PATRONEN VOOR WERKSTROOMBESTURING

Wil van der Aalst

Technische Universiteit Eindhoven, Postbus 513, 5600 MB Eindhoven,
w.m.p.v.d.aalst@tm.tue.nl

In objectgeoriënteerde methoden en ontwikkelomgevingen worden patronen (patterns) gebruikt om veelvuldig voorkomende constructies te onderkennen en te ondersteunen. Ook in werkstroombesturing zijn patronen te onderkennen. De meeste workflow management systemen ondersteunen eenvoudige patronen zoals de sequentiële, parallelle, conditionele en iteratieve routering. Naast deze basispatronen zijn er echter vele andere, veelvuldig voorkomende, patronen die slechts door een deel van de beschikbare systemen ondersteund worden. Aan de hand van deze patronen maken we duidelijk dat er grote verschillen tussen workflow management systemen zijn.

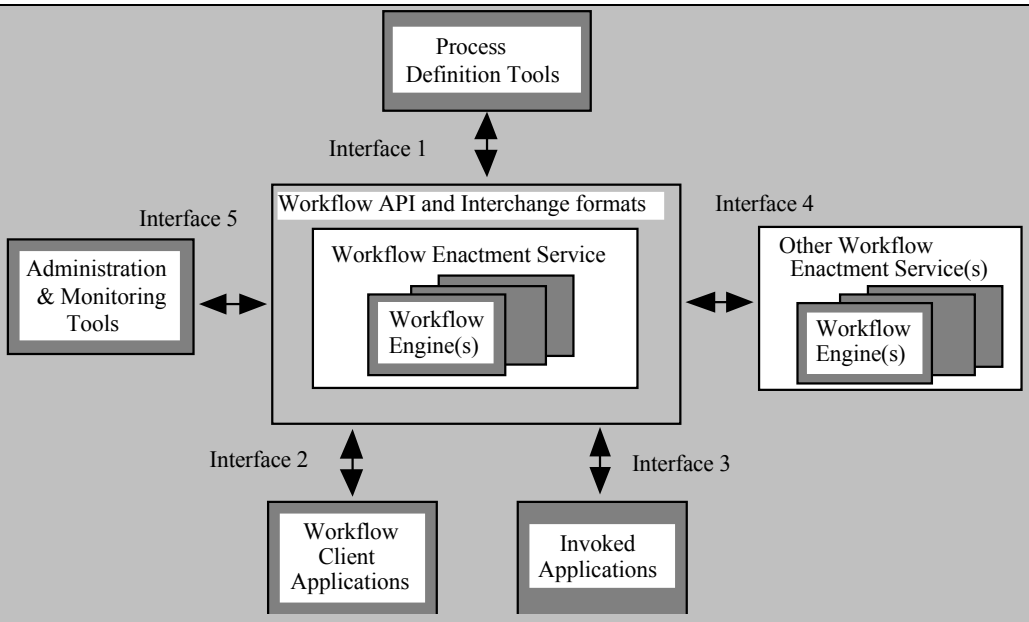
In het afgelopen decennium zijn er vele workflow management systemen beschikbaar gekomen. Deze systemen worden op dit moment vooral gebruikt door grote administratieve organisaties zoals banken, verzekeringsmaatschappijen, uitvoeringsinstanties en overheden. Ook al hebben workflow management systemen nog een bescheiden verspreidingsgraad, het is in de afgelopen jaren duidelijk geworden dat workflow technologie een integraal onderdeel zal zijn van de informatiesystemen van morgen. In ERP-systemen zoals SAP, Baan, Peoplesoft en JD Edwards, maar ook software voor bijvoorbeeld E-commerce en call centers, vinden we workflow componenten terug. Ondanks het belang van workflow management wordt er meestal onzorgvuldig omgesprongen met de vereiste workflow functionaliteit. Dit komt bijvoorbeeld naar voren in selectietrajecten voor een workflow management systeem. In veel gevallen wordt vooral naar de *leverancier* van de software en de *technische eisen* gekeken en minder naar de *functionaliteit* van het gewenste systeem. Het gevolg is dat zeer veel selectietrajecten resulteren in de aanschaf van een workflow management systeem dat in de praktijk niet blijkt te voldoen. Dit is een gevolg van de onbekendheid van veel organisaties met de vrij geavanceerde en complexe workflow technologie. Hierdoor is men niet in staat duidelijke eisen te formuleren ten aanzien van de gewenste functionaliteit. Deze situatie wordt in stand gehouden door de adviesbureaus die bedrijven helpen bij het selecteren van een workflow management systeem. Met enige regelmaat hebben prestigieuze adviesbureaus in de afgelopen 5 jaar rapporten de wereld in gelopen waarin workflow management systemen vergeleken worden aan de hand van een “check list”. Ten aanzien van de workflow routeringsfunctionaliteit wordt in deze rapporten slechts aangegeven of sequentiële, parallelle, conditionele en iteratieve routering mogelijk zijn en of bouwstenen zoals de AND-split, AND-join, OR-split en OR-join aanwezig zijn. Aangezien deze routeringsmogelijkheden en bouwstenen in elk workflow management systeem voorkomen, onderscheiden de systemen zich onvoldoende van elkaar in deze rapporten. Deze rapporten suggereren dan ook dat alle workflow management systemen op het gebied van routering hetzelfde zijn. Niets is minder waar! Er is een wereld van verschil tussen de vele beschikbare systemen. Het is ook erg naïef om te veronderstellen dat complexe bedrijfsprocessen uitgedrukt kunnen worden in termen van sequentiële, parallelle, conditionele en iteratieve routering. Desondanks worden aan de

