Anko Duizer is werkzaam als trainer/coach bij Class-A

State Machine Workflow interessant voor softwareontwikkeling

HET ANDERE WORKFLOWMODEL ONDER DE LOEP GENOMEN

Naast het sequential model is het state machine workflowmodel ook onderdeel van Windows Workflow Foundation. Dit laatste model is een flexibel workflowmodel waarbij niet het proces maar de gebruiker in controle is. Het state machine workflowmodel biedt interessante mogelijkheden voor softwareontwikkeling.

Windows Workflow Foundation levert twee workflowmodellen: sequential workflow en state machine workflow. Er is veel geschreven over het sequential model. In dit artikel besteed ik aandacht aan het andere model, het state machine model. Het state machine model is minstens zo interessant als het sequential model. Ik ga ervan uit dat de lezer enigszins bekend is met sequential workflow. Het state machine model wordt ook wel event-driven workflow genoemd. De twee namen samen geven de kern weer van het model. Het draait om state en events. In tegenstelling tot een sequential model is de volgorde van stappen in de workflow dynamisch. Een event bepaalt de volgende state van de workflow. Een voorbeeld van een workflow is weergegeven in afbeelding 1.

In afbeelding 1 is een state machine workflow weergegeven in Visual Studio 2005. Het workflowvoorbeeld is grafisch weergegeven. De verschillende states en events zijn duidelijk te herkennen. Zo is er een 'OrderOpenState'. Op deze state zijn twee events gedefinieerd: 'OrderUpdatedEvent' en 'OrderProcessedEvent'. Wanneer een order wordt gewijzigd, gaat er wel een event af, maar blijft de state dezelfde. Wanneer de order verwerkt is, gaat het 'OrderProcessedEvent' af. In deze situatie verandert de state naar 'OrderProcessedState'.

Waarom state machine workflows?

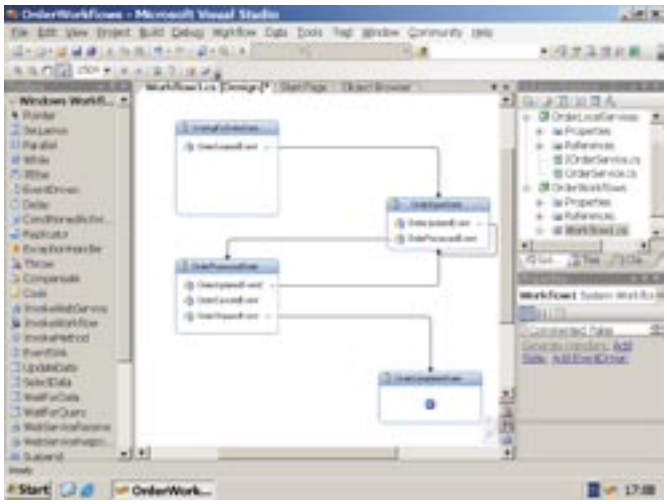
State machine workflows vervullen een specifieke behoefte. Deze behoefte heeft te maken met die businessprocessen die vaak veranderen binnen een organisatie. De kern van de informatiesystemen blijft gelijk, de processen veranderen echter veelvuldig. Proceslogica wordt meestal hard in de code opgenomen. Hierdoor is deze logica minder flexibel en moeilijk aan te passen. Met Windows Workflow Foundation is het mogelijk deze businessproceslogica uit de code te halen en te modelleren in Visual Studio. Het programma kan vervolgens als host optreden voor de Windows Workflow-engine. Wanneer het om een businessproces gaat waar veel menselijke interactie wenselijk is, komt het state machine-model om de hoek kijken. In het state machine-model kan de menselijke interactie worden vertaald in events. Wanneer iemand in het voorbeeld van afbeelding 1 een order wijzigt, dan heeft dit een event tot gevolg. Er zijn diverse processen die goed gemodelleerd kunnen worden in een state-diagram. Het gebruik van state-diagrammen is dan ook een veelgebruikt design pattern in de IT.

Wanneer we bijvoorbeeld het bugproces van MSF bekijken, dan is dit uitstekend te vertalen in een state-diagram. Een voorbeeld state-diagram is weergegeven in afbeelding 2. Wanneer een dergelijk proces in een sequential model gegoten moet worden, levert dit de nodige problemen op. Wanneer de 'Fixing'-state als uitgangspunt wordt genomen, dan wordt de dynamiek van het state-diagram duidelijk. Vanaf de 'Fixing'-state kan de bug veranderen in vier mogelijke andere states. In een sequential model resulteert dit in verscheidene 'if-then-else'-statements. Van nature past een bugproces goed in een state machine workflowmodel.

