Gerard van der Pol is technical consultant bij Borland Benelux.

Design Patterns in de praktijk

APPLICATION LIFECYCLE MANAGEMENT

Naarmate de omvang van projecten groter wordt, nemen ook de complexiteit en de behoeften aan efficiënte coördinatie van activiteiten toe. Om te slagen, hebben bedrijven een effectieve combinatie van mensen, processen en tools nodig. Application Lifecycle Management (ALM) is een industrieel erkend initiatief dat is ontworpen om een set sterk geïntegreerde tools te bieden, waarmee ontwikkelteams eenvoudiger succes kunnen boeken.

ALM vormt de brug tussen de tools die de leden van ontwikkelteams dagelijks gebruiken en de processen die softwareontwikkeling sturen. Ontwikkelteams produceren niet alleen software, vele artefacts zoals requirements, ontwerp, testplannen, documentatie, enzovoort vormen de basis voor het eindproduct. Om te voorkomen dat informatie tussen de verschillende disciplines onnodig gedupliceerd wordt, met alle gevolgen van dien, dient elk teamlid toegang te krijgen tot de informatie die nodig is voor het uitvoeren van zijn of haar taak. ALM omvat daarom de gehele applicatielevenscyclus, onderverdeeld in de zes fasen: definitie, ontwerp, ontwikkeling, test, deployment en management. Alle disciplines binnen het team worden dus ondersteund, de requirements-specialist, verantwoordelijk voor het definiëren van eisen en randvoorwaarden, de architect, die de requirements vertaalt in een blauwdruk van het systeem, de ontwikkelaar, verantwoordelijk voor de implementatie van de requirements, de tester die moet bepalen of de requirements juist worden geïmplementeerd en dus aan de eisen van de gebruiker voldoet, de manager die moet zorgen dat alles tijdig en binnen budget plaatsvindt, tot en met de deployer die ervoor moet zorgen dat het gebouwde systeem in de productieomgeving terechtkomt. Voor elk van deze rollen zijn specifieke tools beschikbaar. ALM richt zich op de integratie van deze tools om synergievoordelen te behalen uit de samenwerking tussen alle teamspelers.

Borland Application Lifecycle Management

De combinatie van tools, de zogenaamde ontwikkelstraat, is vaak opgebouwd rondom de gebruikte IDE. Juist om de synergie tussen de verschillende disciplines in het ontwikkeltraject te verkrijgen is een sterke integratie tussen alle producten van de ontwikkelstraat een must. Zo bestaat er ook voor Visual Studio.NET een ecosysteem van producten die tezamen het gehele traject kunnen ondersteunen. Er is echter helaas niet vaak sprake van een integratie over de gehele linie. De Borland ALM toolset richt zich op het beschikbaar stellen van een geïntegreerde uniforme aanpak voor platforms als .NET, Java, Mobile en C++.

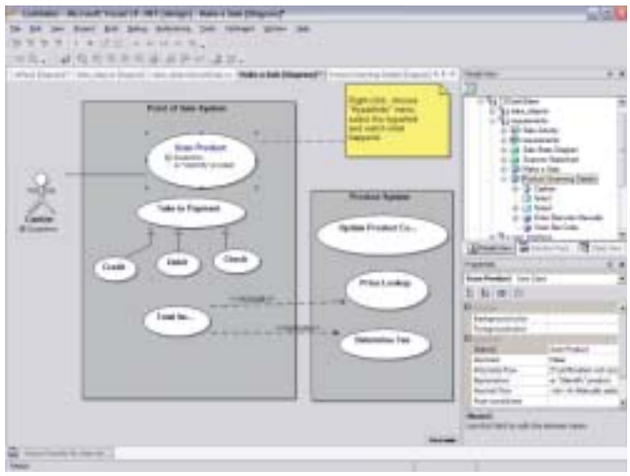
Om de ontwikkelstraat rondom Visual Studio.NET te complementeren zijn er de volgende producten:

- CaliberRM, dat zich richt op requirement-management.
- Borland Together® Edition for Visual Studio.NET, een UML™ modelleertool voor de ontwikkeling van logische en fysieke designmodellen, ondersteuning van design patterns en refactoring en genereren van documentatie.
- Borland Optimizeit™ Profiler, ontworpen om prestatieproblemen aan te pakken, waaronder excessieve tijdelijke objecttoewijzing en CPU-problemen.
- Borland StarTeam, een configuratie- en change managementsysteem met een volledige aanpasbare workflow-ondersteuning, waarin naast revisies ook taken, change request, discussies en requirements een prominente rol spelen.
- Borland Janeva, een Enterprise Application Integration (EAI) middleware-oplossing die interoperabiliteit biedt tussen .NET-, J2EE- en CORBA-omgevingen.