hand van deze zeer ruwe criteria systemen geselecteerd. Het gevolg is dat workflow technologie vaak als een knellend keurslijf wordt ervaren en veel workflow projecten in de pilot fase blijven steken. Dit is de reden dat ik samen met Arthur ter Hofstede (Queensland University of Technology), Bartek Kiepuszewski (Mincom Pty Ltd) en Allistair Barros (University of Queensland) onderzoek ben gaan doen naar "workflow patterns". Dit onderzoek heeft geresulteerd in een vrij volledige repository met een dertigtal patronen en een diepgaande vergelijking van twaalf workflow management systemen (Staffware, IBM MQ Series Workflow, COSA, Visual WorkFlo, Forté Conductor, Meteor, Mobile, Verve, InConcert, I-Flow, SAP R/3 Workflow en HP Changengine). We beschouwen dit onderzoek als het academische antwoord op de eerder genoemde oppervlakkige rapporten van prestigieuze adviesbureaus.

Workflow management systemen

Dankzij workflow management systemen [3,9] is het mogelijk om de procesbesturing uit de toepassingssoftware (applicaties) te 'duwen'. In zekere zin kunnen we een workflow management systeem vergelijken met een database management systeem. Dankzij database management systemen is het immers mogelijk geworden om het beheer van de gegevens uit de toepassingssoftware te duwen. Beide soorten systemen ondersteunen een stukje generieke functionaliteit. Doordat workflow management systemen, in tegenstelling tot database management systemen, pas sinds midden jaren 90 beschikbaar zijn, is het op veel punten nog onduidelijk wat tot de basisfunctionaliteit van een workflow management systeem behoort. Het betreft een nog jonge technologie die nog niet geheel uitgekristalliseerd is. Daarnaast heeft workflow management ook vele gezichten. Workflow management systemen worden ingezet voor flexibilisering, systeemintegratie, procesoptimalisatie, organisatieverandering, verbeteren van de onderhoudbaarheid, evolutionair ontwikkelen, enzovoorts. Dit alles maakt dat er gemakkelijk verwarring kan ontstaan rond de functionaliteit die men van een workflow management systeem mag verwachten. Dit gevaar werd in vroeg stadium onderkend door de *Workflow Management Coalition* (WfMC). De WfMC is een organisatie die zich onder andere bezig houdt met het standaardiseren van de terminologie rond workflow management [10]. Daarnaast is de WfMC bezig met definiëren van standaarden voor de uitwisseling van gegevens tussen workflow management systemen en toepassingen. In 1996 telde de WfMC al 200 leden (waaronder vele leveranciers van workflow management producten).

Een van de uitgangspunten van de WfMC is het zogenaamde *workflow referentiemodel*. Dit referentiemodel is een globale beschrijving van de architectuur van een workflow management systeem waarbij de voornaamste componenten en de bijbehorende interfaces opgesomd worden. Figuur 1 laat een afbeelding van het workflow referentiemodel zien.



Figuur 1: Het referentiemodel van de Workflow Management Coalition (© WfMC).

Het referentiemodel laat zien dat het hart van een workflow systeem gevormd wordt door de zogenaamde *Workflow Enactment Service*. Dit deel van het workflow systeem pompt als het ware de casussen door de organisatie. De Enactment Service zorgt ervoor dat de juiste activiteiten, in de juiste volgorde en door de juiste medewerkers uitgevoerd worden. Om dit te bepalen wordt gebruikt gemaakt van procesdefinities en resource classificaties die met de zogenaamde *Process Definition Tools* gemaakt zijn. Naast het in kaart brengen van het proces en de organisatie bieden deze tools vaak ook analysemogelijkheden zoals simulatie. Via de *Workflow Client Applications* worden taakopdrachten aangeboden aan de medewerkers. Door een taakopdracht te selecteren kan een medewerker beginnen met het uitvoeren van een specifieke taak voor een specifieke casus. Tijdens het uitvoeren van een taak kan het nodig zijn om een toepassing op te starten. Het geheel van toepassingssoftware die vanuit het workflow systeem opgestart kan worden, wordt in het referentiemodel ook wel de *Invoked Applications* genoemd. Het volgen van de werkstroom, het beheren van casussen en het aansturen van medewerkers wordt ondersteund door de zogenaamde *Administration and Monitoring Tools*.

Wat is een patroon?

