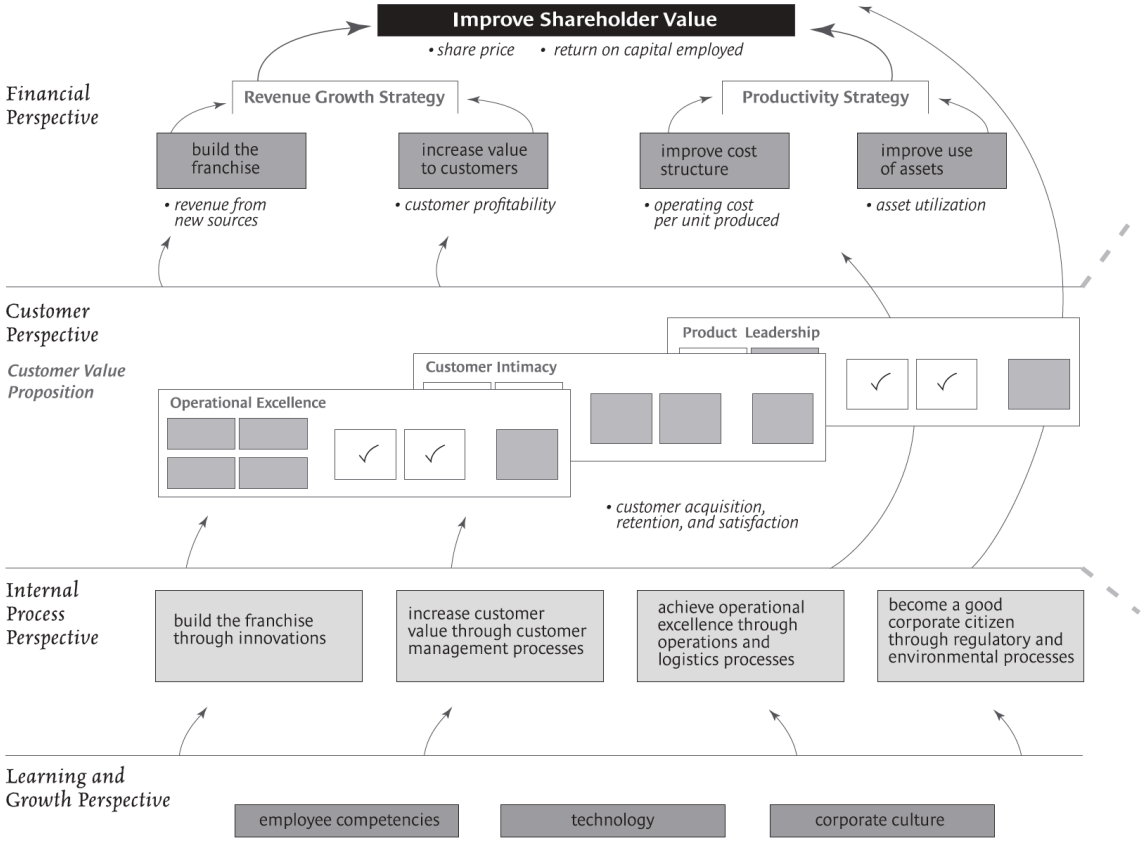
Figuur 2 balanced scorecard : [21]

Nadat de indicatoren per perspectief bepaald zijn op basis van de bijhorende operationele doelstellingen, wordt er zoals te zien op figuur 2 ook nog aangegeven welke specifieke target waarde hieraan gekoppeld wordt. Wanneer de omzet bijvoorbeeld een performantie-indicator is, kan er een target waarde zijn van bijvoorbeeld 100.000. Naast deze targets wordt de balanced scorecard ook aangevuld met initiatieven die door het bedrijf als geheel of door bepaalde afdelingen genomen kunnen worden om de targets te behalen [21].

Op basis van dit framework zal dus de visie gedeeld worden met de rest van het bedrijf en de balanced scorecard voorziet bedrijven van een algemeen model dat de performantie van een bedrijf weergeeft. Deze performantie-indicatoren kunnen door iedereen gebruikt worden en zorgen ervoor dat men dezelfde kijk heeft op performantie op verschillende niveaus en binnen

verschillende afdelingen. Aan de hand van deze indicatoren kijkt men of de gekozen operationele doelstellingen behaald worden en men de strategie die bepaald is op basis van de missie en de visie volgt [23]. Het gaat hier echter slechts om richtlijnen en is er geen garantie op een performantiemeetsysteem dat werkelijk een volledige kijk geeft op performantie [5].

2.1.2 strategy maps



Figuur 3 strategy map: [19]

Om het tweede proces, de communicatie van de strategie, te ondersteunen is er als aanvulling op de balanced scorecard nood aan tools, die de strategie en de noodzakelijke processen en systemen om deze strategie te implementeren, communiceren. Bij het implementeren van een strategie gaat het vaak fout omdat werknemers slechts over beperkte informatie beschikken in verband met taken die ze moeten uitvoeren om de doelstellingen te bereiken [19]. Dit probleem wordt opgelost met behulp van strategy maps.

'Strategy maps' tonen hoe initiatieven in de vier verschillende perspectieven uit de balanced scorecard samen leiden tot bepaalde resultaten. In tegenstelling tot bij de balanced scorecard worden hier ook de relaties tussen de verschillende perspectieven weergegeven en de manier waarop ze bijdragen aan het resultaat. Strategy maps zijn dus een uitbreiding op de balanced

scorecard die gebruikt kunnen worden om de strategie duidelijk te beschrijven en te communiceren [19].

Een strategy map wordt top down ontwikkeld en is dus gebaseerd op de missie en visie van het bedrijf. Net zoals bij de balanced scorecard zijn hier strategische doelstellingen bepaald die men wil behalen op basis van de misse en de visie. In een map begint men dus met de missie en hierna kijkt men welk pad men moet volgen om hier te raken. Nadat de missie bepaald is, gaat men verder met het bepalen van de financiële strategie. Dit is een strategie die focust op het laten toenemen van de aandeelwaarde. Hierbij heeft men twee aspecten, namelijk een strategie die focust op toename van de opbrengsten of een strategie die focust op een productiviteitstoename [19].

Nadat de financiële strategie uitgewerkt is, kijkt men naar het consumentenperspectief en hoe dit hieraan kan bijdragen. In dit perspectief draait het om de klantenwaarde propositie. Deze beschrijft de combinatie van kenmerken van producten en diensten, alsook de bijhorende klantenrelaties en het imago van het bedrijf. Hierbij zijn er drie verschillende categorieën waarvan een bedrijf er één kiest om het sterkst op te focussen. De categorieën zijn 'operationele excellentie', 'klanten intimiteit' en 'product leiderschap'. De kenmerken die hiermee gepaard gaan, zijn weergegeven in tabel 2 [19].

Categorie	**Kenmerken**
Operationele excellentie	"Uitblinken in competitieve prijzen, product kwaliteit en selectie, snelle orderverwerking en op tijd leveren."
Klanten intimiteit	"Kwaliteit van klantenrelaties is enorm belangrijk, men moet exceptionele diensten bieden en volledige oplossingen bieden."
Product leiderschap	"Men moet focussen op de functionaliteit, eigenschappen en algemene performantie van de producten en diensten."

Tabel 2 klantenwaarde propositie: [19]

Nadat het financiële perspectief en klantenperspectief zijn ingevuld, gaat men kijken op welke manier men de beoogde waardenpropositie voor de klanten kan behalen en de productiviteit kan gaan verbeteren zodat de financiële doelstellingen behaald kunnen worden. De activiteiten die uitgevoerd moeten worden zijn cruciaal om de hiervoor beschreven doelen te bereiken. Deze activiteiten gebeuren in de vorm van interne processen, die volgens Kaplan en Norton [19] opgedeeld kunnen worden in de volgende categorieën: "innoveren met nieuwe producten en diensten en door het penetreren van nieuwe markten en klantensegmenten; relaties met bestaande klanten verdiepen; supply chain management, de kost, kwaliteit en doorlooptijd van interne processen, gebruik van activa en capaciteitsmanagement verbeteren; en effectieve relaties met externe stakeholders opbouwen.". Omdat de return van deze processen niet volledig op

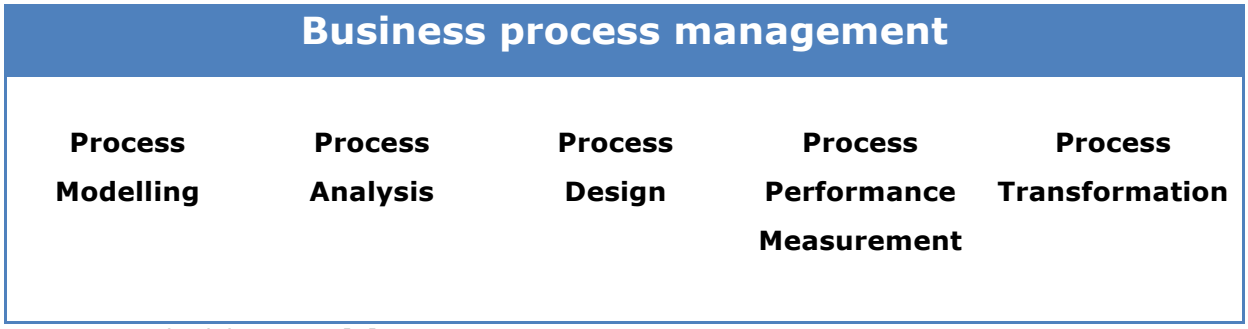
dezelfde termijn gerealiseerd wordt, is het verbeteren van interne processen uit verschillende categorieën aangeraden [19].

De basis van de strategy map ligt vervolgens in het leer- en groeiperspectief. Dit perspectief bevat volgens Kaplan en Norton [19] "kerncompetenties en skills, technologieën en de bedrijfscultuur". Het bepalen van initiatieven die aan deze elementen bijdragen, zijn nodig om de gekozen strategie te ondersteunen. Het complete overzicht van de strategy map met de vier perspectieven die hierboven besproken zijn, is weergegeven in figuur 3 [19].

2.2 Business process management

Het procesaspect komt reeds aan bod in de balanced scorecard, in het interne procesperspectief. Hier wordt echter enkel aangegeven welke interne processen uitgevoerd of vernieuwd moeten worden. De prestaties van de processen worden hier niet gemeten op basis van gegevens die voortvloeien uit de procesuitvoering. Dit is één van de grootste tekortkomingen binnen PM [7]. Bedrijfsprocessen hebben namelijk een grote invloed op heel wat aspecten van organisaties. Zo bepalen zij voor een groot deel de kwaliteit, de hoeveelheid innovatie en productiviteit [24]. Om dit aspect wel te implementeren, wordt business process management geïntroduceerd.

Business process management (BPM) wordt in CBOK [8] gedefinieerd als "een gedisciplineerde aanpak om zowel geautomatiseerde als niet-geautomatiseerde processen te identificeren, ontwerpen, uitvoeren, documenteren, meten, monitoren en controleren". Zoals weergegeven in figuur 4, zijn er vijf kennisdomeinen gedefinieerd die behoren tot business process management. Process performance measurement is één van deze vijf onderdelen. Al deze vijf kennisdomeinen worden gebruikt met als opzet het behalen van de strategische doelstellingen. Door hierop af te stemmen, dragen de processen onder andere beter bij aan de noden en wensen van de klanten. Het meten en monitoren van processen speelt hierin een kritieke rol en daarom is process performance measurement enorm belangrijk binnen BPM [8], [24].



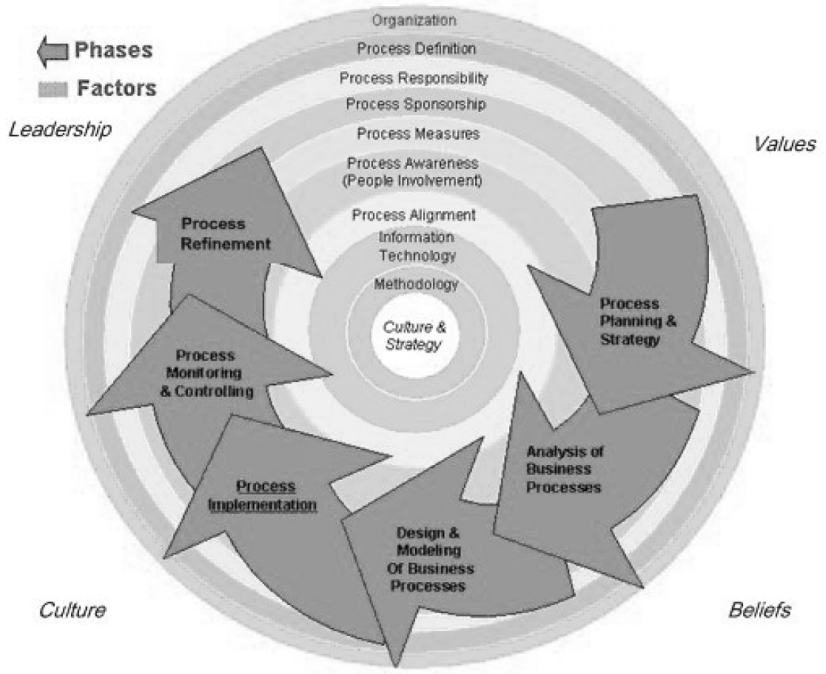
Figuur 4 onderdelen BPM: [8]

Om BPM en de bijhorende activiteiten beter te begrijpen, kijken we naar de BPM lifecycle [24]. Er bestaan verschillende BPM lifecycles in de literatuur, maar hier beschouwen we de BPM lifecycle van ABMP [8]. Deze keuze werd gemaakt op basis van de vergelijking van zeven andere BPM lifecycles die vergeleken werden met ABMP door de Morais et al. [25]. Dit werd gedaan op basis van een tabel waarin de onderdelen van iedere lifecycle weergegeven werden. Deze tabel is te

vinden in bijlage 1. Omdat de strategische doelstellingen een cruciaal element zijn binnen BPM, is het belangrijk dat de strategie verwerkt wordt in de lifecycle. Dit komt echter maar in vier andere modellen voor. Omdat wij hier erg op focussen, komen enkel nog de modellen waar dit wel voorkomt in aanmerking. Verder moeten de processen als laatste stap ook aangepast en geoptimaliseerd worden. Dit gebeurt maar in twee andere lifecycles zoals te zien in bijlage 1. De enige lifecycle die zowel strategie als optimalisatie beschouwt is deze van Houy et al. (2010). Dit model heeft als gebrek dat proces analyse hier niet als stap in verwerkt is. Aangezien proces analyse een enorm belangrijk onderdeel is binnen BPM hebben we dus voor de ABMP BPM lifecycle gekozen [25].

Deze BPM lifecycle op figuur 5 bevat de zes volgende stappen, die verder worden uitgeklaard:

1) proces planning en strategie
2) analyse van bedrijfsprocessen
3) ontwerpen en modelleren van bedrijfsprocessen
4) procesimplementatie
5) proces meten en monitoren
6) procesverfijning



Figuur 5 BPM lifecycle: [8]

1) Proces planning en strategie

BPM activiteiten zorgen er mee voor dat strategische doelstellingen behaald kunnen worden. Net zoals bij process performance measurement wordt er daarom eerst een strategie ontwikkeld die gefocust is op de processen en ook een bijhorend plan. Dit kan bijvoorbeeld aan de hand van de balanced scorecard met een bijhorende strategy map zoals eerder aangegeven. In deze stap moet

ook aangegeven worden welke BPM acties gedaan moeten worden om de doelstellingen te behalen en op welke manier [8], [24].

2) Analyse van bedrijfsprocessen

Om de doelstellingen te kunnen bereiken, is het heel belangrijk om de huidige processen te begrijpen. Hier worden verschillende methodes gebruikt om te kijken in welke mate de processen nu al in lijn liggen met de gestelde doelstellingen. In de analyse wordt er informatie in verband met de processen gehaald uit procesmodellen, strategische plannen, performantie-indicatoren, veranderingen in de omgeving en andere mogelijke factoren [8], [24].

3) Ontwerpen en modelleren van bedrijfsprocessen

In deze stap worden processen en de bijhorende specificaties ontworpen, zodat er waarde gecreëerd wordt voor de klanten. Men gaat een proces zo ontwerpen, zodat dit proces zoveel mogelijk bijdraagt aan de doelstellingen die gesteld zijn. Wanneer de specificaties bepaald zijn, kan het proces gemodelleerd worden [8], [24].

4) Procesimplementatie

Bij de procesimplementatie wordt het model dat ontworpen of aangepast is, geïmplementeerd [24].

5) Proces meten en monitoren

Bedrijfsprocessen moeten voortdurend gemeten en gemonitord worden om beslissingsnemers van de nodige informatie te voorzien. Op basis van deze informatie kijken deze personen of de operationele doelstellingen behaald worden of dat er eventueel aanpassingen moeten gebeuren in de processen om deze wel te behalen [8], [24].

6) Procesverfijning

De processen worden aangepast op basis van de informatie die voortvloeit uit de vorige stappen. Op deze manier wil men ze verbeteren en optimaliseren [8].

Nadat we deze BPM lifecycle en de bijhorende stappen hebben uitgeklaard, situeren we process performance measurement hierin. Process performance measurement is een enorm belangrijk onderdeel van BPM dat zich hoofdzakelijk in de vijfde stap ‘proces meten en monitoren’ bevindt. In deze stap worden processen gemeten en gemonitord aan de hand van de procesperformantie-indicatoren die bepaald zijn op basis van de strategie. In deze stap van de BPM lifecycle situeert zich naast process performance measurement ook business activity monitoring. Hier zullen de PPI’s uit process performance measurement in real-time gemonitord kunnen worden.

Naast deze stap, biedt process performance measurement ook de noodzakelijke informatie aan andere stappen in de lifecycle. Zo wordt de informatie verschaft door PPM en ook BAM gebruikt om de volgende stappen te ondersteunen: ‘analyse van bedrijfsprocessen’, ‘ontwerpen en modelleren van bedrijfsprocessen’ en ‘procesverfijning’. Deze informatie is noodzakelijk in al deze stappen die

doorlopen worden in de BPM lifecycle om veranderingen door te voeren die in lijn liggen met de strategie [8].

2.2.1 Process performance measurement

Process performance measurement is gedefinieerd als "het formele, geplande monitoren van de procesuitvoering en het nagaan van de resultaten hiervan om de effectiviteit en efficiëntie van de processen te bepalen" in CBOK [8]. Dit wordt dan ook de definitie die we hier verder voor zullen gebruiken. Focussen op de processen zal leiden tot een competitief voordeel op lange termijn [26]. Daarom is het belangrijk om hier niet enkel de performantie, maar vooral ook de procesperformantie te meten. Zoals te zien is in figuur 3 zorgen de vier perspectieven er samen voor dat de doelstellingen behaald kunnen worden en hebben ze ook een invloed op elkaar. Het meten van de procesperformantie kan gesitueerd worden binnen het interne procesperspectief. Wanneer we dus process performance measurement hierin introduceren, zal men objectiever kunnen oordelen of het verbeteren van de interne processen in lijn ligt met de strategie [19]. Om deze resultaten ook daadwerkelijk te kunnen behalen, zijn twee dingen enorm belangrijk. Ten eerste moet men de indicatoren bepalen op basis van de operationele doelstelling die gekozen is en verder moeten de processen ook voortdurend gemeten en gemonitord worden [8]. Door procesperformantie voortdurend te meten kunnen problemen herkend worden voor ze uit de hand lopen. Hiervoor ontbreekt in vele bedrijven echter een systeem.

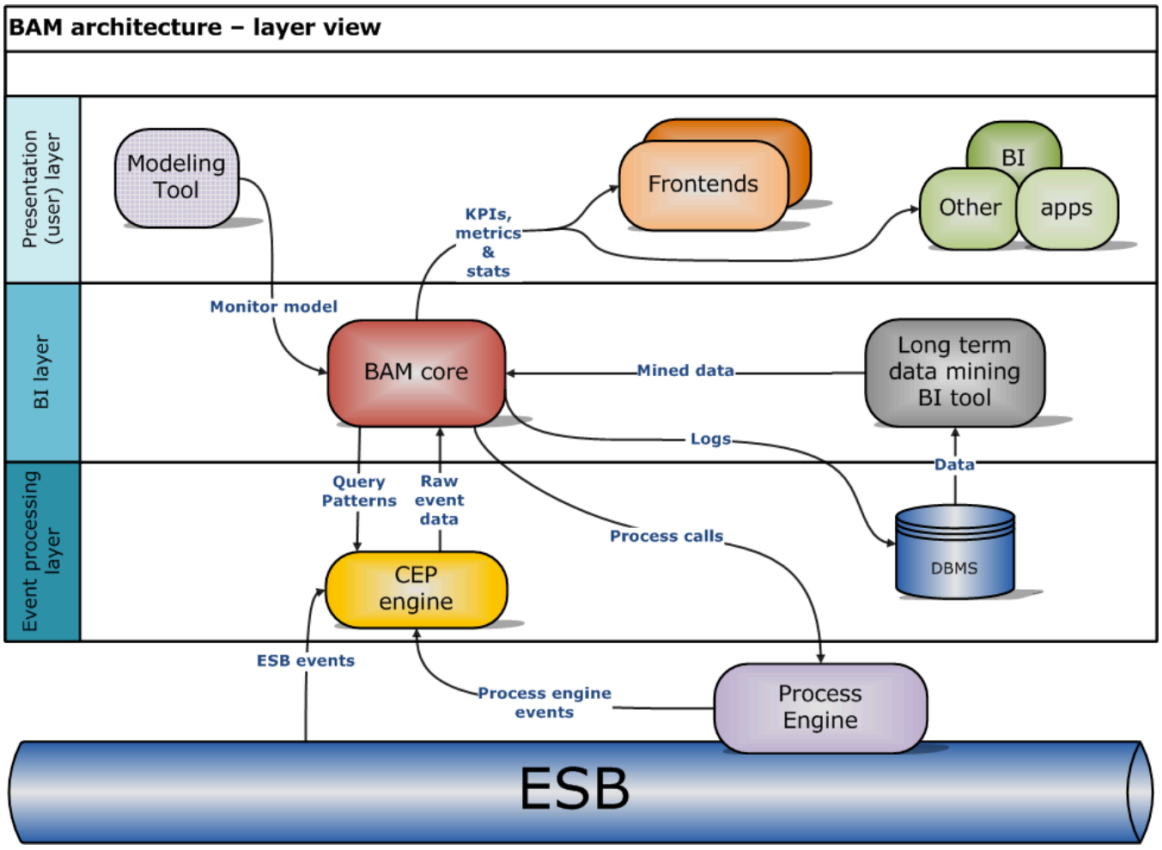
Een systeem dat de procesperformantie meet, moet voldoen aan twee voorwaarden: de focus moet op processen liggen en zowel kwalitatieve als kwantitatieve factoren moeten gemeten worden om de procesperformantie te evalueren. Hoewel de balanced scorecard voldeed aan de tweede eigenschap, was dit voor de eerste niet het geval. Daarom zijn PPMS naast de balanced scorecard gebaseerd op systemen die focussen op het proces-aspect [26]. Dit is een informatiesysteem dat volgens Kueng [26] drie kerneigenschappen bezit op het gebied van functionaliteit. Het verzamelt data in verband met de performantie van verschillende bedrijfsprocessen aan de hand van een set van procesperformantie-indicatoren. De tweede eigenschap is het feit dat dit systeem huidige waarden vergelijkt met historische waarden en targets. En ten slotte verspreidt het systeem ook nog deze resultaten naar de procesbeheerders die verantwoordelijk zijn voor de beschouwde processen.

2.2.2 BAM

De volgorde van stappen binnen een proces en de bijhorende regels hiervan dragen zoals eerder vermeld bij aan de knowhow die een bedrijf bezit. Nadat deze processen dus gedocumenteerd zijn, gaan we ook kijken hoe we ze kunnen monitoren aan de hand van BAM (business activity monitoring) [6]. Business activity monitoring is volgens Friedenstab et al. [12] "de observatie, analyse en presentatie van real-time informatie over bedrijfsactiviteiten over de verschillende systemen van bedrijven heen". Dit onderdeel situeert zich binnen de BPM lifecycle vooral in de stap waar processen gemeten en gemonitord worden. Want hier zullen de PPI's real-time gemeten worden en dit zal bijdragen aan de optimalisatie van de bedrijfsprocessen. Ook zijn er bij BAM vaak tools aanwezig die na de analyses ervoor zorgen dat deze PPI's real-time gevisualiseerd worden.

Op basis hiervan kunnen beslissingsnemers in één oogopslag processen monitoren en zo nodig aanpassen [6], [12].

Zoals reeds vermeld, worden de processen eerst gemodelleerd. Dit gebeurt in het grootste deel van de gevallen via de BPMN 2.0 (business process modelling notation) standaard [18]. Dit is momenteel de meest gebruikte standaard voor het modelleren van bedrijfsprocessen. Om de kloof tussen het modelleren van de bedrijfsprocessen en BAM te verkleinen, gebruiken we voor het modelleren van deze processen, bij voorkeur BPMN2.0 inclusief een BAM extensie [12]. Hiervoor is er in [12] een modelleertaal uitgewerkt, met behulp van zowel een abstracte als een concrete syntax. Om onze PPI's te definiëren, gaan wij enkel gebruik maken van de abstracte syntax, in de vorm van een metamodel waar de concepten van deze BAM extensie voorgesteld worden. De concrete syntrax, namelijk de grafische notatie, zullen wij hier niet gebruiken [12]. Doordat er BAM concepten worden verwerkt in de modelleringstaal, zal de overgang naar BAM analyses minder groot worden.



Figuur 6 BAM architectuur: [6]

2.2.3 BAM applicaties

Alvorens we gaan kijken waarvoor BAM in de praktijk gebruikt wordt, bespreken we hier de onderdelen van de BAM applicaties, zoals aangegeven op figuur 6.

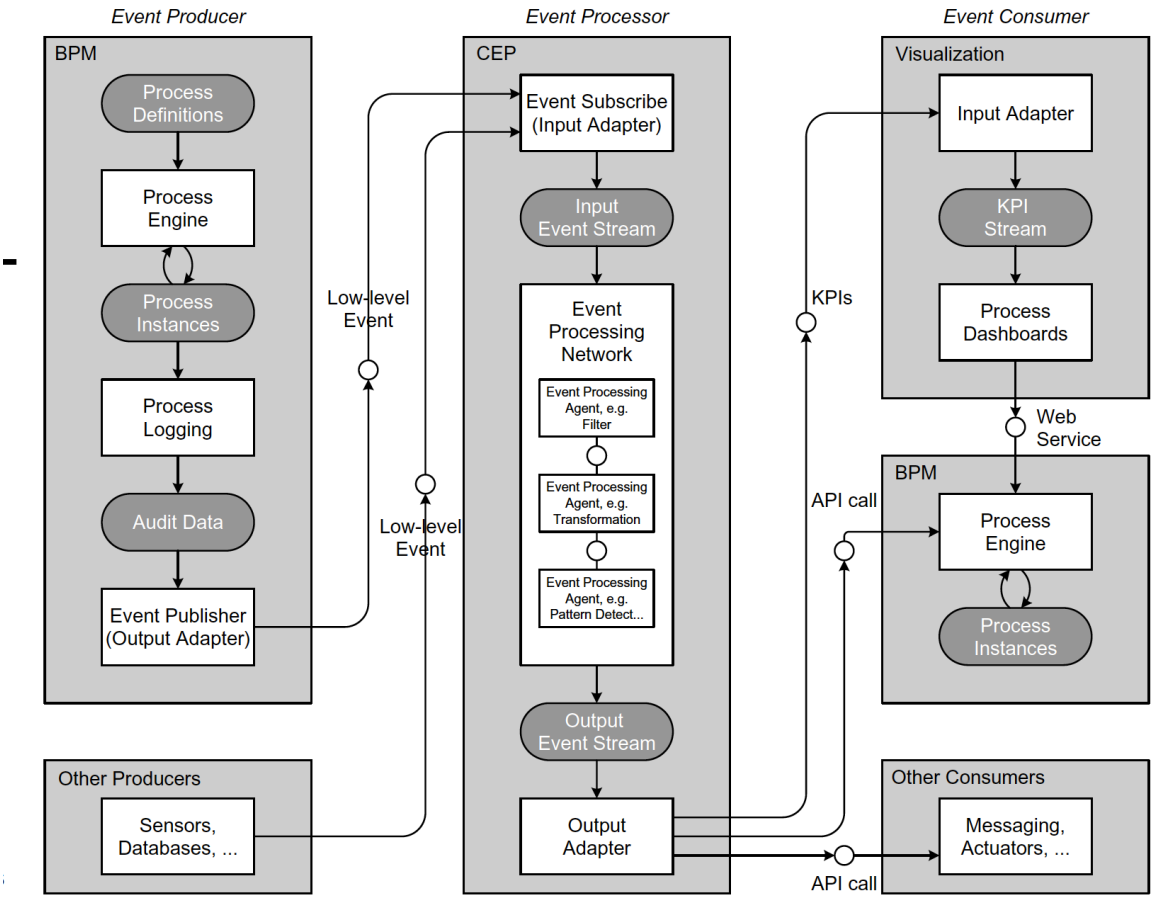
2.2.3.1 Monitoring model

Na de modellering van de processen worden PPI's gemeten en gemonitord [15]. Hiervoor kijkt men eerst naar de doelstellingen die men wil behalen en op basis hiervan bepaalt men dan welke PPI's er gemeten moeten worden. Op basis van deze PPI's kunnen we van deze processen specifieker gaan kijken hoe het zit met tijdsaspecten, kosten, performantie en ook andere procesaspecten. Deze PPI's zijn opgebouwd uit metrieken. Een metriek is "een meetbaar attribuut van een procesinstantie" [6]. Iedere keer dat er een proces uitgevoerd wordt, wordt er één instantie van dit proces gecreëerd. Dus een metriek wordt berekend voor één uitgevoerd proces. PPI's zijn meestal geen simpele metrieken die berekend worden op basis van één instantie, maar een combinatie hiervan. Daarom wordt er in het monitoring model beschreven op basis van welke metrieken deze PPI's samengesteld worden en dit met behulp van formules en/of combinaties van verschillende metrieken. Ook wordt er hier aangegeven op basis van welke events deze metrieken berekend moeten worden. Dit monitor model wordt gebouwd op basis van een modellerinsgstool in de front-end [6].

Om deze events te kunnen monitoren, moeten er events gequeryt worden uit de databases. Nadat deze query's gelopen hebben, moet er in real time gereageerd kunnen worden op deze resultaten. De eenvoudigste manier om dit te doen is op basis van het loggen van events en hierna deze event logs te verwerken. Om steunend op deze event logs bepaalde patronen te vinden, zullen ze vaker verwerkt moeten worden. Dit werd vroeger vooral in batch gedaan, waardoor er heel wat vertragingen optraden. Aangezien er hier een grote hoeveelheid event data aanwezig is, zal dit dus op een andere manier moeten gebeuren.

2.2.3.2 BAM core en complex event processing (CEP)

Hiervoor gebruikt men nu CEP (complex event processing) in samenwerking met de BAM core om patronen uit deze data te halen vlak nadat de data ontstaan. Het event staat hier dus centraal en al de data die opgehaald worden, kunnen dus beschouwd worden als een onderdeel van een event flow [6], [27]. Binnen CEP is een event "een data object dat een opname is van een activiteit die gebeurd is" [28]. De BAM core genereert event patronen op basis van het monitor model dat bepaalt welke events behouden moeten worden om de metrieken te meten waarop de PPI's gebaseerd zijn. Deze patronen worden dan weer doorgegeven aan de CEP engine zodat deze weet welke data geëxtraheerd moeten worden [6].