Borland Together Edition for Visual Studio.NET, gaan we wat nader bekijken in dit artikel.

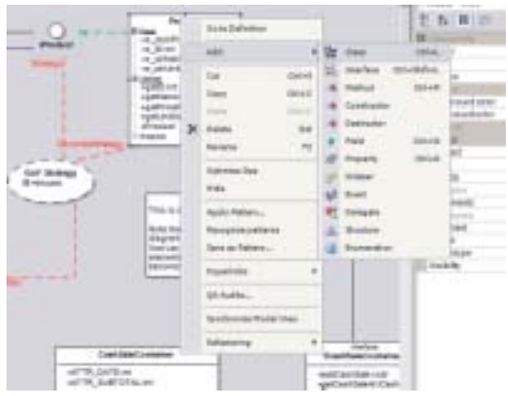
Together?

Belangrijk is om aan te geven is dat we Together kunnen zien als de 'hub' in het project. Het vormt de brug tussen requirements, ontwerp en de implementatie. Together levert dus een belangrijke bijdrage aan verscheidene fasen in de application lifecycle. Delen van UML zijn met name gericht op het vastleggen van de requirements, denk aan Use Case en Activitity diagrammen. De architect zal met name goed uit de voeten kunnen met Together om het design vorm te geven, en de ontwikkelaar heeft een uitstekend hulpmiddel om door een andere bril naar zijn code te kijken. Together bezit hiervoor een aantal eigenschappen, één daarvan is dat het model en de code als één wordt gezien, wat ervoor zorgt dat model en code continu synchroon worden gehouden. Een oud probleem dat code en ontwerp uit elkaar gaan lopen wordt hiermee vermeden. Deze technologie wordt 'lifesource' genoemd en zou je ook wel *simultaneous roundtrip engineering* kunnen noemen. De aanwezigheid van deze technologie maakt het zeer eenvoudig op elk moment in het project te starten met Together; een bestaand project is direct als een model te presenteren. Via het projectmenu of de solution explorer kan de support voor de Together-