Sinds 1995 staan ontwerppatronen (“design patterns”) volop in de belangstelling [8]. Een ontwerppatroon is een niet-triviale abstractie die veelvuldig voorkomt in verschillende situaties. Tot dusver hebben patronen zich vooral toegespitst op objectgeoriënteerde talen. Het doel van patronen is het hergebruik van nuttige en veelvuldig voorkomende constructies. Door te abstraheren van implementatiedetails en context zijn de patronen makkelijk te hergebruiken in zeer uiteenlopende situaties. Op dit moment bestaan er grote verzamelingen van ontwerppatronen voor software ontwikkeling. Voor zover we na kunnen gaan bestond er tot voor kort nog geen gesystematiseerd overzicht van constructies voor werkstroombesturing. Dit is de reden geweest om te beginnen aan het realiseren van een repository bestaande uit uitsluitend werkstroompatronen (“workflow patterns”). Op dit moment beschikken we over een dertigtal patronen. Het doel van elk van de patronen is drieledig. In de eerste plaats, dient de vrij complete verzameling werkstroompatronen een didactisch doel. Een patroon komt overeen met een veelvuldig gewenst stukje routering. Daarom is het belangrijk ontwerpers en

gebruikers van werkstromen bewust te maken van deze patronen. In de tweede plaats vormen de patronen een basis voor het selecteren van een workflow management systeem. Door aan te geven welke patronen relevant zijn voor een bepaalde bedrijfssituatie, kan aan de hand van deze patronen bepaald worden welke systemen voldoen. In de derde plaats bieden de patronen aanknopingspunten voor het indirect realiseren van werkstroomfunctionaliteit die niet rechtstreeks door het workflow management systeem wordt ondersteund.

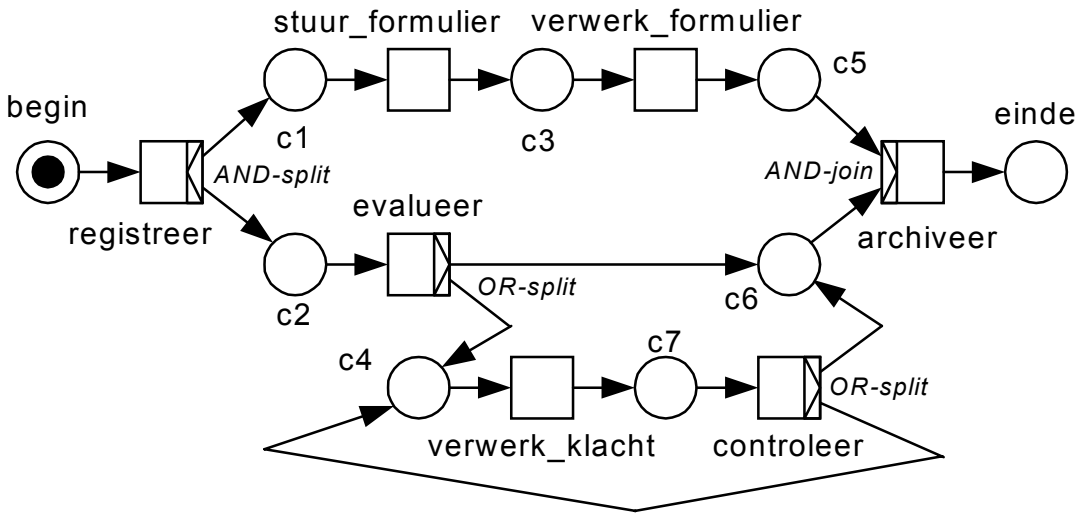
Een patroon bestaat uit de volgende delen:

1. De *naam* van het werkstroompatroon.
2. Een *beschrijving* van de gewenste functionaliteit.
3. De gangbare *synoniemen*.
4. Een aantal *voorbeelden* waarin het patroon ingezet kan worden.
5. De reden waarom sommige systemen moeite hebben met het aanbieden van de gewenste functionaliteit (het *probleem*).
6. De *oplossingen* die gebruikt kunnen worden om binnen de context van een workflow systeem of klasse van workflow systemen de gewenste functionaliteit te realiseren.

Het voert te ver om alle patronen hier in veel detail te behandelen. Voor een overzicht van de patronen verwijzen we naar [4,5] en http://www.tm.tue.nl/it/research/patterns/. Voordat we twee patronen nader toelichten, introduceren we een voorbeeldmodel. Dit model wordt gebruik om de twee patronen te verduidelijken.

Workflow model

In een bedrijfsproces moeten er voor elke *casus* (bijvoorbeeld een bestelling of een verzekeringsclaim) *taken* (bijvoorbeeld ‘controleer polisgegevens’) uitgevoerd worden. Met behulp van een procesmodelleringstechniek kan worden aangegeven welke taken er uitgevoerd moeten worden en in welke volgorde. We spreken in dit verband ook wel over *routering*. Door de WfMC worden de volgende routeringsvormen onderkend: sequentiële, parallelle, conditionele en iteratieve routering. De meeste WFM-systemen ondersteunen deze vier routeringsvormen. Om deze routeringsvormen toe te lichten gebruiken we het voorbeeld in Figuur 2. Dit voorbeeld zullen we later gebruiken om een aantal meer geavanceerde patronen toe te lichten.



Figuur 2: Een voorbeeld van een werkstroom waarin sequentiële, parallelle, conditionele en iteratieve routering voorkomen.