Figuur 7 onderdelen CEP: [28]

Binnen CEP zijn er drie verschillende componenten, namelijk event producenten, event processoren en event consumenten [28]. Deze componenten en hun bijhorende onderdelen zijn weergegeven in figuur 7.

1. Event producent

Event producenten zijn entiteiten die events genereren die als input zullen dienen voor het echte CEP systeem. De event producent is meestal een onderdeel van het BPM (business process management) systeem. BPM is volgens [28] "een verzameling van tools en methodes om processen te begrijpen en ook te verbeteren". Binnen het BPM systeem is het de 'process engine' die procesinstanties automatisch uitvoert, gebaseerd op bestaande proces definities. Naast deze process engine worden de processen ook door een afzonderlijke component gelogd zoals te zien is in figuur 6. Dit gebeurt op basis van de informatie uit het monitor model. De data die hier gelogd zijn, worden hierna met behulp van een event 'publisher' doorgegeven aan de event processor [28].

2. Event processor

Event processoren zijn entiteiten die de events verwerken die ze van event producenten verkregen.

Dit gebeurt specifieker aan de hand van een complex event processor engine (CEP engine). De event processor, hier de CEP engine, beschouwt enkel de gewenste events en de rest wordt dus ook niet meer gelogd zoals voordien. Deze CEP engine bestaat uit drie onderdelen: de input adapter, het event processing netwerk (EPN) en de output adapter [28].

2.1 Input adapter

De input adapter transformeert inkomende events die afkomstig zijn van de event producent naar het format dat gelezen kan worden door het CEP systeem [28].

2.2 Event processing netwerk

Wanneer de events in het juiste format staan dat verwerkt kan worden door de CEP, zullen deze in een event stroom terechtkomen die als input zal dienen voor het event processing netwerk (EPN). Dit netwerk bestaat uit verschillende event processing agenten (EPA) die gelinkt zijn aan elkaar. Deze EPA's zullen de input event stroom analyseren om bepaalde voorwaarden en patronen te herkennen en indien nodig te melden [28].

Binnen deze EPA's zijn er verder nog drie verschillende soorten: de filter EPA, de transformatie EPA en de patroon herkennende EPA. De filter EPA is beperkt tot het louter filteren van events. De transformatie EPA baseert zich op event patronen die vooraf gedefinieerd zijn. Op basis van deze patronen verwerkt hij input events en transformeert deze naar output events die een functie zijn van deze initiële events. Bij de patroon herkennende EPA wordt er gekeken of er bepaalde patronen aanwezig zijn in de input stroom. De events waarbij deze patronen herkend zijn, zullen dan weer als output dienen. Wanneer we kijken naar het order proces, kan een filter EPA bijvoorbeeld filteren op een specifieke customer ID. De patroon herkennende EPA kan kijken of er dubbele customer ID's zijn in een bepaalde korte periode. De transformatie EPA kan dan weer na de vorige EPA's de events groeperen voor één bepaalde klant, bij wie er bijvoorbeeld de som van het bedrag van de orders als resultaat uitkomt. Zo kunnen events ook samengevoegd worden zodat er patronen herkend worden in een groep van events. Wat de EPA's allen gemeen hebben, is het feit dat ze de input stroom van events analyseren en op basis hiervan zelf ook een output genereren die gebruikt zal worden door de event consument [28].

Aangezien er bij CEP data gehaald worden uit de process engine en men hierbij patronen wil herkennen en enkel de gewenste events wil behouden uit het grotere geheel, kan dit niet gerealiseerd worden met een eenvoudige datataal zoals SQL. Bij SQL worden er data opgehaald op basis van alleenstaande events en bepaalde voorwaarden waar zij aan voldoen. Voor BAM is dit echter niet voldoende aangezien er niet enkel op alleenstaande events gereageerd moet worden, maar op een combinatie van events en hun onderlinge relaties over de tijd heen [6], [29]. Het is dus nodig om een query taal te gebruiken die niet enkel alleenstaande, maar ook samengestelde events kan queryen. Daarom gebruikt de CEP engine CQL (complex query languages) waardoor de data opgehaald kunnen worden op basis van een complexere selectie. Een voorbeeld van zo'n taal is de XCHANGE^{EQ} die voorgesteld wordt in [29]. Aan de hand van deze taal kan men informatie

halen uit websystemen. Deze systemen communiceren events naar andere systemen meestal in de vorm van een XML boodschap. Maar om hier de juiste informatie uit te halen zouden de events ook in combinatie met elkaar gedetecteerd moeten kunnen worden [29].

2.3 Output-adapter

Opdat de event consumenten deze outputstroom van events zouden kunnen gebruiken, moeten deze wel nog getransformeerd worden naar het juiste format. Dit gebeurt door middel van de output-adapter [28].

3. Event consument

Event consumenten zijn entiteiten die de events ontvangen nadat deze verwerkt zijn door de event processor [28]. In de architectuur van Kolar [6] is dit bijvoorbeeld de front-end. Hier kunnen de eindgebruikers de nodige informatie verkrijgen. Aan de hand van de PPI's die voortvloeien uit de CEP engine, kunnen er dus dashboards ontwikkeld worden, waarop beslissingsnemers in één oogopslag deze gegevens kunnen bekijken [6]. Deze dashboards kunnen gekoppeld worden aan onderliggende doelstellingen op basis waarvan de PPI's zijn gekozen. Zo kan men in één opslag zien of de doelstellingen behaald zijn door bijvoorbeeld de waardes van de PPI's te vergelijken met een target waarde. Naast deze visuele weergave van PPI's, kan er op de event data afkomstig uit de CEP engine ook data mining worden toegepast om complexere inzichten te vergaren.

Verder kan er op basis van de output van de CEP een melding gestuurd worden naar de process engine waardoor een proces bijvoorbeeld gestart of gestopt kan worden. Verder kunnen er ook meldingen gestuurd worden naar procesbeheerders van de bijhorende processen wanneer er zich uitzonderingen voordoen. Wanneer er een bepaalde target niet behaald wordt, krijgt de procesbeheerder hier een melding van. Dit kan bijvoorbeeld via mail of een ander kanaal. De regels hiervoor zijn gedefinieerd in het monitor model [6], [28].

2.2.4 Doel BAM

Het doel van BAM is ervoor zorgen dat processen beter gemeten en geoptimaliseerd kunnen worden. Na iedere significante verandering die optreedt in de omgeving of in de processen zelf, zouden deze dus geoptimaliseerd moeten kunnen worden. Doordat men BAM gebruikt, heeft men real-time informatie over de processen. Hierdoor kunnen er sneller beslissingen genomen worden zodat de processen voldoen aan de huidige marktvereisten en kunnen processen ook sneller aangepast worden wanneer er optimalisatie nodig is vanuit het bedrijfsperspectief. Wanneer een proces bijvoorbeeld niet efficiënt genoeg verloopt en dit kosten met zich meebrengt, kan men aan de hand van BAM bekijken waar dit aan ligt. Het snel identificeren van relevante proces events is dan ook noodzakelijk om voldoende snel te kunnen reageren en veranderen [6], [12], [27].

3. Definiëring performantie-indicatoren

Procesperformantie-indicatoren zijn de bouwstenen van business process intelligence. Op basis van deze indicatoren worden er meetsystemen gebouwd.

In het ontwikkelen van deze performantiemeetsystemen zijn er volgens Bourne et al. [10] drie belangrijke fases, namelijk:

1) Het definiëren van performantie-indicatoren
2) Het implementeren van performantie-indicatoren
3) Het gebruik van performantie-indicatoren

In ons onderzoek zullen we voornamelijk op het definiëren van de indicatoren focussen. Deze fase kan opgedeeld worden in twee belangrijke onderdelen: het 'identificeren van hoofddoelstellingen en bijhorende performantie-indicatoren' en 'het modelleren van de performantie-indicatoren zelf'. De eerste deelfase hiervan vertaalt de behoeften van de belangrijkste stakeholders in specifieke doelstellingen en performantie-indicatoren die gelinkt kunnen worden aan deze doelstellingen. Dit wordt gedaan door eerst de missie, visie en strategie van het bedrijf te bepalen. Op basis van deze strategie kiest het management strategische doelstellingen die men wil behalen. Om dit echter in de praktijk te kunnen doorvoeren, worden er voor de verschillende afdelingen operationele doelstellingen gezet. Om te kijken of deze gehaald worden, worden er performantie-indicatoren uitgekozen voor ieder van hen [5], [10].

In dit werk ligt de focus op de tweede deelfase, het modelleren van de performantie-indicatoren. Hier worden de procesperformantie-indicatoren gedefinieerd op basis van een metamodel. Een metamodel is een model dat de concepten en de relaties hiertussen definieert die later gebruikt worden in een toegepast model. Het metamodel dat wij zullen bouwen is gebaseerd op drie metamodellen: PPINOT, MetricML en BAM. In PPINOT worden PPI's gedefinieerd en in MetricML ligt de focus hoofdzakelijk op de definiëring van KPI's. Om te zorgen dat de PPI's die we gaan definiëren later ook real-time geanalyseerd en gevisualiseerd kunnen worden, gaan we ook gebruik maken van een metamodel dat een uitbreiding van Business Activity Monitoring (BAM) op BPMN voorstelt [12]. Bij BAM wordt de bedrijfsperformantie gemeten aan de hand van de real-time monitoring van bedrijfsprocessen. Dit gebeurt bij BAM ook aan de hand van indicatoren en aangezien het hier om een BPMN extensie gaat, zijn dit ook PPI's die gemodelleerd worden [30]. Daarom zal deze abstracte syntax ook gebruikt worden in ons metamodel naast deze van PPINOT en MetricML om tot een beter resultaat te komen. Ieder van deze drie modellen wordt verder besproken. Erna gaan we deze modellen met elkaar vergelijken om de belangrijkste verschillen te identificeren. Deze verschillen zullen we ook gebruiken om uiteindelijk een nieuw metamodel te bekomen.

3.1 MetricM

In de literatuur focust men vooral op performantie-indicatoren die de operationele performantie van de organisatie representeren, namelijk de key performantie-indicatoren (KPI's). In de paper van Frank et al. wordt er gefocust op het modelleren van deze KPI's [14].

3.1.1 MetricM modelleermethode

Om deze KPI's te kunnen modelleren, wordt er in [14] een modelleermethode, MetricM, ontwikkeld met een bijhorende domein-specifieke modelleertaal, MetricML. MetricM is een methode die het ontwerp en gebruik van performantie-indicatoren en overeenkomstige performantiemeetsystemen ondersteunt. Deze MetricM modelleermethode is gebouwd op basis van MEMO 'een multi-perspective enterprise modeling method'. Modelleermethodes zoals MEMO en MetricM die gebruik maken van een domein-specifieke modelleertaal, zorgen er in het algemeen voor dat performantiemeetsystemen effectiever en efficiënter gebouwd en geïnterpreteerd kunnen worden. Wanneer men hierbij de relatie tussen verschillende performantie-indicatoren niet uit het oog verliest, zullen er zich minder fouten voordoen in deze systemen. Fouten die hierdoor vermeden kunnen worden zijn bijvoorbeeld het fout interpreteren van de indicatoren zelf en de relaties onderling [14], [16].

3.1.2 MetricML

Net zoals MetricM bevat de MEMO-modelleermethode een bijhorende modelleertaal, namelijk de MEMO meta modelleertaal. Op deze meta taal is de MetricML taal gebaseerd. MetricML is zoals eerder al vermeld een domein-specifieke modelleertaal. Bij een domein-specifieke modelleertaal worden er vaak bepaalde technische beperkingen vooropgesteld, die anders pas manueel toegevoegd konden worden door de ontwikkelaar. Dit zorgt voor een nauwkeuriger model en minder kans op manuele fouten. Het bouwen van een dergelijke domein-specifieke modelleertaal is echter een moeilijke en belangrijke taak die zorgvuldig moet worden uitgevoerd. Op voorhand moet er namelijk bepaald worden voor welk domein deze taal in de praktijk zal dienen. Dit zorgt ervoor dat deze modellen enorm expressief zijn, maar ook moeilijker aan te passen zijn naar een ander domein. We zullen dus de complexiteit van het bouwen van een dergelijke taal moeten afwegen tegen de voordelen.

3.1.3 Vereisten MetricML

In MetricM worden enkel vereisten opgesteld in verband met de methode die deze definiëring ondersteunt. Aangezien wij hier geen methode zullen ontwikkelen, bekijken wij deze vereisten hier niet. Wel zullen we kijken naar de vereisten die voorgesteld zijn in MEMO aangezien MetricML hierop gebaseerd is.

In MEMO [31] worden er vereisten voorgesteld waaraan een meta modelleertaal moet voldoen, wanneer we voor een domein-specifieke taal kiezen [14], [31]. Deze vereisten kunnen opgedeeld worden in drie categorieën, namelijk formele vereisten, vereisten die gefocust zijn op de gebruiker en applicatie georiënteerde vereisten. Formele vereisten zijn vereisten die opgesteld worden om

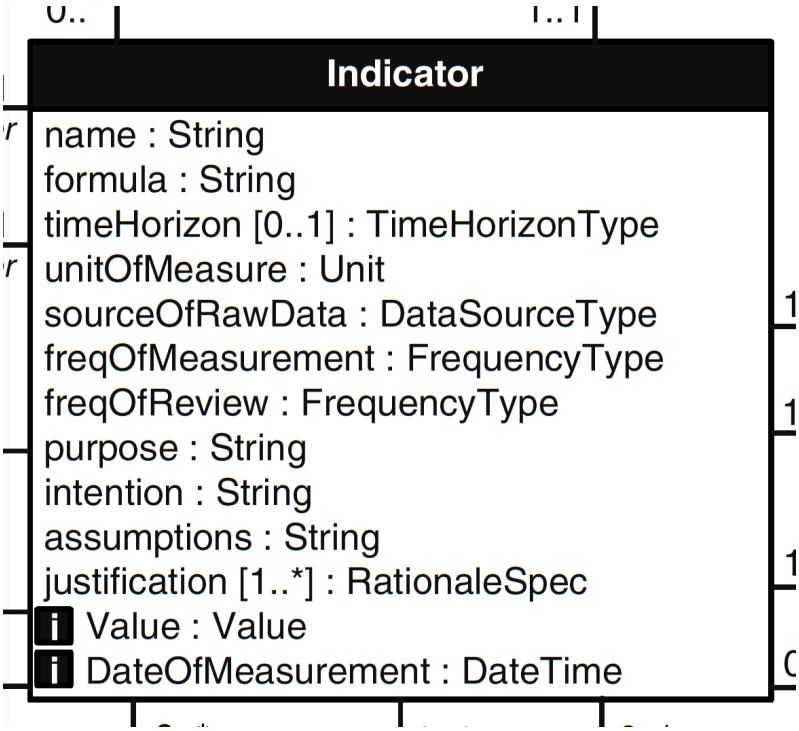
ervoor te zorgen dat de modelleertalen formeel gespecifieerd kunnen worden. Vereisten die gefocust zijn op de gebruiker hebben betrekking op de interpretatie van de concepten van de taal en de voorstelling hiervan. De applicatie georiënteerde vereisten tenslotte draaien om het feit dat de metamodelleertaal verschillende talen moet kunnen definiëren. Deze vereisten zijn weergegeven in tabel 3 [31].

Formele vereisten	
F1	De specificatie van een meta modelleertaal moet een formele specificatie van zijn abstracte syntax bevatten.
F2	In het ideale geval, zou er een formele specificatie van de semantiek van een meta modelleertaal moeten zijn. De specificatie moet dus compleet en correct zijn. Aangezien een complete formalisatie van semantiek soms te veel moeite zal vereisen, kan het voldoende zijn om de semantiek te specifiëren op een manier die als ondubbelzinnig beschouwd wordt door experten.
F3	Om formalisatie en begrijpbaarheid te bevorderen, zou een meta modelleertaal simpel genoeg moeten zijn.
F4	Om bij te dragen aan een precieze of formele semantiek, moet de meta meta modelleertaal die gebruikt is om de meta modelleertaal te specifiëren een formele taal zijn. De meta modelleertaal moet duidelijk eenvoudiger zijn dan de modelleertaal die hij moet beschrijven.
Gebruiker vereisten	
U1	De concepten van een meta modelleertaal zouden moeten overeenkomen met concepten waar modelleerexperten vertrouwd mee zijn. Aangezien concepten die gebruikt worden om datamodellen of klassediagrammen te maken gekend zijn bij de toekomstige gebruikers, lijken deze erg geschikt voor dit doel.
U2	De talen die gebruikt worden op verschillende niveaus van abstractie, zoals een meta modelleertaal of een modelleertaal, moeten duidelijk gescheiden worden. Eén taal gebruiken voor verschillende niveaus van abstractie zou vermeden moeten worden.
Applicatie vereisten	
A1	Een meta modelleertaal zou alle concepten moeten aanbieden die nodig zijn om talen te specifiëren die de structuur, processen etc. binnen een bedrijf modelleren.
A2	Een meta modelleertaal zou beperkt moeten zijn tot concepten die vereist zijn om een taal te ontwerpen.

Tabel 3 vereisten MEMO: [31]

Wanneer we kijken naar de domein-specifieke modelleertaal, MetricML, zien we dat deze taal bestaat uit zowel een metamodel als een grafische notatie. Hier gaan we verder op het metamodel dat zal bijdragen tot ons eigen model. In dit model worden voornamelijk performantie-indicatoren

gedefinieerd. Indicatoren die gelinkt zijn aan processen kan men hier ook definiëren, maar deze zijn niet rechtstreeks gelinkt aan de proceselementen [14], [31].



Figuur 8 centrale entiteit MetricML: [14]

Het metamodel van MetricML leggen we hier verder uit. Het centrale concept hiervan is dat van de indicator, dat weergegeven is in het deel van het metamodel dat te zien is op figuur 8. Deze entiteit beschrijft de belangrijkste eigenschappen van een indicator type. De verschillende attributen hiervan worden hieronder besproken [14].

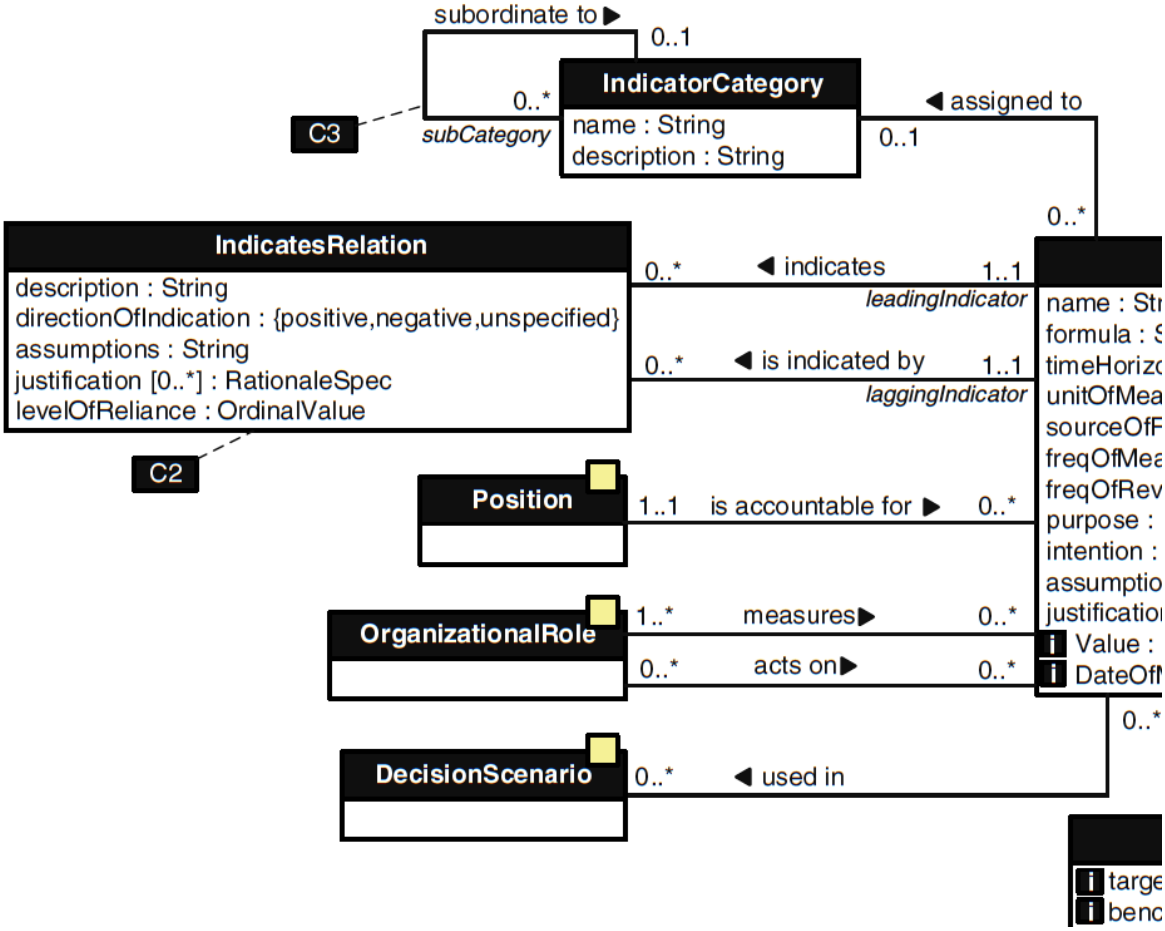
Maar eerst bespreken we nog de bijzondere datatypes die in figuur 8 te vinden zijn. Hierbij heb je de 'TimeHorizonType', 'Unit', 'DataSourceType' en 'FrequencyType'. Dit zijn eigenlijk gewoon String uitdrukkingen die een waarde moeten hebben uit een bepaalde lijst of een waarde die aan bepaalde voorwaarden voldoet. Zo heb je bijvoorbeeld de 'Unit' dat een percentage, seconde, dollar etc. kan zijn. Echter is er voor deze datatypes wel de exacte achterliggende betekenis niet gedefinieerd in MetricM [14].

Verder heb je ook nog het datatype 'RationaleSpec' dat een apart geval is. Aan de hand van dit datatype wil men bepalen hoe goed de hypothese die achterliggend is aan een indicator gerechtvaardigd kan worden. Dit datatype is een combinatie van drie justificatie procedures op basis waarvan de justificatie gespecifieerd kan worden [14].

- De naam beschrijft de naam van de indicator.
- De formule beschrijft in woorden hoe de KPI berekend moet worden.
- De tijdshorizon kan optioneel gebruikt worden ter ondersteuning van de meeteenheid. Dit kan uitgedrukt worden als bijvoorbeeld dag, week, maand of jaar. Hier kan een PPI dus

uitgedrukt worden als een bedrag per jaar ($ / jaar) bijvoorbeeld.

- De meeteenheid geeft de bijhorende eenheid weer van de waarde van de indicator. Dit kan dus bijvoorbeeld een percentage, een munteenheid, een tijdseenheid, een index of geen eenheid zijn.
- De bron van de ruwe data geeft de databron weer waar de data voor het berekenen van de PPI afkomstig van is. Dit kan zowel via een beschrijving of een 'fully qualified data manipulation language statement' zoals bijvoorbeeld SQL.
- De meetfrequentie geeft aan hoe vaak de PPI gemeten moet worden. Dit kan dus bijvoorbeeld om de twee weken zijn.
- De review frequentie bepaalt hoe vaak de PPI en de bijhorende eigenschappen herbekeken moeten worden.
- Het doel geeft aan wat het doel is dat men wil bereiken en waar deze PPI op gebaseerd is.
- De intentie geeft weer wat het management wil bereiken op basis van de indicator definitie.
- De assumpties geven de assumpties weer die bepaald zijn voor de PPI.
- De justificatie geeft de beschrijving weer die verklaart waarom het nuttig is om deze PPI te gebruiken.
- De waarde is de waarde van de PPI. Dit is een attribuut dat niet op het type niveau wordt bepaald maar wel op het instantie niveau. Omdat er hier per instantie een waarde gekoppeld moet kunnen worden.
- De meetdatum tenslotte is de datum waarop de PPI gemeten wordt. Dit is ook een attribuut dat bepaald is per instantie [14].



Figuur 9 metamodel MetricML deel 1: [14]

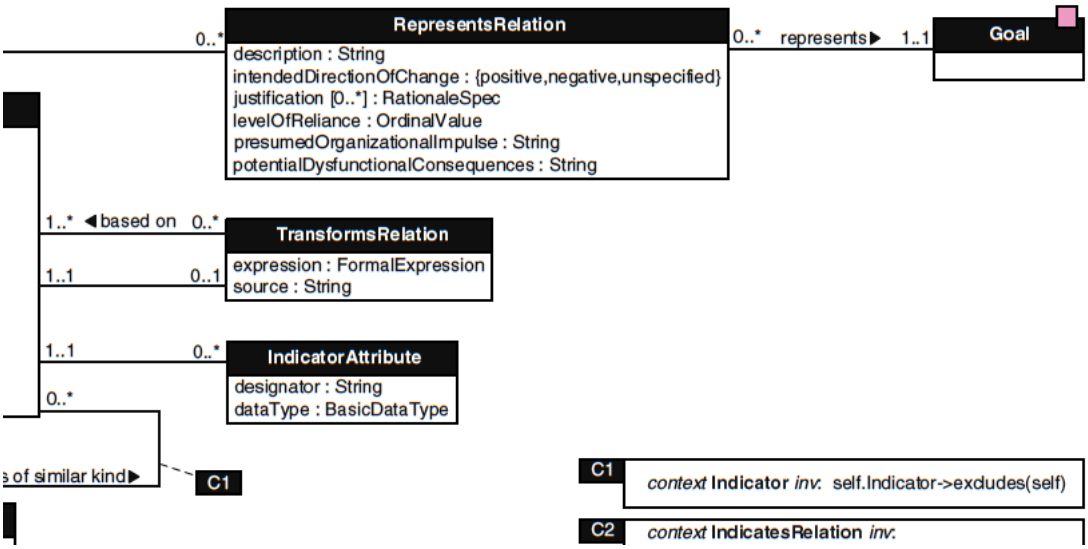
Het volgende onderdeel van het metamodel dat we zullen bespreken is weergegeven op figuur 9 en verdere informatie hieromtrent volgt hieronder.

De indicator types die bepaald zijn aan de hand van een indicator entiteit kunnen ook toegewezen worden aan categorieën. Deze categorieën worden bepaald aan de hand van de indicator categorie en groeperen indicator types op basis van criteria die bepaald zijn door de gebruiker [14].

Ook zijn er vaak relaties tussen de verschillende indicatoren. Dit wordt aangegeven door 'IndicatesRelation'. Hierbij is er een indicator ('leadingIndicator') die de waarde van een andere indicator ('laggingIndicator') beïnvloedt. De verdere informatie in verband met deze relatie is aangegeven door de attributen van 'IndicatesRelation'. Ten eerste wordt de relatie beschreven aan de hand van een beschrijving. Ook geeft men aan of de relatie positief, negatief of onbepaald is bij richting van de indicatie. Hierna geeft men aan in assumpties welke assumpties er gemaakt zijn over deze relatie. Dit kan bijvoorbeeld zijn 'PPI1 en PPI2 zijn positief gecorreleerd enkel wanneer de omzet lager is dan 400.000 euro'. In justificatie kan er indien dit gewenst is, een reden aangegeven worden waarom deze relatie voldoende nut heeft. Tenslotte wordt er nog meegegeven hoe betrouwbaar het veronderstellen van deze relatie daadwerkelijk is. Dit doet men aan de hand van een schaal (bv. laag – gemiddeld – hoog) gebruikt in het niveau van betrouwbaarheid [14].

Verder worden er ook resources aangegeven die betrekking hebben tot de PPI. Ten eerste wordt er in positie aangegeven wie er verantwoordelijk is voor de PPI. Dit is altijd één resource die wel verantwoordelijk kan zijn voor meerdere PPI's. Deze persoon neemt de finale beslissingen over het achterliggend proces en wordt geacht al de beslissingen van anderen goed- of af te keuren. Naast de verantwoordelijke worden ook resources aangegeven die de PPI meten en erop reageren. Dit wordt aangegeven met behulp van de organisationele rol [14].

Verder kunnen de PPI's ook nog gebruikt worden in een beslissingsscenario. Hier kan men dus de PPI's gebruiken in een uitgewerkt scenario om bepaalde beslissingen te nemen in verband met bijvoorbeeld het aantal resources.



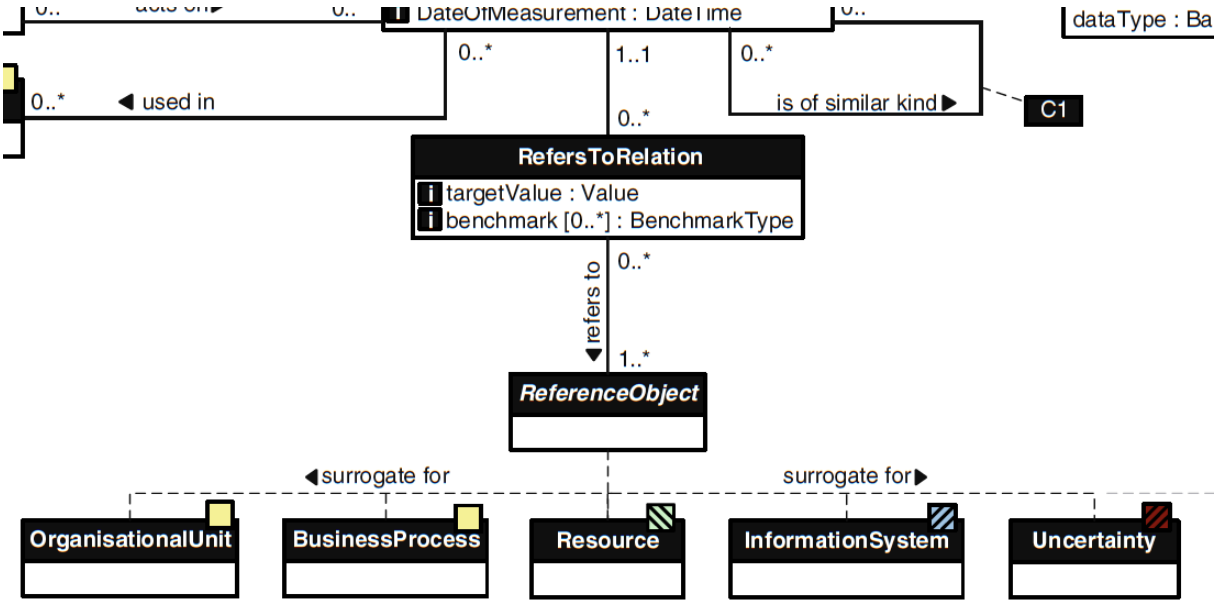
Figuur 10 metamodel MetricML deel 2: [14]

Vervolgens zien we op figuur 10, vier elementen die gekoppeld worden aan de indicator. Deze elementen zijn 'RepresentsRelation', 'Goal', 'TransformsRelation' en 'IndicatorAttribute'. Het eerste onderdeel dat enorm belangrijk is, is de doelstelling die bepaald is in het bijhorende onderdeel. Aangezien iedere PPI gebaseerd is op een doelstelling, wordt de relatie tussen de PPI en de doelstelling aangegeven door 'RepresentsRelation'. Hier wordt de relatie beschreven in de beschrijving en de verwachte verandering van het behalen van de doelstelling in 'intentedDirectionOfChange'. Hier wordt aangegeven of een verandering in de waarde van de PPI een invloed heeft op het behalen van de doelstelling en of deze correlatie positief of negatief is. Verder kan hier de relatie ook weer gerechtvaardigd worden, met behulp van een verklaring in justificatie. Ook wordt hier weer de mate van betrouwbaarheid aangegeven in het niveau van betrouwbaarheid. Verder wordt er in 'presumedOrganizationalImpulse' aangegeven welke invloed het gebruik van de indicator heeft op het gedrag binnen de organisatie. Dus zou men door het gebruik van een bepaalde indicator bijvoorbeeld 'schuldgraad' misschien minder leningen aangaan als bedrijf omdat men hier nu bewuster van is. Tenslotte kunnen ook nog mogelijke problematische gevolgen die ontstaan door het gebruik van een indicator meegegeven worden in

'potentialDysfunctionalConsequences' [14].