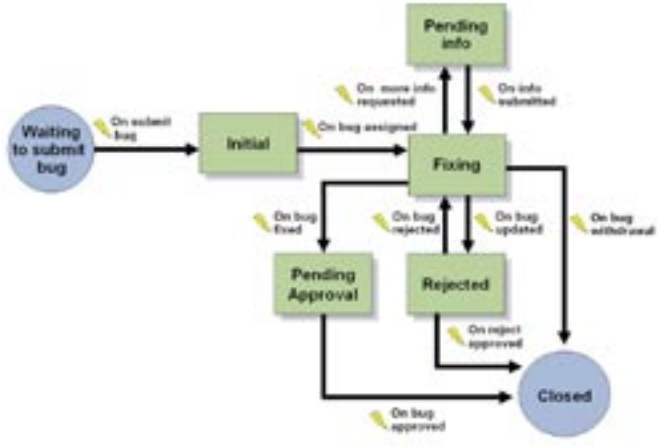
Basiselementen

Het state machine model in Windows Workflow Foundation kent vier basiselementen:

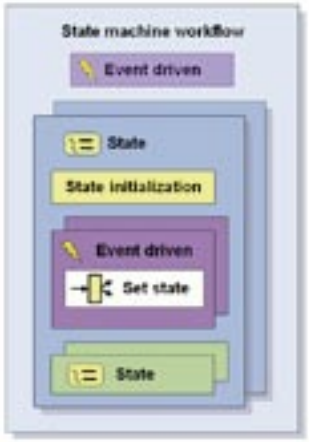
- State
- Event driven
- Set state
- State-initialisatie



Afbeelding 1. Voorbeeld van een state machine workflow



Afbeelding 2. MSF bug proces state-diagram



Afbeelding 3. Basiselementen in een state machine workflow

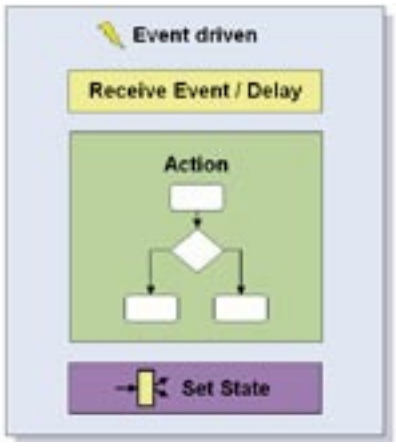
Iedere workflow is gebaseerd op deze vier basiselementen. De state machine zelf is een zogenaamde root activity. Windows Workflow Foundation maakt gebruik van state activities als bouwstenen om een workflow vorm te geven. Iedere state activity vertegenwoordigt een status. Een state activity wordt zelf weer opgebouwd uit een optionele state-initialisatie activity en één of meer event-driven activities. Een event-driven activity handelt de events af en zorgt voor een transitie in state (set state). Het formeren van een Windows Workflow is vergelijkbaar met lego, waarbij de activities zijn te vergelijken met legostenen. De basiselementen zijn weergegeven in afbeelding 3.

Querying een state machine workflow

Windows Workflow Foundation biedt de infrastructuur om in runtime vragen te stellen over de situatie waarin de workflow verkeert. Dit is cruciale informatie voor visualisatie en interactie met de gebruiker. Wanneer dynamisch wordt bepaald wat de status is dan kan ook worden bepaald wat de mogelijke events zijn die kunnen optreden. Deze events kunnen bijvoorbeeld worden vertaald in buttons.

In het bugprocesvoorbeeld uit afbeelding 2 kan een GUI worden gestuurd op basis van het resultaat van een query op de state machine workflow. Wanneer de workflow zich in de state Rejected bevindt, dan zijn alleen de buttons Accept en Update enabled. Op deze manier kan de controller uit het Model-View-Controler-pattern worden geïmplementeerd met een state machine workflow. Het uitvragen van de workflow zorgt ervoor dat bijvoorbeeld de volgende vragen kunnen worden beantwoord:

- Wat is de huidige status van de workflow?
- Welke transities zijn er mogelijk vanuit de huidige status?
- Wat zijn al de mogelijke states van de workflow?