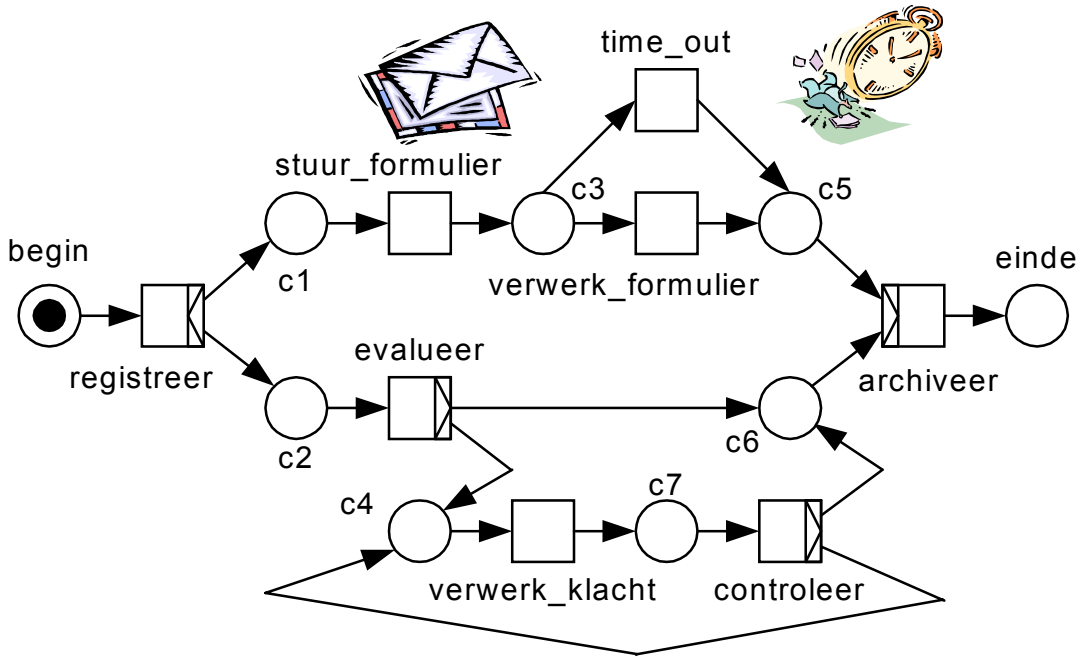
In figuur 2 wordt een schematechniek op basis van Petri-netten gebruikt [1,3,11]. Sinds de zeventiger jaren worden Petri-netten gebruikt voor het in kaart brengen van werkstromen [6,7]. Het verschil tussen figuur 2 en een klassiek Petri-net is de aanwezigheid van gegevens op basis waarvan keuzes gemaakt kunnen worden. Ook wordt van taken met meerdere in- of uitgangen aangegeven of het om een AND/OR-split/join gaat. Het fictieve proces modelleert het afhandelen van klachten. Elke casus start met een token in conditie *begin*. Een token wordt weergegeven door de zwarte stip en een conditie, ook wel plaats genoemd, wordt weergegeven door een cirkel. In Figuur 2 is één token afgebeeld. Dit token stelt een net geïnitieerde klacht voor. Deze klacht wordt eerst geregistreerd. Taken, in het klassieke Petri-net ook wel transities genoemd, worden weergeven door een vierkant. Taak *registreer* is een AND-split. Door de uitvoering van deze taak wordt het token in *begin* verwijderd en wordt er zowel een token in *c1* als *c2* gelegd. Na het uitvoeren van de taak *registreer* wordt er in parallel gewerkt aan het versturen en verwerken van een klachtenformulier en het evalueren en verwerken van de klacht. Het token dat in *c1* is neergelegd gaat na uitvoering van de taken *stuur_formulier* en *verwerk_formulier* via conditie *c3* naar conditie *c5*. Het token dat in *c2* is neergelegd heeft een meer complexe levenscyclus. Uiteindelijk komt het token in *c6* te liggen. In de taken *evalueer* en *controleer* wordt steeds een expliciete keuze gemaakt: óf er wordt een token in *c6* gelegd óf er wordt een token in *c4* gelegd waarna de klacht (nog een keer) verwerkt wordt. Op het moment dat er zowel in *c5* als *c6* een token ligt kan de taak *archiveer* uitgevoerd worden. Deze taak is een zogenaamde AND-join en synchroniseert de beide parallelle stromen. Na het uitvoeren van deze taak komt er een token in conditie *einde* te liggen.

Twee voorbeeld patronen

Van het dertigtal werkstroompatronen die we in de loop der jaren hebben verzameld wil ik er twee toelichten. Voor beide patronen speelt het *toestandsbegrip* een grote rol.

In figuur 2 wordt taak *stuur_formulier* altijd gevolgd door *verwerk_formulier*. Dit verondersteld dat elk klachtenformulier uiteindelijk door de klant wordt geretourneerd. Het is immers pas mogelijk om taak *verwerk_formulier* uit te voeren nadat het formulier door de klagende partij is teruggestuurd. Omdat in menig geval het formulier niet of veel te laat wordt teruggestuurd, vindt er na twee weken een zogenaamde *time_out* plaats. Indien het formulier

niet binnen twee weken is ontvangen, wordt taak *time_out* uitgevoerd in plaats van taak *verwerk_formulier*. Figuur 3 laat het aangepaste proces zien. Met een token in conditie *c3* zijn zowel *time_out* als *verwerk_formulier* nog mogelijk. Desondanks wordt slechts één van deze twee taken uitgevoerd.

