Edward Bakker en Adrie Geelhoed
zijn werkzaam als softwarearchitect voor LogicaCMG

Ontwikkelen van serverapplicaties voor mobile devices

MMIT, .NET Compact Framework en User-procescomponenten

De afgelopen jaren is er een enorme toename geweest in het aantal mobiele telefoons. Nu bijna iedereen beschikt over een mobiele telefoon, komen fabrikanten op de markt met steeds geavanceerdere PDA's en 'smart phones'. De hoogste tijd dus om eens te kijken naar de mogelijkheden die het Microsoft .NET-platform aan ontwikkelaars biedt bij de ontwikkeling van mobiele applicaties. Naast een blik op de beschikbare .NET-tools besteden de auteurs van dit artikel aandacht aan een aantal ontwerprichtlijnen dat hergebruik van businesslogica tussen mobile en desktop devices ondersteunt.

Ook voor de introductie van Microsoft .NET waren er al tools beschikbaar voor de ontwikkeling van applicaties voor mobile devices. Zo bestaat er bijvoorbeeld 'Embedded Visual Basic' (EVB) en 'Embedded Visual C++' (EVC). Beide tools bieden een totaal nieuwe ontwikkelomgeving die anders is dan de omgeving die gebruikt wordt voor desktop-ontwikkelingen. Hiermee hebben we dan eigenlijk direct het grootste nadeel van deze tools te pakken. Bovendien geldt voor 'EVB' dat de op VB Script gebaseerde omgeving vaak te weinig mogelijkheden biedt om stabiele, geavanceerde toepassingen te ontwikkelen.

Nieuwe generatie technologie

Microsoft heeft als onderdeel van het .NET-platform een aantal producten ontwikkeld dat het 'any device'-aspect ondersteunt, namelijk: Microsoft Mobile Internet Toolkit (MMIT) en Compact .NET Framework en Smart Device Extensions. Deze producten zijn gebaseerd op nieuwe technologie en bieden een oplossing voor bovenstaande problemen. De tools zijn opgenomen in Visual Studio .NET 2003 dat nu beschikbaar is. Bovenstaande tools dienen beide een apart doel en kunnen dan ook worden gezien als elkaar complementerende tools. Om de verschillen en overeenkomsten tussen deze producten duidelijk te maken dienen we de verschillende tools in meer detail te bekijken.

Microsoft Mobile Internet Toolkit (MMIT)

De Mobile Internet Toolkit is een uitbreiding op de ASP.NET 'Web Forms'. Na installatie van MMIT verschijnt in de IDE van Visual Studio.NET een extra projecttemplate (Mobile Web Application). Deze template dient als uitgangspunt voor een toepassing gebaseerd op de Mobile Internet Toolkit. Naast deze template biedt MMIT een verzameling *mobile controls*. Deze verzameling bestaat uit een aantal vaak gebruikte controls zoals: form, label, textbox, button en selectionlist.

Device-specifieke rendering

De Mobile Internet Toolkit runtime stelt de functionaliteit van ASP.NET beschikbaar aan mobile devices. De mobile userinterface wordt in Visual Studio .NET ontwikkeld met behulp van de mobile controls. De runtime herkent het type mobile device dat een request doet voor de mobile webpagina en genereert het meest geschikte outputformaat (WML, HTML of cHTML). Het kan echter zijn dat de technische beperkingen van het device, zoals de schermgrootte, een specifiek userinterface-ontwerp noodzakelijk maken. Alle mobile controls hebben een device-specifieke 'default rendering' (zie afbeelding 1). Daarnaast heeft de ontwikkelaar de mogelijkheid een specifiekere rendering toe te passen, die is gebaseerd op eigenschappen van een bepaald device. Zo kan men bijvoorbeeld gebruik maken van kleur of lettertype in het geval de toepassing benaderd wordt

vanuit de Internet Explorer van de PDA. Dit in tegenstelling tot een veel eenvoudigere output in het geval van een WAP-telefoon. Standaard wordt een groot aantal devices ondersteund. Het eenvoudig uitbreidbare model van de Mobile Internet Toolkit maakt het echter ook mogelijk zelf een adapter te ontwikkelen voor een totaal nieuw device.