Afbeelding 4. Event-driven activity

De state activity

De state activity is het belangrijkste element van een state machine workflow. De state activity representeert de status van de workflow. Een workflow verkeert op een moment in tijd in één state. De workflow gaat van de ene state naar een andere state. Een state activity kan event-driven activities bevatten. De states moeten overeenkomen met de stadia waarin een businessproces kan verkeren. Het businessproces is dus leidend voor het modelleren van de states in een workflow. Een veel gebruikte vuistregel is: iedere milestone in het businessproces wordt vertaald in een state activity. Iedere state machine workflow moet een initialisatie-state hebben, dit is een 'normale' state activity. Een workflow mag een 'completed state' hebben. Een workflow hoeft dus niet een gedefinieerd einde te hebben. In een dergelijk geval stopt de workflow zodra de host (het proces) wordt stopgezet. Wanneer een state machine workflow wordt ingezet als controller in een MVC, dan bestaat de workflow zolang de applicatie draait. De initialstate en completedstate zijn properties van de workflow. Zodra een transitie wordt gedaan naar de completedstate wordt de executie van de workflow-instantie afgebroken. The instantie gaat 'shut down'.

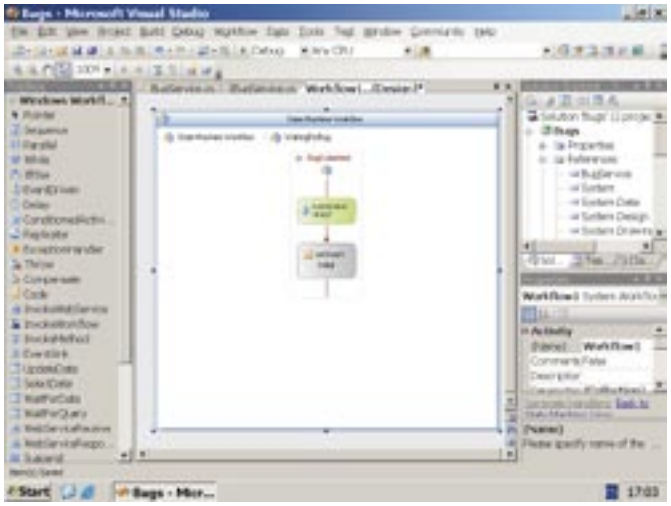
De event-driven activity

Een belangrijk andere activity is de event-driven activity. Deze activity wordt gebruikt om events af te handelen. De event-driven activity kan onderdeel zijn van een state activity of direct worden gekoppeld aan de root van een workflow. De structuur van een event-driven activity komt overeen met hoe deze wordt gebruikt in een sequential activity. Een event-driven activity bestaat uit een sequentiële set van activiteiten. De eerste moet een event sink of een delay zijn. Er kan maar één event sink of delay worden gebruikt. Dit is anders dan in een sequential workflow. Om events te kunnen afhandelen is het noodzakelijk om een interface te definiëren voor de mogelijke events. Een dergelijke interface wordt een DataExchangeService genoemd. In voorbeeldcode 1 is een interface gedefinieerd voor het bugprocesvoorbeeld.
De interface moet vervolgens worden geïmplementeerd in een class voor de werkelijke afhandeling van events. Deze class wordt geregistreerd bij de host, zodat de events ook werkelijk worden

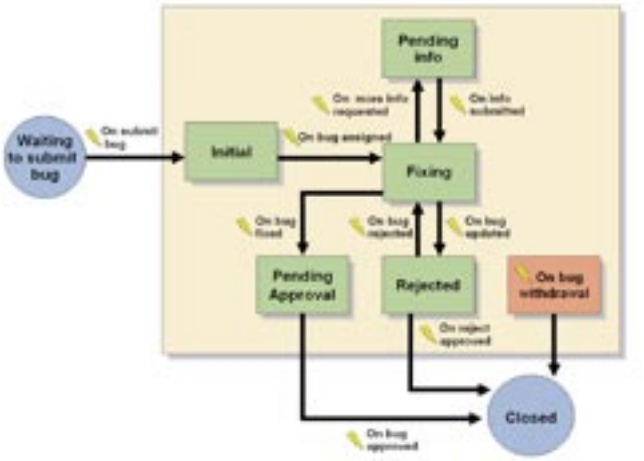
```
using System;
using System.Collections.Generic;
using System.Text;
using System.Workflow.ComponentModel;
using System.Workflow.Runtime.Messaging;

namespace BugService
{
   public class BugEventArgs : WorkflowMessageEventArgs
   {
      private string _bugId;
      public BugEventArgs(Guid instanceId, string bugId) :
      base(instanceId)
      {
         BugId = bugId;
      }
      public string BugId
      {
         get { return _bugId; }
         set { _bugId = value; }
      }
   }
   [DataExchangeService]
   public interface IBugService
   {
      event EventHandler<BugEventArgs> BugSubmitted;
      event EventHandler<BugEventArgs> BugAssigned;
      event EventHandler<BugEventArgs> BugInfoRequested;
      event EventHandler<BugEventArgs> BugInfoSubmitted;
      event EventHandler<BugEventArgs> BugWithdrawed;
      event EventHandler<BugEventArgs> BugFixed;
      event EventHandler<BugEventArgs> BugUpdated;
      event EventHandler<BugEventArgs> BugRejected;
      event EventHandler<BugEventArgs> BugApproved;
      event EventHandler<BugEventArgs> BugRejectApproved;
   }
}
```