Verder kunnen er op basis van de indicatoren ook nieuwe indicatoren gebouwd worden door gebruik te maken van wiskundige functies die toegepast worden op bestaande indicatoren. Dit wordt uitgevoerd in 'TransformsRelation'. Hierbij is er ook een attribuut bron dat aangeeft waar de logische relatie gedocumenteerd is. Wanneer men bijvoorbeeld het kost-efficiëntie ratio wil bekijken als indicator, weet men uit de literatuur dat dit berekend kan worden door het delen van de kost door de efficiëntie [14].

De laatste klasse van figuur 10 is 'IndicatorAttribute'. Aan de hand van deze klasse kan men nieuwe attributen toevoegen aan de indicator. De naam van het attribuut wordt dan meegegeven in het attribuut 'designator'. Het datatype ervan, wordt gedefinieerd in 'dataType' [14].



Figuur 11 metamodel MetricML deel 3: [14]

In het laatste onderdeel van het metamodel kunnen PPI's gekoppeld worden aan verschillende referentie objecten. Deze geven de setting weer van de organisatie zodat de PPI beter geïnterpreteerd kan worden. De relatie tussen de indicator en het referentie object wordt aangegeven door 'RefersToRelation'. Hierbij wordt er ook een targetwaarde bepaald en indien gewenst ook een benchmark. Bij de benchmark wordt de waarde van de indicator vergeleken met de waarde van een gelijkaardige indicator in bijvoorbeeld een ander bedrijf of andere afdeling. De referentie objecten die hier aanwezig zijn en dus de organisatorische context bepalen, zijn organisationele eenheid, bedrijfsproces, resource, informatiesysteem en onzekerheid. Dit deel is hierboven weergegeven op figuur 11 [14].

3.2 PPINOT

Aan de hand van PPI's kunnen bedrijfsprocessen geoptimaliseerd worden en daarom zijn deze ook van groot belang. Hieronder bespreken we dan ook het onderzoek van Del Rio Ortega [15] in verband met het definiëren van procesperformantie-indicatoren. Deze definiëring zullen we ook zelf ondersteunen aan de hand van een metamodel dat we zullen bouwen in hoofdstuk 4.

In de paper van Del Rio Ortega, is het doel: "Een set van technieken ontwikkelen om het definiëren en analyseren van procesperformantie-indicatoren te ondersteunen.". Om dit doel te bereiken wordt een metamodel ontwikkeld dat de definiëring van deze procesperformantie-indicatoren ondersteunt. Om deze procesperformantie-indicatoren te definiëren is er eerst een typologie bepaald. Deze typologie stelt de verschillende soorten indicatoren voor die gemodelleerd kunnen worden aan de hand van het bijhorende PPINOT metamodel. Dit metamodel is gebouwd aan de hand van 'unified modeling language'. Naast de abstracte syntax in de vorm van de PPINOT-modelleertaal is er ook een concrete syntax, namelijk de PPINOT grafische notatie. Bij deze notatie worden PPI's grafisch voorgesteld op een bedrijfsprocesdiagram in de BPMN-taal om een duidelijk beeld te schetsen [9].

Verder is er hier ook nog ondersteuning voor het definiëren en analyseren van de PPI's aanwezig in de vorm van de 'PPINOT tool suite'. Deze tool suite bevat tools zoals een grafische editor om PPI's weer te geven op bedrijfsprocesdiagrammen zoals BPMN modellen. Verder is er ook nog een tool die de analyse ondersteunt, maar hier wordt enkel een rapport gecreëerd en is er dus nog geen sprake van dashboardcreatie. Dit kan echter later nog wel gebeuren aan de hand van de informatie die er gebruikt wordt om het rapport te maken. Want een dashboard helpt bij de ondersteuning van het nemen van beslissingen [9]. Echter gaan wij ons hier focussen op de typologie en de abstracte syntax, namelijk het PPINOT metamodel dat hieronder beschreven is.

3.2.1 PPINOT typologie

Eerst en vooral beschrijven we hier de typologie met de soorten indicatoren die tevens in het metamodel gemodelleerd zijn. In PPINOT wordt de waarde van een PPI gemeten aan de hand van een meetdefinitie. Deze meetdefinitie verschilt naargelang het type indicator en daarom is een typologie hier ook noodzakelijk.

De typologie van [15] Del Rio Ortega, is hier weergegeven op figuur 12. Hier is te zien dat er twee dimensies zijn die bepalen tot welk type de indicator behoort. Deze types worden dan verder gegroepeerd in drie klasses.

Op de Y-as wordt er bepaald of de indicator voor één of meerdere procesinstanties gedefinieerd wordt. Op de X-as staan er verschillende categorieën die aangeven wat voor soort indicator er berekend wordt [15].

De eerste klasse, de basisindicator, bevat tijdsindicatoren, telindicatoren, conditie indicatoren en data indicatoren die gemeten worden voor één procesinstantie [15].

Een **tijdsindicator** meet de tijdsduur tusen twee tijdstippen waarop zich bepaalde gebeurtenissen voordoen. Deze tijdsduur kan hier gemeten worden voor een specifieke activiteit, proces of een deel van een bepaald proces. Een voorbeeld hiervan kan de tijd zijn tussen het moment waarop een activiteit A toegewezen is en het moment waarop deze activiteit beëindigd is [15].

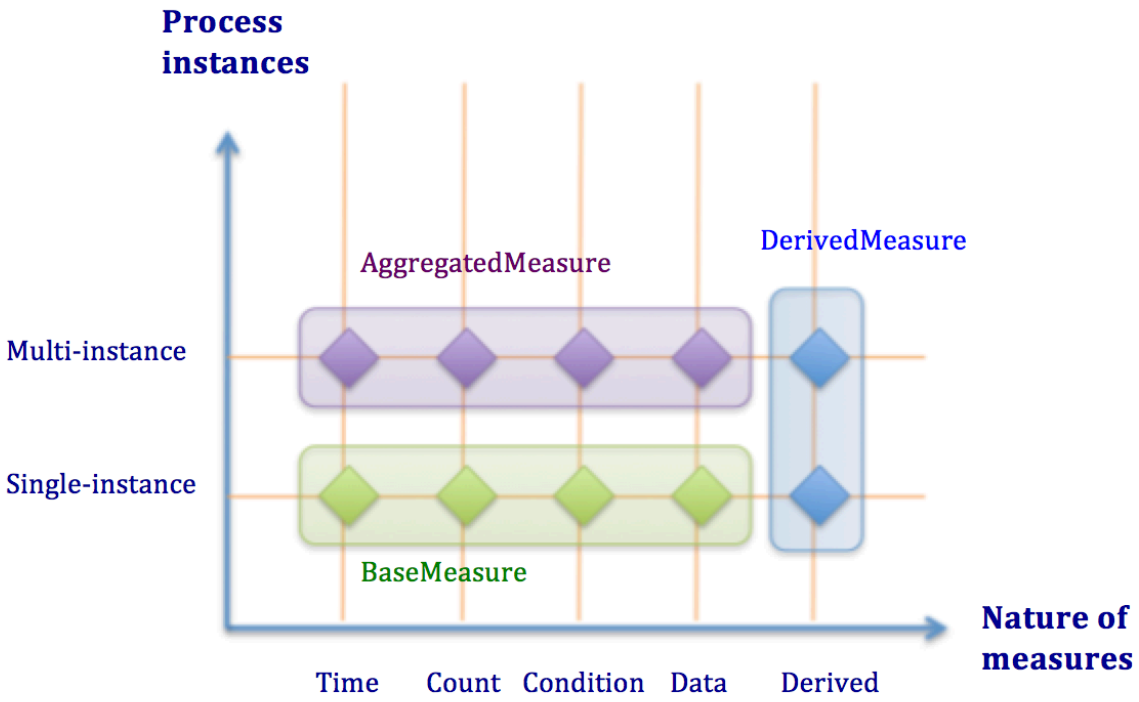
Een **telindicator** telt hoe vaak een BP element verandert naar een bepaalde staat of hoe vaak een event getriggerd is. Hier kan men bijvoorbeeld tellen hoe vaak activiteit B binnen een procesinstantie opgeschort is geweest [15].

Een **conditie indicator** meet of lopende procesinstanties aan een bepaalde conditie voldoen of dat beëindigde procesinstanties aan deze conditie voldeden. Er zijn hierbij twee types mogelijk qua condities, namelijk condities die verwijzen naar de staat van een BP (bedrijfsproces) element of een restrictie van een bepaald data object. Een voorbeeld hiervan kan zijn 'het voldoen aan de conditie prijs>500 voor het data object offerte'. Voor het andere type kan een voorbeeld 'het voldoen aan de conditie dat activiteit C momenteel in de staat voltooid is' [15].

Een **data indicator** meet de waarde van een specifiek onderdeel van een data object. Hier kan er dus een PPI bepaald worden die de waarde bevat van de leverancier in het data object offerte [15].

De tweede klasse is deze van de geaggregeerde indicator. Een **geaggregeerde indicator** wordt berekend door het aggregeren van één van de hiervoor besproken indicatoren in verschillende process. Dit gebeurt op basis van verschillende functies zoals het minimum, maximum, de som, het gemiddelde etc. Men kan dus bijvoorbeeld de gemiddelde duur van activiteit D bepalen op basis van de duur van verschillende instanties [15].

De laatste klasse bevat de afgeleide indicatoren. Een **afgeleide indicator** wordt berekend aan de hand van een wiskundige functie die uitgevoerd wordt op één of meerdere indicatoren. Hierbij maakt men een onderscheid tussen het toepassen van een formule op single-instance indicatoren of multi-instance indicatoren. Een voorbeeld hiervan kan zijn dat men wil weten hoeveel langer activiteit A duurt ten opzichte van activiteit B. Dit kan dan bepaald worden als (duurA / duurB). Dit zijn allebei de tijdsindicatoren die de volledige duur (tussen de staat toegewezen en voltooid) van activiteit A en B bepalen [15].



Figuur 12 typologie PPINOT categorisatie: [15]

3.2.2 Vereiste eigenschappen PPINOT

Voor we het metamodel bestuderen, kijken we eerst nog naar de vereisten. Zo worden hier in [15] enkele belangrijke eigenschappen voorgesteld waaraan het PPINOT metamodel moet voldoen. Er zijn hier richtlijnen voor de drie aspecten die we hierboven hebben vermeld, namelijk de definiëring, voorstelling en analyse van de PPI's. Aangezien we ons hier toeleggen op de definiëring van PPI's, bekijken we de bijhorende eigenschappen [15].

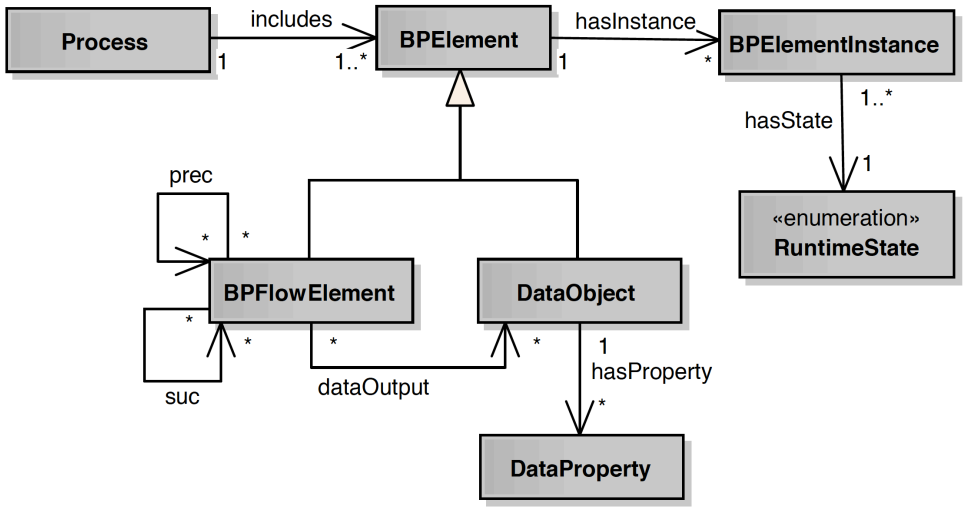
De eerste eigenschap is "ondubbelzinnigheid en volledigheid". Wanneer er enkel een taal uitdrukking gebruikt wordt om PPI's te definiëren, ontbreekt er vaak informatie voor de berekening van de PPI's of is de uitdrukking dubbelzinnig. Bijvoorbeeld wanneer er een PPI is met de volgende definitie "duratie van het maken van een offerte". Als er een offerte gemaakt kan worden door verschillende rollen in het proces, is het al niet duidelijk om welke activiteit binnen het proces het hier gaat. Hier kan er dan bijvoorbeeld twee maal een activiteit zijn binnen het proces waarin de tekst "maak offerte" voorkomt als onderdeel van de activiteitnaam. Waar de informatie vandaan gehaald kan worden staat hier ook niet in de definitie vermeld. Daarom is het dus noodzakelijk om een ondubbelzinnige en volledige definitie te hebben voor de PPI's [15].

De tweede eigenschap is "traceerbaarheid naar bedrijfsprocessen". Wanneer PPI's gedefinieerd worden zonder een link naar de overeenkomstige bedrijfsprocessen, ontstaan er verschillende problemen. Het eerste probleem is de moeilijkheid om PPI's en de bedrijfsprocessen op elkaar af te stemmen. Een voorbeeld hiervan kan zijn dat een PPI gemeten wordt op basis van een activiteit

die verwijderd wordt. Wanneer de PPI niet gelinkt is aan het proces, ziet men dus niet onmiddellijk dat deze activiteit verwijderd is. Zonder een duidelijke link is het ook moeilijk om informatie te verzamelen die nodig is om de PPI te berekenen aangezien men niet weet waar men in het proces moet meten. Het linken van de PPI met het bedrijfsproces is dus noodzakelijk om bijvoorbeeld de waardes van de PPI's automatisch te kunnen berekenen [15].

De derde eigenschap is "expressivieit". Bij het definiëren van PPI's moet men sterk rekening houden dat het model dat de PPI's gaat definiëren voldoet aan drie eigenschappen. De eerste eigenschap draait om het feit dat het nodig is dat de PPI's voldoen aan de SMART (specifc – measurable – assignable – realistic – time-related) criteria. Verder is het ook belangrijk voor de tweede eigenschap dat de type indicatoren die volgens Del Rio Ortega [15] gevonden zijn in de literatuur en in organisaties uitgedrukt kunnen worden aan de hand van het metamodel. Tenslotte voor de laatste deeleigenschap die bijdraagt aan de expressiviteit is het nodig dat het metamodel voldoende filters kan definiëren. Deze filters bepalen op basis van welke instanties de PPI's berekend worden [15].

3.2.3 PPINOT metamodel



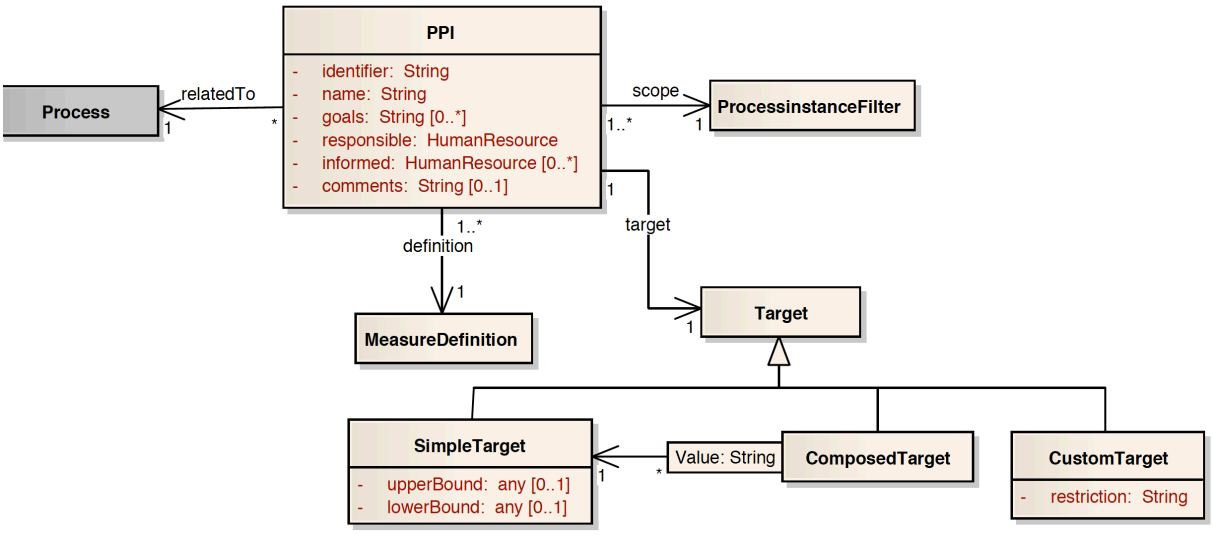
Figuur 13 metamodel BPMN: [15]

Als eerste onderdeel van het PPINOT metamodel wordt er een abstracte BPM (business process modelling) taal voorgesteld. Deze is weergegeven op figuur 13. Hier wordt aangegeven dat een proces bestaat uit één of meer BP (bedrijfsproces) elementen. Deze BP elementen hebben verschillende instanties, waarbij een instantie steeds gekoppeld wordt aan exact één staat.

Binnen deze BP elementen is er een onderscheid gemaakt tussen twee types, namelijk een BP flow element en een data object. In dit werk is niet aangegeven welke elementen onder de categorie BP flow element vallen, maar wanneer men zich baseert op BPMN 2.0 gaat het hier om events (vb. start event), gateways en activiteiten. In deze syntax wordt ook de volgorde van deze elementen

binnen een proces aangegeven door 'prec', het element dat ervoor komt en 'suc' het element dat erna komt [15].

Verder heb je een data object dat bepaalde eigenschappen heeft. De output van een BP flow element wordt bewaard in een bijhorend data object [15].



Figuur 14: PPINOT metamodel kern [15]

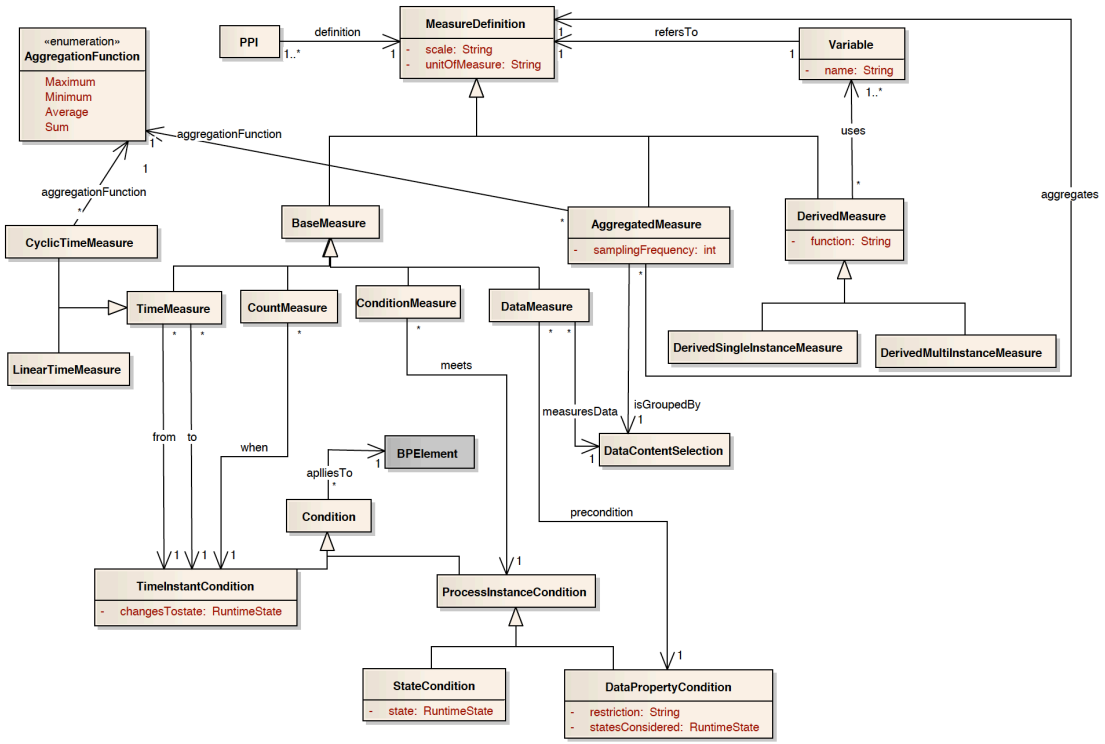
Nadat BPM gedefinieerd wordt met behulp van een abstracte syntax, worden de PPI's hier gemodelleerd aan de hand van een abstracte syntax. Dit is weergegeven op figuur 14. Hier heb je de PPI entiteit waarbij er kerngegevens worden meegegeven in de vorm van attributen. Hieronder is een overzicht te vinden van de attributen met de bijhorende uitleg zoals meegegeven in [15].

De identificeerder is de unieke code die meegegeven wordt aan een PPI zodat er geen verwarring kan optreden met een andere gelijkaardige PPI.
De naam is de naam die gegeven wordt aan de PPI.
De doelstellingen zijn de strategische of operationele doelstellingen waaraan een PPI gekoppeld is.
Verantwoordelijke geeft weer welke persoon verantwoordelijk is voor de PPI.
Geïnformeerde zijn dan weer de personen die naaste de verantwoordelijke ook geïnformeerd moeten worden over deze PPI.
Tenslotte heb je ook nog commentaar waar een willekeurige tekst kan meegegeven worden over andere informatie [15].

Deze PPI is ook gekoppeld aan het bijhorende proces. Verder is er ook een koppeling met een filter die de scope bepaalt van de PPI. Hier wordt dus bepaald op basis van welke instanties de PPI gemeten zal worden. Dit wordt in een volgend onderdeel van het metamodel nog verder uitgewerkt [15].

Ook kan iedere PPI gekoppeld worden aan één of meerdere targets. In dit model zijn er drie verschillende soorten namelijk de simpele target, de samengestelde target en de gepersonaliseerde target. Bij de simpele target wordt er een bereik gedefinieerd waar de waarde van de PPI in hoort te vallen. Dit wordt gedefinieerd aan de hand van een onder- en/of bovengrens. Wanneer er bijvoorbeeld enkel een ondergrens wordt gedefinieerd, is het in orde wanneer de PPI een waarde heeft groter dan deze grens. Voor de samengestelde target kunnen verschillende bereiken gecombineerd worden. Tenslotte is er ook nog de gepersonaliseerde target waarbij de target voorgesteld wordt aan de hand van een bepaalde wiskundige functie waaraan de PPI moet voldoen. Men kan dus bijvoorbeeld als target bepalen dat de waarde van de PPI moet voldoen aan de volgende vergelijking '7x > 500' [15].

Tenslotte heb je de meetdefinitie, die definieert hoe een indicator gemeten wordt. Dit wordt hieronder verder uitgewerkt in het onderdeel van het metamodel weergegeven op figuur 15 [15].



Figuur 15: PPINOT metamodel meetdefinitie [15]

In dit onderdeel van het model worden de meetdefinities van de verschillende types PPI's gemodelleerd. De types die hier beschouwd worden zijn bepaald in de typologie van [15] Del Rio Ortega. De gemodelleerde types zijn de tijdsindicator, de telindicator, de conditie indicator, de data indicator, de geaggregeerde indicator en de afgeleide indicator. De typologie is hierboven reeds beschreven. Voor de duidelijkheid van de meetdefinities worden de beschrijvingen van deze types hier echter herhaald [15].

De **tijdsindicator** meet de duur tussen twee tijdstippen. Deze tijdstippen kunnen overeenkomen met de verandering van een bepaalde staat van een BP element activiteit, pool of data object of

met het triggeren van een bepaald event. Dit is hier gemodelleerd met behulp van de 'TimeInstantCondition' die gekoppeld is aan een BP element [15].

Verder is er ook bepaald hoe men de tijd wil meten wanneer er sprake is van een loop. Dit kan lineair of cyclisch gebeuren. Wanneer de tijd lineair gemeten wordt, meet men de duur tussen de eerste keer dat de beginconditie voorkomt en de laatste keer dat de eindconditie voorkomt. Men kan de tijd ook cyclisch meten, waarbij de tijd tussen de beginconditie en eindconditie voor iedere uitvoering van de loop gemeten wordt. Deze waarden worden dan later geaggregeerd door bijvoorbeeld het gemiddelde hiervan te nemen [15].

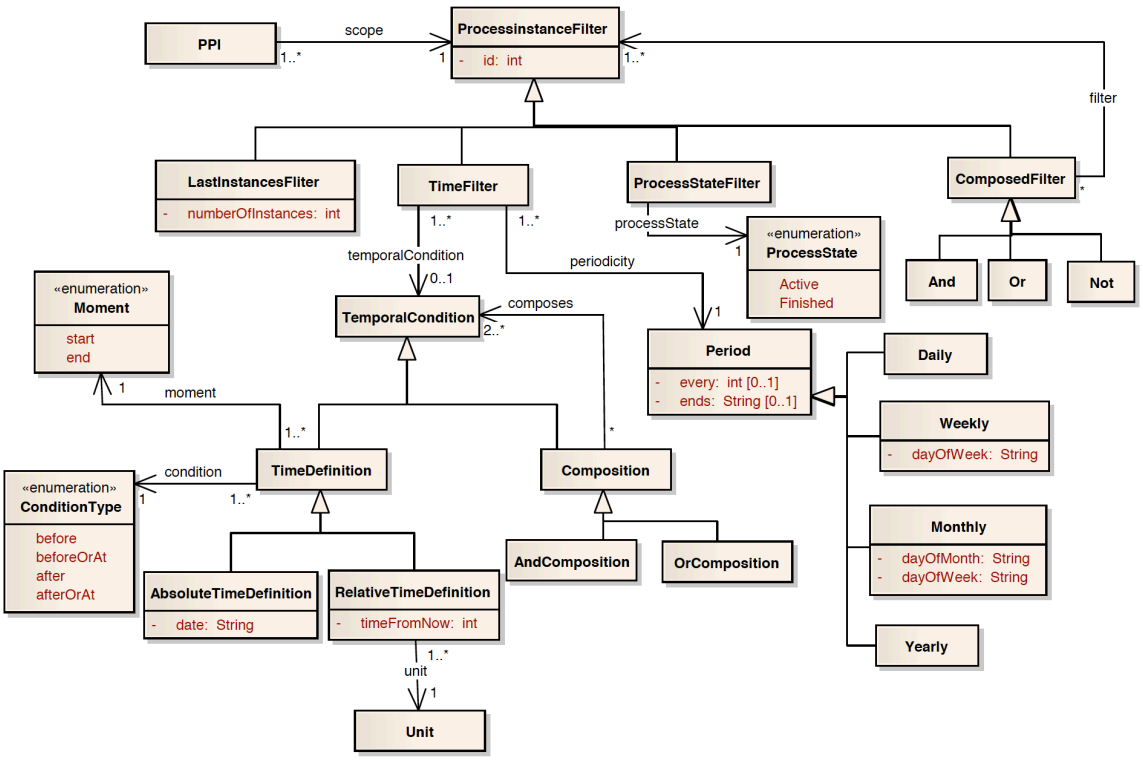
De **telindicator** telt hoe vaak iets gebeurt, namelijk hoe vaak een BP element zich in een bepaalde staat bevindt of hoe vaak een bepaald event getriggerd is [15].

De **conditie indicator** kijkt of zowel lopende als beëindigde procesinstanties voldoen aan een bepaalde conditie. Hier zijn twee types condities gedefinieerd, namelijk de 'StateCondition' en 'DataPropertyCondition'. Hier wordt gekeken of een BP element zich in een bepaalde staat bevindt of dat een data object een bepaalde eigenschap bezit [15].

De **data indicator** meet de waarde van een bepaalde eigenschap van een data object. Via 'dataContentSelection' wordt er bepaald welk deel van het data object gemeten wordt. De 'precondition' bepaalt verder aan welke voorwaarde het data object moet voldoen [15].

De **afgeleide indicator** combineert andere indicatoren op basis van een wiskundige functie. Deze andere indicatoren worden hier gebruikt aan de hand van de meetdefinitie via de variabelen die in de functie zijn bepaald. Verder is er bij deze indicatoren ook nog een onderscheid tussen het combineren van single-instance indicatoren en multi-instance indicatoren, die berekend zijn aan de hand van het volgende type [15].

De **geaggregeerde indicator** aggregeert de vorige vier types over verschillende instanties heen. Dit kan zonder voorwaarde gedaan worden, maar dit kan ook gebeuren op basis van de inhoud van een data object. Hier kunnen dus bijvoorbeeld alle instanties geaggregeerd worden waarvan een data object "order" de waarde "geleverd" heeft. Net zoals bij de afgeleide indicator gebeurt dit op basis van de meetdefinitie van de andere indicatoren. De manier waarop de indicator geaggregeerd wordt, is aangeduid met behulp van de aggregatie functie. Dit kan hier het minimum, maximum, gemiddelde of de som zijn. Tenslotte kan deze indicator ook gegroepeerd worden op basis van de waarde van een data object [15].



Figuur 16: PPINOT metamodel filter [15]

In het laatste onderdeel van het PPINOT metamodel wordt de scope bepaald. Hier modelleert men dus welke instanties er gebruikt worden voor de berekening van de indicatoren. Dit gebeurt hier aan de hand van vier type filters [15].

Het eerste type is de laatste instanties filter waar er een vooraf bepaald aantal instanties weerhouden wordt voor de berekening. Het gaat hier om de laatste aantal keren dat het proces wordt uitgevoerd [15].

Het tweede type is de tijdsfilter waarbij er een tijdsconditie wordt bepaald waaraan de proces instanties moeten voldoen. Bij deze filter zijn er verschillende onderdelen waarop ze gebaseerd kan worden. Enerzijds heb je de tijdsvoorwaarde die gedefinieerd kan worden en ten tweede heb je de periodiciteit, dus de regelmaat waarop de indicatoren gemeten zullen worden. Bij de tijdsvoorwaarde kan je bepalen dat enkel de instanties beschouwd worden voor of na een bepaald moment. Dit moment kan zowel absoluut als in relatie tot de dag van de berekening bepaald worden. Verder kan er een onderscheid gemaakt worden tussen dagelijkse, wekelijkse, maandelijkse en jaarlijkse metingen. Aan de hand van een getal dat gedefinieerd kan worden in de periode, kunnen de metingen ook om de drie weken gebeuren bijvoorbeeld. Dit stelt dus de periodiciteit van de metingen voor [15].