Programmeermodel en toepassing van MMIT

Hoewel er kleine verschillen zijn, lijkt het programmeermodel van de Mobile Internet Toolkit erg veel op het model van ASP.NET. Een ontwikkelaar die bekend is met ASP.NET zal zich dan ook snel thuis voelen in de MMIT-omgeving. Zoals aangegeven is MMIT een *extension* op ASP.NET en kan daarom gezien worden als de Web Forms-variant voor een mobile device. Dit betekent dat er geen software nodig is op het device, een connectie naar een webserver is voldoende. Om deze reden is een MMIT-applicatie uitermate geschikt voor de huidige generatie mobiele telefoons. De beperkingen die gelden voor het display en toetsenbord van de mobiele telefoon vormen dan ook de belangrijkste beperkingen aan de mobile applicatie. Bovendien geldt dat er een continue verbinding dient te zijn met de server omdat de applicatie niet op het device kan draaien.

Afbeelding 1. Default rendering mobile controls op een WAP-telefoon

.NET Compact Framework en Smart Device Extensions

Is MMIT een uitbreiding op het bestaande ASP.NET -model, voor het .NET Compact Framework geldt dat het juist een subset is van het normale .NET Framework. Het .NET Compact Framework is speciaal ontwikkeld voor de beperkte resources van de huidige smart devices (PDA's en smart phones). Vooral de runtime en garbage collector zijn aangepast zodat ze goed presteren op devices die slechts enkele MB's aan geheugen bevatten. De Smart Device Extensions (SDE) is een add-in voor Visual Studio.NET en biedt de ontwikkelaar de visuele tools die nodig zijn om applicaties te ontwikkelen met het .NET Compact Framework. Daarnaast bevat de SDE een aantal zogenaamde *device profiles* die de nodige device specifieke settings bevatten.

.NET Compact Framework Programmeermodel

Zoals gezegd is het .NET Compact Framework een subset van het .NET Framework. Dit betekent dat de klassen uit het .NET Compact Framework wat betreft interface identiek zijn aan het equivalent in het .NET Framework. Alleen functionaliteit die een te grote impact heeft op de performance, te veel geheugen vereist of tegen de beperkingen van het besturingsysteem aanloopt zijn niet opgenomen in het .NET Compact Framework.

Het .NET Compact Framework biedt voornamelijk functionaliteit op de volgende vlakken:

- *User interface*: het Compact Framework biedt een subset van *System.Windows.Forms* classes. Met behulp van deze classes kunnen geavanceerde Windows CE user-interface ontwikkeld worden.
- *Database en XML*: het .NET Compact Framework bevat een verzameling klassen die communicatie met zowel relationele (SQL Server CE) als niet relationele databronnen mogelijk maakt.
- *XML Webservices*: volledige ondersteuning wordt geboden voor communicatie met XML Webservices.

Toepassing van het .NET Compact Framework

De aanwezigheid van het .NET Compact Framework op het mobile device biedt een aantal voordelen ten opzichte van devices waarbij dit niet geval is. Zo kunnen bijvoorbeeld applicaties ontwikkeld worden die niet continue een verbinding met de server nodig hebben. Gegevens kunnen lokaal worden opgeslagen op het device in een XML-bestand of zelfs een SQL Server CE database. Op de momenten dat er connectie met de server is, kan synchronisatie plaatsvinden tussen bijvoorbeeld een SQL Server CE op het device en SQL Server 2000 database op de enterprise server. Een nadeel dat een op het .NET Compact Framework gebaseerde applicatie kan hebben ten opzichte van een MMIT-applicatie is: deployment. Zowel de ontwikkelde mobile applicatie als het .NET Compact Framework zullen namelijk op het device geïnstalleerd dienen te worden. Dit brengt een aantal deployment-issues met zich mee.