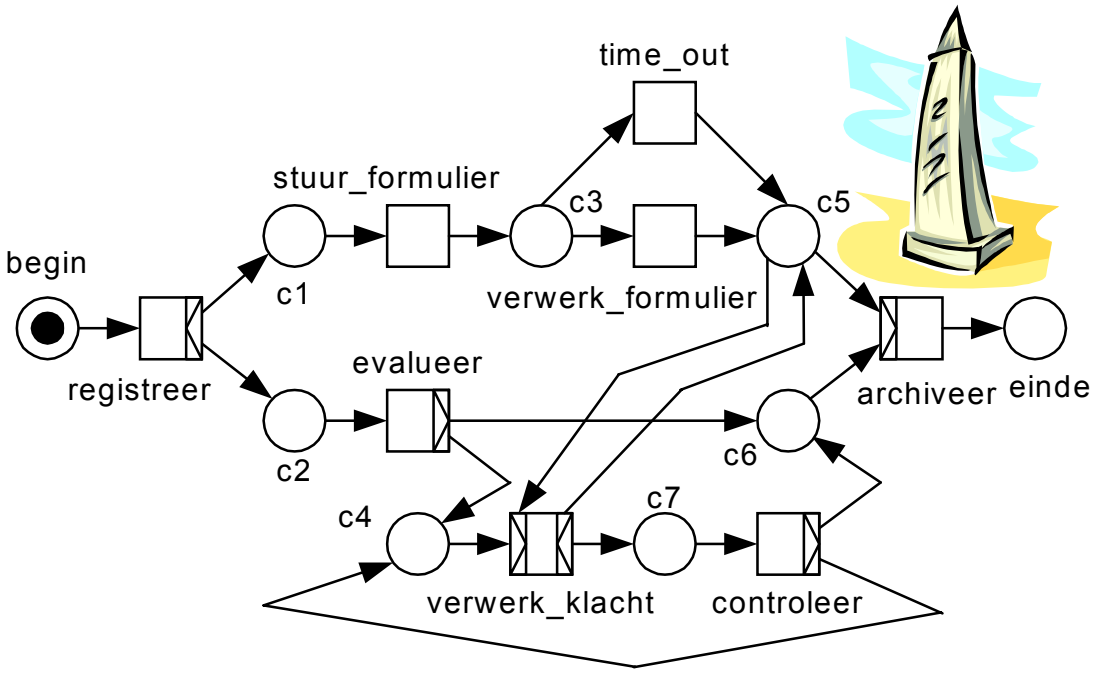


Figuur 3: Voorbeeld van het "Deferred choice pattern": de impliciete OR-split tussen verwerk_formulier en time_out.

De impliciete keuze tussen twee of meer alternatieve taken is een voorbeeld van een patroon waarbij het toestandbegrip een grote rol speelt. Dit patroon word de "Uitgestelde keuze" ("Deferred choice") genoemd. De keuze wordt niet gemaakt op het moment dat de taak *stuur_formulier* wordt uitgevoerd maar om het moment dat het formulier binnenkomt of op het moment dat de twee weken verstreken zijn. Dit is duidelijk een andere soort keuze dan de keuze die gemaakt wordt in *evalueer* of *controleer*. Direct na het uitvoeren van *evalueer* of *controleer* is reeds vastgelegd of *verwerk_klacht* of *archiveer* de volgende processtap is. Het verschil tussen de uitgestelde keuze en de normale OR-split lijkt subtiel maar is van cruciaal belang. Indien in figuur 3 de uitgestelde keuze door een normale OR-split wordt vervangen, moet direct na uitvoering van *stuur_formulier* reeds vastgelegd worden of *verwerk_formulier* of *time_out* uitgevoerd gaat worden. Dit vereist dat de medewerker die de taak *stuur_formulier* uitvoert vooraf kan bepalen of de klagende partij het formulier terug gaat sturen. Dit is natuurlijk niet het geval! Omdat het schema in figuur 3 expliciet toestanden onderkent, is het vrij eenvoudig een uitgestelde keuze te realiseren. Indien, zoals in de meeste workflow management systemen, geabstraheerd wordt van toestanden wordt, is het patroon echter erg lastig te realiseren. Systemen zoals Staffware, IBM MQ Series Workflow, Visual WorkFlo, Forté Conductor, Verve, InConcert, I-Flow, SAP R/3 Workflow en HP Changengine kunnen slechts op een ad-hoc of zeer indirecte wijze een uitgestelde keuze bewerkstelligen. COSA is een van de weinige systemen dat dit patroon direct ondersteund. In [4,5] geven we aan hoe in andere systemen de gewenste functionaliteit met enig kunst-en-vliegwerk kan worden benaderd.

In figuur 3 kan de klacht verwerkt worden voordat het formulier ontvangen is of de twee weken verstreken zijn. In figuur 4 is door middel van het "Mijlpaal patroon" ("Milestone

pattern") afgedwongen dat de *verwerk_klacht* pas uitgevoerd kan worden nadat óf *verwerk_formulier* óf *time_out* heeft plaatsgevonden. Taak *verwerk_klacht* is een AND-join en kan dus pas plaatsvinden nadat er een token in *c4* en *c5* ligt. Na uitvoering van *verwerk_klacht* wordt er een token in *c5* teruggelegd en wordt via *c7* de taak *controleer* klaargezet. Het mijlpaal patroon komt in de praktijk veelvuldig voor. Steeds als er in een parallel proces in de ene tak een taak moet wachten tot er in de andere tak een bepaalde toestand is bereikt, is het patroon nodig. Ook hier is de oplossing weer eenvoudig zolang toestanden expliciet onderkend worden. Indien gebruik gemaakt wordt van een workflow management systeem dat abstraheert van toestand *c5* is het onmogelijk de gewenste routering direct te realiseren.