Voorbeeldcode 1:
Interface-definitie voor de data exchange-service



Afbeelding 5: Event-driven activity sequence



Afbeelding 6. Voorbeeld van een recursive composition

afgehandeld. In voorbeeldcode 2 wordt een class weergegeven met een gedeeltelijke implementatie. In het voorbeeld is alleen de BugSubmitted volledig geïmplementeerd.

De set state activity

Binnen een state machine workflow is het belangrijk om van state te kunnen veranderen. Om de state te veranderen wordt gebruikgemaakt van een set state activity. De set state activity kent een property TargetState waarmee een gewenste state in de workflow kan worden aangegeven.

Een set state activity wordt als onderdeel gebruikt van een event handler. Het is altijd de laatste activity in een event handler. Hierdoor wijzigt daadwerkelijk de state. In afbeelding 5 is een voorbeeld uitgewerkt voor het event BugSubmitted. Naast het werkelijk afvangen van het event wordt als laatste stap een set state uitgevoerd. In dit voorbeeld is de volgende state Fixing. Wanneer de set state activity wordt uitgevoerd, zorgt dit ervoor dat de activity waar het een onderdeel van is wordt beëindigd. Verder zorgt de set state ervoor dat de status van de workflow in de gewenste state wordt veranderd.

Recursive composition

Wanneer een event door meer states afgehandeld moet worden, dan heeft dat tot gevolg dat het event in alle states gedefinieerd moet zijn. Dit kan ook worden opgelost door een zogenaamd recursive composition pattern. In Windows Workflow Foundation is het mogelijk om een event slechts één keer te definiëren en vervolgens te gebruiken in meer states. In Windows Workflow Foundation worden meer state activities gegroepeerd om een event te delen. Het model is te vergelijken met een base class waarvan overerving plaatsvindt. Op het base class-niveau wordt het event gedefinieerd, de andere classes die overerven krijgen het event 'cadeau'.

```
using System;
using System.Collections.Generic;
using System.Text;

namespace BugService
{
  class BugService : IBugService
  {
    public BugService() { }

    public void RaiseBugSubmittedEvent(string bugId, Guid instanceId)
    {
      if (BugSubmitted != null)
        BugSubmitted(null, new BugEventArgs(instanceId, bugId));
    }

    public event EventHandler<BugEventArgs> BugSubmitted;
    public event EventHandler<BugEventArgs> BugAssigned;
    public event EventHandler<BugEventArgs> BugInfoRequested;
    public event EventHandler<BugEventArgs> BugInfoSubmitted;
    public event EventHandler<BugEventArgs> BugWithdrawed;
    public event EventHandler<BugEventArgs> BugFixed;
    public event EventHandler<BugEventArgs> BugUpdated;
    public event EventHandler<BugEventArgs> BugRejected;
    public event EventHandler<BugEventArgs> BugApproved;
    public event EventHandler<BugEventArgs> BugRejectApproved;
  }
}
```

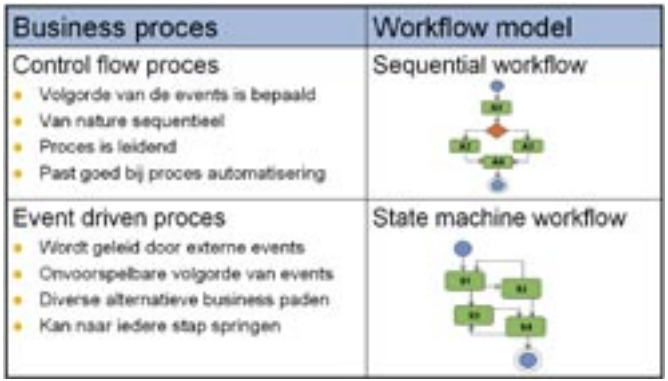
Voorbeeldcode 2:
Implementatie van de data exchange-interface

Wanneer we het bugproces van MSF bekijken, moet het op ieder moment mogelijk zijn om de bug door een bevoegd persoon terug te trekken. In afbeelding 6 is dit opgelost door een recursive composition te gebruiken. In afbeelding 6 is aangegeven dat een bug op ieder moment kan worden teruggetrokken en dat de status closed is bereikt.