Samenvattend

Beide tools bieden uitstekende mogelijkheden voor het ontwikkelen van mobile applicaties. De tools worden geleverd met een uitgebreid helpbestand en voorbeelden die het eenvoudig maken snel met deze producten aan de slag te gaan. Welk product gebruikt wordt voor de ontwikkeling van de applicatie is voornamelijk afhankelijk van het type device dat ondersteund dient te worden, en de gewenste functionaliteit van de mobile toepassing. Met behulp van een emulator is het mogelijk mobile applicaties te ontwikkelen en te testen zonder een echt mobile device te gebruiken. Voor beide tools geldt dat de applicatiel-

	Mobile Internet Toolkit	.NET Compact Framework
Installatie op device	Nee	Ja
Continue connectie	Ja	Nee
Lokale data-storage	Nee	Ja
Code behind pages	Ja	Ja
Visual Basic.NET, C#, J#, etc.	Ja	Ja

Tabel. Eigenschappen van MMIT en .NET Compact Framework.

ogica kan worden ontwikkeld in de zogenaamde *code behind pages*. Ontwikkelen kan in iedere '.NET taal' zoals Visual Basic.NET en C#.

Een stap verder.....

Uit wat voorgaand is beschreven blijkt dat er geen technische beperkingen zijn om de reeds bestaande businesslogica te gebruiken vanaf een mobile device. Zowel applicaties die zijn ontwikkeld met MMIT als applicaties ontwikkeld met het .NET Compact Framework kunnen namelijk communiceren met .NET-componenten of webservices op de company server. Voor het ontwerp van de applicatie dient echter rekening gehouden te worden met specifieke device-eigenschappen die vooral voor de userinterface gevolgen hebben. Deze worden voornamelijk veroorzaakt door het beperkte formaat van het device-display.

Een userinterface is verantwoordelijk voor de interactie met de gebruiker en voert de volgende taken uit:

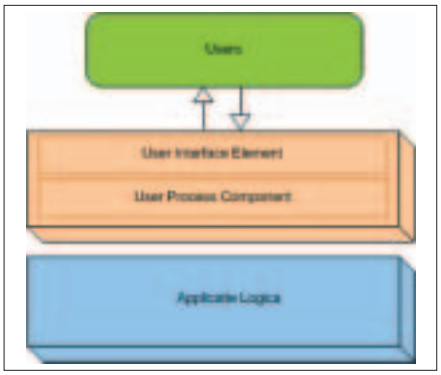
- Verzamelt user-input
- Valideert user-input
- Toont data aan de gebruiker
- Reageert op events die door een user gestart worden.

Zodra de user een event afvuurt (door te klikken op een button) wordt logica aangeroepen die de verzamelde en gevalideerde user-input doorgeeft aan een businesscomponent. Deze logica biedt de zogenaamde *controller* functionaliteit op de userinterface en wordt idealiter in een aparte methode op de userinterface geïmplementeerd. Zodra deze controller antwoord krijgt van de businesscomponent zal deze de userinterface-elementen updaten met het resultaat, bijvoorbeeld door een listbox bij te werken met nieuwe data. Er is hierover echter nog wel een drukke ontwerpdiscussie gaande.

Een scherm voor een mobile device als een WAP-telefoon zal over het algemeen veel minder user-input kunnen verzamelen dan een scherm dat is ontwikkeld voor een desktop. Dit betekent dat er meer mobile schermen nodig zijn voordat voldoende user-input verzameld is en de communicatie met de businesscomponent gestart kan worden. Het tussenresultaat van de aparte schermen dient bewaard te worden tot het moment dat er voldoende input is om te communiceren met de businesscomponent. Een zelfde situatie kan ontstaan voor de verwerking van het resultaat van de businesscomponent, het verversen van de userinterface. Dit is het moment om eens te kijken naar de mogelijkheden van zogenaamde 'User Process Componenten'.