Figuur 4: Voorbeeld van het "Milestone pattern": verwerk_klacht mag alleen uitgevoerd worden als er een token c5 ligt.

Een volledige beschrijving van een werkstroompatroon bestaat uit de *naam,* een *beschrijving,* gangbare *synoniemen*, een aantal *voorbeelden*, een *probleembeschrijving* en mogelijke *oplossingen.* Aan de hand van figuren 3 en 4 hebben we van twee patronen de naam, een korte beschrijving en een voorbeeld gegeven. Voor de overige onderdelen van het patroon (synoniemen, meer voorbeelden, probleembeschrijving en oplossingen) verwijzen we naar [4,5] en http://www.tm.tue.nl/it/research/patterns/.

De twee patronen laten zien dat er goede redenen zijn voor het expliciet modelleren van toestanden. In de eerste plaats hebben toestanden meestal een duidelijke betekenis die helder en zinvol is voor de personen die participeren in het proces. Een bestelling kan bijvoorbeeld in de toestand 'geaccepteerd' zijn en bepaalde taken mogen alleen uitgevoerd worden indien de bestelling in deze toestand is. Met name in omgevingen waar men te maken heeft met stringente regelgeving is het onderkennen van toestanden zinvol. Binnen veel overheidsinstellingen (justitie, belastingdienst, douane) heeft men bijvoorbeeld te maken met wettelijke toestanden. Ook vanuit een logistiek standpunt is het onderkennen van toestanden van belang. Als we kijken naar de doorlooptijd van een casus, dan is deze vaak vele malen groter dan de tijd die feitelijk besteed wordt aan de casus. De afhandelen van een

belastingaangifte kost bijvoorbeeld weken terwijl de feitelijke bedieningsduur misschien maar enige minuten is. Dit betekent dat een casus meestal in rusttoestand is. Het is dus merkwaardig om juist dit aspect niet expliciet te modelleren. Ook zijn bepaalde routeringsvormen niet of moeilijk te modelleren indien men abstraheert van toestanden. Dit twee besproken patronen illustreren dit.

Overige patronen

De twee toestandgeoriënteerde patronen zijn slechts twee voorbeelden uit een verzameling van ongeveer 30 patronen [4,5]. Een groot deel van deze patronen zijn te vinden op een special daarvoor ingerichte website: http://www.tm.tue.nl/it/research/patterns/. Figuur 5 laat één van de pagina's op deze website zien. De afgebeelde pagina beschrijft de uitgestelde keuze ("Deferred choice pattern").

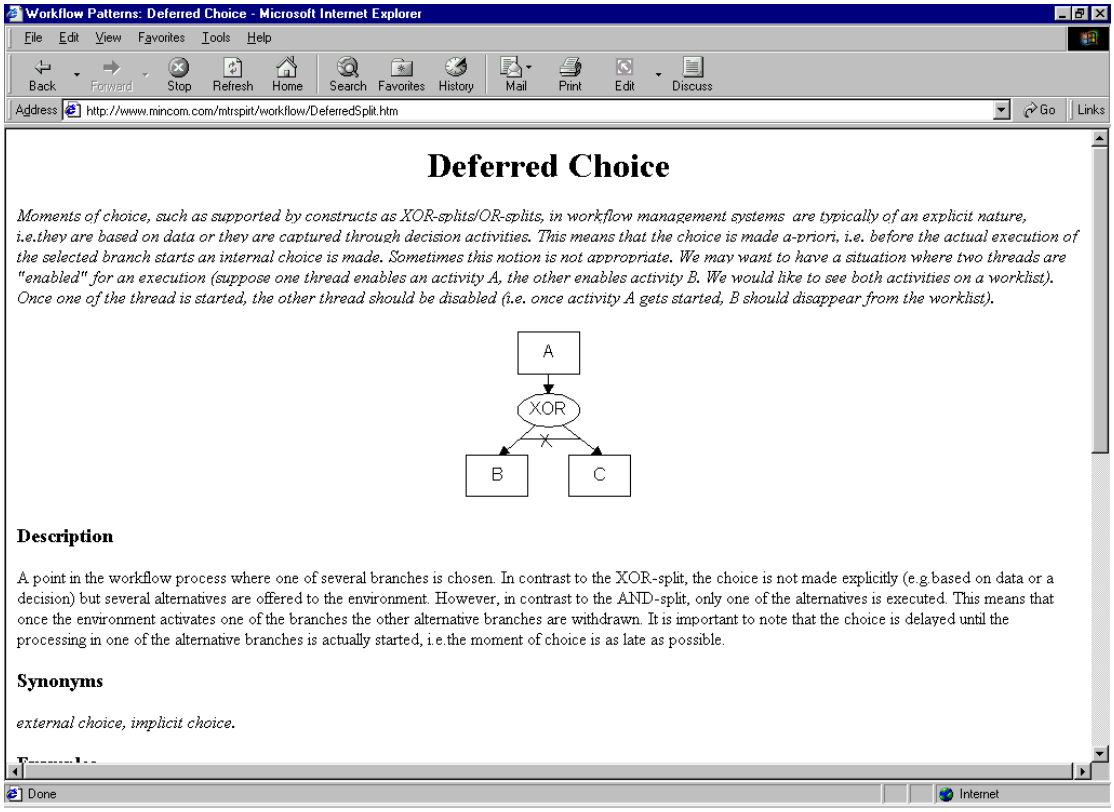


Figure 5: Een van de vele patronen ("Deferred choice pattern") op http://www.tm.tue.nl/it/research/patterns/