Het derde type is de processtaatfilter die filtert op de staat van procesinstanties. Hier zijn de twee mogelijke staten actief of beëindigd. Dus men kan kiezen om de actieve of beëindigde instanties te behouden [15].

Het laatste type is de samengestelde filter waar de vorige drie types gecombineerd kunnen worden aan de hand van and, or of not operatoren [15].

3.3 BAM

Om het real-time analyseren en monitoren van PPI's mogelijk te maken, wordt er gebruik gemaakt van BAM. Om dit te koppelen aan de processen, wordt er in [12] een BAM extensie op BPMN voorgesteld. Hierbij is er gebruik gemaakt van een metamodel alsook een grafische notatie om de extensie voor te stellen. Aan de hand van dit metamodel worden procesperformantie-indicatoren gedefinieerd. Op de grafische notatie gaan we hier niet verder in [12].

De abstracte syntax wordt voorgesteld aan de hand van verschillende UML klasse diagrammen. Op deze diagrammen zijn de BPMN concepten grijs ingekleurd om het onderscheid te verduidelijken [12].

In dit werk wordt geen expliciete typologie voorgesteld, maar er wordt wel gemodelleerd op basis van verschillende types indicatoren. Deze types zijn de volgende [12]:

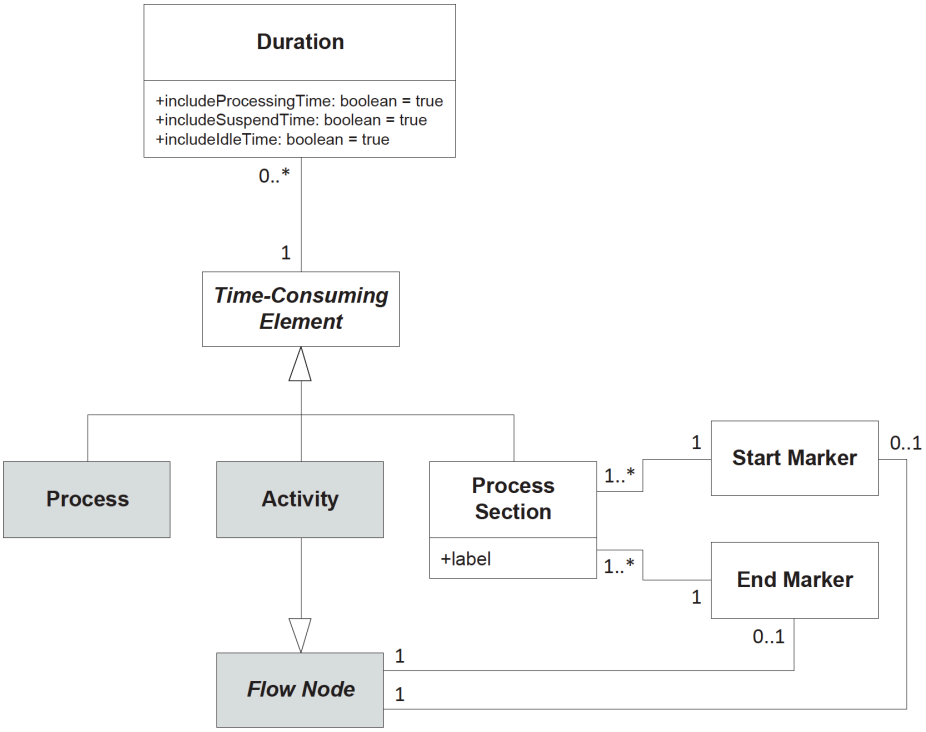
De **tijdsindicator** meet de tijd van processen op verschillende niveaus. Verder gaat het hier om een indicator die gemeten wordt voor één bepaalde instantie [12].

Bij de **frequentie indicator** wordt er geteld hoe vaak iets gebeurt binnen één procesinstantie [12]. Deze komt overeen met de telindicator uit PPINOT.

De **samengestelde indicator** wordt gemeten door verschillende basisindicatoren (tijdsduur indicator en/of frequentie indicator) te combineren aan de hand van wiskundige operaties. Dit type indicator wordt eveneens op het instantie niveau gemeten [12]. Deze indicator wordt in PPINOT vermeld als de afgeleide indicator.

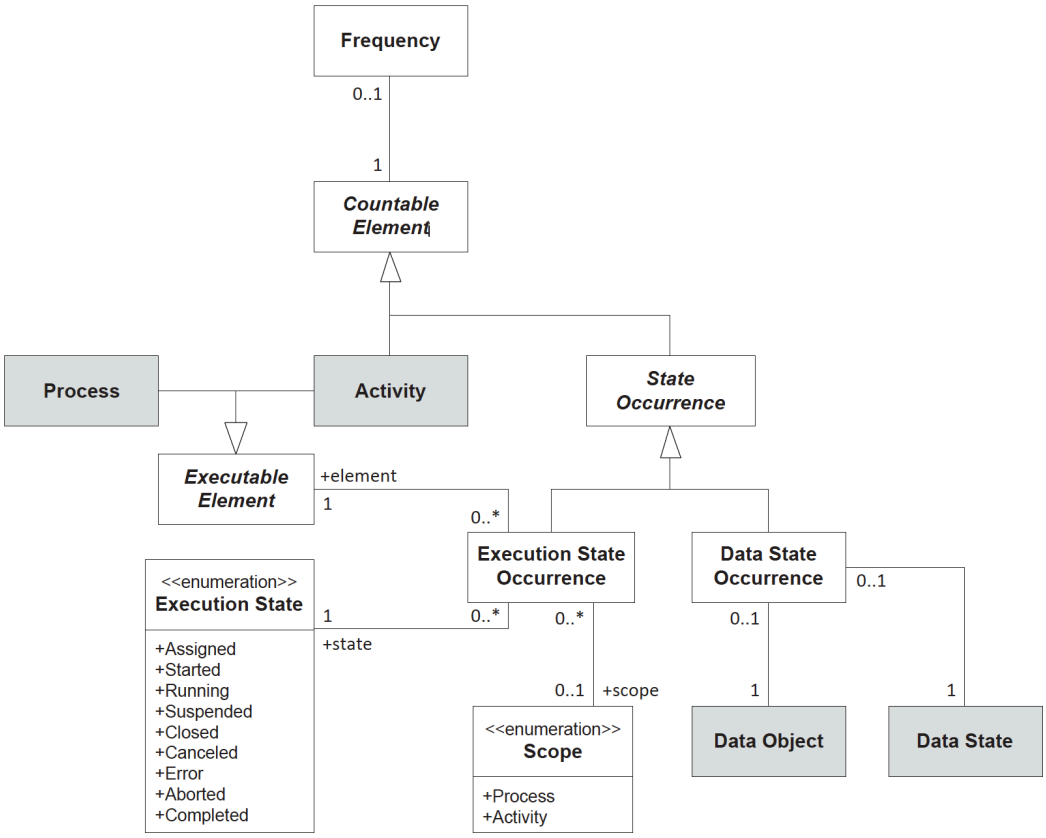
Bij de **geaggregeerde indicator** kunnen de vorige drie types gecombineerd worden over verschillende instanties heen. Dit kan bijvoorbeeld door het minimum, maximum, de som … te nemen van deze verschillende instanties [12].

Hieronder wordt besproken hoe deze types in deze taal gemodelleerd worden.



Figuur 17 metamodel BAM duratie: [12]

De eerste indicator die gemodelleerd wordt, is de tijdsduur indicator zoals te zien op figuur 17. In dit model wordt de tijdsduur in de entiteit 'Duration' berekend op basis van een tijdsconsumerend element. Zo een element kan een proces, activiteit of proces sectie zijn. Wanneer dit een proces is, wordt de tijd tussen het begin en het einde van het proces berekend. Bij een activiteit wordt de duur van begin tot einde gemeten. Tenslotte is er ook nog de proces sectie die gebaseerd is op flow nodes. Flow nodes kunnen activiteiten, gateways en events (vb. start event) zijn. Bij een proces sectie kan dus bijvoorbeeld de tijdsduur van activiteit A tot activiteit B gemeten worden. Verder zijn de attributen van de entiteit 'Duration' ook erg belangrijk omdat zij bepalen welk deel van de tijd gemeten wordt. Zo kan er bepaald worden op basis van 'includeProcessingTime', 'includeSuspendTime' en 'IncludeIdleTime' of de verwerkingstijd, opschortingstijd en 'idle time' van bijvoorbeeld een activiteit mee opgenomen worden in de berekening [12].



Figuur 18 metamodel BAM frequentie: [12]

De volgende indicator die gemodelleerd wordt, is de frequentie indicator. De abstracte syntax hiervan is weergegeven op figuur 18.

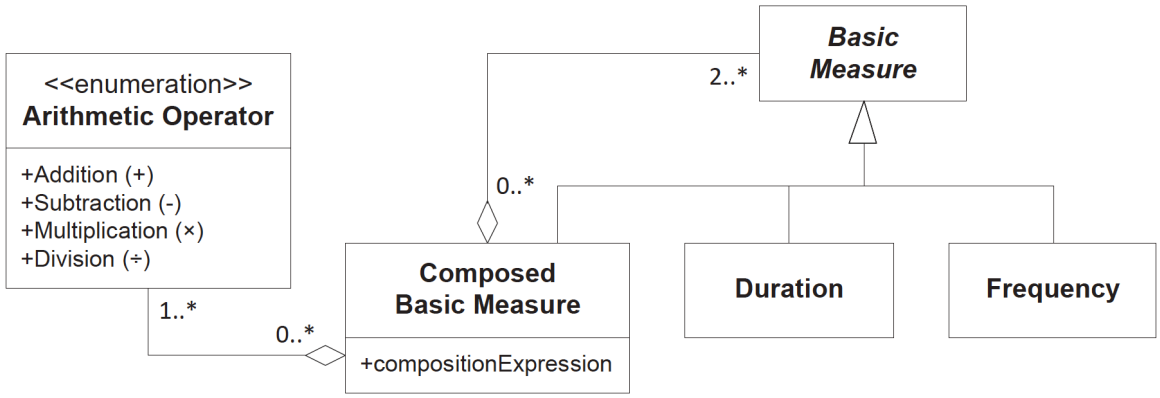
De frequentie wordt geteld op basis van één telbaar element. Hier zijn twee mogelijke telbare elementen, namelijk een activiteit of het voorkomen van een staat 'state occurrence'.
Bij de activiteit wordt er geteld hoe vaak een activiteit succesvol uitgevoerd is binnen één proces.
Bij 'state occurrence' wordt er geteld hoe vaak een BPMN element zich in een bepaalde staat bevindt tijdens procesuitvoering.

Binnen 'state occurrence' kan er een onderscheid gemaakt worden tussen twee subklasses, namelijk 'data state occurrence' en 'execution state occurrence'. Bij de eerste subklasse 'data state occurrence' wordt er gemeten hoe vaak een BPMN data object zich in een specifieke staat bevindt. Dus hier kan men tellen hoe vaak een data object 'bouwvergunning' zich in de staat 'goedgekeurd' bevidnt.

Voor de tweede subklasse 'execution state occurrence' wordt er gemeten hoe vaak een uitvoerbaar element (proces of activiteit) zich in een bepaalde staat bevindt. Bij een activiteit wordt er gekeken hoe vaak een activiteit zich in een bepaalde staat bevindt. Dus kan men bijvoorbeeld tellen hoe vaak de activiteit 'analyseer bouwvergunning' in de staat 'error' is.

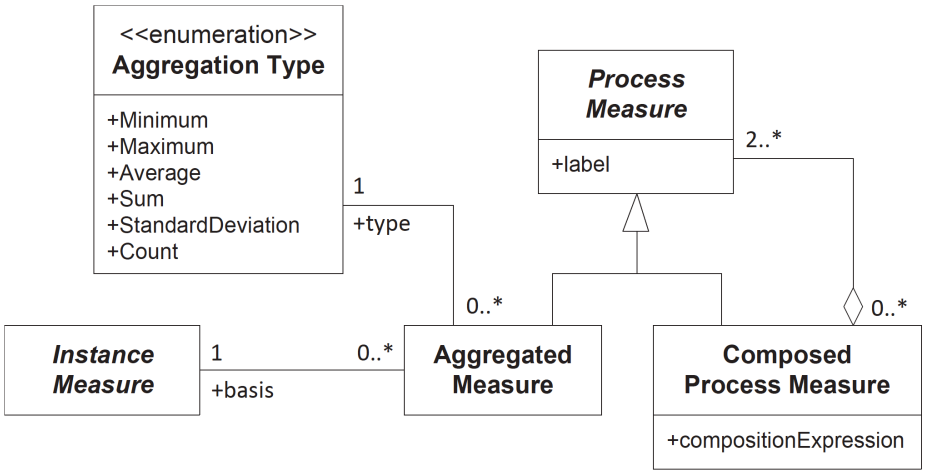
Bij het proces is het nodig om een scope te definiëren. Deze scope kan zowel een proces als een activiteit zijn. Wanneer de scope het proces is, telt men hoe vaak de procesinstantie zich in een bepaalde staat bevond. Dus kijkt men hoe vaak de uitvoering van proces X in de staat lopend was tijdens een specifieke instantie.

Wanneer de scope een activiteit is, telt men hoe veel activiteiten van het proces zich in een bepaalde staat bevonden. Bijvoorbeeld hoeveel activiteiten geschorst zijn geweest [12].



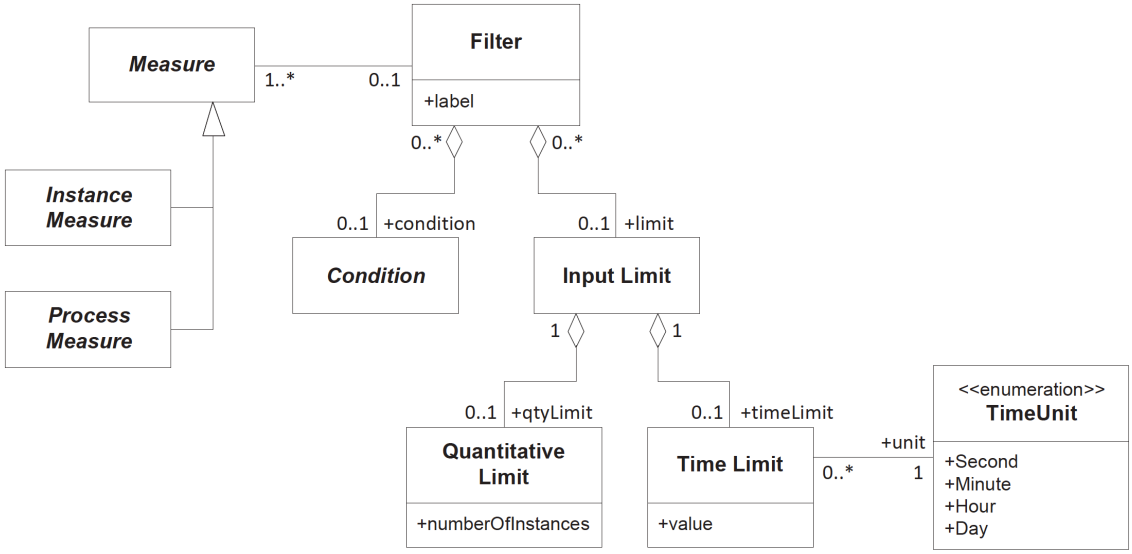
Figuur 19 metamodel BAM samengestelde basis indicator: [12]

De derde indicator is de samengestelde basis indicator, zoals voorgesteld op figuur 19. Hier worden twee of meer basis indicatoren gecombineerd op basis van wiskundige operaties. Hierbij kan eveneens een reeds samengestelde basis indicator gebruikt worden voor een nieuwe samenstelling. Op welke manier de indicatoren gecombineerd worden, is gedefinieerd in de compositie expressie [12].



Figuur 20 metamodel BAM geaggregeerde indicator: [12]

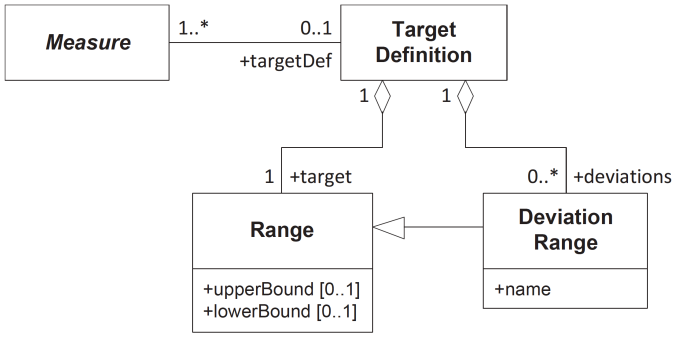
De vierde indicator is de geaggregeerde indicator, zoals voorgesteld op figuur 20. Bij de vorige drie types wordt de indicator gemeten voor één procesinstantie. Het is echter vaak niet voldoende om de indicatoren te berekenen op procesinstantieniveau, daarom is de geaggregeerde indicator nodig. Iedere geaggregeerde indicator baseert zich op een instantie indicator die als basis dient voor het aggregeren. Dit is dus een indicator die berekend is op basis van één specifieke procesinstantie. Het aggregatie type bepaalt verder op welke manier de instantie indicatoren gecombineerd worden. Wanneer indicatoren geaggregeerd zijn, behoren ze tot de superklasse proces indicator. Deze kunnen dan weer verder gebruikt worden om op basis van wiskundige operaties gecombineerd te worden tot een samengestelde proces indicator [12].



Figuur 21 metamodel BAM filter: [12]

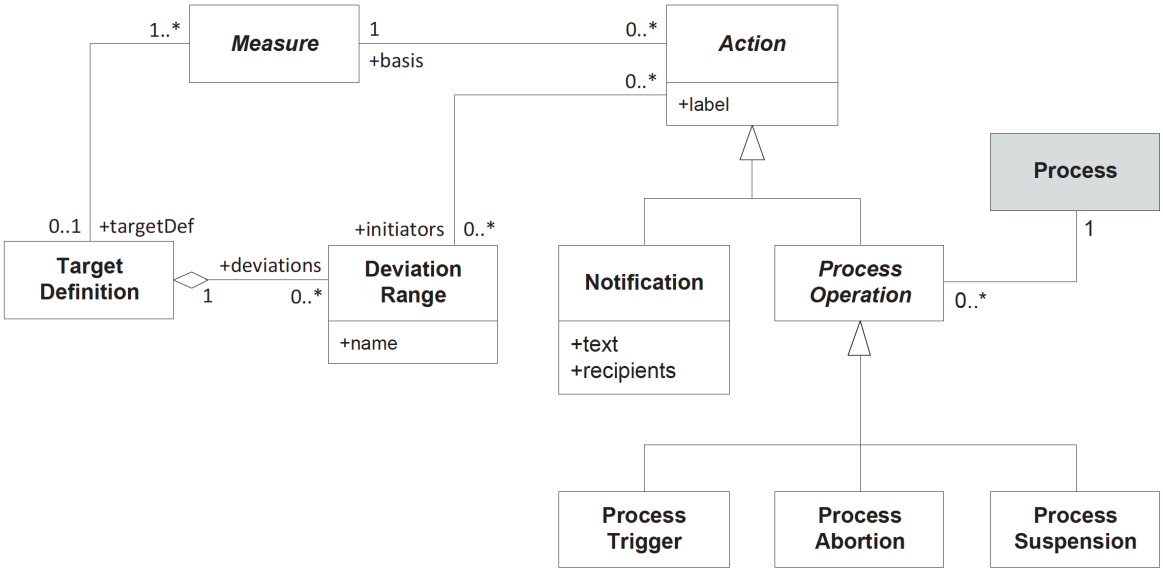
Nadat deze verschillende types indicatoren gemodelleerd zijn, zijn er ook nog andere elementen die gemodelleerd worden aan de hand van deze abstracte syntax. Het eerste dat hier beschouwd wordt, is een filter zoals te zien op figuur 21. Aan de hand van een filter worden indicatoren berekend op basis van een deel van de beschikbare procesinstanties. Deze filters kunnen toegepast worden op zowel instantie-indicatoren als procesindicatoren [12].

Deze filter zal instanties filteren die aan een bepaalde conditie voldoen en/of op basis van een vooropgestelde input limiet. Bij de input limiet kan men een kwantitatieve en/of tijdslimiet instellen. Bij een kwantitatieve limiet wordt enkel een bepaalde hoeveelheid recente instanties weerhouden om de indicator te meten. Hier kunnen dus bijvoorbeeld de zeventig laatst uitgevoerde instanties gebruikt worden voor de meting. Bij een tijdslimiet bepaalt men in welke periode de indicator berekend moet worden. Wanneer beide limieten gebruikt worden, telt degene die als eerste bereikt wordt [12].



Figuur 22 metamodel BAM target definitie: [12]

Verder worden er ook targets bepaald die gekoppeld worden aan indicatoren. Iedere indicator kan gekoppeld worden aan een target definitie. Binnen deze definitie wordt er een bereik bepaald waarbinnen men wenst dat de waarde van de indicator valt. Verder kan er ook een bereik gekozen worden voor de afwijkingen [12].

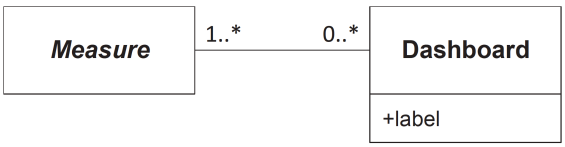


Figuur 23 metamodel BAM acties: [12]

Om de targets in de praktijk gebruiken, worden de deviatie bereiken gekoppeld aan een bijhorende actie. Een actie is een automatische reactie die gebeurt wanneer een indicator buiten het target bereik valt. Er kunnen verschillende afwijkingsbereiken bepaald worden waardoor er verschillende acties kunnen gekoppeld worden aan één indicator [12].

Binnen de abstracte klasse 'actie' heeft men twee onderdelen, namelijk notificatie en proces operatie. De notificatie klasse geeft weer welke boodschap naar welke ontvangers gestuurd moet worden. Bij de proces operatie worden er verschillende acties gegroepeerd die gerelateerd zijn aan een proces. Dit is dus een superklasse van proces trigger, proces abortie en proces suspensie. Bij de proces trigger wordt er een nieuwe instantie van een proces gestart. Wanneer de actie een

proces abortie is, kan een proces instantie beëindigd worden. Bij een proces suspensie wordt een instantie gepauzeerd. Proces abortie en suspensie kunnen enkel toegepast worden op instantie indicatoren [12].



Figuur 24 metamodel BAM dashboard element: [12]

Het laatste element dat voorgesteld wordt in dit metamodel, is het dashboard element. Hier kan aangegeven worden welke indicatoren gebruikt worden in een dashboard en in welke dashboards [12].

3.4 Linken en verschillen tussen de modellen

Verschil 1 : PPI's vs KPI's

Het eerste grote verschil dat er is tussen de drie modellen is de soort performantie-indicator die gemodelleerd kan worden aan de hand van het metamodel. Bij PPINOT en BAM zijn dit procesperformantie-indicatoren en bij MetricM gaat het hoofdzakelijk om performantie-indicatoren. Het gaat enkel om procesperformantie-indicatoren wanneer het referentie object een proces is. In het MetricML metamodel zijn er echter geen BP elementen verwerkt. Door het ontbreken van de koppeling met de proceselementen kan de PPI dus nooit real-time gemeten worden.

Verschil 2: typologie

Het volgende verschil dat aanwezig is, gaat om de aanwezigheid van een typologie. In PPINOT is er duidelijk een typologie aanwezig die de types voorstelt die gemodelleerd worden aan de hand van het metamodel. In BAM daarentegen is er geen expliciete typologie voorgesteld, maar worden er wel aparte metamodellen gebouwd voor verschillende types indicatoren. We stellen dus vast dat er hier een impliciete typologie aanwezig is. Bij MetricM is er geen typologie maar worden er enkel indicatorcategorieën gemaakt in de entiteit 'IndicatorCategory' op basis van beslissingen van gebruikers. Hier is er dus geen onderscheid tussen voorafbepaalde types. Hierbij is er dus meer vrijheid voor de gebruikers om eigen keuzes te maken.

Naast de aanwezigheid van een typologie bij PPINOT en BAM, is er ook een onderscheid op basis van de types die aanwezig zijn in de typologieën. Van tabel 4 kan men aflezen welke types er gemodelleerd kunnen bij PPINOT en BAM. In tabellen 5 tot en met 8 worden de types die in beide metamodellen gemodelleerd worden vergeleken met elkaar.

	PPINOT	**BAM**
Tijdsindicator	Ja	Ja
Telindicator	Ja	Ja
Conditie indicator	Ja	Nee
Data indicator	Ja	Nee
Geaggregeerde indicator	Ja	Ja
Afgeleide indicator	Ja	Ja

tabel 4: vergelijking aanwezige types PPINOT en BAM

Tijdsindicator	**PPINOT**	**BAM**
Overeenkomst	Bij PPINOT wordt de duur gemeten op basis van de entiteit 'TimeInstantCondition'. Deze bepaalt de twee momenten waartussen de tijd gemeten moet worden. Ieder moment wordt bepaald door een conditie (verandering van staat) van een BP element.	In BAM komt dit overeen met het tijdsconsumerend element, de proces sectie die een start- en eindmarkering bevat. Deze refereren ook beide naar de verandering van een staat van een BP element. In BAM is er ook nog sprake van een proces en activiteit als tijdsconsumerend element, maar deze kunnen perfect gemodelleerd worden op basis van de proces sectie.
Verschillen	Bij PPINOT kan de tijd op twee manieren gemeten worden wanneer er sprake is van een loop in het proces. Dit kan zowel lineair (dus vanaf de eerste activiteit in de eerste loop tot de laatste activiteit van de laatste loop) of cyclisch (hier wordt de tijd gemeten per afzonderlijke loop en hierna geaggregeerd zoals men wenst).	
		Bij BAM zijn er in de duratie entiteit drie attributen die de mogelijkheid geven om zelf de keuze te maken of de verwerkingstijd, opschortingstijd en 'idle time'

		al dan niet op te nemen in de indicator. Hierdoor kan men bijvoorbeeld de duur van een proces meten waarbij men enkel meet hoelang activiteiten daadwerkelijk uitgevoerd worden.

Tabel 5: vergelijking tijdsindicator BAM en PPINOT

Telindicator	**PPINOT**	**BAM**
Overeenkomst	Bij PPINOT wordt er geteld hoe vaak een BP element (BP flow element / data object) zich in een bepaalde staat bevindt aan de hand van de 'TimeInstantCondition'.	Dit komt overeen bij BAM met het telbaar element 'state occurrence' met als onderdelen 'execution state occurrence' en 'data state occurrence'.
Verschillen		Bij BAM is het proces ook een uitvoerbaar element, waardoor men kan tellen hoe vaak een proces of hoe veel activiteiten zich binnen het proces in een bepaalde staat bevinden. Bij PPINOT kan men enkel tellen op niveau van het BP flow element, waardoor dit geen mogelijkheid is.

Tabel 6: vergelijking telindicator BAM en PPINOT

Geaggregeerde indicator	**PPINOT**	**BAM**
Overeenkomst	Een geaggregeerde indicator wordt berekend door middel van het aggregeren van een 'BaseMeasure', 'DerivedMeasure' of zelfs een andere 'AggregatedMeasure' op basis van de meetdefinitie. De aggregatie wordt bepaald door middel van een aggregatie functie.	In BAM wordt er op een gelijkaardige manier een instantie-indicator geaggregeerd over meerdere instanties op basis van een aggregatie type.
Verschillen	Bij PPINOT kan een geaggregeerde indicator	

	gegroepeerd worden op basis van een eigenschap van een data object met behulp van 'DataContentSelection'.	
		In BAM worden er ook wiskundige formules toegepast op geaggregeerde indicatoren, dit hoort bij PPINOT bij de afgeleide indicatoren. Verder zijn er ook twee functies 'count' en 'standaardafwijking' die wel inbegrepen zijn bij BAM. De count is vooral noodzakelijk voor het aggregeren van data indicatoren, aangezien deze waarden niet zomaar opgeteld kunnen worden.

Tabel 7: vergelijking geaggregeerde indicator BAM en PPINOT

Afgeleide indicator	**PPINOT**	**BAM**	**MetricML**
	Bij PPINOT wordt de afgeleide indicator berekend door middel van een functie in de entiteit 'DerivedMeasure'. Deze functie gebruikt variabelen die gebaseerd zijn op de meetdefinitie. Deze variabelen kunnen zowel basisindicatoren als geaggregeerde indicatoren.	De naam van de afgeleide indicator bij BAM is 'ComposedBasicMeasure' aangezien er een wiskundige formule toegepast kan worden hier op de twee basis indicatoren (duratie en frequentie). Dit gebeurt door middel van een wiskundige operator in de entiteit 'arithmetic operator'. De formule zit verwerkt in de entiteit 'ComposedBasicMeasure' als attribuut 'compositionExpression'.	In MetricML kan men ook wiskundige formules toepassen op andere indicatoren om op deze manier een nieuwe indicator te bekomen. Dit gebeurt aan de hand van de entiteit 'TransformsRelation'. Deze heeft als attribuut 'expression' en 'source'. 'expression' komt dus overeen met de 'function' uit PPINOT en 'compositionexpression' uit BAM.
	Bij PPINOT is er binnen dit type dus ook de mogelijkheid om een		

	functie toe te passen op geaggregeerde indicatoren. Bij BAM wordt dit verwerkt in het vorige type, de geaggregeerde indicator.		

Tabel 8: vergelijking afgeleide indicator BAM en PPINOT

Verschil 3: duidelijkheid modellering BP elementen

Verder zijn de BP elementen ook duidelijker gemodelleerd in BAM dan in PPINOT. In PPINOT is er een model voorgesteld voor BPM. Hierbij heb je een BP element dat een BP flow element of een data object kan zijn. In het modelleren van de verschillende types wordt er soms geen duidelijk onderscheid gemaakt tussen deze BP elementen.

Verschil 4: overzichtelijkheid

Verder zijn de BAM metamodellen ook overzichtelijker dan het MetricM en PPINOT metamodel aangezien deze per type apart worden gemodelleerd. Dit zorgt ervoor dat het model makkelijker te begrijpen valt.