User-procescomponenten

Een 'user-procescomponent' kan worden geïmplementeerd als een normale .NET-klasse die logisch gezien een extra laag vormt tussen de user-interface (scherm met daarop de controls) en de businesscomponent; zie afbeelding 2. De user-procescomponent kan worden gezien als een dialoogmanager – die afhankelijk van het device – in staat is de juiste dialoog te verzorgen noodzakelijk voor de uitvoering van een bepaalde actie op een businesscomponent.



Afbeelding 2. User-procescomponent als onderdeel van de architectuur

De user-component implementeert een publieke methode voor iedere actie die op de businesscomponent uitgevoerd kan worden. Deze methode wordt ook wel *action*-methode genoemd. Binnen deze methode kan een device-specifieke dialoog worden geïmplementeerd. In het geval van een mobile device kan het bijvoorbeeld voorkomen dat verscheidene schermen achtereenvolgens worden aangeroepen. In dit geval wordt het resultaat van de afzonderlijke schermen op de user-procescomponent opgeslagen. Op het moment dat de volledige dialoog doorlopen is, wordt de call gemaakt naar de businesscomponent. De user-procescomponent dient dus in staat te zijn alle tussentijdse resultaten te bewaren benodigd voor de uiteindelijke method-call op de businesscomponent. Het type device en de daarmee samenhangende dialoog bepaalt het aantal stappen (schermen) dat nodig is om de informatie op te leveren.

State van een user-procescomponent

Het bijhouden van tussentijdse resultaten maakt het mogelijk te bepalen in welke *state* het user-proces zich bevindt. Door ervoor te zorgen dat de user-procescomponent in staat is deze state te persisteren, wordt het bijvoorbeeld mogelijk *pause*-, *restart*- en *cancel*-functionaliteit te implementeren binnen de component. Hierdoor wordt het mogelijk het proces te continueren in het geval de verbinding met de server onverhoopt verbroken wordt; iets wat niet geheel ondenkbaar is in het geval van een mobile applicatie. Men moet in dit geval wel een unieke id toekennen aan de instantie van het user-

proces. Als deze gekoppeld wordt aan een gebruiker van de toepassing kan het user-proces opnieuw gecreëerd worden op het moment dat de gebruiker aanlogt op de applicatie. De gebruiker zal in dit geval 'doorgaan' waar hij gebleven was. In het geval van een .NET Compact Framework-applicatie kan de state van de user-procescomponent lokaal op het device gepersisteerd worden. Indien er synchronisatiefunctionaliteit aanwezig is tussen het mobile device en de centrale server kan de state hergebruikt worden vanaf de gebruiker. Op deze manier kan bijvoorbeeld een vertegenwoordiger *data-entry*-intensieve acties uitvoeren met de desktopvariant van de dialoog, deze opslaan en afmaken met de mobile versie van de dialoog op zijn PDA op het moment dat hij bij de klant is.

Implementatie van een user-procescomponent

Hoewel de implementatie van user-procescomponenten extra complexiteit oplevert tijdens het ontwerp en ontwikkeling van een applicatie biedt het aanzienlijk meer flexibiliteit. Deze flexibiliteit is vooral gewenst in het geval de applicatie vanaf meer soorten devices benaderd dient te worden. Daarnaast draagt het bij aan de onderhoudbaarheid van de user-interface-laag van de applicatie. De introductie van nieuwe schermen of aanbrengen van wijzigingen in de dialoogstructuur wordt door het gebruik van user-procescomponenten een stuk eenvoudiger.

Tot slot

Hierboven is slechts een globale beschrijving gegeven van user-procescomponenten. Daarbij is getracht duidelijk te maken welke rol deze kunnen spelen bij de ontwikkeling van applicaties die benaderd dienen te worden vanaf een mobile device. Meer informatie over user-procescomponenten kan gevonden worden in het document 'Application architecture for .NET' dat beschikbaar is voor download vanaf de Microsoft-site.

Nuttige internetadressen

- http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnbda/html/distapp.asp
- http://www.devbuzz.com/content/default.asp
- http://msdn.microsoft.com/library/default.asp?url=/msdnmag/issues/02/11/mmit/toc.asp