De werkstroompatronen vallen uiteen in acht categorieën:

- *Basispatronen*. Onder deze patronen vallen de standaard constructies voor het realiseren van sequentiële, parallelle, conditionele en iteratieve routering. Naast de standaard OR-join wordt hier ook de zogenaamde "Simple merge" en "Multi choice" onderkend.
- *Geavanceerde synchronisatiepatronen*. Naast de standaard synchronisatie via de AND-join zijn vele andere vormen van synchronisatie denkbaar. Een voorbeeld is het zogenaamde "N-out-of-M join" patroon. Dit patroon start M parallelle taken of subprocessesen op en gaat verder zodra er N van deze parallelle takken zijn afgerond. Denk bijvoorbeeld aan een document dat door drie mensen moet worden getekend maar dat verder verwerkt mag worden zodra de tweede handtekening binnen is. Andere

geavanceerde synchronisatiepatronen zijn de “Synchronizing merge”, de “Multi merge” en de “Discriminator”.

- *Structuurpatronen*. Veel workflow management systemen kennen beperkingen ten aanzien van de structuur van het workflow proces. Veel systemen staan bijvoorbeeld geen cyclische verbindingen toe en beperken iteratie tot blokvormachtige constructies. Andere systemen eisen een unieke eindtaak. De structuurpatronen laten zien hoe met deze beperkingen omgegaan kan worden.
- *Patronen voor meerdere instanties van een taak of deelproces*. In veel situaties is het nodig dat een taak of deelproces meerdere keren uitgevoerd moet worden. Denk bijvoorbeeld aan een bestelling waarbij binnen de bestelling taken per bestelregel uitgevoerd moeten worden. Het aantal instanties kan vast of variabel zijn en de kennis over deze instanties kan vooraf of pas tijdens uitvoering bekend zijn. Ook kan het zijn dat er wel of niet gesynchroniseerd moet worden. In totaal onderkennen we een viertal patronen om met meerdere instanties van een taak of deelproces om te gaan.
- *Temporele patronen*. Meestal mag de uitvoering van een vervolgtaak pas starten nadat de voorgaande taak volledig is afgerond. Het is echter ook mogelijk om deze eisen wat te verzwakken. De vervolgtaak mag bijvoorbeeld al starten terwijl de voorgaande taak nog uitgevoerd wordt. Een ander voorbeeld is een taak die meteen opgestart mag worden maar pas na afronding van een voorgaande taak afgesloten mag worden.
- *Toestandgebaseerde patronen*. De “Uitgestelde keuze” (“Deferred choice”) en de “Mijlpaal” (“Milestone”) zijn voorbeelden van toestandgebaseerde patronen. Een ander patroon in deze categorie is de “Interleaved parallel routing” waar door mutual exclusion wordt afgedwongen dat taken in verschillende parallelle takken niet tegelijk uitgevoerd worden.
- *Intrekpatronen*. Sommige systemen bieden de mogelijkheid een activiteit (d.w.z. de uitvoering van een taak voor een specifieke casus) of zelfs een gehele casus in te trekken.
- *Interworkflow patronen.* De laatste categorie van patronen bestaat uit constructies voor het op elkaar afstemmen van casussen in verschillende werkstromen. Veel workflow management systemen bieden de mogelijkheid om via het uitwisselen van triggers casussen op elkaar af te stemmen. Slechts een beperkt aantal systemen bieden meer gestructureerde interworkflow patronen.

Evaluatie van 12 systemen

Aan de hand van de werkstroompatronen zijn de volgende 12 systemen geëvalueerd: COSA (Thiel Logistik AG/Ley GmbH/COSA Solutions BV), HP Changengine (Hewlett-Packard), Forté Conductor (Forte/SUN), I-Flow (Fujitsu), InConcert (TIBCO), Meteor (University of Georgia), Mobile (University of Erlangen), MQ Series Workflow (IBM), R/3 Workflow (SAP AG), Staffware (Staffware PLC), Verve (Verve Inc.) en Visual WorkFlo (FileNET). Tabel 1 laat op hoofdlijnen de resultaten van de evaluatie zien. In de tabel komen de kolommen overeen met de twee beschreven patronen en een zevental categorieën. Per patroon/categorie wordt aangegeven of er sprake is van directe ondersteuning (+), indirecte of partiële ondersteuning (+/-) of geen directe ondersteuning (-). Alle systemen ondersteunen de basispatronen. De uitgestelde keuze en mijlpaal worden alleen direct ondersteund door COSA. Duidelijk is dat geen enkele systeem directe ondersteuning levert voor alle relevante werkstroompatronen!

Tabel 1: Een evaluatie van 12 systemen aan de hand van patronen.