Verschil 5: vergelijking filters

PPINOT	**BAM**	**MetricML**
Bij PPINOT wordt de entiteit 'ProcessInstanceFilter' gekoppeld aan de indicatoren. Deze heeft als attribuut 'id', waardoor er een cijfercode gegeven kan worden aan iedere filter.	Bij BAM kan je een filter koppelen aan een indicator op basis van de entiteit 'filter' met als attribuut 'label'. Dit kan zowel op instantie- als procesindicatoren toegepast worden.	/
Hier zijn er vier deelfilters, namelijk 'LastInstancesFilter', 'TimeFilter', 'ProcessStateFilter' en 'ComposedFilter'. De laatste hiervan is een combinatie van de vorige vier.	Hierbij zijn er twee deelfilters, een 'condition' en 'input limit'. Beide kunnen gecombineerd worden. Binnen de 'input limit' heb je zowel een 'quantitative limit' als een 'time limit' (integer) die gekoppeld wordt aan een 'TimeUnit' (seconde, minuten…)	/
De eerste filter die hetzelfde is bij beide modellen is de 'LastInstancesFilter' bij PPINOT en de 'Quantitative limit' bij BAM. Deze hebben beide als attribuut 'numberofinstances', het aantal instanties die gebruikt moeten worden om de indicator te berekenen.		/
De volgende filters die slechts gedeeltelijk overeenkomen, zijn de 'ProcessStateFilter' uit PPINOT en de 'Condition' filter uit BAM. Bij de 'ProcessStateFilter' filtert men op de staat van een proces aan de hand van 'ProcessState'. De 'Condition' filter, zal filteren op basis van een conditie. Dus dit kan bijvoorbeeld ook de staat van een bepaald proces zijn. Dit kunnen echter ook andere condities zijn (zoals de eigenschap van een data object), dus daarom komen ze niet volledig overeen.		/
De 'ComposedFilter' uit PPINOTcombineert de drie andere filters op basis van logische operatoren zoals 'AND', 'NOT' en 'OR'. Deels komt dit overeen met het feit dat in BAM de filters gecombineerd worden. Dit is dus enkel van toepassing op het 'AND' deel. Voor de andere operatoren is dit niet het geval en is PPINOT dus uniek.		
Dan heb je tenslotte nog de 'TimeFilter' uit PPINOT en de 'Time Limit' uit BAM die deels overeenkomen. Bij de 'Time Limit' in combinatie met de 'Time Unit' kan men bepalen tot hoeveel tijd geleden er instanties beschouwd moeten worden. Dit wordt in PPINOT gedaan door middel van de 'RelativeTimeDefinition' en een bijhorende 'Unit'. Op dit gebied zijn de filters identiek aan elkaar.		

Bij PPINOT daarentegen is de 'TimeFilter' nog wat uitgebreider dan dit aspect alleen. De 'TimeFilter' heeft twee subklasses, namelijk 'Period' en 'TemporalCondition'. Bij de 'Period' wordt er bepaald om de hoeveel tijd een PPI gemeten wordt. De 'TemporalCondition' heeft verder nog twee subklasses, de 'TimeDefinition' en 'Composition'. Aan de hand van de Composition kunnen twee of meer tijdscondities gecombineerd worden. De 'TimeDefinition' kan zowel relatief als absoluut bepaald worden. Bij de 'AbsoluteTimeDefinition' wordt er een datum bepaald. Deze wordt dan gecombineerd met een 'ConditionType' (bv. before) en een moment (start of end). Hier kan men dus zeggen dat men al de instanties beschouwd voor 4 maart 2017 om de indicator te meten.		

Tabel 9: vergelijking filters BAM en PPINOT

Verschil 6: target

PPINOT	**BAM**	**MetricML**
Bij PPINOT en BAM wordt de indicator gekoppeld aan een target. Hierbij wordt er een range bepaald waarbinnen de waarde van de PPI geacht wordt te liggen. In BAM noemt deze entiteit 'range' en in PPINOT 'SimpleTarget', deze hebben beide als attribuut een upperbound en/of een lowerbound. In BAM is er echter ook een 'deviation range' die bepaalt hoeveel er afgeweken mag worden van het bereik. PPINOT heeft verder ook nog het verschil met BAM dat bij PPINOT de relatie zo bepaald is, dat er sowieso een target bepaald moet worden. Verder heb je hier ook nog de 'ComposedTarget' die meerdere targets combineert en de		Bij MetricML wordt er in tegenstelling tot bij de andere twee modellen geen bereik maar een targetwaarde bepaald. Dit wordt gedaan in de entiteit 'RefersToRelation' die gekoppeld wordt aan de 'Indicator'. Deze is optioneel te bepalen en heeft als attributen 'TargetValue' en 'Benchmark'.

'CustomTarget'. Bij deze laatste wordt er een specifieke restrictie bepaald zoals bijvoorbeeld het voldoen aan een functie.	

Tabel 10: vergelijking target BAM, MetricML en PPINOT

Verschil 7: acties

In alle drie de modellen kunnen verder targets gekoppeld worden aan de indicatoren. Echter is er enkel in het BAM metamodel de mogelijkheid om acties te koppelen aan deze target. Hierdoor kan er dus onmiddellijk gereageerd worden wanneer de waarde van de indicator afwijkt van deze target.

PPINOT	**BAM**	**MetricML**
/	BAM is het enige model van de drie dat modelleert om automatisch te reageren wanneer er afwijkingen zijn van het target bereik. Dit kunnen bijvoorbeeld notificaties of proces operaties (het beëindigen van een procesinstantie) zijn.	Bij MetricML is er een 'DecisionScenario' waar men de indicator kan gebruiken in bijvoorbeeld een simulatie om beslissingen op te baseren. De acties die hierna volgen gebeuren echter niet automatisch.

Tabel 11: vergelijking acties BAM en MetricML

Verschil 8: dashboard

PPINOT	**BAM**	**MetricML**
/	Bij BAM wordt een indicator gekoppeld aan de entiteit 'Dashboard'. Hierdoor is er de mogelijkheid om reeds te modelleren bij welk dashboard of welke dashboards een indicator zal voorkomen.	/

Tabel 12: dashboard element BAM

Verschil 9: hoofdentiteit

Ook is er nog een belangrijk verschil tussen PPINOT en MetricM enerzijds en BAM anderzijds. In PPINOT en MetricM wordt er namelijk een centrale entiteit gemodelleerd die de indicator en zijn kernattributen definieert. Dit ontbreekt er bij BAM. Dus daardoor is de link tussen de verschillende metamodellen hier niet duidelijk gebaseerd op een gemeenschappelijk centraal concept.

In PPINOT noemt de centrale entiteit 'PPI' en bij MetricML is er sprake van 'Indicator'. Echter zijn de attributen ervan niet allemaal hetzelfde, dus vergelijken we deze hieronder.

PPINOT	**MetricML**
Identifier	/
Name	Name
Goals	**Niet in de centrale entiteit**
Responsible	**Niet in de centrale entiteit**
Informed	/
Comments	/
/	TimeHorizon
Niet in de centrale entiteit	**UnitOfMeasure**
/	sourceOfRawData
Niet in de centrale entiteit	**freqOfMeasurement**
/	freqOfReview
/	purpose
/	intention
/	assumptions
/	justification
/	value
/	dateOfMeasurement

Tabel 13: vergelijking centrale entiteiten BAM en PPINOT

Zo ontbreekt er bij MetricM bijvoorbeeld een unieke code als attribuut voor de indicator. Dit kan tot verwarring leiden wanneer er twee dezelfde indicatoren met andere eigenschappen gedefinieerd worden. De overige elementen die ontbreken in één van beide modellen, zijn weergegeven op de bovenstaande tabel. Hieronder bespreken we nog de attributen die in één van beide modellen buiten de centrale entiteit gemodelleerd zijn.

In verband met de attributen die resource-gerelateerd zijn, 'Responsible' en 'Informed gaan we hieronder verder op in.

'Goals' is het eerste attribuut dat in MetricML buiten de centrale entiteit gemodelleerd is. In MetricML is dit namelijk gemodelleerd door 'RepresentsRelation' te koppelen aan de 'Indicator' entiteit en erna hier 'Goals' aan te koppelen. De doelstellingen worden dus in beide modellen op een andere manier gemodelleerd. Bij BAM is er geen sprake van het koppelen van de strategische doelstellingen.

Het volgende attribuut 'UnitOfMeasure' dat voorkomt in de hoofdentiteit van MetricML wordt in PPINOT gemodelleerd als een attribuut van de meetdefinitie.

Het laatste attribuut dat anders gemodelleerd wordt buiten de hoofdentiteit in PPINOT is 'freqOfMeasurement' uit MetricML. Dit wordt hier namelijk gemodelleerd in de 'TimeFilter' in het onderdeel 'Period' dat we bij de filter reeds besproken hebben.

Verschil 10: resources

In BAM zijn er geen resources gemodelleerd, maar in PPINOT en MetricML is dit wel het geval. Echter zijn de resources niet op dezelfde manier gemodelleerd en zijn er hier verschillen tussen de soorten resources.

PPINOT	**MetricML**
Verantwoordelijke: Attribuut 'responsible' in 'PPI'	'Position' gekoppeld aan 'Indicator'
Geïnformeerde: Attribuut 'informed' in 'PPI'	/
Persoon die de PPI meet: /	'Organizationalrole' gekoppeld aan 'Indicator' pijl : measures
Persoon die handelt naar de waarde van de PPI: /	'Organizationalrole' gekoppeld aan 'Indicator' pijl : acts on (optioneel)

Tabel 14: vergelijking resources MetricML en PPINOT

Verschil 11: overige extra's MetricML

Verder zijn er nog bepaalde dingen gemodelleerd in MetricML die niet voorkomen in de andere twee modellen.

Het eerste aspect is de entiteit 'IndicatesRelation' die de relatie tussen verschillende indicatoren weergeeft.

Verder heb je ook nog de entiteit 'IndicatorAttribute' waardoor de gebruikers zelf nog attributen kunnen toevoegen aan de 'Indicator' entiteit.

En tenslotte heb je nog het 'ReferenceObject' dat gekoppeld wordt aan de Indicator en dat de context bepaalt. De opties hierbij zijn 'OrganisationalUnit', 'BusinessProcess', 'Resource', 'InformationSystem' en 'Uncertainty'. Bij PPINOT en BAM gaan we ervanuit dat het referentie object steeds een proces is.

3.5 Scenario's die gemodelleerd moeten kunnen worden

Hieronder hebben we een heel aantal scenario's opgesteld die gemodelleerd moeten kunnen worden door ons metamodel. Deze scenario's zijn gebaseerd op basis van de drie basismodellen. Sommige hiervan kunnen door alle modellen, meerdere of één van de modellen gemodelleerd worden.

3.5.1 Scenario : mogelijk bij alle modellen

Scenario 1:
Men wil een wiskundige formule toepassen op bestaande indicatoren om op deze manier een nieuwe indicator te bekomen.

3.5.2 Scenario : mogelijk bij BAM en PPINOT

Scenario 2:
Men wil de tijd meten tussen twee tijdstippen van het proces. Bijvoorbeeld de tijd meten tussen de start van activiteit A en het einde van activiteit B.

Scenario 3:
Men wil tellen hoe vaak activiteit A of het proces zich in de staat 'lopend' (of een andere staat) bevindt. Een activiteit of proces bevindt zich in een bepaalde staat doordat er een event voorkomt.
Men wil tellen hoe vaak een data object G zich in datastaat J bevindt.

Scenario 4:
Ze willen de gemiddelde duur van activiteit A berekenen over verschillende instanties heen. Of men wil andere indicatoren aggregeren over meerdere instanties met behulp van functies zoals het gemiddelde, minimum, maximum etc.

Scenario 5:
Men wil een indicator berekenen op basis van de X aantal laatste procesinstanties.

Scenario 6:
Men wil een indicator berekenen op basis van instanties vanaf bijvoorbeeld 20 dagen geleden tot nu.

Scenario 7:
Men wil een targetbereik bepalen waarbinnen de waarde van een PPI hoort te vallen.

3.5.3 Scenario : mogelijk bij MetricML en PPINOT

Scenario 8:
Men wil bepalen hoe vaak een indicator gemeten moet worden. Bij PPINOT wordt er ook nog verder de exacte dag in de week of maand bepaald waarop ze berekend moet worden.

Scenario 9:
Men wil de doelstelling koppelen aan de indicator zodat men ziet op welke doelstelling de PPI's gebaseerd zijn.

Scenario 10: deels MetricML, deels PPINOT

Men wil kunnen aangeven welke resources verantwoordelijk zijn voor de PPI, geïnformeerd moeten worden, de PPI meten en reageren op de waarde van de PPI.

3.5.4 Scenario : mogelijk bij BAM

Scenario 11:
Men wil de tijd meten tussen start activiteit A en einde activiteit D. Hierbij wil men echter enkel de verwerkingstijd (of idle time of opschortingstijd) meten.

Scenario 12:
Men wil tellen hoe vaak het proces X zich in de staat 'error' (of een andere staat) bevindt.

Scenario 13:
Men wil een indicator kunnen aggregeren op basis van een count of standaardafwijking. Count gaat vooral nuttig zijn voor de data en conditie indicator uit PPINOT.

Scenario 14:
Men wil een indicator 'duur van activiteit A' berekenen op basis van de instanties waar een data object een bepaalde waarde voor een eigenschap bezit.

Scenario 15:
Men wil automatische acties (bijvoorbeeld notificaties en procesabortie) koppelen wanneer er afgeweken wordt van het targetbereik.

Scenario 16:
Men wil reeds weergeven op welk dashboard de PPI zal verschijnen.

3.5.5 Scenario : mogelijk bij MetricML

Scenario 17:
Men wil verder ook de relaties tussen de verschillende indicatoren kunnen weergeven omdat ze een invloed hebben op elkaars waarde.

Scenario 18:
Wanneer men extra attributen wil toevoegen aan de 'indicator' klasse, moet dit ook achteraf mogelijk zijn.

3.5.6 Scenario : mogelijk bij PPINOT

Scenario 19:
Er bevindt zich een loop in het proces waarbij activiteit A en activiteit B meerdere keren achtereenvolgens uitgevoerd kunnen worden. Men wil de gemiddelde tijd meten tussen de start van activiteit A en het einde van activiteit B.

Scenario 20:
Men wil een geaggregeerde indicator groeperen per eigenschap van data object G. Zo wil men dus bijvoorbeeld de gemiddelde duur van activiteit A berekenen per groep instanties waarbij data object G een andere waarde heeft voor eigenschap I.

Scenario 21:
Men wil een indicator die ons zegt welke waarde een specifieke eigenschap van een data object. Deze indicator kan dan ook later geaggregeerd worden over verschillende instanties heen op basis van een count.

Scenario 22:
Men wil een indicator die in de vorm van een boolean waarde weergeeft of een instantie aan een voorafbepaalde voorwaarde voldoet. Ook dit kan later geaggregeerd worden over verschillende instanties heen.

Scenario 23:
Men wil een indicator berekenen op basis van de X laatste instanties en op basis van een data conditie. Het kan ook zijn dat men een combinatie wil maken van de overige soorten filters.

Scenario 24:
Men wil een indicator berekenen met de procesinstanties vanaf/voor een exacte datum.

Scenario 25:
Men wil zowel een target dat bestaat uit een bereik als een target dat bepaald is op basis van een functie kunnen combineren.

4. Artefact

In dit hoofdstuk wordt de typologie bepaald en het metamodel voorgesteld. Nadat het metamodel beschreven is, demonstreren we het nut hiervan aan de hand van een voorbeeldproces en bijhorende indicatoren.

4.1 Vereisten voor het artefact

Om achteraf ons metamodel te kunnen evalueren, stellen we hier vereisten op waaraan we willen dat ons model voldoet. Deze vereisten zijn opgesteld aan de hand van de verschillen tussen de modellen die besproken zijn in 3.4. Hierdoor worden alle noodzakelijke kenmerken die ontbraken in sommige modellen opgenomen in ons metamodel. Alsook zijn hier twee vereisten gekozen die reeds voorgesteld waren in PPINOT [15] en MEMO [31].

Vereiste 1:
De link tussen de PPI's en de bijhorende processen moet duidelijk aangegeven worden bij de definiëring van deze PPI's.

PPI's moeten berekend worden op basis van bedrijfsprocessen. Wanneer deze link bij de definiëring echter niet goed wordt aangegeven, zal dit de berekening van de waarde van de PPI verstoren. De PPI's moeten namelijk worden berekend op basis van proceselementen. Wanneer deze elementen niet duidelijk aangegeven worden in de definitie, dan kan er later geen automatisering van deze berekening komen [15]. Het is dus ook noodzakelijk om aan te geven om welk proceselement het gaat om de definiëring te verduidelijken.

Vereiste 2:
Gebruik concepten in de modelleertaal die toekomstige gebruikers gemakkelijk begrijpen.

De voornaamste gebruikers van een metamodel zijn ontwerpers van andere modelleertalen en de ontwikkelaars van modelleertools. Er moeten dus concepten gebruikt worden die zij gewoon zijn om te gebruiken. Zowel data modellen als klasse diagrammen zijn deze doelgroep welbekend, dus er kan best voor deze opties geopteerd worden [31].

Vereiste 3:
Er moet een typologie gedefinieerd worden die al de types bevat die gemodelleerd worden in onze basismodellen.

De typologie die we gebruiken om de types PPI's voor te stellen moet al de types bevatten die in PPINOT en BAM voorgesteld worden. BAM bevat slechts een subset van de typologie van PPINOT, dus daarom is het voldoende om de types te gebruiken uit PPINOT. Om tot deze types te komen is Del Rio Ortega op onderzoek gegaan in de literatuur en in organisaties.

Vereiste 4:
Het metamodel moet in staat zijn om al de types die bepaald zijn in de typologie te modelleren.

Al de types die er bepaald zijn in de typologie moeten duidelijk gemodelleerd worden in het metamodel. Om het overzicht te behouden worden deze types het best gemodelleerd op basis van aparte metamodellen die samen gecombineerd worden tot een groter geheel.

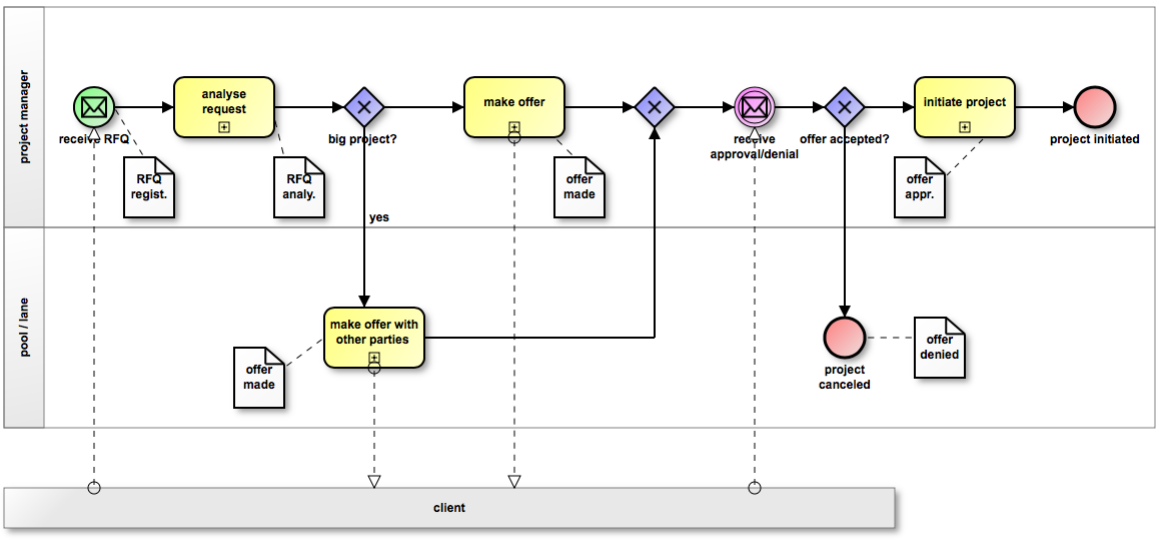
Vereiste 5:
Het metamodel moet een centraal indicator concept bevatten waardoor andere onderdelen hieraan gekoppeld kunnen worden.

Het is nodig dat er een centraal indicator concept is dat eigenschappen van de PPI bevat. Op basis van dit concept kunnen de onderdelen van het metamodel aan elkaar gelinkt worden en kan men indicatoren weer hergebruiken in de definiëring van andere PPI's.

Vereiste 6:
De scenario's die voorgesteld zijn in 3.5 moeten door het metamodel gemodelleerd kunnen worden.

4.2 Voorbeeldproces

Om het model dat we hierna modelleren te verduidelijken en de definiëring van de PPI's te demonstreren, is er hier een voorbeeldproces voorgesteld. Aan de hand van dit proces zullen we voorbeelden geven van de verschillende types indicatoren. Hierna zullen deze indicatoren ook gedefinieerd worden aan de hand van ons eigen metamodel.



Figuur 25: voorbeeldproces offertes

Dit proces situeert zich in een bedrijf dat projectgericht werkt. Het proces dat wij bekijken loopt vanaf het moment dat er een offerteaanvraag ingediend is door een klant tot de project annulering

of initiatie. De eerste stap is dus het ontvangen van de offerteaanvraag. Hierna wordt deze aanvraag geanalyseerd om te kijken van welke grootte het project zal zijn en hoe groot de bijkomende risico's zijn. Op basis hiervan wordt erna een ander pad gevolgd voor projecten die groter zijn dan een bepaald vastgelegd bedrag. Hier wordt namelijk een tijdelijke projectgroep voor opgezet waarin verschillende bedrijven samenwerken. De offerte zal dan ook binnen deze groep opgesteld worden aangezien er goedkeuring nodig is van al de partijen. Wanneer er geen sprake is van een groot project, maakt de project manager zelf de offerte. Nadat er een offerte is gestuurd naar de klant door de project manager of de tijdelijke projectgroep, wacht men op een bevestiging of afwijzing. Wanneer de offerte niet aanvaard wordt, wordt het project beëindigd. Na aanvaarding van de offerte door de klant, zal de projectmanager het project initialiseren door het toe te wijzen aan verschillende personen en de details vast te leggen. De uitvoering van het project zelf behoort niet tot de scope van ons voorbeeldproces.

4.3 Typologie

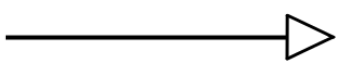
De typologie die weergeeft welke types indicatoren we zullen modelleren aan de hand van ons metamodel, wordt overgenomen van Del Rio Ortega [15]. Deze hebben we besproken in '3.2.1 PPINOT typologie'. De types die hier bepaald zijn, zijn de tijdsindicator, telindicator, conditie indicator, afgeleide indicator en geaggregeerde indicator. Dit zijn dan ook de types die terugkomen in ons metamodel in het volgende deel.

4.4 Metamodel

Het metamodel dat we voorstellen is gebouwd op basis van [12], [14] en [15]. In ons metamodel houden we rekening met de vereisten bepaald in 4.1. We proberen tevens zo veel mogelijk de sterktes te gebruiken van ieder van de oorspronkelijke modellen die ontbraken bij één of twee van de andere modellen. We focussen hier op de abstracte syntax van het metamodel dat weergegeven wordt aan de hand van UML (unified modelling language) klasse diagrammen. Hierbij verschaffen we ook voor ieder onderdeel een bijhorende beschrijving ter verduidelijking.

Om de klasse diagrammen hierna duidelijker te maken, geven we hier reeds enkele UML elementen die we gebruiken en hun bijhorende betekenis aan.

generalisatie



A en B hebben een ouder-kind relatie, waarbij A een subklasse is van de superklasse B.

associatie

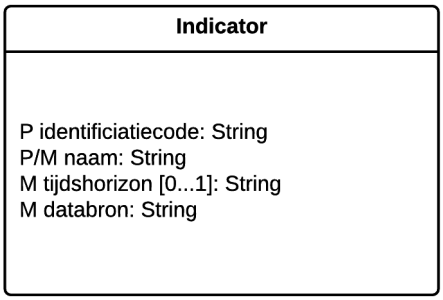
A en B zijn gerelateerd aan elkaar.

Wanneer er een ander soort relatie is, wordt dit aangeduid met behulp van een volle pijl en een tekst die de relatie beschrijft.

klasse

Klassen met bijhorende attributen die data weergeven, worden op de bovenstaande manier voorgesteld.

Bij het opbouwen van dit metamodel beginnen we ten eerste bij de centrale klasse, namelijk de 'indicator'. Binnen deze klasse zijn er verschillende attributen die gehaald zijn uit het PPINOT en MetricML metamodel. Het metamodel waar het attribuut afkomstig van is, is aangegeven door middel van een P (PPINOT) en/of een M (MetricML). Wanneer er later in het model elementen uit BAM gebruikt worden, zal dit aangegeven worden door middel van een B.



Figuur 26: centrale entiteit metamodel

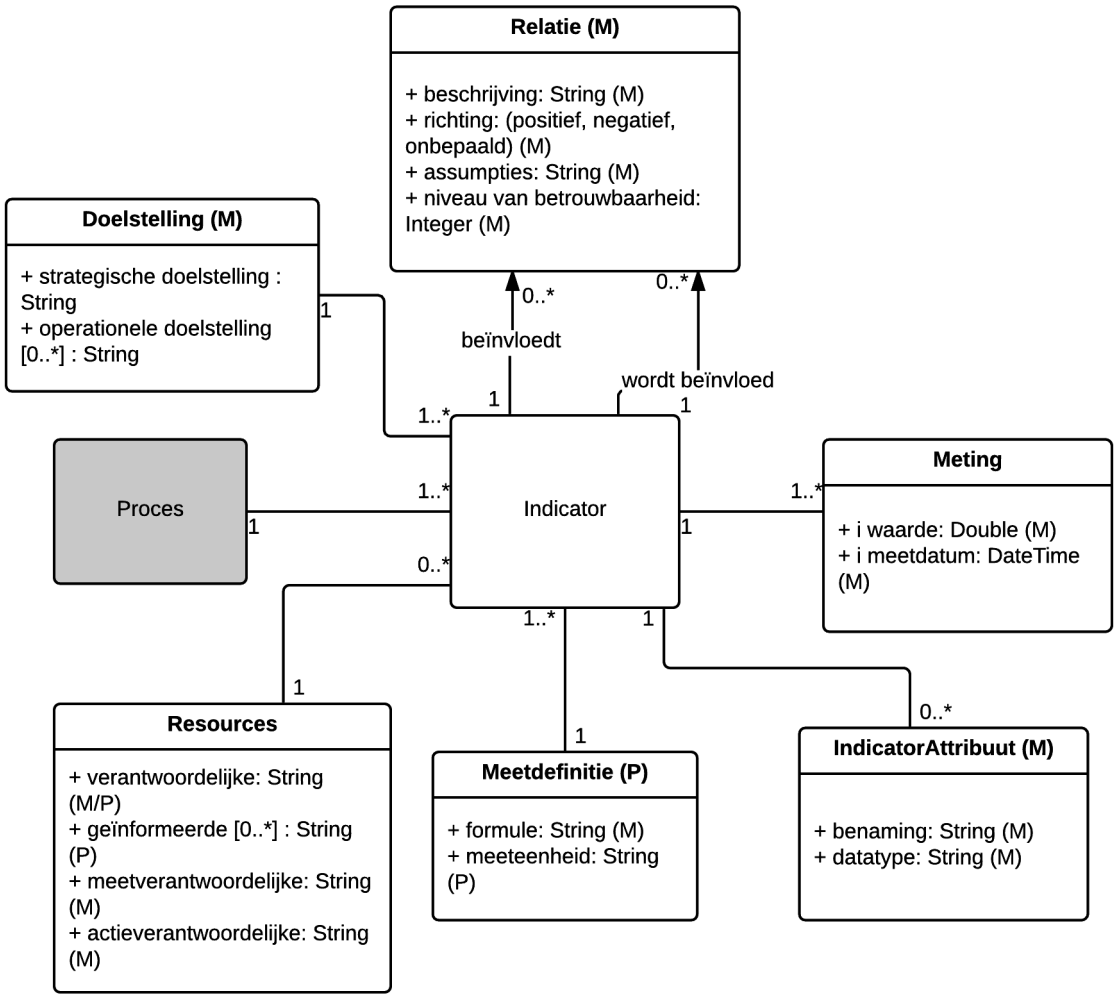
Hier volgt een uitleg over de attributen die aanwezig zijn binnen de indicator klasse. Na de naam van het attribuut, volgt ook het datatype van deze attributen.

Identificatiecode: *String*. Voor iedere procesperformantie-indicator (PPI) is er een unieke code waardoor iedere PPI gemakkelijk geïdentificeerd kan worden. Omdat de namen soms te sterk op elkaar gelijken, zorgt deze code ervoor dat er geen verwarring ontstaat tussen de verschillende PPI's [15].

Naam: *String*. Op basis van de naam wordt de PPI beschreven. De naam geeft dus al samengevat weer wat er berekend wordt bij deze PPI [14], [15].

Tijdshorizon: *String*. Hier wordt bepaald of de indicator per dag, week, maand (of een andere tijdseenheid) gemeten wordt. Wanneer het bijvoorbeeld gaat om een percentage, weet men door de tijdshorizon dat dit bijvoorbeeld een percentage per dag is. De tijdshorizon is optioneel en moet dus niet altijd gespecifieerd worden. Men kan er hier voor kiezen dat enkel bepaalde waarden mogen ingegeven uit een vooraf bepaalde lijst, hier 'tijdswaarde' genoemd. Zo kan men hier geen waarde ingeven die niet relevant is als tijdshorizon [14].

Databron: *String*. Hierbij geven we aan uit welke databron de data gehaald wordt die nodig is om de PPI te meten. Dit geven we aan in woorden, zodat er duidelijk voor iedere variabele uit de formule kan aangegeven worden waar de data vandaan komt. Dit kan ook later met behulp van een data manipulatie uitdrukking zoals een SQL statement weergegeven worden. Bijvoorbeeld: 'SELECT * FROM inschrijvingsproces', met behulp van deze uitdrukking haalt men dan alle data op uit de tabel 'inschrijvingsproces'. Deze statements moeten dan wel gekoppeld worden aan de variabelen uit de formule, aangezien iedere variabele afkomstig is uit een andere specifieke databron[14].



Figuur 27: metamodel kern

De eerste koppeling die we maken vanaf de indicator, is deze naar het bijhorende proces. Dit is een onderdeel van business process modelling (BPM) en wordt hierdoor net als de andere BPM elementen die we zullen gebruiken in het grijs gearceerd. Dit proces behoort niet tot de modelleertaal, maar is een echt proces dat zich binnen het bedrijf afspeelt. Aangezien het hier om PPI's gaat, is het belangrijk om altijd te weten aan welk specifiek proces binnen het bedrijf deze

indicator gekoppeld is. De indicator 'duur van de analyse van het verzoek' kan hier bijvoorbeeld gekoppeld worden aan het proces 'voorbeeldproces offerte'.

Vervolgens wordt de indicator ook gekoppeld aan de doelstellingen waar ze op gebaseerd is. Hierbij zijn de attributen 'strategische doelstelling' en 'operationele doelstelling'. Een indicator is rechtstreeks verbonden aan een operationele doelstelling die gekozen is. Deze is dan weer gebaseerd op een strategische doelstelling. Op deze manier ziet men duidelijk welke strategie er achterliggend is aan de indicator. Hierbij is de strategische doelstelling een verplicht attribuut en de operationele doelstelling is optioneel.

Een ander belangrijk aspect dat gemodelleerd moet worden is de relatie tussen de verschillende indicatoren. Dit wordt gedaan aan de hand van de 'relatie' entiteit met de attributen 'beschrijving', 'richting', 'assumpties' en 'niveau van betrouwbaarheid'. Bij het attribuut 'beschrijving' wordt de relatie tussen twee indicatoren in woorden beschreven. Bij de 'richting' bepaalt men of de ene indicator de andere positief of negatief beïnvloedt of dat dit onbepaald is. Verder worden er ook assumpties bepaald die gaan over deze relatie. Hier zegt men bijvoorbeeld dat er een relatie is tussen de twee indicatoren wanneer er aan bepaalde omgevingsfactoren voldaan is. En tenslotte wordt er ook het 'niveau van betrouwbaarheid' bepaald. Dit is echter een vrij subjectieve score op 10 die bepaalt hoe zeker men is van de relatie.

Verder wordt de klasse 'meting' ook nog gekoppeld aan de indicator. Hier worden al de metingen voor een bepaalde PPI bijgehouden. Deze klasse heeft als attributen 'waarde' en 'meetdatum'.

