Een overzicht van het .NET platform

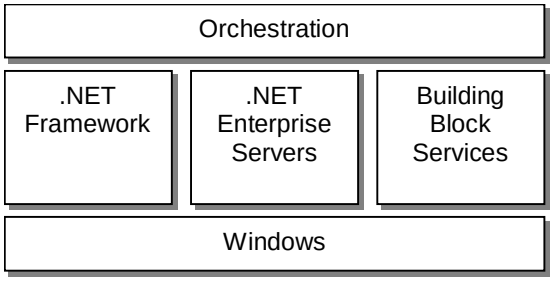
Microsofts visie op Internet

Danny Greefhorst

Microsoft geeft met het .NET platform zijn visie op het Internet en is daarmee een regelrechte concurrent van Java gerelateerde alternatieven. .NET is het platform dat gebruikers in de toekomst in staat moet stellen om gebruik te maken van alle diensten die via het Internet beschikbaar zijn. Het accent zal hierbij verschuiven van het bieden van websites naar het bieden van webdiensten, die programmatisch beschikbaar zijn voor andere webapplicaties en eenvoudig verhuurt kunnen worden. Een belangrijk onderdeel van het .NET platform is het .NET framework; een programmeertaal- en apparaat-onafhankelijke ontwikkel- en run-time omgeving voor applicaties. Dit artikel geeft een overzicht van het .NET platform, het .NET framework en de daarbij behorende ontwikkelgereedschappen.

.NET platform

Het .NET platform, zoals dat in de zomer van 2000 op de Amerikaanse Professional Developers Conference aan het publiek is gepresenteerd, zal bestaan uit een aantal onderdelen (zie figuur 1). In de eerste plaats is er een besturingssysteemlaag, die zal worden ingevuld door de verschillende smaken Windows die Microsoft voert. Deze variëren van de voor kleine apparaten bedoelde Windows-CE tot de enterprise versie van Windows 2000. In de toekomst zullen deze besturingssystemen mogelijk plaats maken voor een speciaal voor het .NET platform ontwikkeld besturingssysteem; Windows.NET. Daarnaast is het ook zeer waarschijnlijk dat derde partijen zullen komen met implementaties op andere besturingssystemen.



Figuur 1 .NET platform

Bovenop het besturingssysteem is het .NET framework beschikbaar. Dit raamwerk wordt gebruikt om .NET applicaties mee te ontwikkelen, waarvoor een groot aantal diensten worden aangeboden; maar hierover dadelijk meer.

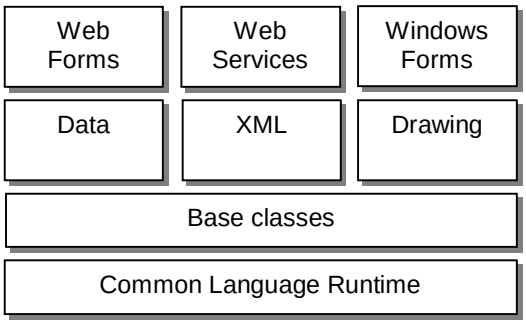
.NET applicaties worden ondersteund door verschillende enterprise servers die door Microsoft worden aangeboden. Onder deze noemer vallen veel van de bestaande server producten van Microsoft zoals SQL Server en Exchange Server, maar ook de nieuwere BizTalk server. Deze laatste is een op XML-gebaseerd applicatie-integratie product, dat onder de noemer “information broker” kan worden geschaard.

Naast deze enterprise servers zullen .NET applicaties gebruik kunnen maken van commerciële webdiensten van Microsoft en andere partijen. Voorbeelden van dergelijke “building-block services” zijn identiteit, notificatie, personalisatie, opslag en agenda.

De bovenste laag van het .NET platform is de “orchestration” laag, die een workflow-gebaseerde abstractie biedt bovenop de .NET diensten. Hiermee kunnen bedrijfsprocessen onafhankelijk van hun implementatie worden gedefinieerd. De eerder genoemde BizTalk server speelt ook hier een belangrijke rol. Middels een koppeling met Visio is het mogelijk om op een visuele manier bedrijfsprocessen te modelleren en te koppelen aan implementaties. Er zijn vanuit BizTalk verschillende soorten kopellingen mogelijk waaronder koppelingen gebaseerd op XML, COM en SOAP.