	Basispatronen	**Geavanceerde synchronisatiepatronen**	**Structuurpatronen**	**Patronen voor meerdere instanties**	*Uitgestelde keuze*	*Mijlpaal*	**Toestandsgebaseerde patronen**	**Intrekpatronen**	**Interworkflow patronen**
COSA	+	-	+/-	-	+	+	+	+/-	+/-
HP Changengine	+	+/-	+/-	-	-	-	-	+/-	-
Forté Conductor	+	+	+/-	+	-	-	-	+/-	-
I-Flow	+	-	+/-	+	-	-	-	+/-	-
InConcert	+	+/-	+/-	-	-	-	-	-	-
Meteor	+	+/-	+/-	-	-	-	-	-	+/-
Mobile	+	+/-	+/-	-	+/-	+/-	+/-	-	-
MQ Series Workflow	+	+/-	+/-	+/-	-	-	-	-	-
R/3 Workflow	+	+/-	-	-	-	-	-	-	+/-
Staffware	+	-	+	-	-	-	-	+/-	-
Verve	+	+/-	+	+	-	-	-	+/-	+/-
Visual WorkFlo	+	-	-	+	-	-	-	-	+

Conclusie

De werkstroompatronen kunnen op een aantal manieren ingezet worden. In de eerste plaats kunnen de patronen gebruikt worden om werkstroomontwerpers te trainen in veel voorkomende niet-triviale constructies. Het trainingsmateriaal dat nu door de softwareleveranciers gebruikt wordt gaat meestal niet verder dan de eerder genoemde basispatronen (sequentiële, parallelle, conditionele en iteratieve routering). In de tweede plaats kunnen de patronen gebruikt worden bij het selecteren van een workflow management systeem. In [5] is per werkstroompatroon aangegeven welke workflow management systemen het patroon ondersteunen. Op deze wijze kan een snelle selectie op functionele gronden gemaakt worden. Zoals in de inleiding is aangegeven worden in selectietrajecten deze functionele kenmerken niet of nauwelijks meegenomen. Het gevolg is dat veel workflow projecten in de pilot fase blijven steken. In de derde plaats kunnen de patronen gebruikt worden om de gewenste functionaliteit via een omweg te realiseren. Per patroon is

aangegeven of er alternatieve manieren zijn om via indirecte wijze de gewenste functionaliteit te benaderen.

Voor meer informatie over de patronen verwijzen we naar [4] en http://www.tm.tue.nl/it/research/patterns/. Voor de vergelijking van twaalf workflow management systemen op basis van deze patronen verwijzen we naar het rapport "Workflow patterns" [5]. In dit rapport worden de 12 eerder genoemde systemen vergeleken.

Over de auteur

Prof.dr.ir. W.M.P. van der Aalst is hoogleraar binnen zowel de faculteit Technologie Management als de faculteit Wiskunde en Informatica van de Technische Universiteit Eindhoven. Hij is voorzitter van de capaciteitsgroep Informatie en Technologie en zijn onderzoeksinteresses gaan uit naar workflow management, procesmodellen en informatiesystemen.

Referenties

[1] W.M.P. van der Aalst. The Application of Petri Nets to Workflow Management. *The Journal of Circuits, Systems and Computers*, 8(1):21-66, 1998.

[2] W.M.P. van der Aalst en T. Basten. Inheritance of Workflows: An approach to tackling problems related to change. *Theoretical Computer Science*, 2001 (to appear).

[3] W.M.P. van der Aalst en K.M. van Hee. *Workflow Management: Modellen, Methoden en Systemen*. Academic Service, Schoonhoven, 1997.

[4] W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, en A.P. Barros. Advanced Workflow Patterns. In O. Etzion en P. Scheuermann, editors, *7th International Conference on Cooperative Information Systems (CoopIS 2000)*, volume 1901 of *Lecture Notes in Computer Science*, pages 18-29. Springer-Verlag, Berlin, 2000.

[5] W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, en A.P. Barros. Workflow Patterns. BETA Working Paper Series, WP 47, Eindhoven University of Technology, Eindhoven, 2000. http://tmitwww.tm.tue.nl/staff/wvdaalst/Publications/publications.html

[6] C.A. Ellis. Information Control Nets: A Mathematical Model of Office Information Flow. In *Proceedings of the Conference on Simulation, Measurement and Modeling of Computer Systems*, pages 225-240, Boulder, Colorado, 1979. ACM Press.

[7] C.A. Ellis en G.J. Nutt. Modelling and Enactment of Workflow Systems. In M. Ajmone Marsan, editor, *Application and Theory of Petri Nets 1993*, volume 691 of Lecture Notes in Computer Science, pages 1-16. Springer Verlag, Berlin, 1993.

[8] E. Gamma, R. Helm, R. Johnson, en J. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, Reading, Massachusetts, 1995.

[9] S. Jablonski en C. Bussler. *Workflow Management: Modeling Concepts, Architecture, and Implementation*. International Thomson Computer Press, 1996.

[10] P. Lawrence, editor. *Workflow Handbook 1997*, Workflow Management Coalition. John Wiley and Sons, New York, 1997.

[11] W. Reisig en G. Rozenberg, editors. *Lectures on Petri Nets I: Basic Models*, volume 1491 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, 1998.