Waarde: *Waarde*. Dit geeft de waarde aan van de PPI die we meten. We geven het datatype hier aan als 'Waarde' aangezien de waarde in verschillende datatypes kan voorkomen. In ons geval kan dit een waarde zijn met de volgende datatypes: boolean, string of double. Dit attribuut is echter niet gedefinieerd op het type niveau, maar op het instantieniveau (aangegeven door de **i**). Dus dit attribuut hoort specifiek bij een bepaald geval van het proces waarbij de PPI hoort. Een voorbeeld hiervan op basis van ons proces kan de waarde zijn van 'duur van de analyse van het verzoek' voor één specifieke procesuitvoering. Wanneer er bijvoorbeeld een geaggregeerde indicator is, zullen deze individuele waardes gecombineerd worden. Als het gemiddelde van de 'duur van de analyse van het verzoek' gemeten moet worden voor de 50 laatste procesinstanties, worden eerst deze 50 waardes apart gemeten en erna gecombineerd [14].

Meetdatum: *DateTime*. Dit geeft de datum en het tijdstip weer waarop de PPI berekend is. Ook dit wordt bepaald op het instantie niveau en niet op het type niveau. De waarde van iedere instantie van de PPI zal gemeten worden op een specifiek moment [14].

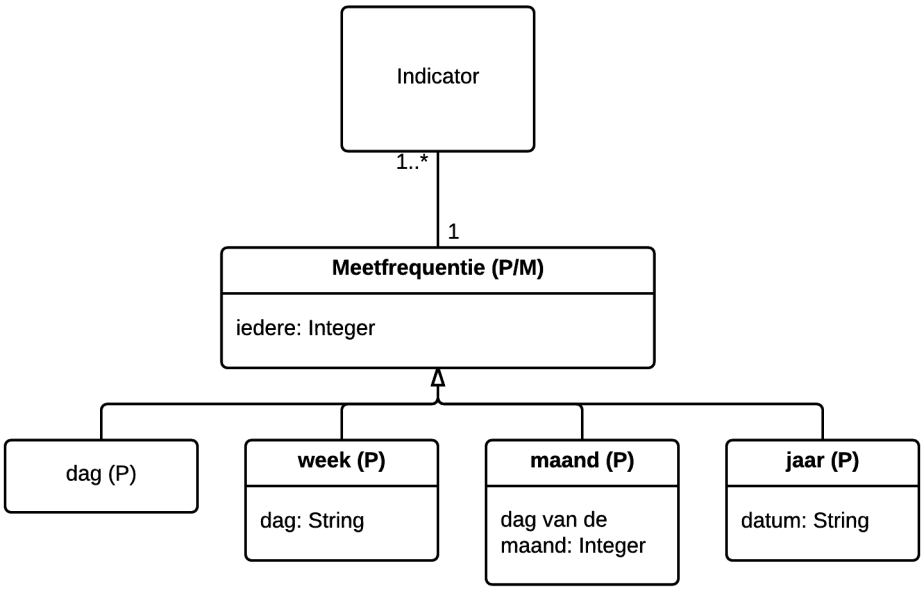
De entiteit 'Indicatorattribuut' zorgt ervoor dat er extra attributen kunnen toegevoegd worden aan de 'indicator' entiteit inclusief een bepaling van het datatype van dit attribuut.

Vervolgens wordt een belangrijk onderdeel, de meetdefinitie, die bepaalt hoe de indicator gemeten moet worden gekoppeld aan de indicator klasse. Het eerste attribuut is dan ook de formule, waarbij men in woorden uitdrukt hoe de PPI gemeten moet worden.
De meetdefinitie heeft als tweede attribuut de meeteenheid van de indicator. Dit bepaalt de eenheid waarin de PPI gemeten wordt. Dit kan bijvoorbeeld 'uren', 'euro', 'percent' of geen eenheid zijn.

Verder wordt aan dit proces ook nog aan een groep resources gekoppeld. Hier zijn er vier verschillende attributen, namelijk: 'verantwoordelijke', 'geïnformeerde', 'meetverantwoordelijke' en 'actieverantwoordelijke'.
De verantwoordelijke is de resource die verantwoordelijk is voor het proces en die op de hoogte gehouden moet worden wanneer de indicator opmerkelijke waarden vertoont. De verantwoordelijke is telkens één van de procesbeheerders en deze wordt hier gekoppeld aan de indicator, zodat mogelijke meldingen later bij de juiste persoon terechtkomen.
De geïnformeerde moet naast de verantwoordelijke ook op de hoogte gehouden worden wanneer er zich merkwaardigheden voordoen. Deze persoon is echter optioneel en er kan sprake zijn van meerdere personen die geïnformeerd moeten worden.
De meetverantwoordelijke is de persoon die verantwoordelijk is voor de meting van de PPI.
De actieverantwoordelijke tenslotte is de persoon die actie moet ondernemen op basis van de waarde van de PPI.

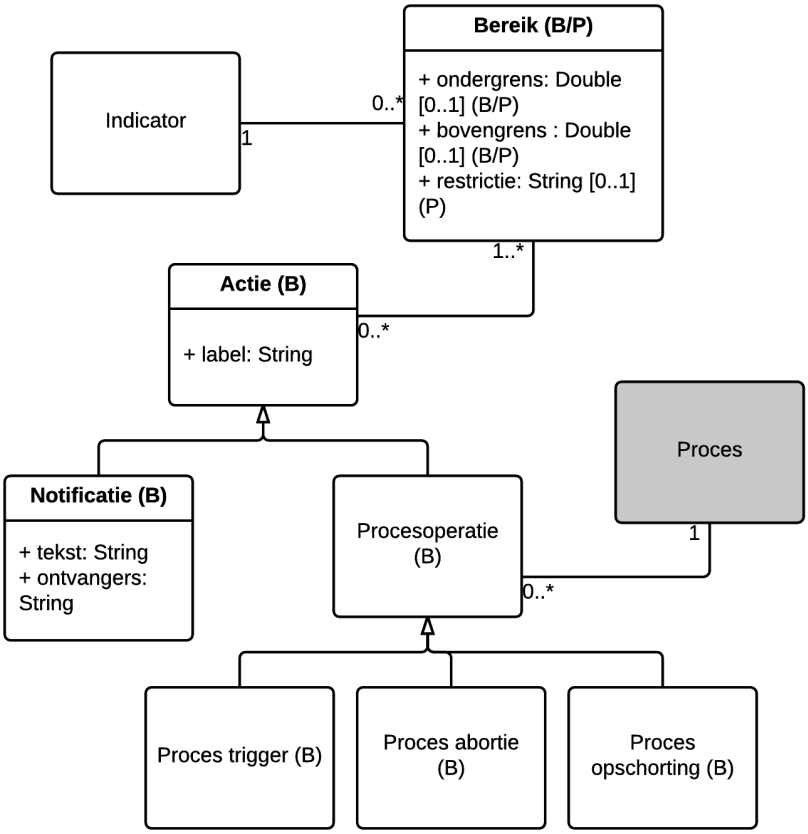
Meetfrequentie

Een ander element dat gekoppeld moet worden aan de indicator is de meetfrequentie zoals weergegeven op figuur 28. Hier wordt bepaald hoe vaak de meting gedaan zal worden, bijvoorbeeld iedere week, maand, jaar etc. Dit kan aangegeven worden door een combinatie van een integer en een tijdswaarde (dag/week/maand/jaar). Dus de combinatie van 2 en maand, wijst op een frequentie van iedere twee maanden [14],[15]. Wanneer het getal gekoppeld wordt aan een week, dan bepaalt men hiernaast ook nog dat de indicator bijvoorbeeld iedere drie weken op maandag gemeten wordt. Dit doet men aan de hand van het attribuut 'dag'. Voor de maand geld hetzelfde, maar dan geeft men weer de hoeveelste dag van de maand de indicator gemeten moet worden. Dit gebeurt met behulp van 'dag van de maand'. Voor het jaar tenslotte geldt dat er een 'datum' bepaald moet worden zoals bijvoorbeeld '7 januari', waarop de indicator jaarlijks gemeten zal worden.



Figuur 28: meetfrequentie

Target



Figuur 29: metamodel target

Hier wordt het targetbereik gedefinieerd waarvan men wil dat de waarde van de PPI zich hierbinnen situeert of hieraan voldoet. In de entiteit 'bereik' heb je een ondergrens en een bovengrens. Hiervan moet er telkens minimum één van beide gespecifieerd worden of er moet een restrictie bepaald zijn [12], [15]. Een restrictie is bijvoorbeeld '2x > 9', waarbij x de waarde van de indicator is. Maar een restrictie kan ook een specifieke waarde zijn, in het geval van een data indicator.

Ondergrens: De minimale waarde die een PPI zou mogen hebben. Wanneer men enkel de ondergrens definieert, moet de PPI gewoon een waarde groter hebben dan deze.
Bovengrens: De maximale waarde die een PPI mag hebben, zodat het bijhorend doel behaald wordt. Wanneer enkel de bovengrens gedefinieerd is, is iedere waarde hieronder goed [12], [15].

Echter heeft het definiëren van een bereik vrij weinig nut wanneer er geen acties op volgen. Daarom breiden we ons model uit met acties die kunnen volgen wanneer de waarde van een indicator afwijkt van het bereik. Deze acties zijn gemodelleerd zoals eerder gedaan in [12]. Binnen de klasse 'actie' zijn er twee mogelijke opties, namelijk een notificatie sturen naar bepaalde ontvangers of een procesoperatie uitvoeren. Mogelijke operaties die automatisch uitgevoerd kunnen worden zijn een proces trigger, proces abortie en proces opschorting.

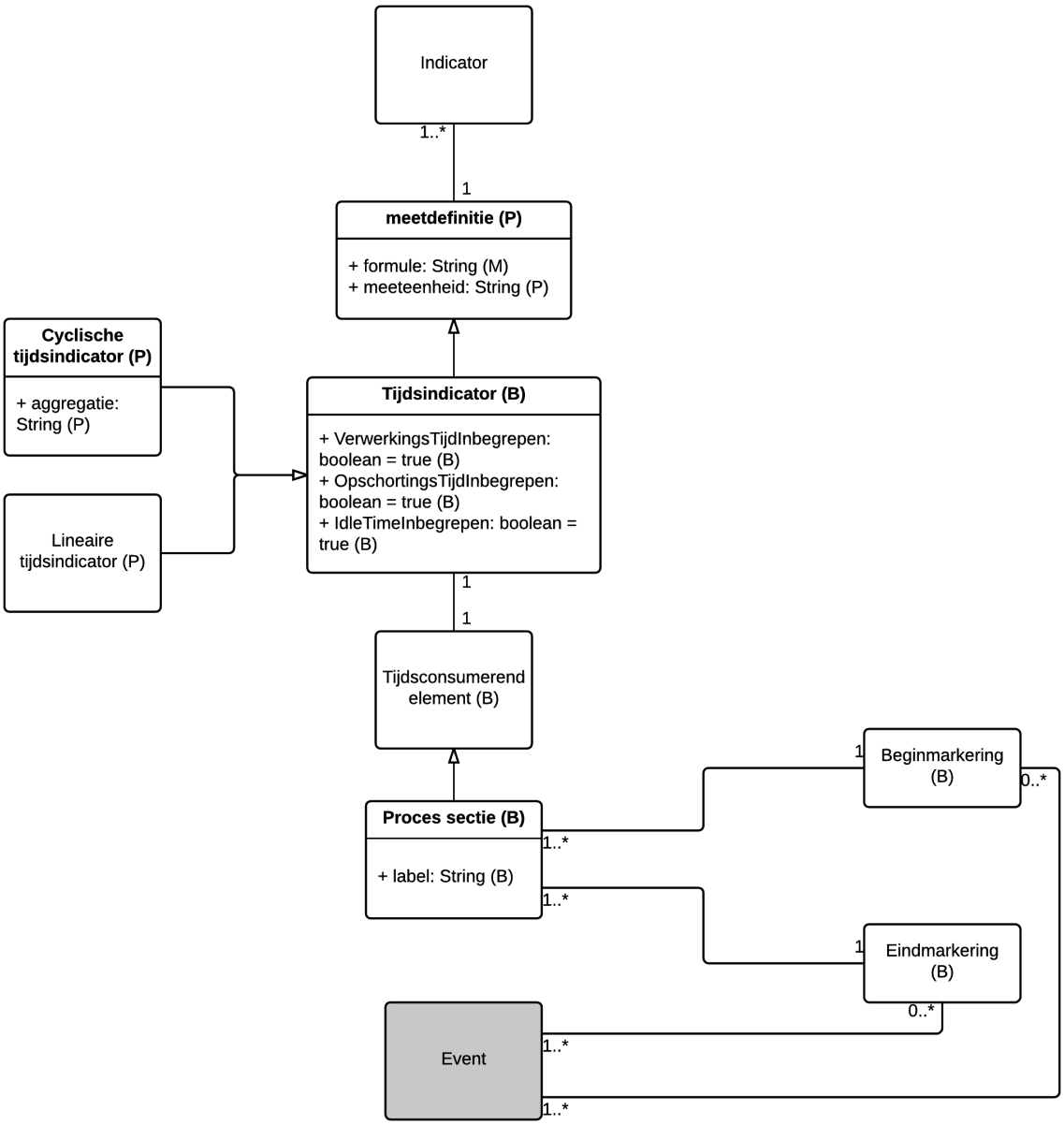
Bij de proces trigger kan men een nieuwe procesinstantie starten. Hierbij moet als attribuut de 'procesnaam' (String) meegegeven worden om te weten voor welk proces dit moet gebeuren. Wanneer het gaat om proces abortie, kan de huidige procesinstantie beëindigd worden. Bij een proces opschorting tenslotte, wordt een procesinstantie gepauzeerd. Proces abortie en proces suspensie kan men wel enkel gebruiken bij basisindicatoren of een single-instance afgeleide indicator. In andere gevallen weet men niet welke instantie beëindigd moet worden. Verder moet bij deze twee entiteiten zowel de 'procesnaam' (String) als de 'procesinstantiecode' (String) meegegeven worden [12].

Nu gaan we verder de verschillende types PPI'S modelleren in ons metamodel. Deze types zijn bepaald in onze typologie. Er worden in ons metamodel daarom basisindicatoren (tijdsindicator, telindicator, conditie indicator, data indicator), geaggregeerde indicatoren en afgeleide indicatoren gemodelleerd. Dit doen we aan de hand van meetdefinities die gelinkt worden aan de indicator klasse. Voor ieder type indicator, is er een afzonderlijke meetdefinitie.

Hier beginnen we met de vier basisindicatoren.

Tijdsindicator

Om de eerste basis tijdsindicator te definiëren, gaan we ons voornamelijk baseren op [12].



Figuur 30: metamodel tijdsindicator

De tijdsindicator wordt berekend op basis van een tijdsconsumerend element namelijk een proces sectie. Hier wordt een sectie binnen het proces aangeduid aan de hand van een begin- en eindmarkering. Deze markeringen zijn gebaseerd op een event. Events zijn gebeurtenissen die leiden tot een verandering van staat in zowel activiteiten als het proces zelf. Men kan dus bijvoorbeeld de tijd meten tussen het moment waarop het start event van 'analyse request'

getriggerd is en het moment waarop het event 'receive approval' getriggerd is. Deze PPI geven we dan de naam 'duur vanaf analyse verzoek tot ontvangst van bevestiging'. Wanneer er tijdens een proces twee of meerdere activiteiten parallel uitgevoerd worden gaan we de tijd meten tussen het start event dat als eerste getriggerd is en het eind event dat als laatste getriggerd is van deze activiteiten.

Wanneer men de duur van een proces wil berekenen, wordt de duur tussen proces start (start event getriggerd) en proces einde (end event getriggerd) gemeten. In ons voorbeeldproces kan hier dus de PPI 'duur voorbeeldproces offertes' gemeten worden. Hier wordt de duur gemeten vanaf het moment dat het start event 'receive RFQ' getriggerd is tot het moment dat één van beide end events ('project canceled' of 'project initiated') getriggerd is. Wanneer er meerdere start events zijn, beschouwt men het start event dat (als eerste) getriggerd is geweest.

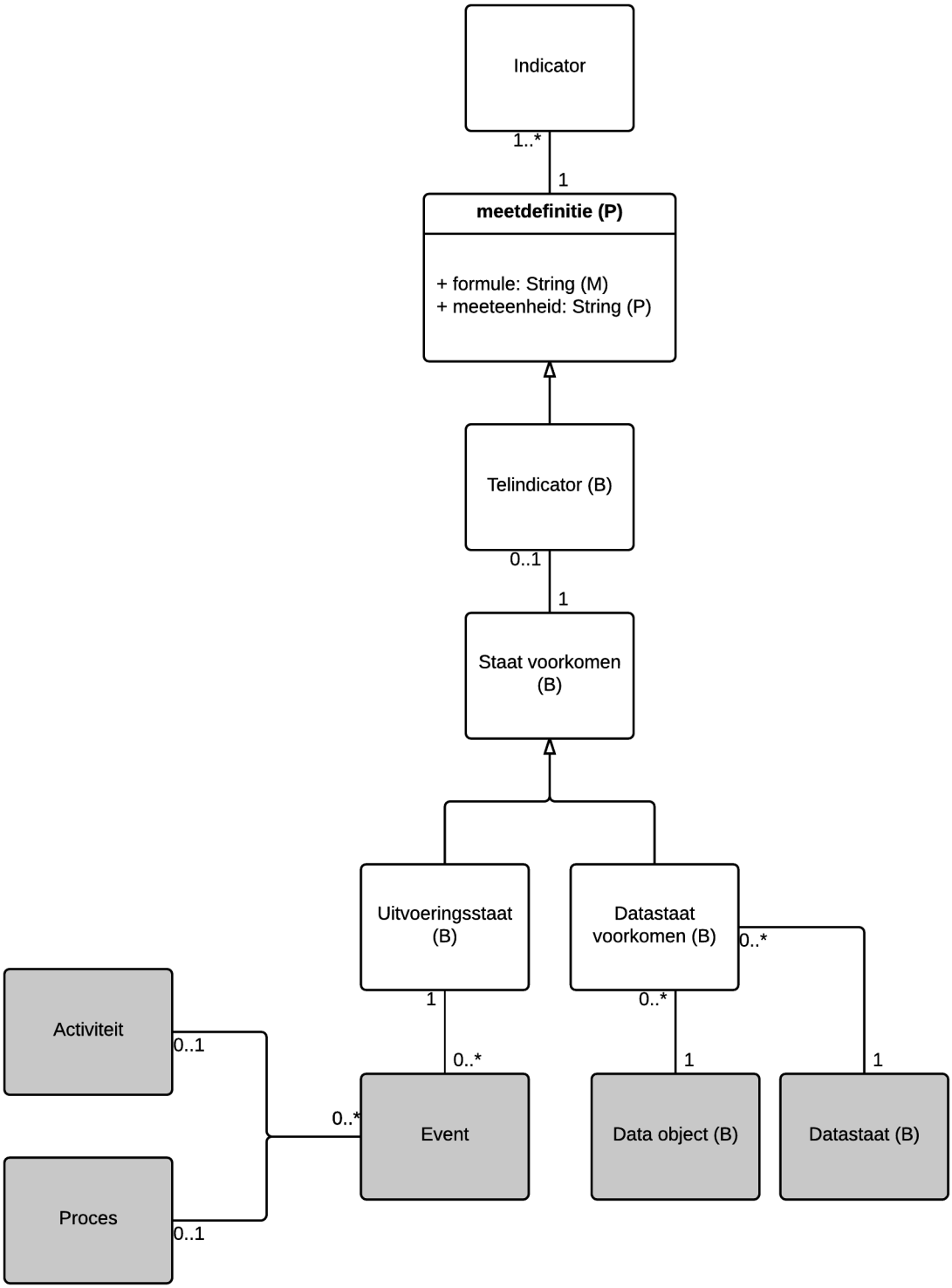
Bij de activiteit zal de duur tussen het begin (start event) en het einde (end event) van de activiteit gemeten worden. Voor ons proces kan men bijvoorbeeld 'duur offerte maken' meten.

Bij de duur die berekend wordt kan men bepalen aan de hand van events of de 'Verwerkingstijd', 'Idle Time' en 'Opschortingstijd' inbegrepen worden. De 'Verwerkingstijd is de tijd dat het proces of de activiteit werkelijk wordt uitgevoerd en de 'Idle Time', de tijd dat een activiteit is toegewezen aan een bepaald persoon voor ze wordt gedaan. De 'Opschortingstijd' is de tijd wanneer een proces of activiteit bijvoorbeeld tijdelijk opgeschort wordt. Wanneer het echter over een processectie gaat die meerdere activiteiten omvat, kan dit niet bepaald worden aan de hand van twee events. Daarom hebben we deze elementen als attribuut van de tijdsindicator verwerkt en kan men bepalen welke opgenomen worden.

Tenslotte is er ook nog een belangrijk aspect verwerkt, namelijk het feit of de tijdsindicator cyclisch of lineair wordt berekend wanneer er sprake is van een loop in het proces. Cyclisch wil zeggen dat men per loop de duur berekend. Hiervan kan men dan bijvoorbeeld het gemiddelde, minimum, maximum of de som nemen. Dit kan aan de hand van de 'cyclische tijdsindicator' en het bijhorende attribuut 'aggregatie' dat de soort aggregatie bepaalt. Wanneer men de tijd lineair meet, zal dit gedaan worden van de eerste activiteit tot de laatste activiteit in de laatste loop.

Telindicator

Om de telindicator te modelleren zullen we ons ook voornamelijk baseren op [12].



Figuur 31: metamodel telindicator

De telindicator telt hoe vaak een bepaalde gebeurtenis zich voordoet binnen één procesinstantie. Deze telindicator is altijd geassocieerd met exact één 'staat voorkomen'. Een voorbeeld hiervan kan dus de volgende indicator zijn: 'aantal keren dat het project succesvol geïnitialiseerd is'. Deze PPI telt dan hoe vaak de activiteit 'initiate project' succesvol beëindigd is binnen één procesinstantie.

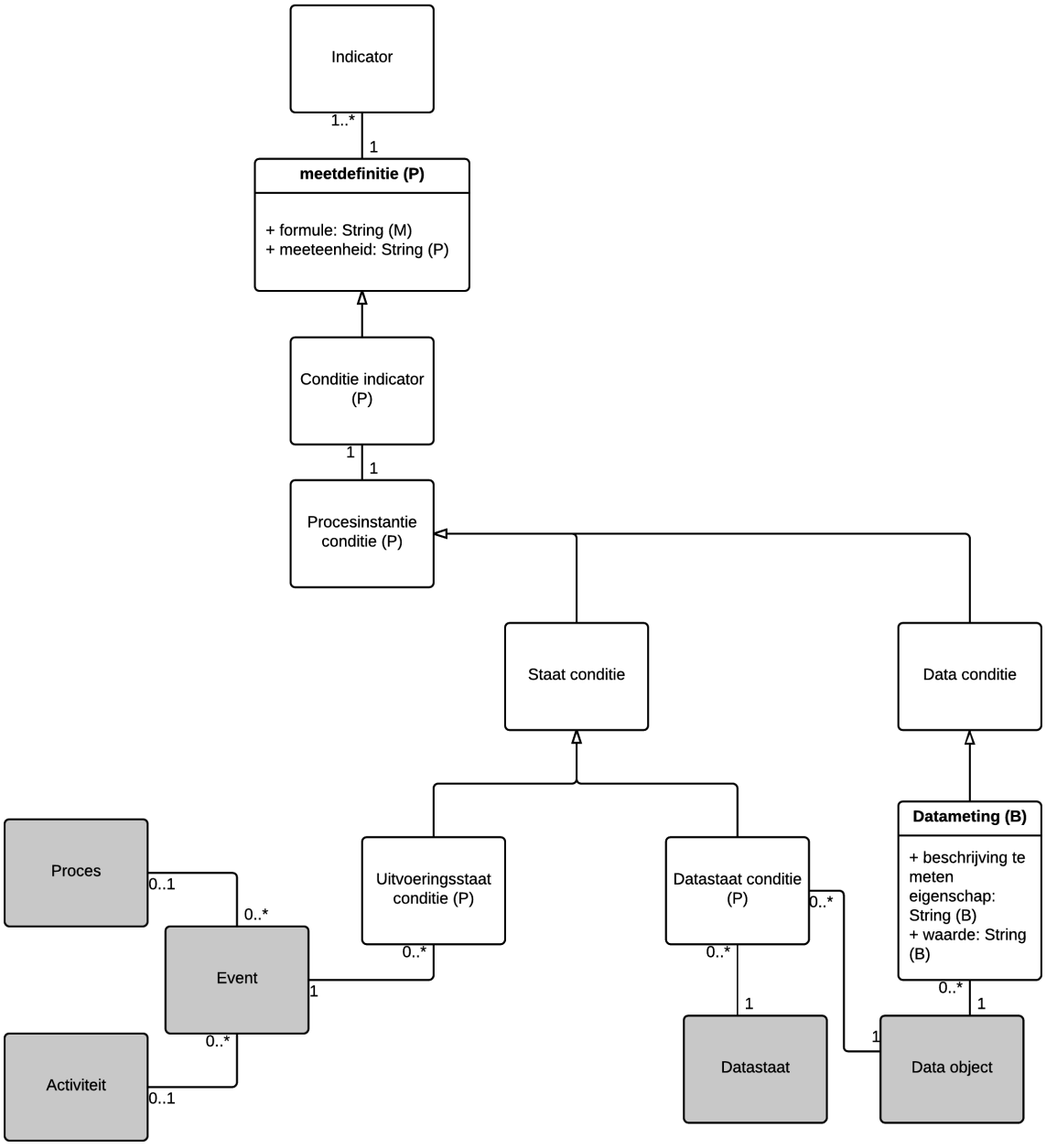
Bij staat voorkomen telt men hoe vaak een staat voorkomt tijdens procesuitvoering van een procesinstantie. Binnen staat voorkomen wordt er een onderscheid gemaakt tussen 'uitvoeringsstaat' en 'datastaat voorkomen'. Bij 'datastaat voorkomen' wordt er geteld hoe vaak een BPMN data object zich in een bepaalde staat bevindt. In ons voorbeeldproces kan men dus bijvoorbeeld tellen hoe vaak het data object 'offer' zich in de staat 'denied' bevindt voor één procesinstantie. Dit gaat hier dus 0 of 1 zijn, omdat deze staat maar één maal voor kan komen binnen een procesinstantie.

De 'uitvoeringsstaat' wordt gekoppeld aan een event. Door dit event zal een proces of activiteit gaan naar een bepaalde staat. Hierdoor kan men dus tellen hoe vaak een proces of activiteit zich in een bepaalde staat bevindt. Wanneer dit een activiteit is, wordt er geteld hoe vaak een activiteit zich in een bepaalde staat bevindt tijdens een procesinstantie. Een voorbeeld hiervan is de indicator 'aantal keren dat het maken van de offerte met andere partijen tijdelijke opgeschort is'. Hier wordt er dus gekeken hoe vaak de activiteit 'make offer with other parties' zich in de staat 'geschorst' bevindt tijdens één procesinstantie door het voorkomen van een specifiek event.

Wanneer het over een proces gaat, kan men tellen hoe vaak het proces zich in een bepaalde staat bevond. Wij kunnen bijvoorbeeld een PPI bepalen 'aantal keren dat het offerte proces geannuleerd wordt'. Hier berekent men hoe vaak het proces zich in de staat 'geannuleerd' bevindt binnen één procesinstantie doordat er een pause event voorkwam.

Conditie indicator

Om de conditie indicator te modelleren hebben we ons vooral gebaseerd op [12] en [15].



Figuur 32: metamodel conditie indicator

Wanneer een conditie indicator gemeten wordt, is het resultaat een boolean waarde. De enige twee waardes die deze indicator kan hebben, zijn dus ‘true’ of ‘false’. Bij deze indicatoren wordt gemeten of er voldaan is aan een bepaalde voorwaarde in zowel lopende als beëindigde procesinstanties. Deze voorwaarde vindt men in ‘proces instantie conditie’ en op basis hiervan wordt de waarde van de indicator bepaald.

Hierbij is er een onderscheid tussen twee soorten voorwaardes namelijk een staat conditie of een data conditie. Bij een staat conditie zijn er twee types condities namelijk een uitvoeringsstaat

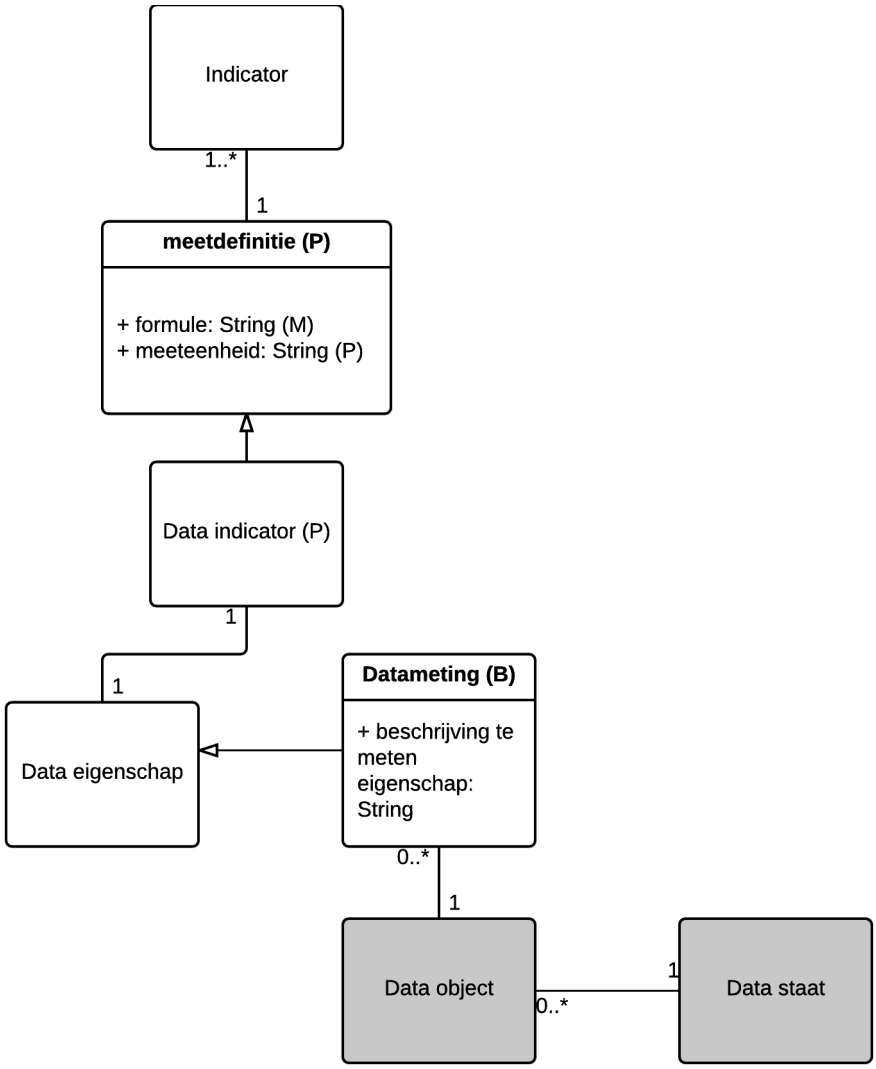
conditie en een datastaat conditie. Bij de uitvoeringsstaat conditie wordt er gekeken of een BP (bedrijfsproces) element zich in een bepaalde staat bevindt aan de hand van een event dat voorkomt. Net zoals bij de telindicator maakt men een onderscheid tussen twee bedrijfsproceselementen waar een event een invloed op kan hebben, namelijk een proces of een activiteit. Een bedrijfsproceselement kan zich namelijk enkel in een staat bevinden wanneer een event zich voordoet. Een proces kan dus tijdelijk geannuleerd worden wanneer het pause event voorkomt. Een PPI kan hier zijn het voldoen aan de voorwaarde dat de activiteit 'initiate project' zich momenteel in de staat toegewezen bevindt.