.NET framework

Het belangrijkste nieuws in het .NET platform is het .NET framework (zie figuur 2). Centraal in dit raamwerk staat de Common Language Runtime (CLR). Deze CLR definieert een virtuele machine, vergelijkbaar met de Java virtuele machine. Het belangrijkste verschil met de Java virtuele machine is dat de CLR geheel gebaseerd is op Just In Time (JIT) compilatie, waarbij de instructies zo snel mogelijk worden vertaald naar machine-specifieke code. De voertaal van de CLR is Intermediate Language (IL); wederom erg vergelijkbaar met Java bytecode. Door het gebruik van een virtuele laag wordt het mogelijk .NET applicaties op elk platform te executeren waarvoor een CLR-implementatie beschikbaar is. Alhoewel Microsoft hier niet direct over spreekt, lijkt dit in de toekomst toch een belangrijk speerpunt te kunnen worden.



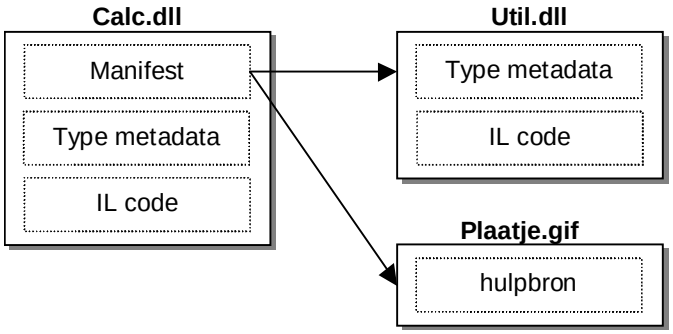
Figuur 2 .NET framework

Het andere belangrijke voordeel van zo'n virtuele laag en de CLR in het bijzonder, is dat deze programmeertaal-onafhankelijk is, waardoor een veel hechtere vorm van integratie tussen verschillende programmeertalen mogelijk wordt. Microsoft heeft dat dan ook in verre vorm doorgevoerd; alle talen in Visual Studio, het ontwikkelgereedschap van Microsoft, kunnen omgaan met IL. Hierbij wordt er niet alleen IL gegenereerd; ook wordt er gebruik gemaakt van bestaande IL en de bijbehorende meta-informatie. Dat laatste maakt het mogelijk dat een klasse in Visual Basic direct gebruik kan maken van een klasse in Visual C++, bijvoorbeeld door er van te erven. Naast de door Visual Studio ondersteunde talen, Visual Basic, C++ en de nieuwe taal C# (uitgesproken als CSharp), hebben andere bedrijven gewerkt aan .NET versies van een twintigtal andere programmeertalen zoals Eiffel, COBOL, Python en Perl.

De CLR biedt verder een aantal diensten aan de applicaties die erop executeren, zoals beheer van processen, threads, security en geheugen. In het bijzonder biedt de CLR "garbage collection" waarbij objecten die niet meer worden gebruikt automatisch worden opgeruimd. Merk op dat .NET programmeertalen deze voorzieningen dus niet zelf hoeven te regelen. Een belangrijke voorwaarde om gebruik te maken van deze garbage collection is dat de code zich houdt aan bepaalde gedragsregels, zoals het vermijden van het gebruik van pointers, ook wel aangeduid als "managed code".

Een belangrijk concept in het .NET framework is dat van een "assembly"; een logische eenheid van types en bijbehorende hulpbronnen. Deze assemblies zijn de eenheid van installatie, security, versionering en zichtbaarheid in .NET. Hiertoe zijn zij voorzien van zogenaamde "manifests" waarin hun meta-data is beschreven zoals hun naam, versienummer, bijbehorende bestanden, geëxporteerde elementen, benodigde permissies en hun afhankelijkheden naar specifieke versies van andere assemblies. Merk op dat een assembly niet noodzakelijk in één bestand hoeft te zijn gedefinieerd (zie bijvoorbeeld

figuur 3). Daarnaast wordt door het toestaan van meerdere versies van één assembly en het kunnen definiëren van versie policies de bekende "DLL hell"[1] opgelost.