Hoe maak je een state machine workflow?

Om een state machine workflow te definiëren wordt Visual Studio 2005 gebruikt. Het grootste deel kan worden gemodelleerd in de visual designer. In Visual Studio 2005 is een projecttype beschikbaar voor state machine workflows. Wanneer een project op basis van deze template wordt gemaakt, wordt een zogenaamde workflow1.cs source-file aangemaakt. Deze file kan worden gebruikt om een workflow te modelleren. Om een workflow werkend te maken, moeten de volgende stappen worden genomen:

1. Een project maken van het type state machine workflow
2. In de visual designer de state en event-driven activities definiëren
3. Definiëren van een argumenten-class en een data exchange-interface
4. Implementeren van de interface in een class die gebruikt kan worden door de host
5. Per event-driven activity de implementatie verzorgen. In ieder geval moeten er events worden afgevangen die worden gekoppeld aan de data exchange-interface
6. Wanneer noodzakelijk, moet de state worden veranderd in een event-driven activity door gebruik te maken een set state activity
7. Er moet een host worden gemaakt (of worden gebruikt) om de workflow in te draaien. Dit kan bijvoorbeeld een WinForms-applicatie zijn. Deze applicatie moet ook de class gebruiken die gedefinieerd is voor het opwerpen van de events.
8. De host kan worden uitgebreid om in runtime de status van een workflow te analyseren. Er is een API beschikbaar om de gewenste informatie op te vragen. Deze informatie kan bijvoorbeeld worden gebruikt voor het enablen/disablen van buttons in een form.



Afbeelding 7. Wanneer moet welk model worden gebruikt?

Dynamisch karakter

Windows Workflow Foundation biedt naast het sequential model ook een state machine model. Dit model is een workflowmodel dat dynamisch van karakter is en goed kan worden gebruikt wanneer er veel menselijke interactie wordt verwacht. In tegenstelling tot het sequential model is het verloop van het bedrijfsproces niet voorspelbaar. Windows Workflow Foundation kan worden gehost door een proces. Op deze manier kan de businessproceslogica uit de .NET-code worden gemodelleerd in een workflow. Dit brengt de wenselijke scheiding aan tussen de proceslogica en de businesslogica. Wanneer moet een bepaald workflow-model worden gebruikt? In afbeelding 7 wordt een vergelijking gemaakt tussen de twee workflow-modellen. Belangrijke eigenschappen van het state machine model zijn:

- Flexibiliteit, een businessproces zit niet vast aan vastomlijnd proces. De gebruiker is in controle.
- Transparant, op ieder moment kan de status van de workflow worden gevraagd en worden bepaald wat de mogelijke vervolg-states zijn.
- Controle, dankzij events en een recursive composition pattern heeft de gebruiker de controle over het businessproces.

Het state machine workflowmodel is een zeer interessant workflowmodel dat op veel plaatsen kan worden toegepast. In dit artikel is het model beknopt besproken. Er zijn details en mogelijkheden niet besproken. Zo biedt het state machine model de mogelijkheid het model dynamisch te 'updaten' of om een state 'over te slaan'. Het gaat te ver om dit allemaal in één artikel te bespreken. De hoofdlijn is besproken en er blijven genoeg interessante aspecten over om zelf te bestuderen. Windows Workflow Foundation is zo interessant dat je er vandaag mee moet beginnen. Het is de volgende stap in softwareontwikkeling. Het programmeren verandert langzaam, maar zeker in assembleren. Het is verstandig om de concepten vandaag tot je te nemen zodat je ze morgen kan gaan toepassen.

Anko Duizer is werkzaam als trainer/coach bij Class-A. www.class-a.nl Daarvoor heeft hij vijf jaar gewerkt bij Microsoft als consultant. Onder zijn klantenkring bevinden zich voornamelijk Top100-bedrijven in Nederland. Sinds begin 2001 is hij bezig met .NET. Speciale interesse heeft Anko voor de architectuur en het ontwerp van gedistribueerde databaseapplicaties en servicegeoriënteerde systemen. De blog van Anko is te vinden www.ankoduizer.nl en zijn mailadres is anko.duizer@class-a.nl

Referenties

www.windowsworkflow.net/
msdn.microsoft.com/windowsvista/building/workflow/default.aspx
msdn.microsoft.com/library/en-us/dnlong/html/WWFIntro.asp
www.ankoduizer.nl