Bij de datastaat conditie kijkt men of een data object zich in een bepaalde staat bevindt. Dus aan de hand van ons voorbeeldproces kan men dus nagaan of het data object 'offer' zich in de staat 'denied' bevindt.

Tenslotte bij de data conditie, kijkt men of een data object een bepaalde waarde heeft. Dit doet men aan de hand van de klasse 'datameting' waarin bepaald wordt welke eigenschap van het data object gemeten moet worden en welke waarde hij zou moeten hebben. Een voorbeeld van dit eerste is het voldoen aan de voorwaarde dat het data object 'offer' een bedrag groter dan 20000 heeft. Hierbij moet men dus kijken in de regel 'bedrag' in het data object, wat de waarde van het bijhorende veld is.

Data indicator

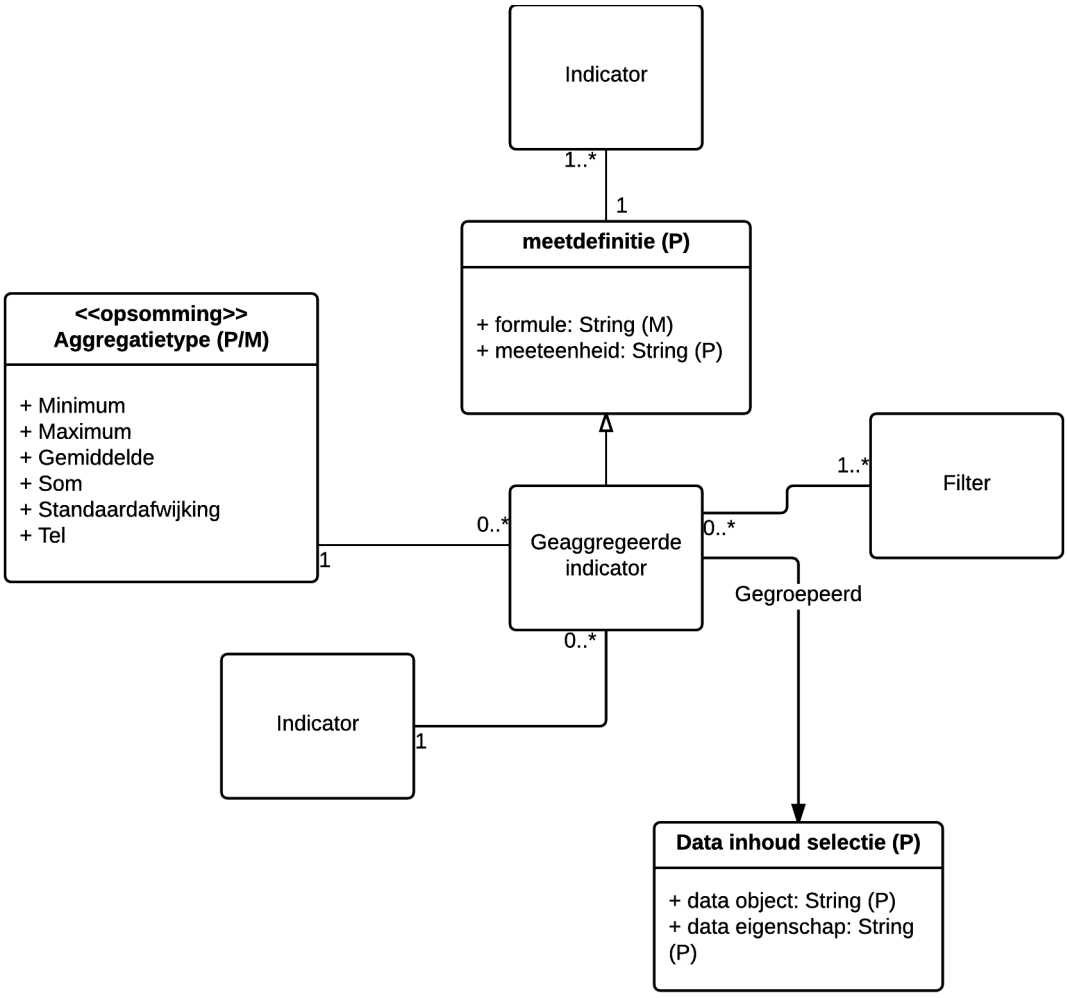
De laatste basisindicator tenslotte is de data indicator. Deze meet de waarde van een bepaalde eigenschap van een data object.



Figuur 33: metamodel data indicator

De data indicator wordt gemodelleerd door de klasse ‘Datameting’ te koppelen aan een data object. In ‘Datameting’ wordt er bepaald welke eigenschap er gemeten moet worden van het data object. Hier kan er dus een PPI bepaald worden die de waarde bevat van de eigenschap ‘leverancier’ in het data object ‘offer’. Men moete echter ook bepalen in welke datastaat het object zich bevindt bij de meting aangezien de waarde van de eigenschap kan verschillen naargelang de staat.

Geaggregeerde indicator



Figuur 34: metamodel geaggregeerde indicator

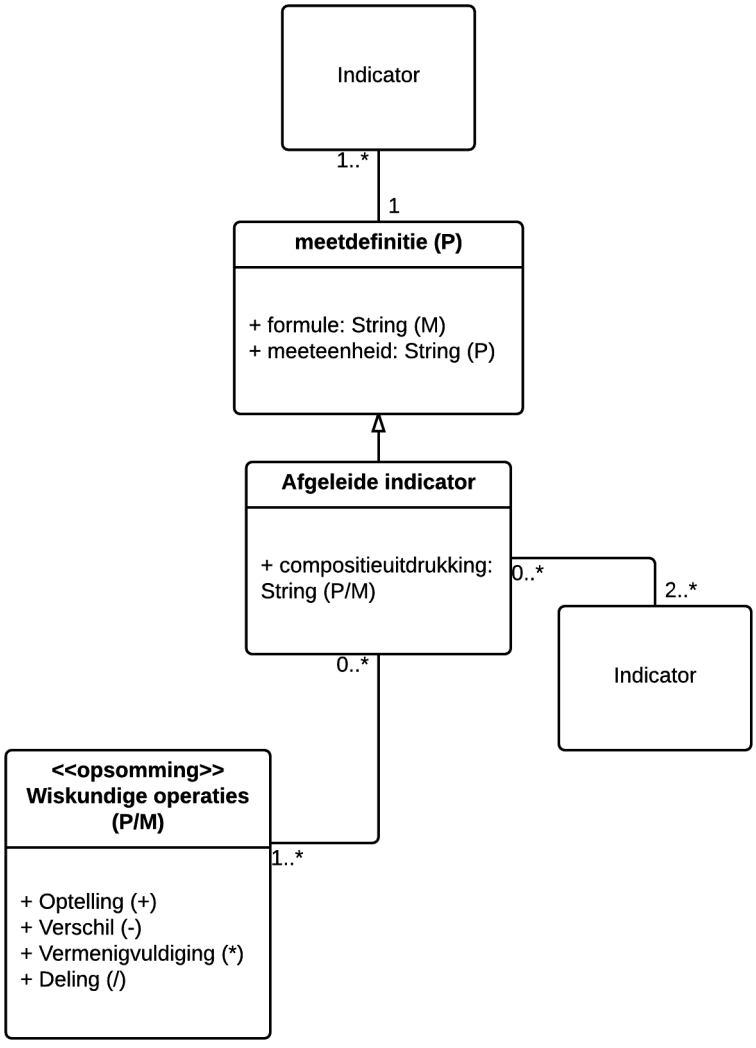
Een geaggregeerde indicator groepeert een indicator over verschillende instanties heen. Om deze aggregatie te kunnen uitvoeren, is deze indicator gebaseerd op een indicator, meestal een basisindicator of een afgeleide indicator. Hierdoor weten we voor welk proces er indicatoren geaggregeerd zullen worden. Om te bepalen welke instanties er gebruikt worden om de aggregatie te berekenen is het bij deze indicator noodzakelijk om een filter te koppelen. Per geaggregeerde indicator is er dus de koppeling van minimum één filter.

Deze geaggregeerde indicatoren kunnen gegroepeerd worden op de verschillende manieren die weergegeven zijn in het aggregatie type. Er kan bijvoorbeeld een minimumwaarde worden genomen van de indicator voor de verschillende instanties, een som van de indicator voor de verschillende instanties etc. Deze geaggregeerde indicator is nodig omdat een basis indicator of een afgeleide indicator in verband met één procesinstantie vaak weinig zeggend is. Dit is ook het geval voor de telindicator ‘aantal keren dat het project succesvol geïnitialiseerd is’. Deze PPI telt voor één instantie hoe vaak de activiteit ‘initiate project’ succesvol beëindigd is. Aangezien deze activiteit maar één maal voorkomt, zegt dit niet veel. Daarom willen we kijken hoe vaak deze

activiteit succesvol beëindigd is in de laatste maand. Daarvoor maken we de som van deze basis telindicator over alle instanties heen van de laatste maand. Op deze manier wordt er een geaggregeerde indicator verkregen die hetzelfde zegt maar dan over verschillende instanties heen.

Als extra uitbreiding kan de indicator verder ook gegroepeerd worden per waarde van een data eigenschap van een data object. Dit gebeurt op basis van de 'data inhoud selectie' met als attributen het data object dat beschouwd wordt en de eigenschap waarop gegroepeerd moet worden. Wanneer er dus bijvoorbeeld een eigenschap A is met mogelijke waardes B, C en D, dan wordt de indicator berekend per groep van instanties die dezelfde waarde bezitten.

Afgeleide indicator



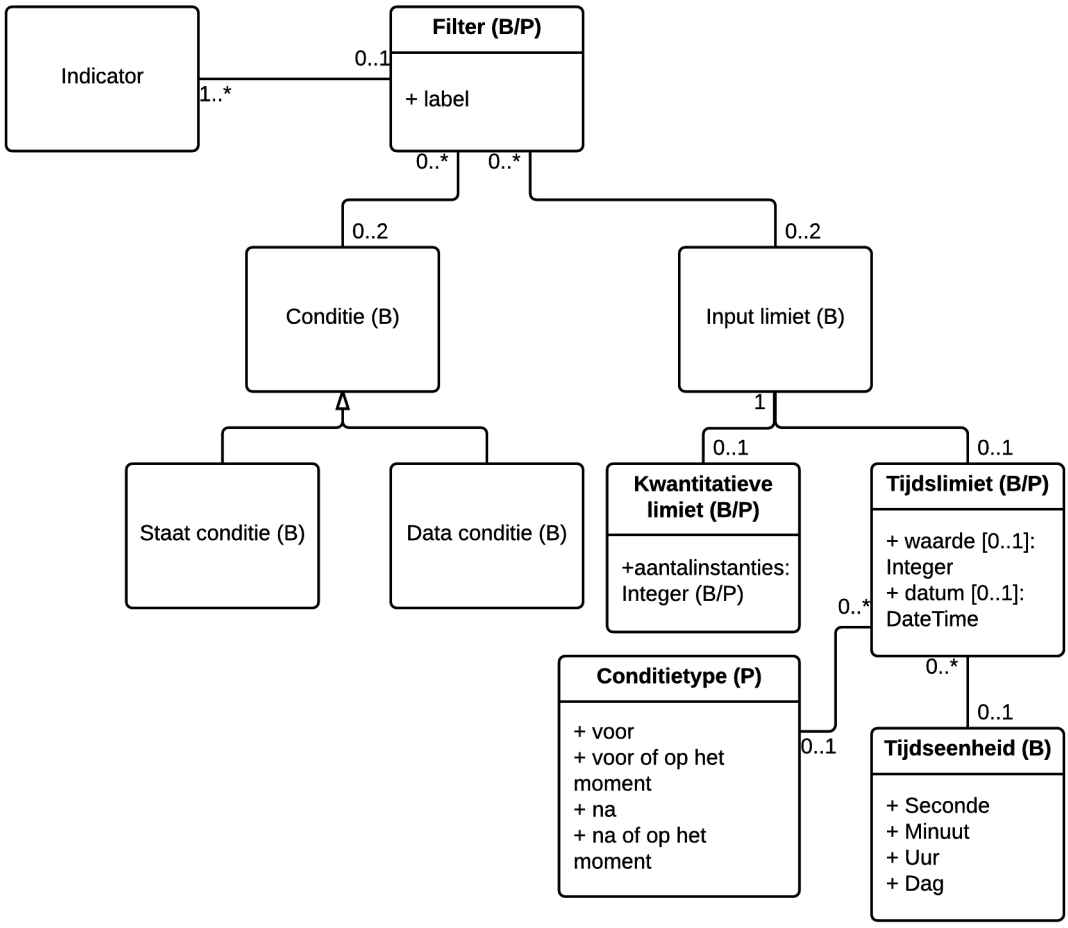
Figuur 35: metamodel afgeleide indicator

Een afgeleide indicator is gebaseerd op twee of meer indicatoren. Deze indicatoren worden gecombineerd op basis van wiskundige operaties (zoals optelling, verschil, vermenigvuldiging en

deling). In de compositie expressie wordt er weergegeven hoe de indicatoren gecombineerd worden aan de hand van deze operaties. Deze indicatoren kunnen zowel basisindicatoren als geaggregeerde indicatoren zijn. Deze worden verder geplaatst in een superklasse 'indicator voor afleiding'. Een voorbeeld van een PPI voor ons proces kan de deling zijn van twee telindicatoren. We kunnen een PPI maken die het aantal geaccepteerde offertes vergelijkt met het aantal gemaakte offertes. Dit gebeurt op basis van de volgende formule: aantal keer dat het data object 'offer' in de staat 'approved' is / aantal keer dat het data object 'offer' in de staat 'made' is. Aangezien deze staten slechts één maal voorkomen in het proces, is het echter nuttiger om deze PPI te berekenen op basis van de geaggregeerde indicatoren. De formule wordt dan som(aantal keer dat het data object 'offer' in de staat 'approved' is) / som(aantal keer dat het data object 'offer' in de staat 'made' is). Op basis hiervan kunnen we dus bijvoorbeeld voor de laatste week kijken hoeveel gemaakte offertes relatief geaccepteerd worden.

Filter

Nu dat we de meetdefinities gemodelleerd hebben voor de verschillende types indicatoren, gaan we verder naar het modelleren van de filters. Het model hiervan is grotendeels gebaseerd op [12].

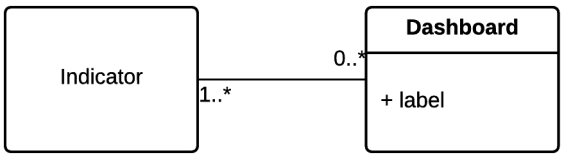


Figuur 36: metamodel filter

Wanneer we indicatoren enkel willen definiëren op basis van een deel proces instanties, maken we gebruik van een filter of meerdere filters. Deze filters kunnen zowel voor basis als geaggregeerde indicatoren gebruikt worden. Een filter kan bestaan uit zowel een conditie en/of input limiet. Hiervan moet er minimum één limiet gedefinieerd zijn. De conditie limiet zorgt ervoor dat enkel indicatoren gemeten worden van procesinstanties waar er aan een bepaalde conditie voldaan is. Een voorbeeld hiervan kan zijn dat een data object een bepaalde waarde moet hebben. Binnen de conditie limiet is er een opdeling tussen staat condities en data condities. De manier waarop deze gemodelleerd worden is aangegeven in het model van de conditie indicator op figuur 31.

Bij de input limiet wordt er nog een onderscheid gemaakt tussen een kwantitatieve limiet en een tijdslimiet.
Bij de kwantitatieve limiet wordt er bepaald hoeveel recente instanties er gebruikt worden om de indicatoren te definiëren. Dus hierbij worden bijvoorbeeld enkel de 500 laatste instanties gebruikt.
Bij de tijdslimiet bepaalt men hoe lang men terug gaat in de tijd om de indicator te berekenen. Wanneer men de indicator vandaag definieert, kan men zeggen dat we instanties gebruiken vanaf vijf dagen geleden bijvoorbeeld. Verder kan er ook bepaald worden dat men instanties beschouwt na of voor een specifieke datum. Deze datum wordt gekozen in 'datum' binnen de tijdslimiet. Dit wordt gecombineerd met een conditietype, bijvoorbeeld 'na' dat ons zegt dat we al de instanties na de gekozen datum bekijken. Bij de combinatie van deze twee limieten, zal de eerste die men tegenkomt gelden. Dus wanneer er meer dan 500 instanties zijn in deze vijf dagen, zal men de laatste 500 instanties gebruiken.

Dashboard element (B)



Figuur 37: metamodel dashboard element

Bij het laatste element in ons metamodel, wordt er een dashboard element aan een indicator gekoppeld. Hierbij wordt aangegeven voor welke dashboards de indicatoren gebruikt zullen worden. Ieder van deze dashboards zal overeenkomen met één of meerdere doelstellingen. Zo kan men kijken of deze doelstellingen behaald zijn. Om dit element nog verder uit te breiden op langere termijn, zou men hier ook al kunnen aangeven op welke manier deze indicator gevisualiseerd kan worden [12].

4.5 Demonstratie metamodel

Ter demonstratie en evaluatie van ons metamodel, zullen we PPI's definiëren op basis van dit model. Dit zullen we doen voor de voorbeelden die hierboven reeds zijn aangegeven voor al de types te definiëren.

Het bedrijf dat gekozen is om ons model op te demonstreren is een fictief IT-consultancy bedrijf 'BRAS' dat projectmatig werkt. De projecten draaien vooral om het netwerk- en storage aspect.

De missie van dit bedrijf is "ervoor zorgen dat klanten efficiënter kunnen werken met behulp van onze IT-oplossingen".
De visie is "behoren tot de tien meest innovatieve bedrijven binnen België door voortdurend te vernieuwen".
De strategie om dit te bereiken is "investeren in jonge personen die nog veel ideeën hebben en nooit stoppen met het veranderen van de processen zodat deze voortdurend innovatief blijven".

Enkele van de strategische doelstellingen die aan deze strategie gekoppeld kunnen worden zijn:

- Binnen de twee jaar 25 nieuwe krachten aannemen die minder dan vijf jaar werkervaring bezitten
- De eigen interne processen optimaliseren
- Een innovatie lab opstarten binnen de twee jaar
- Omzet verhogen zodat er weer geïnvesteerd kan worden in innovatie

Een deel belangrijke processen uit dit bedrijf zijn:

- het aanwervingsproces
- het offerte proces
- het proces in verband met de uitvoering van de IT-projecten
- …

Tijdsindicatoren

Om PPI's te definiëren aan de hand van ons metamodel, gebruiken we eerst de centrale entiteit op figuur 26. Op basis hiervan komen we voor onze eerste voorbeeldindicator 'duur voorbeeldproces offertes' tot de volgende tabel:

Indicator	
Identificatiecode:	PPI-001
Naam:	Duur voorbeeldproces offertes
Tijdshorizon:	/
Databron:	Deze info wordt gehaald uit een file genaamd 'duur offerte proces' dat uit het ERP systeem geëxtraheerd wordt

Tabel 15: demonstratie PPI-001 centrale entiteit

Aan deze centrale entiteit worden hierna de elementen uit figuur 27 gekoppeld. Het resultaat hiervan is te zien op de onderstaande tabellen:

Doelstelling	
Strategische doelstelling:	De eigen processen optimaliseren.
Operationele doelstelling:	Het offerteproces sneller laten verlopen dan 60 dagen.

Tabel 16: demonstratie PPI-001 doelstelling

Meetdefinitie	
Formule:	Tijd tussen trigger start event en trigger end event
Meeteenheid:	Dagen

Tabel 17: demonstratie PPI-001 meetdefinitie

Resources	
Verantwoordelijke:	PM03 (projectmanager) – Jean-Jacques Duchateau
Geïnformeerde:	/
Meetverantwoordelijke:	BI04 (BI medewerker) – Jef Terwingen
Actieverantwoordelijke:	OM02 (operationeel manager) – Pieter Goossens

Tabel 18: demonstratie PPI-001 resources

Proces:	Voorbeeldproces offertes

Tabel 19: demonstratie PPI-001 proces

Verder wordt aan de indicator ook nog gekoppeld hoe vaak ze gemeten moet worden. Dit gebeurt op basis van figuur 28.

Meetfrequentie	
Iedere:	1
Tijd:	maand
Dag van de maand:	7

Tabel 20: demonstratie PPI-001 meetfrequentie

Bovenstaande gegevens moeten kunnen opgehaald worden voor al de types indicatoren. Nu gaan we verder op het deel dat specifiek is aan de tijdsindicator. Dit deel wordt gedefinieerd op basis van figuur 30. Deze onderdelen worden aan de indicator gekoppeld via de meetdefinitie.

Proces sectie	
Label:	Volledig proces
Beginmarkering:	Start event 'receive RFQ'
Eindmarkering:	End event 'project initiated' OR 'project canceled'

Tabel 21: demonstratie PPI-001 tijdsindicator

Aan de indicator zelf worden ook nog een bereik en acties gekoppeld zoals gemodelleerd in figuur 29.

Bereik	
Ondergrens:	/
Bovengrens:	60

Tabel 22: demonstratie PPI-001 bereik

Actie:	
Actie:	Proces opschorting
Procesnaam:	Voorbeeldproces offertes
Procesinstantiecode:	1254

Tabel 23: demonstratie PPI-001 actie

Tenslotte kan men ook bepalen op welk dashboard de indicator weergegeven zal worden.

Dashboard	
Label:	Dashboard offerteproces

Tabel 24: demonstratie PPI-001 dashboard

Al de bovenstaande gegevens kunnen dan samen gegroepeerd worden tot één tabel, zoals we hieronder voor de andere indicatoren gedaan hebben.

De volgende indicatoren die we gedefinieerd hebben, zijn gebaseerd op de scenario's uit 3.5. Op deze manier kunnen we aantonen dat al deze scenario's gemodelleerd kunnen worden door ons eigen model.

Scenario 1:
Men wil een wiskundige formule toepassen op bestaande indicatoren om op deze manier een nieuwe indicator te bekomen.

Dit scenario modelleren we op basis van het type 'afgeleide indicator' uit ons model. De PPI die op basis hiervan gedefinieerd is, vinden we in tabel 25. De elementen die specifiek ervoor zorgen dat dit scenario gemodelleerd kan worden, duiden we in het vet aan. Dit zal tevens het geval zijn voor de andere scenario's.

Indicator	
Identificatiecode:	PPI-013
Naam:	Verhouding geaccepteerde offertes ten opzichte van het aantal gemaakte offertes
Tijdshorizon:	/
Databron:	Deze info wordt gehaald uit een file genaamd 'proces offertes' dat uit het project management systeem wordt geëxtraheerd
Strategische doelstelling:	Omzet verhogen zodat er weer geïnvesteerd kan worden in innovatie
Operationele doelstelling:	D04 – zoveel mogelijk offertes die goedgekeurd worden
Formule:	Aantal keren dat het data object 'offer' in de staat 'approved' is / aantal keren dat het data object 'offer' in de staat 'made' is
Meeteenheid:	Geen
Verantwoordelijke:	PM03 – Jean-Jacques Duchateau
Geïnformeerde:	PA05 (proces assistent) – Filip Aerts
Meetverantwoordelijke:	BI04 (BI medewerker) – Jef Terwingen
Actieverantwoordelijke:	OM02 (operationeel manager) – Pieter Goossens
Proces:	Voorbeeldproces offertes
Iedere:	1
Tijd:	maand
Dag vd maand:	5
Ondergrens:	/
Bovengrens:	/
Restrictie:	x = 1
Actie:	notificatie
Tekst:	X
Compositieuitdrukking:	**PPI-016 / PPI-017** **Ervan uitgaande dat PPI-016 = aantal keren dat het data object 'offer' in de staat 'approved' is en PPI-017 = aantal keren dat het data object 'offer' in de staat 'made' is**
Wiskundige operatie:	**Deling**
Indicatoren:	**PPI-016 en PPI-017**
Dashboard:	Dashboard offerteproces

Tabel 25: demonstratie PPI-013

Aangezien deze formule gebaseerd is op twee basisindicatoren, zegt deze ons weinig. Daarom wordt de volgende PPI een afgeleide indicator gebaseerd op twee geaggregeerde indicatoren.

Indicator	
Identificatiecode:	PPI-014
Naam:	Verhouding geaccepteerde offertes in een bepaalde periode ten opzichte van het aantal gemaakte offertes
Tijdshorizon:	/
Databron:	Deze info wordt gehaald uit een file genaamd 'proces offertes' dat uit het project management systeem wordt geëxtraheerd
Strategische doelstelling:	Omzet verhogen zodat er weer geïnvesteerd kan worden in innovatie
Operationele doelstelling:	D04 – zoveel mogelijk offertes die goedgekeurd worden
Formule:	(Som(Aantal keren dat het data object 'offer' in de staat 'approved' is) / Som(aantal keren dat het data object 'offer' in de staat 'made' is)) x100
Meeteenheid:	Percentage
Verantwoordelijke:	PM03 – Jean-Jacques Duchateau
Geïnformeerde:	PA05 (proces assistent) – Filip Aerts
Meetverantwoordelijke:	BI04 (BI medewerker) – Jef Terwingen
Actieverantwoordelijke:	OM02 (operationeel manager) – Pieter Goossens
Proces:	Voorbeeldproces offertes
Iedere:	1
Tijd:	maand
Dag vd maand:	13
Ondergrens:	75
Bovengrens:	/
Restrictie:	/
Actie:	notificatie
Tekst:	X
Ontvangers:	OM02
Compositieuitdrukking:	PPI-018 / PPI-019 Ervan uitgaande dat PPI-018 = som aantal keren dat het data object 'offer' in de staat 'approved' is en PPI-019 = som aantal keren dat het data object 'offer' in de staat 'made' is

Wiskundige operatie:	Deling
Indicatoren:	PPI-018 en PPI-019
Dashboard:	Dashboard offerteproces

Tabel 26: demonstratie PPI-014

Scenario 2:

Men wil de tijd meten tussen twee tijdstippen van het proces. Bijvoorbeeld de tijd meten tussen de start van activiteit A en het einde van activiteit B.

Indicator	
Identificatiecode:	PPI-002
Naam:	Duur offerte maken
Tijdshorizon:	/
Databron:	Deze info wordt gehaald uit een file genaamd 'duur offerte proces' dat uit het ERP systeem geëxtraheerd wordt
Strategische doelstelling:	De eigen processen optimaliseren en efficiënter maken
Operationele doelstelling:	D05 – offertes maken moet bij kleine projecten minder lang dan 4 uur duren
Formule:	Tijd tussen het moment dat de activiteit 'make offer' toegewezen is aan een resource en het moment dat deze activiteit voltooid is
Meeteenheid:	Uren
Verantwoordelijke:	PM03 – Jean-Jacques Duchateau
Geïnformeerde:	PA05 (proces assistent) – Filip Aerts
Meetverantwoordelijke:	BI04 (BI medewerker) – Jef Terwingen
Actieverantwoordelijke:	OM02 (operationeel manager) – Pieter Goossens
Proces:	Voorbeeldproces offertes
Iedere:	2
Tijd:	week
Dag:	dinsdag
Ondergrens:	/
Bovengrens:	4
Restrictie:	/
Actie:	Notificatie
Tekst:	Duur offerte maken van 4 uur overschreden
Ontvangers:	OM02
Label:	**Offerte duur**
Beginmarkering:	**Start event 'make offer'**

Eindmarkering:	**End event 'make offer'**
Dashboard:	Dashboard offerteproces

Tabel 27: demonstratie PPI-002

Scenario 3:
Men wil tellen hoe vaak activiteit A of het proces zich in de staat 'lopend' (of een andere staat) bevindt. Een activiteit of proces bevindt zich in een bepaalde staat doordat er een event voorkomt. Men wil tellen hoe vaak een data object G zich in datastaat J bevindt.

Indicator	
Identificatiecode:	PPI-004
Naam:	Aantal keren dat het project succesvol geïnitialiseerd is
Tijdshorizon:	/
Databron:	Deze info wordt gehaald uit een file genaamd 'proces offertes' dat uit het project management systeem wordt geëxtraheerd
Strategische doelstelling:	De eigen processen optimaliseren.
Operationele doelstelling:	D02 – optimalisatie projectinitiatie
Formule:	Aantal keren dat het event 'project initiated' getriggerd is
Meeteenheid:	Geen
Verantwoordelijke:	PM03 – Jean-Jacques Duchateau
Geïnformeerde:	PA05 (proces assistent) – Filip Aerts
Meetverantwoordelijke:	BI04 (BI medewerker) – Jef Terwingen
Actieverantwoordelijke:	OM02 (operationeel manager) – Pieter Goossens
Proces:	Voorbeeldproces offertes
Iedere:	1
Tijd:	maand
Dag vd maand / week / ect (optie):	15
Ondergrens:	/
Bovengrens:	/
Restrictie:	x = 1
Actie:	notificatie
Tekst:	X
Ontvangers:	OM02
Staat voorkomen:	**Uitvoeringsstaat**
Event:	**End event 'project initiated'**
Proces:	**Voorbeeldproces offertes**

Dashboard:	Dashboard offerteproces

Tabel 28: demonstratie PPI-004

Indicator	
Identificatiecode:	PPI-005
Naam:	Aantal keren dat de offerte niet goedgekeurd wordt
Tijdshorizon:	/
Databron:	Deze info wordt gehaald uit een file genaamd ‘proces offertes’ dat uit het project management systeem wordt geëxtraheerd
Strategische doelstelling:	De eigen processen optimaliseren.
Operationele doelstelling:	D04 – zoveel mogelijk offertes die goedgekeurd worden
Formule:	Aantal keren dat het data object ‘offer’ zich in de staat ‘denied’ bevindt
Meeteenheid:	Geen
Verantwoordelijke:	PM03 – Jean-Jacques Duchateau
Geïnformeerde:	PA05 (proces assistent) – Filip Aerts
Meetverantwoordelijke:	BI04 (BI medewerker) – Jef Terwingen
Actieverantwoordelijke:	OM02 (operationeel manager) – Pieter Goossens
Proces/activiteit:	Voorbeeldproces offertes
Iedere:	1
Tijd:	week
Dag vd week:	dinsdag
Ondergrens:	/
Bovengrens:	/
Restrictie:	x = 1
Actie:	notificatie
Tekst:	X
Ontvangers:	OM02
Staat voorkomen:	**Datastaat voorkomen**
Data object:	**Offer**
Datastaat:	**Denied**
Dashboard:	Dashboard offerteproces

Tabel 29: demonstratie PPI-005

Indicator	
Identificatiecode:	PPI-006
Naam:	Aantal keren dat het maken van de offerte met andere partijen tijdelijk opgeschort is
Tijdshorizon:	/
Databron:	Deze info wordt gehaald uit een file genaamd 'proces offertes' dat uit het project management systeem wordt geëxtraheerd
Strategische doelstelling:	De eigen processen optimaliseren.
Operationele doelstelling:	D03 – communicatie met externe partijen zo efficiënt mogelijk laten verlopen
Formule:	Hoe vaak de activiteit 'make offer with other parties' zich in de staat 'geschorst' bevindt
Meeteenheid:	Geen
Verantwoordelijke:	PM03 – Jean-Jacques Duchateau
Geïnformeerde:	PA05 (proces assistent) – Filip Aerts
Meetverantwoordelijke:	BI04 (BI medewerker) – Jef Terwingen
Actieverantwoordelijke:	OM02 (operationeel manager) – Pieter Goossens
Proces:	Voorbeeldproces offertes
Iedere:	1
Tijd:	maand
Dag vd maand:	25
Ondergrens:	/
Bovengrens:	/
Restrictie:	x < 1
Actie:	notificatie
Tekst:	X
Ontvangers:	OM02
Staat voorkomen:	**Uitvoeringsstaat**
Event:	**Schors event**
Proces/activiteit:	**Make offer with other parties**
Dashboard	Dashboard offerteproces

Tabel 30: demonstratie PPI-006

Scenario 4:

Ze willen de gemiddelde duur van activiteit A berekenen over verschillende instanties heen. Of men wil andere indicatoren aggregeren over meerdere instanties met behulp van functies zoals het gemiddelde, minimum, maximum etc.