Figuur 3 assembly bestaande uit drie bestanden

Voorgedefinieerde .NET klassen

Bovenop de CLR zijn een groot aantal voorgedefinieerde .NET klassen beschikbaar. Naast ondersteuning voor de basisbehoeften van .NET applicaties is er hier gedacht aan standaard klassen voor tekenen, databasetoegang, XML, windows en web-gebaseerde user interfaces en webdiensten. Het moge duidelijk zijn dat overal in deze klassen XML een belangrijke rol speelt. Zo is er een nieuwe programmeer-interface (API) voor database toegang gedefinieerd onder de naam ADO+. In tegenstelling tot wat de naam doet vermoeden gaat het hier om een geheel nieuwe interface waarbij XML wordt gebruikt voor het versturen en opslaan van databasegegevens. ADO+ promoot het gebruik van de database op een connectieloze manier, waarbij gehele gegevensverzamelingen worden overgehaald en later kunnen worden gesynchroniseerd met de originele database, om de schaalbaarheid te vergroten.

Voor het definiëren van web-diensten wordt er sterk geleund op de mede door Microsoft ingediende standaard SOAP (Simple Object Access Protocol). SOAP biedt een XML-gebaseerde codering voor het aanroepen van een dienst op een andere machine. De operaties die een dienst aanbiedt worden hierbij in het op XML-gebaseerde Web Service Description Language (WDSL) gedefinieerd. Het adverteren en vinden van diensten vindt vervolgens plaats middels de Universal Description, Discovery and Integration (UDDI) standaard. SOAP is in principe onafhankelijk van de gebruikte transportlaag, maar in de praktijk zal dat voornamelijk HTTP zijn.

Een andere belangrijke component in het .NET framework is ASP+, dat wordt gebruikt om web-gebaseerde gebruikersinterfaces te definiëren. ASP+ is een geheel nieuwe implementatie van het ASP web-server raamwerk waarbij gestreeft is naar browser-

[1] Deze DLL hell treedt op wanneer nieuwe applicaties bestaande DLL's overschrijven met nieuwe versies, waardoor oude applicaties potentieel niet meer goed functioneren.

onafhankelijkheid. Hiertoe zijn er standaard klassen beschikbaar die automatisch detecteren wat de mogelijkheden van de gebruikte web (of WAP) browser zijn en op basis daarvan HTML (of WML) genereren. Hierdoor hoeven webpagina's dus maar één keer te worden gedefinieerd en zijn ze automatisch voor alle soorten browsers beschikbaar. Een andere verandering die ASP+ sinds ASP heeft ondergaan is de ondersteuning voor alle mogelijke .NET programmeertalen, waardoor ASP-scripts nu ook gebruik kunnen maken van de voordelen van getypeerde talen. Tenslotte worden ASP+ pagina's nu om snelheidsredenen ook gecompileerd.

Natuurlijk is het nog steeds mogelijk om applicaties met een Windows-specifieke gebruikersinterface te ontwikkelen. Hiervoor zijn een groot aantal standaard klassen aanwezig die gebruikt kunnen worden vanuit alle .NET programmeertalen. Nieuw hierin is het gebruik van "anchoring" en "docking" om de positie van elementen op het scherm aan te geven, waardoor formulieren er ook goed uit zien als ze van grootte veranderen. Verder wordt er gebruik gemaakt van GDI+ waardoor er technologisch hoogstaande effecten mogelijk zijn en er ondersteuning is voor bijvoorbeeld transparante en gelaagde windows in Windows 2000.