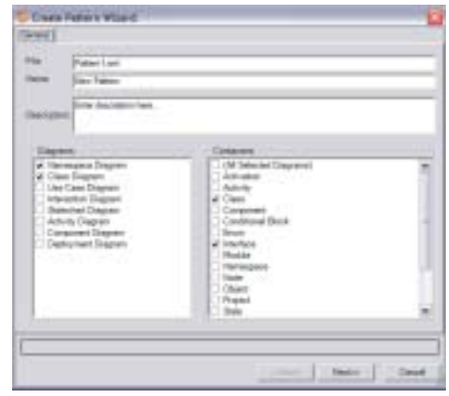


Afbeelding 1. Together integratie in Visual Studio.NET

Afbeelding 2. De Model View geeft een nieuw perspectief op het project



Afbeelding 4. Optimale ondersteuning voor visual C# F.Net en Visual Basic .NET

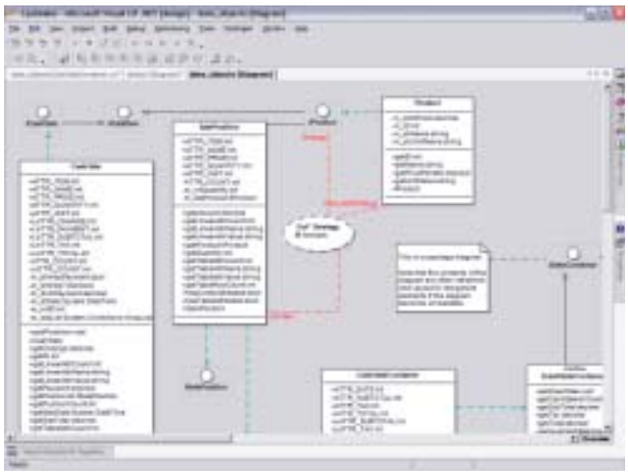


Afbeelding 5. Eigen design-patterns kunnen eenvoudig worden toegevoegd

technologie in het project geactiveerd worden. Het default-gedrag kan met de Visual Studio-opties ingesteld worden. Aangezien de Together-support per project kan worden ingesteld, is het mogelijk in een Visual Studio solution per project te bepalen of er modeling-support nodig is. Vanaf het moment dat de Together-support geactiveerd is, is er een aantal extra views beschikbaar. De toolbox geeft afhankelijk van het geselecteerde soort diagram toegang tot UML-elementen, de Model View biedt identiek aan de solution explorer hulp om door het project te navigeren en de diagram designer die naast class diagrammen ook use case-, sequence-, collaboration-, statechart-, activity-, component- en deployment-diagrammen ondersteunt. Naast de eigen code kunnen ook classes van bijvoorbeeld het Framework of third party library worden weergeven via een class diagram of eenvoudig in het model worden opgenomen.

Vanuit de editor kan snel genavigeerd worden naar het betreffende element op de class diagram. De Model view is met name hiervoor geïntroduceerd. Identiek aan de optie Synchronize Class View om vanuit de editor de Class View synchroon te laten lopen met het geselecteerde type in de editor, biedt de Together-integratie een Synchronize Model View-optie. Vanuit de model view kan dan direct het geselecteerde element op het diagram worden weergegeven. Together bezit over een aantal handige navigatiehulpmiddelen, zoals hyperlinks en shortcuts. Met hyperlink kunnen diverse UML-diagrammen aan elkaar worden gekoppeld zodat gemakkelijk het gehele model kan worden verkend, shortcuts faciliteren hergebruik, ze maken het mogelijk om library classes op diagrammen weer te geven en kunnen relaties tussen diagrammen in het model tonen.

Door middel van een class diagram krijgt men een statische overview van het systeem, het diagram presenteert de classes



Afbeelding 3. Toegepaste design-patterns zijn traceerbaar

en de relaties daartussen. Een complex project kan in Together opgedeeld worden in verschillende views, waar via shortcuts kan worden genavigeerd door het gehele systeem. Together kan automatisch class diagrammen tonen van de classes, interfaces en relaties binnen een namespace. Niets echter beperkt de gebruiker zijn eigen views te vervaardigen waarin een subset van de informatie wordt getoond.

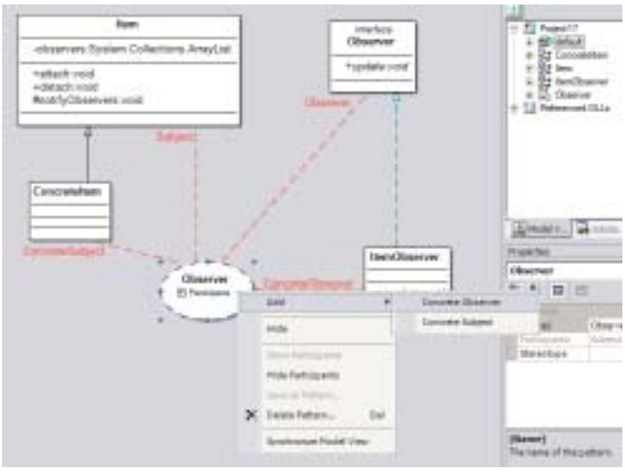
De functionaliteit die wordt geboden is contextgevoelig en geeft ondersteuning voor elementen van de taal zoals delegates, indexers en events.