Aangezien we in ons model ervoor gezorgd hebben dat een filter verplicht is bij de geaggregeerde indicator, passen we deze hier ook toe. Dus hier combineren we scenario 4 met de filter uit scenario 6.

Scenario 6:
Men wil een indicator berekenen op basis van instanties vanaf bijvoorbeeld 20 dagen geleden tot nu.

Indicator	
Identificatiecode:	PPI-012
Naam:	Som aantal keren dat het project succesvol geïnitialiseerd is
Tijdshorizon:	/
Databron:	Deze info wordt gehaald uit een file genaamd 'proces offertes' dat uit het project management systeem wordt geëxtraheerd
Strategische doelstelling:	De eigen processen optimaliseren.
Operationele doelstelling:	D02 – optimalisatie projectinitiatie
Formule:	Som (PPI-004)
Meeteenheid:	Geen
Verantwoordelijke:	PM03 – Jean-Jacques Duchateau
Geïnformeerde:	PA05 (proces assistent) – Filip Aerts
Meetverantwoordelijke:	BI04 (BI medewerker) – Jef Terwingen
Actieverantwoordelijke:	OM02 (operationeel manager) – Pieter Goossens
Proces:	Voorbeeldproces offertes
Iedere:	1
Tijd:	maand
Dag vd maand:	17
Ondergrens:	/
Bovengrens:	/
Restrictie:	/
Aggregatietype:	**Som**
Indicator:	**PPI-004**
Label:	**Filter PPI-012**
Filter:	**Input limiet**
Input limiet:	**Tijdslimiet**
Waarde:	**30**
Tijdseenheid:	**Dag**
Dashboard:	Dashboard offerteproces

Tabel 31: demonstratie PPI-012

Scenario 5:
Men wil een indicator berekenen op basis van de X aantal laatste procesinstanties.

Naast de filter die bij de bovenstaande PPI is toegepast, is het ook mogelijk om de laatste X instanties beschouwen om een indicator te definiëren.

Filter	
Label:	**Filter 2 PPI-012**
Filter:	**Input limiet**
Input limiet:	**Kwantitatieve limiet**
Aantalinstanties:	**300**

Tabel 32: demonstratie filter 2 PPI-012

Scenario 20:
Men wil een geaggregeerde indicator groeperen per eigenschap van data object G. Zo wil men dus bijvoorbeeld de gemiddelde duur van activiteit A berekenen per groep instanties waarbij data object G een andere waarde heeft voor eigenschap I.

Hierbij breiden we PPI-012 uit met een 'Data inhoud selectie'. Dit gaat op de volgende manier:

Data inhoud selectie	
Data object:	Offer
Data eigenschap:	Klant

Tabel 33: demonstratie PPI-012 – inhoud selectie

Scenario 7:
Men wil een targetbereik bepalen waarbinnen de waarde van een PPI hoort te vallen.

De indicator hieronder is tevens een toepassing van scenario 2. Maar verder wordt hier ook een targetbereik bepaald met bovengrens 40. De waarde van de indicator hoort dus kleiner of gelijk te zijn dan 40.

Dit wordt hier ook nog gekoppeld aan scenario 15, die zorgt voor een bijhorende actie.

Scenario 15:
Men wil automatische acties (bijvoorbeeld notificaties en procesabortie) koppelen wanneer er afgeweken wordt van het targetbereik.

Indicator	
Identificatiecode:	PPI-003
Naam:	Duur vanaf analyse verzoek tot ontvangst van bevestiging

Tijdshorizon:	/
Databron:	Deze info wordt gehaald uit een file genaamd 'duur offerte proces' dat uit het ERP systeem geëxtraheerd wordt
Strategische doelstelling:	De eigen processen optimaliseren.
Operationele doelstelling:	D03 – communicatie met externe partijen zo efficiënt mogelijk laten verlopen
Formule:	Tijd tussen het moment waarop 'analyse request' toegewezen is en het moment waarop het event 'receive approval/denial' getriggerd is
Meeteenheid:	Dagen
Verantwoordelijke:	PM03 – Jean-Jacques Duchateau
Geïnformeerde:	PA05 (proces assistent) – Filip Aerts
Meetverantwoordelijke:	BI04 (BI medewerker) – Jef Terwingen
Actieverantwoordelijke:	OM02 (operationeel manager) – Pieter Goossens
Proces:	Voorbeeldproces offertes
Iedere:	2
Tijd:	maand
Dag van de maand:	17
Ondergrens:	**/**
Bovengrens:	**40**
Restrictie:	/
Actie:	**notificatie**
Tekst:	**X**
Ontvangers:	**OM02**
Label:	Deel analyse tot bevestiging
Beginmarkering:	Start event 'analyse request'
Eindmarkering:	Message event 'receive approval/denial'
Dashboard:	Dashboard offerteproces

Tabel 34: demonstratie PPI-003

Scenario 8:
Men wil bepalen hoe vaak een indicator gemeten moet worden. We willen ook de exacte dag in de week of maand bepalen waarop ze berekend moet worden.

In onderstaand voorbeeld zien we dus dat PPI-007 iedere maand gemeten moet worden op de vierde dag van de maand.

Scenario 12:
Men wil tellen hoe vaak het proces X zich in de staat 'error' (of een andere staat) bevindt.

PPI-007 is een toepassing van de combinatie van scenario 8 en 12.

Indicator	
Identificatiecode:	PPI-007
Naam:	Aantal keren dat het offerte proces geannuleerd wordt
Tijdshorizon:	/
Databron:	Deze info wordt gehaald uit een file genaamd 'proces offertes' dat uit het project management systeem wordt geëxtraheerd
Strategische doelstelling:	De eigen processen optimaliseren.
Operationele doelstelling:	D39 – men wil zo weinig mogelijk annulaties van processen hebben aangezien dit verloren tijd veroorzaakt
Formule:	Hoeveel keren het proces 'voorbeeldproces offertes' zich in de staat 'geannuleerd' bevindt
Meeteenheid:	Geen
Verantwoordelijke:	PM03 – Jean-Jacques Duchateau
Geïnformeerde:	PA05 (proces assistent) – Filip Aerts
Meetverantwoordelijke:	BI04 (BI medewerker) – Chelisa Manohar
Actieverantwoordelijke:	OM02 (operationeel manager) – Pieter Goossens
Proces:	Voorbeeldproces offertes
Iedere:	**1**
Tijd:	**maand**
Dag vd maand / week / ect (optie):	**4**
Ondergrens:	/
Bovengrens:	/
Restrictie:	x < 1
Actie:	notificatie
Tekst:	X
Ontvangers:	OM02
Staat voorkomen:	Uitvoeringsstaat
Event:	Pause event
Proces/activiteit:	Voorbeeldproces offertes
Dashboard:	Dashboard offerteproces

Tabel 35: demonstratie PPI-007

Scenario 9:

Men wil de doelstelling koppelen aan de indicator zodat men ziet op welke doelstelling de PPI's gebaseerd zijn.

In ons geval hebben we ervoor gekozen om zowel een strategische als een operationele doelstelling te mogen definiëren. In de definiëring van PPI-009, vind je de definiëring van beide soorten.

Scenario 10:
Men wil kunnen aangeven welke resources verantwoordelijk zijn voor de PPI, geïnformeerd moeten worden, de PPI meten en reageren op de waarde van de PPI.

Dit is ook aangegeven in de definiëring van PPI-009. Hierbij is enkel de geïnformeerde optioneel.

Indicator	
Identificatiecode:	PPI-009
Naam:	Voldoen aan het feit dat de offerte afgewezen is
Tijdshorizon:	/
Databron:	Deze info wordt gehaald uit een file genaamd 'proces offertes' dat uit het project management systeem wordt geëxtraheerd
Strategische doelstelling:	**Omzet verhogen zodat er weer geïnvesteerd kan worden in innovatie**
Operationele doelstelling:	**D04 – zoveel mogelijk offertes die goedgekeurd worden**
Formule:	Voldoen aan de voorwaarde dat het data object 'offer' zich in de staat 'denied' bevindt
Meeteenheid:	Geen
Verantwoordelijke:	**PM03 – Jean-Jacques Duchateau**
Geïnformeerde:	**PA05 (proces assistent) – Filip Aerts**
Meetverantwoordelijke:	**BI04 (BI medewerker) – Jef Terwingen**
Actieverantwoordelijke:	**OM02 (operationeel manager) – Pieter Goossens**
Proces:	Voorbeeldproces offertes
Iedere:	1
Tijd:	maand
Dag vd maand:	17
Ondergrens:	/
Bovengrens:	/
Restrictie:	x = FALSE
Actie:	notificatie
Tekst:	X
Ontvangers:	OM02
Procesinstantie conditie:	Staat conditie

Staat conditie:	Datastaat conditie
Datastaat:	denied
Data object:	Offer
Dashboard:	Dashboard offerteproces

Tabel 36: demonstratie PPI-009

Scenario 11:
Men wil de tijd meten tussen start activiteit A en einde activiteit D. Hierbij wil men echter enkel de verwerkingstijd (of idle time of opschortingstijd) meten.

Bij PPI-003 wordt de tijd vanaf het analyse verzoek tot de ontvangst van de bevestiging gemeten. Wanneer men hier echter enkel de verwerkingstijd wil beschouwen, kan dit door de waarden van de attributen van 'Tijdsindicator' te veranderen.

Tijdsindicator	
VerwerkingsTijdInbegrepen	True (is al default, dus is in principe niet nodig)
OpschortingsTijdInbegrepen	False
IdleTimeInbegrepen	False

Tabel XX: demonstratie PPI-003 – uitbreiding scenario 11

Scenario 22:
Men wil een indicator die in de vorm van een boolean waarde weergeeft of een instantie aan een voorafbepaalde voorwaarde voldoet. Ook dit kan later geaggregeerd worden over verschillende instanties heen.

Indicator	
Identificatiecode:	PPI-008
Naam:	Voldoen aan het feit dat de projectinitiatie momenteel loopt
Tijdshorizon:	/
Databron:	Deze info wordt gehaald uit een file genaamd 'proces offertes' dat uit het project management systeem wordt geëxtraheerd
Strategische doelstelling:	Omzet verhogen zodat er weer geïnvesteerd kan worden in innovatie
Operationele doelstelling:	D12 – Het bedrijf wil op ieder moment minimum 20 projecten die zich in de initiatiefase bevinden
Formule:	Boolean die weergeeft of de activiteit 'initiate project' momenteel loopt of niet
Meeteenheid:	Geen
Verantwoordelijke:	PM03 – Jean-Jacques Duchateau

Geïnformeerde:	PA05 (proces assistent) – Filip Aerts
Meetverantwoordelijke:	BI04 (BI medewerker) – Jef Terwingen
Actieverantwoordelijke:	OM02 (operationeel manager) – Pieter Goossens
Proces:	Voorbeeldproces offertes
Iedere:	1
Tijd:	maand
Dag vd maand:	16
Ondergrens:	/
Bovengrens:	/
Restrictie:	/
Actie:	/
Tekst:	/
Ontvangers:	/
Procesinstantie conditie:	**Staat conditie**
Staat conditie:	**Uitvoeringsstaat conditie**
Event:	**Running event**
Proces/activiteit:	**Activiteit 'initiate project'**
Dashboard:	Dashboard offerteproces

Tabel 37: demonstratie PPI-008

Scenario 25:
Men wil zowel een target dat bestaat uit een bereik als een target dat bepaald is op basis van een functie kunnen combineren.

Bij de onderstaande PPI-010, wordt er een conditie indicator gedefinieerd. Hierbij zijn er enkel de waardes true of false. Dus kan er geen gebruik gemaakt worden van een bereik, maar wel van een restrictie. In sommige gevallen, kunnen beide wel gecombineerd worden, indien de waarde van de PPI een getal is.

Indicator	
Identificatiecode:	PPI-010
Naam:	Voldoen aan het feit dat de offerte een bedrag heeft groter dan 20000 euro
Tijdshorizon:	/
Databron:	Deze info wordt gehaald uit een file genaamd 'proces offertes' dat uit het project management systeem wordt geëxtraheerd
Strategische doelstelling:	Omzet verhogen zodat er weer geïnvesteerd kan worden in innovatie
Operationele doelstelling:	D27 – We willen zoveel mogelijk offertes maken

	die een bedrag groter dan 20000 euro hebben wegens de projectkosten achteraf
Formule:	Voldoen aan de voorwaarde dat het data object 'offer' een waarde in het veld 'bedrag' heeft dat groter is dan 2000
Meeteenheid:	Geen
Verantwoordelijke:	PM03 – Jean-Jacques Duchateau
Geïnformeerde:	PA05 (proces assistent) – Filip Aerts
Meetverantwoordelijke:	BI04 (BI medewerker) – Jef Terwingen
Actieverantwoordelijke:	OM02 (operationeel manager) – Pieter Goossens
Proces:	Voorbeeldproces offertes
Iedere:	1
Tijd:	maand
Dag vd maand:	23
Ondergrens:	/
Bovengrens:	/
Restrictie:	X = true
Actie:	/
Tekst:	/
Ontvangers:	/
Procesinstantie conditie:	**Data conditie**
Beschrijving te meten eigenschap:	**bedrag**
Waarde:	**> 2000**
Data object:	**offer**
Dashboard:	Dashboard offerteproces

Tabel 38: demonstratie PPI-010

Scenario 21:
Men wil een indicator die ons zegt welke waarde een specifieke eigenschap van een data object. Deze indicator kan dan ook later geaggregeerd worden over verschillende instanties heen op basis van een count.

Scenario 16:
Men wil reeds weergeven op welk dashboard de PPI zal verschijnen.

Indicator	
Identificatiecode:	PPI-011
Naam:	De naam van de klant inclusief klantencode
Tijdshorizon:	/
Databron:	Deze info wordt gehaald uit een file genaamd

	'proces offertes' dat uit het project management systeem wordt geëxtraheerd
Strategische doelstelling:	Omzet verhogen zodat er weer geïnvesteerd kan worden in innovatie (klantenbinding leidt tot meer vervolg aankopen)
Operationele doelstelling:	D17 – Kortingen geven op de volgende offertes van klanten die de vorige maand meer dan 3 keer voorkwamen
Formule:	De waarde van de eigenschap 'klant' uit het data object 'offer'
Meeteenheid:	Geen
Verantwoordelijke:	PM03 – Jean-Jacques Duchateau
Geïnformeerde:	PA05 (proces assistent) – Filip Aerts
Meetverantwoordelijke:	BI04 (BI medewerker) – Jef Terwingen
Actieverantwoordelijke:	OM27 (operationeel manager) – Marlouz Dillen
Proces:	Voorbeeldproces offertes
Iedere:	1
Tijd:	maand
Dag vd maand:	22
Ondergrens:	/
Bovengrens:	/
Restrictie:	x < 3
Actie:	notificatie
Tekst:	X
Ontvangers:	OM02
Beschrijving te meten eigenschap:	**klant**
Data object:	**Offer**
Dashboard:	**Dashboard offerteproces**

Tabel 39: demonstratie PPI-011

Scenario 13:
Men wil een indicator kunnen aggregeren op basis van een count of standaardafwijking. Count gaat vooral nuttig zijn voor de data en conditie indicator uit PPINOT.

Wanneer men bovenstaande indicator wil aggregeren, gaat dit op basis van een count. Dit wordt gedefinieerd op de onderstaande manier.

Aangezien het om een geaggregeerde indicator gaat, moet er ook een filter toegepast worden. Hierbij voegen we dus ook de filter toe uit scenario 24.

Scenario 24:
Men wil een indicator berekenen met de procesinstanties vanaf/voor een exacte datum.

Aggregatietype:	**Count**
Indicator:	**PPI-011**
Label:	**Filter PPI-011**
Filter:	**Input limiet**
Input limiet:	**Tijdslimiet**
Datum:	**7/01/2017**
Conditietype:	**na**

Tabel 40: demonstratie PPI-011 filter tijd

Scenario 17:
Men wil verder ook de relaties tussen de verschillende indicatoren kunnen weergeven omdat ze een invloed hebben op elkaars waarde.

Dit kan gedefinieerd worden door middel van 'Relatie' dat aan de twee indicatoren wordt gekoppeld die een invloed hebben op elkaar.

Relatie	
Beschrijving	Wanneer PPI-003 een grotere waarde heeft, zal de waarde van PPI-001 ook groter worden.
Richting:	positief
Assumpties:	Dit is in ieder geval zo.
Niveau van betrouwbaarheid:	8

Tabel 41: demonstratie PPI-001 en PPI-003 relatie

Scenario 18:
Wanneer men extra attributen wil toevoegen aan de 'indicator' klasse, moet dit ook achteraf mogelijk zijn.

Men kan bijvoorbeeld het attribuut 'departement' toevoegen met het datatype 'String', dat weergeeft bij welk departement een PPI hoort.

IndicatorAttribuut	
Benaming:	departement
Datatype:	String

Tabel 42: demonstratie IndicatorAttribuut

Scenario 19:
Er bevindt zich een loop in het proces waarbij activiteit A en activiteit B meerdere keren achtereenvolgens uitgevoerd kunnen worden. Men wil de gemiddelde tijd meten tussen de start van activiteit A en het einde van activiteit B.

We hebben in ons geval geen loop in het voorbeeldproces, maar wanneer dit wel het geval is en men wil de tijd meten, heb je in ons model twee keuzes. Ofwel behandel je het als een 'Cyclische tijdsindicator' (tijd berekenen per loop) of als een 'Lineaire tijdsindicator'. Bij de cyclische tijdsindicator moet men verder bepalen op welke manier de tijdsduur van de verschillende loops geaggregeerd kan worden. Dit doet men aan de hand van een attribuut 'aggregatie'.

Scenario 23:
Men wil een indicator berekenen op basis van de X laatste instanties en op basis van een data conditie. Het kan ook zijn dat men een combinatie wil maken van de overige soorten filters.

In ons geval willen we bijvoorbeeld de kwantitatieve limiet en de tijdslimiet combineren met elkaar. Hiervoor moeten we dan twee maal een filter definiëren. Wanneer we twee filters combineren, is de regel dat de filter waar we het eerst geraken geldt. Dus bijvoorbeeld een combinatie van de 200 laatste instanties en instanties vanaf vijf dagen geleden, leidt ertoe dat enkel de 150 laatste instanties beschouwd worden wanneer er in de laatste vijf dagen er zoveel zijn uitgevoerd.

Filter	
Label:	Filter (voorbeeld) deel 1
Filter:	Input limiet
Input limiet:	Kwantitatieve limiet
Aantalinstanties:	200

Tabel 43: demonstratie filter voorbeeld deel 1

Filter	
Label:	Filter (voorbeeld) deel 2
Filter:	Input limiet
Input limiet:	Tijdslimiet
Waarde:	5
Tijdseenheid:	Dag

Tabel 44: demonstratie filter voorbeeld deel 2

Scenario 14:
Men wil een indicator, bijvoorbeeld 'duur van activiteit A' berekenen op basis van de instanties waar een data object een bepaalde waarde voor een eigenschap bezit.

Dit gebeurt bij ons aan de hand van een data conditie. Deze definiëren we in de tabel hieronder. Deze filter kan gewoon toegepast worden op iedere indicator.

Filter:	
Label:	Filter voorbeeld data conditie
Filter:	Conditie
Conditie:	Data conditie

Beschrijving te meten eigenschap:	Klant
Waarde:	734 - AEX

Tabel 45: demonstratie filter voorbeeld data conditie

5. Conclusie

De onderzoeksvraag luidt als volgt: "Wat is het verschil tussen de drie modelleertalen uit MetricM, PPINOT en BAM en wat kunnen ze bijdragen aan elkaar?". Om deze vraag te beantwoorden, hebben we de drie modelleertalen uit MetricM, PPINOT en BAM vergeleken met elkaar.

Het meest opvallende verschil is het feit dat er niet bij al de metamodellen procesperformantie-indicatoren gemodelleerd worden. Bij MetricM worden er namelijk KPI's gemodelleerd, terwijl dit bij PPINOT en BAM wel PPI's zijn. In MetricM ontbreken er dus BP elementen.

Verder is er ook in PPINOT en BAM sprake van een typologie. MetricM bevat echter geen onderscheid tussen verschillende types. Aangezien PPINOT de types die voorgesteld zijn in BAM en hiernaast ook nog andere bevat, is er beslist om deze typologie te gebruiken.

Een volgend aspect dat enorm belangrijk is bij het modelleren van de PPI's zijn de BP elementen. Deze zijn veel duidelijker gemodelleerd binnen BAM dan binnen PPINOT. Om deze reden hebben we ook de methode waarop ze gemodelleerd zijn in BAM op ons eigen metamodel toegepast.

Een element dat enorm belangrijk is voor het monitoren van de PPI's, is de koppeling van acties wanneer er afgeweken wordt van een target. Dit is enkel een onderdeel van het BAM metamodel. Hierdoor kan men onmiddellijk problemen melden aan procesbeheerders of automatisch processen laten beëindigen of starten.

Op basis van deze en ook nog andere verschillen die besproken zijn in hoofdstuk 4, hebben wij ons metamodel gebouwd. Deze verschillen zijn vertaald naar vereisten voor het model. Al de verschillen die we hebben aangehaald zijn dan ook op een bepaalde manier verwerkt in het metamodel.

Dit hebben we achteraf ook nog eens geëvalueerd aan de hand van scenario's. Dit zijn scenario's die reeds gemodelleerd konden worden door de andere modellen. Deze hebben wij dan toegepast op ons eigen model om aan te tonen dat ons de model de mogelijkheid heeft om al deze scenario's te kunnen definiëren.

6. Bibliografie

[1] M. J. Lebas, "Performance measurement and performance management," *Int. J. Prod. Econ.*, vol. 41, no. 1, pp. 23–35, Oct. 1995.
[2] M. Bourne, A. Neely, J. Mills, and K. Platts, "Implementing performance measurement systems: a literature review," *Int. J. Bus. Perform. Manag.*, vol. 5, no. 1, pp. 1–24, Jan. 2003.
[3] Andy Neely, "The performance measurement revolution: why now and what next?," *Int. J. Oper. Prod. Manag.*, vol. 19, no. 2, pp. 205–228, Feb. 1999.
[4] S. Chauduri, U. Dayal, and V. Narasayya, "An overview of business intelligence technology," *Communications of the ACM*, vol. 54, no. 8, pp. 88–98, 2011.
[5] A. Neely *et al.*, "Performance measurement system design: developing and testing a process-based approach," *Int. J. Oper. Prod. Manag.*, vol. 20, no. 10, pp. 1119–1145, Oct. 2000.
[6] J. Kolář, "Business Activity Monitoring," Master's thesis, Masarykova univerzita, Fakulta informatiky, 2009.
[7] Tobias Bucher, Anke Gericke, and Stefan Sigg, "Process-centric business intelligence," *Bus. Process Manag. J.*, vol. 15, no. 3, pp. 408–429, Jun. 2009.
[8] "BPM CBOK® - ABPMP International." 2009.
[9] A. del-Rio-Ortega, M. Resinas, C. Cabanillas, and A. Ruiz-Cortes, "On the definition and design-time analysis of process performance indicators," *Inf. Syst.*, vol. 38, no. 4, pp. 470–490, Jun. 2013.
[10] M. Bourne, J. Mills, M. Wilcox, A. Neely, and K. Platts, "Designing, implementing and updating performance measurement systems," *Int. J. Oper. Prod. Manag.*, vol. 20, no. 7, pp. 754–771, 2000.
[11] D. Grigori, F. Casati, M. Castellanos, U. Dayal, M. Sayal, and M. C. Shan, "Business Process Intelligence," *Comput. Ind.*, vol. 53, no. 3, pp. 321–343, 2004.
[12] J.-P. Friedenstab, C. Janiesch, M. Matzner, and O. Muller, "Extending BPMN for Business Activity Monitoring," *Hawaii International Conference on Systems Sciences*, vol. 1, no. 45, pp. 4158–4167, 2012.
[13] W. M. P. van der Aalst, A. H. M. ter Hofstede, and M. Weske, "Business Process Management: A Survey," in *Business Process Management*, 2003, pp. 1–12.
[14] S. Strecker, U. Frank, D. Heise, and H. Kattenstroth, "MetricM: a modeling method in support of the reflective design and use of performance measurement systems," *Inf. Syst. E-Bus. Manag.*, vol. 10, no. 2, pp. 241–276, Jun. 2012.
[15] A. del-Río-Ortega, "On the definition and analysis of process performance indicators," University of Seville, 2012.
[16] P. Johannesson and E. Perjons, *An introduction to design science*. New York: Springer International Publishing, 2014.
[17] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A Design Science Research Methodology for Information Systems Research," *J. Manag. Inf. Syst.*, vol. 24, no. 3, pp. 45–77, Winter2007/2008 2007.
[18] A. R. Hevner, S. T. March, J. Park, and S. Ram, "Design Science in Information Systems Research 1," *MIS Q. Minneap.*, vol. 28, no. 1, pp. 75–105, Mar. 2004.
[19] R. S. Kaplan and D. P. Norton, "Having Trouble with Your Strategy? Then Map It," *Harv. Bus. Rev.*, vol. 78, no. 5, pp. 167–176, Oct. 2000.
[20] "About Us | Google." [Online]. Available: //www.google.com/about/. [Accessed: 17-Jul-2017].
[21] R. S. Kaplan and D. P. Norton, "Using the Balanced Scorecard as a Strategic

Management System," *Harv. Bus. Rev.*, vol. 74, no. 1, pp. 75–85, Feb. 1996.
[22] R. S. Kaplan and D. P. Norton, "Putting the Balanced Scorecard to Work," *Harv. Bus. Rev.*, vol. 71, no. 5, pp. 134–147, Oct. 1993.
[23] R. S. Kaplan and D. P. Norton, "Strategic learning & the balanced scorecard," *Strategy Leadersh. Chic.*, vol. 24, no. 5, p. 18, Oct. 1996.
[24] Ronaldo Bernardo, Simone Vasconcelos Ribeiro Galina, and Silvia Inês Dallavalle de Pádua, "The BPM lifecycle: How to incorporate a view external to the organization through dynamic capability," *Bus. Process Manag. J.*, vol. 23, no. 1, pp. 155–175, Jan. 2017.
[25] Rinaldo Macedo de Morais, Samir Kazan, Silvia Inês Dallavalle de Pádua, and André Lucirton Costa, "An analysis of BPM lifecycles: from a literature review to a framework proposal," *Bus. Process Manag. J.*, vol. 20, no. 3, pp. 412–432, May 2014.
[26] P. Kueng, "Process performance measurement system: A tool to support process-based organizations," *Total Qual. Manag. Abingdon*, vol. 11, no. 1, pp. 67–85, Jan. 2000.
[27] Julian Krumeich, Benjamin Weis, Dirk Werth, and Peter Loos, "Event-Driven Business Process Management: where are we now?: A comprehensive synthesis and analysis of literature," *Bus. Process Manag. J.*, vol. 20, no. 4, pp. 615–633, Jul. 2014.
[28] Christian Janiesch, Martin Matzner, and Oliver Müller, "Beyond process monitoring: a proof-of-concept of event-driven business activity management," *Bus. Process Manag. J.*, vol. 18, no. 4, pp. 625–643, Jul. 2012.
[29] M. Eckert and F. Bry, "Rule-based composite event queries: the language XChangeEQ and its semantics," *Knowl. Inf. Syst.*, vol. 25, no. 3, pp. 551–573, Dec. 2010.
[30] "IBM Knowledge Center." [Online]. Available: https://www.ibm.com/support/knowledgecenter/en/SS7NQD_6.2.0/com.ibm.btools.help.monitor.install.doc/intro/keyconcepts.html. [Accessed: 16-Jun-2017].
[31] U. Frank, "The MEMO meta modelling language (MML) and language architecture. 2nd Edition," ICB-Research Report, Working Paper 43, 2011.

7. Bijlagen

7.1 Bijlage 1: vergelijking BPM lifecycles

Cycle steps BPM (ABPMP)	Hallerbach *et al.* (2008)	Netjes *et al.* (2006)	Houy *et al.* (2010)	Zur Muehlen and Ho (2006)	Van der Aalst (2004a)	Verma (2009)	Weske (2007)
Planning and strategy			Development of strategy	Specification of objectives and analysis of environment		Define objectives	Administration and Stakeholders
Analysis		Design	Definition and Modeling	Design	Design	Identify process	Design and Analysis
Design and modeling	Modeling	Configuration	Implementation	Implementation	Configuration	Classify Process	Configuration
Implementation	Frequency and Selection	Execution	Execution	Monitoring	Execution	Choose process	Operation
Monitoring and control	Execution and monitoring	Control	Monitoring and control	Evaluation	Diagnosis	Define tool and implement process	Performance Evaluation
Refining	Optimization	Diagnosis	Optimization and Improvement			Monitor process	

[25]

Auteursrechtelijke overeenkomst

Ik/wij verlenen het wereldwijde auteursrecht voor de ingediende eindverhandeling:
De ontwikkeling van Business Process Intelligence Dashboards

Richting: **master in de toegepaste economische wetenschappen: handelsingenieur in de beleidsinformatica**
Jaar: **2017**

in alle mogelijke mediaformaten, - bestaande en in de toekomst te ontwikkelen - , aan de Universiteit Hasselt.

Niet tegenstaand deze toekenning van het auteursrecht aan de Universiteit Hasselt behoud ik als auteur het recht om de eindverhandeling, - in zijn geheel of gedeeltelijk -, vrij te reproduceren, (her)publiceren of distribueren zonder de toelating te moeten verkrijgen van de Universiteit Hasselt.

Ik bevestig dat de eindverhandeling mijn origineel werk is, en dat ik het recht heb om de rechten te verlenen die in deze overeenkomst worden beschreven. Ik verklaar tevens dat de eindverhandeling, naar mijn weten, het auteursrecht van anderen niet overtreedt.

Ik verklaar tevens dat ik voor het materiaal in de eindverhandeling dat beschermd wordt door het auteursrecht, de nodige toelatingen heb verkregen zodat ik deze ook aan de Universiteit Hasselt kan overdragen en dat dit duidelijk in de tekst en inhoud van de eindverhandeling werd genotificeerd.

Universiteit Hasselt zal mij als auteur(s) van de eindverhandeling identificeren en zal geen wijzigingen aanbrengen aan de eindverhandeling, uitgezonderd deze toegelaten door deze overeenkomst.

Voor akkoord,

Breuls, Samina

Datum: **16/08/2017**