Visual Studio.NET

Natuurlijk hebben we gereedschappen nodig om software te ontwikkelen voor het nieuwe .NET platform. Bij Microsoft zijn ze daarom druk bezig met de nieuwste versie van Visual Studio. Een eerste beta van deze nieuwe omgeving is voor het eerst beschikbaar gekomen tijdens de Professional Developers Conference. Het belangrijkste kenmerk van Visual Studio 7 (of Visual Studio.NET) is het feit dat het één geïntegreerde omgeving is voor alle programmeertalen. Niet alleen de door Microsoft meegeleverde programmeertalen als Visual Basic, C++ en C# kunnen daaroor dezelfde omgeving delen; ook programmeertalen van andere partijen die .NET ondersteunen kunnen vanuit dezelfde omgeving gebruikt worden.

Alle tools die in de Visual Studio omgeving beschikbaar zijn kunnen dus door alle programmeertalen gebruikt worden. Hieronder vallen ondermeer de editor, debugger, helpfunctionaliteit en gebruikersinterface builders. Van de laatste zijn er in Visual Studio.NET twee; een "windows forms designer" die gebruikt wordt om Windows interfaces te definiëren en een "web forms designer" waarmee gebruikersinterfaces voor het web kunnen worden gedefinieerd (zie figuur 4). Deze laatste zal erg herkenbaar zijn voor gebruikers van Visual InterDev; deze web-ontwikkelomgeving is namelijk geheel geïntegreerd en geoptimaliseerd om samen te werken met het nieuwe ASP+. Het debuggen van applicaties in deze omgeving is ook bijzonder; zo kan de debugger moeiteloos wisselen van programmeertaal.

Figuur 4 Visual Studio web forms designer

Verder zijn er wat extra tools voor de ontwikkelaar bijgekomen. Zo is er speciaal voor XML ondersteuning in de vorm van een XML-editor, waarmee ook grafisch XML schemadefinities (XSD) kunnen worden gemaakt. Daarnaast is er een “server explorer” bijgekomen waarmee de Microsoft server programmatuur kan worden beheerd. De nieuwe “component designer” kan samen met de server explorer worden gebruikt om op een visuele manier server-componenten te configureren en te assembleren, vergelijkbaar met grafische gebruikersinterface builders.

Naast deze nieuwe tools is de omgeving zelf wat vriendelijker gemaakt. Zo start Visual Studio standaard met de startpagina waarvandaan de meest gebruikte functionaliteit kan worden geactiveerd. De gebruiker wordt nu herinnerd aan de dingen die hij nog moet doen door een “task list”, die kijkt naar annotaties in de code. Verder is de helpfunctionaliteit een stuk intelligenter geworden, evenals de debugger, die nu meer soorten breakpoints kent. Andere uitbreidingen betreffen ondermeer ondersteuning voor macro’s, een uitgebreidere toolbox en een solution explorer waarmee bestanden kunnen worden beheerd.

Taalvernieuwingen

Interessanter wellicht zijn de vernieuwingen die de talen in de Visual Studio suite hebben gekregen, naast uiteraard de volledig nieuwe programmeertaal C#. Om te beginnen bij

Visual Basic; deze is nu voor het eerst volledig object-geörienteerd, zodat objecten nu kunnen erven van andere objecten. Hierbij hoort ook ondersteuning voor overloading, klasse variabelen ("shared members" in Visual Basic terminologie) en geparametriseerde constructors. De taal is een stuk rijker geworden door de verregaande integratie met het .NET framework. Onder deze noemer vallen ondermeer de ondersteuning van "delegates" (object-geörienteerde functie-pointers), "namespaces" en reflectie. Bestaande Visual Basic gebruikers worden geholpen met de migratie naar de .NET versie middels een upgrade tool. Helaas kan dit tool niet alle veranderingen automatisch doorvoeren.