Patterns & practices-ondersteuning in Visual Studio

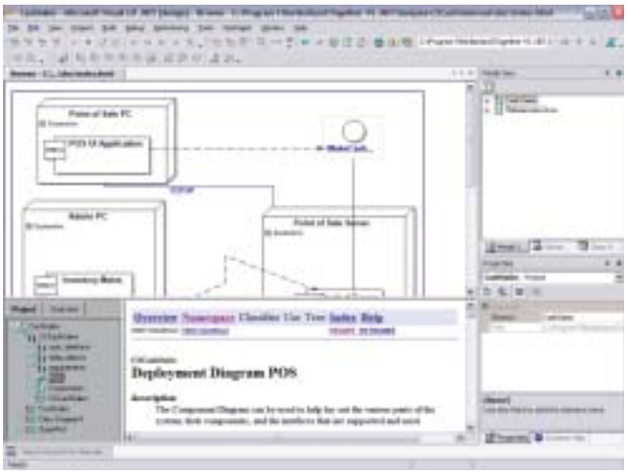
Together ondersteunt het gebruik van design patterns, en sluit hiermee heel goed aan bij Microsoft Patterns & Practices, aanbevelingen op het gebied van ontwerp, implementatie en deployment. Zie onder andere http://msdn.microsoft.com/practices en Design Patterns van de 'Gang of Four' voor meer informatie omtrent het onderwerp design patterns. Met design patterns wordt als het ware een abstractielaag toegevoegd die het mogelijk maakt programmeertaalonafhankelijk over oplossingen te communiceren. De patterns worden in verscheidene programmeertalen beschreven. Met behulp van Together kunnen deze patterns toegepast worden in een C# of Visual Basic .NET Visual Studio project. De default Together-installatie bevat al een behoorlijke hoeveelheid patterns van diverse bronnen. Ook beschikt de ontwikkelaar over de mogelijkheid om nieuwe patterns aan de collectie toe te voegen en reeds opgeslagen patterns aan te passen aan nieuwe inzichten. Een groot deel van de patterns zoals is gedocumenteerd in patterns & practices zijn reeds aanwezig en gereed voor gebruik. Ook is Together in staat om het gebruik van design patterns te herkennen, en dit weer te geven in een class diagram.

Om een pattern toe te voegen aan de collectie is er de Create Pattern Wizard. Nadat de betreffende UML-elementen in het class diagram zijn geselecteerd kan de wizard assisteren bij het toevoegen aan de Pattern Registry. Aangegeven kan worden waar het pattern beschikbaar is, of het nieuwe pattern parameters bevat, die het gebruik van het pattern flexibeler maken en welke locatie het pattern in de pattern registry dient te krijgen. Aangezien er al een grote hoeveelheid Design Patterns gedocumenteerd zijn, en er bijna dagelijks nieuwe toegevoegd worden, is het belang deze patterns te rubriceren om het overzicht te behouden. De Pattern Registry en Organizer richten zich met name op managen van de beschikbare patterns.

Je kunt patterns gebruiken om classes en relaties te vervaardigen of te wijzigen. In het volgende voorbeeld is een mechanisme nodig dat nadat de Item-class een item heeft geretourneerd



Afbeelding 6. Together biedt perfecte ondersteuning voor design-patterns



Afbeelding 7. Up-to-date technische documentatie binnen handbereik

de teller wordt opgehoogd en andere delen van het systeem worden ingelicht. Het Observer-pattern is nuttig als oplossing voor een dergelijk probleem. Om het Observer-pattern op de Item-class toe te passen, gebruiken we de Pattern Wizard. Together VS .NET herkent het pattern en visualiseert de patterns op het diagram als een ovaal. Bovendien worden de deelnemers van het pattern aangegeven en gekoppeld aan het pattern. De pattern-elementen op het diagram hebben specifieke pattern-acties die relevant zijn voor het pattern.

Together VS .NET heeft een UML documentatie wizard die HTML-documentatie kan genereren van het project. De gegeneerde documentatie wordt na vervaardiging in de Visual Studio .NET web-browser geopend.

Together complementeert Visual Studio .NET niet alleen met de standaard UML modeling-capaciteiten, maar biedt tevens ondersteuning aan het maken en toepassen van design patterns en het genereren van een volledige set van documentatie. Deze uitbreidingen maken van Visual Studio .NET een meer hybride tool die inzetbaar is in meer fasen van de hierboven besproken Application Lifecycle.

Nuttige internetadressen
http://www.borland.nl/alm
http://bdn.borland.com
http://msdn.microsoft.com/practices

Gerard van der Pol is technical consultant bij Borland Benelux. www.borland.nl