Minder schokkend veranderd is Visual C++. Hieronder vallen de verbeterde compiler en de betere ondersteuning voor MFC en ATL wel. Veel interessanter is natuurlijk de ondersteuning voor het .NET framework. Zo zijn er nu "managed extensions" voor .NET waarmee het mogelijk is om "managed code" in Visual C++ te schrijven. Merk op dat C++ de enige taal is die standaard "unmanaged code" genereert. Verder biedt Visual C++ ondersteuning voor attribuut-gebaseerd programmeren, waarbij bestaande code kan worden gedecoreerd met bepaalde eigenschappen (attributen) die anders zouden moeten worden uitgeprogrammeerd. Een opvallende verschijning is de ATL-server die een C++ specifieke manier biedt om webapplicaties te schrijven. Deze ATL-server biedt een grote overlap met ASP+, maar hanteert een eigen syntax. Volgens Microsoft zou ATL server interessant zijn als er net even een betere performance nodig is.

C#

De belangrijkste noviteit in Visual Studio is de nieuwe programmeertaal C#, die qua eigenschappen erg dicht bij het .NET framework ligt. Op het eerste gezicht lijkt de taal erg op Java en dat is waarschijnlijk geen toeval (zie ook figuur 5). In de nieuwste versie van Visual Studio vervangt C# dan ook de Java-gebaseerde Visual J++ omgeving. Wel zullen er via het aangekondigde Java User Migration Path (JUMP) tools beschikbaar komen om bestaande Java code te gebruiken in-, of te converteren naar de .NET omgeving. Wat C# met Java deelt is een groot deel van de syntax, ondersteuning voor garbage collection, interfaces, inner classes en uiteraard de compilatie naar tussencode. Daarnaast zijn er echter ook een groot aantal verschillen te onderkennen, die tevens het erfgoed van de taal (C, C++) aan het licht brengen. Zo is er de mogelijkheid om "unmanaged code" te schrijven door het gebruik van pointers, of "P/Invoke" om bestaande DLL's aan te roepen. Ook zijn er in C# structs en functie-pointers (delegates) en kunnen operatoren worden geherdefinieerd. Waar in C# duidelijk goed over na is gedacht is het gebruik van primitieve types. In C# wordt er een onderscheid gemaakt tussen value types (op de stack gealloceerd) en reference types (op de heap gealloceerd). Primitieve types zijn value types, maar kunnen transparant naar een reference type worden vertaald middels een mechanisme dat "boxing" heet. Hierbij wordt er automatisch een wrapper-object gecreëerd voor de waarde. Het omgekeerde proces wordt ook wel aangeduidt als "unboxing". Een andere verbetering ten opzichte van Java is de ondersteuning voor properties, wel bekend voor gebruikers van Delphi. Hierbij kunnen eigenschappen worden gedefinieerd die analoog aan een attribuut kunnen worden benaderd, maar waaraan "get"

en “set” methoden kunnen worden geassocieerd. Het is in C# verder mogelijk om middels XML toelichtingen in de code aan te brengen, die kan worden gebruikt als basis voor XML-gebaseerde documentatie, vergelijkbaar met het voor Java beschikbare Javadoc.

```
class Hello {
  public static void Main() {
    System.Console.WriteLine(“Hello World!”);
  }
}
```

Figuur 5 voorbeeld C# programma

Conclusies

Met .NET lanceert Microsoft een interessant platform dat in de toekomst een belangrijke rol zal gaan spelen in de Internetwereld. Het is niet eens zo dat de onderliggende technologie zo baanbrekend is; veel belangrijker is het feit dat het een geïntegreerd platform is. Op dit moment is het .NET framework al behoorlijk compleet en in de komende tijd zullen we zien dat steeds meer producten van Microsoft en andere partijen onder de .NET paraplu zullen gaan vallen. De recentelijk door Sun aangekondigde SunONE strategie voor het realiseren van Java-gebaseerde webdiensten lijkt voorlopig een veel minder vaste vorm te hebben. Alhoewel Microsoft het niet zodanig positioneert zal .NET, met C# als belangrijkste voertaal, een voorzetting zijn van de tweesplitsing in de wereld tussen Microsoft en Java-gebaseerde standaarden.

Danny Greefhorst is werkzaam als senior adviseur bij het Software Engineering Research Centre te Utrecht.

