.net column

Worden Smart-Clients een nieuwe hype?

Wie vandaag de dag met .NET technologie een applicatie ontwikkelt, moet grofweg een keuze maken tussen een Windows Forms (Winform) applicatie of een Webforms (ASP.NET) applicatie. Veelal wordt deze keuze bepaald door de aard van de applicatie en de eisen die aan de distributie ervan worden gesteld. Ook spelen de eisen die aan de client worden gesteld op het gebied van hard- en software voor de te kiezen applicatiesoort een belangrijke rol.

Vanuit dit perspectief wordt op dit moment veel gekozen voor Webform applicaties vanwege de eenvoudige distributie via het intranet of het internet en de beperkte eisen die aan de client worden gesteld, namelijk een moderne webbrowser.

Toch hebben internet applicaties ook nadelen, die voor sommige projecten zo zwaar wegen, dat men noodgedwongen toch overstapt op Winform applicaties. Hierbij valt te denken aan: Speciale user-interface features die moeilijk met (D)HTML te bouwen zijn, snelheid van ontwikkelen, offline kunnen werken en toegang tot andere resources zoals de Graphic Device Interface (GDI). Voor deze categorie van applicaties zijn Smart-Clients wellicht een uitkomst.

Een Smart-Client is in principe een gewone Windows Forms applicatie, die via HTTP gedownload kan worden. Als de Smart-Client applicatie modulair is opgebouwd, worden alleen de assembly's gedownload, die op dat moment benodigd zijn.

Er zijn verschillende manieren om een Smart-Client te starten:

1. De gebruiker start de applicatie door middel van een URI. Deze kan bijvoorbeeld in de runbox worden ingetypt.
2. Er wordt gebruik gemaakt van een loader applicatie
   a. De loader applicatie kan met XCOPY worden geïnstalleerd op de locale machine.
3. Er wordt gebruik gemaakt van een ASP.NET (web) applicatie, die als loader fungeert.
   a. De gebruiker benadert de URI met behulp van een webbrowser.
   b. Via de webpagina wordt een lijst met applicaties gepresenteerd.

Nadat een Smart-Client gestart is, worden de gedownloade assembly's (dll/exe) in een lokale cache bewaard, die zelfs meerdere versies van dezelfde assembly kan bevatten. Als de Smart-Client applicatie gestart wordt, wordt het versienummer van de assembly op de remote server gecontroleerd. De assembly wordt alleen gedownload indien de versie die op de remote server staat, nog niet lokaal in de cache aanwezig is. Het fraaie van dit systeem is dat, als er op de server een nieuwe versie van de applicatie wordt uitgerold, deze automatisch gedownload wordt door alle clients op het moment dat de Smart-Client applicatie gestart wordt.

Natuurlijk hoeft een Smart-Client niet te betekenen dat het een 'fat client' is. Zo kunnen vanuit een Smart-Client met diverse technieken als .NET remoting, DCOM en Webservices, de bedrijfscomponenten worden benaderd op een remote machine. De benodigde remoting files kunnen met de Smart-Client worden gedistribueerd.

Uiteraard hebben Smart-Clients ook nadelen. Zo dient de .NET Runtime op elke client machine geïnstalleerd te zijn, hetgeen extra software eisen voor de client betekent. In .NET is het executeren van een gedownloade assembly afkomstig van een andere machine onderhevig aan Code Access Security. Welke assembly's wel rechten hebben op de lokale machine resources en welke niet, wordt bepaald door de configuratie-instellingen van het Code Access Security Model. Voldoende kennis op het vlak van .NET security is dus noodzakelijk voor een correcte werking van Smart-Clients.

Smart-Clients vullen het gat tussen Winform- en Webform applicaties. Zo heeft men het gemak van automatische distributie en versiebeheer via een URI (centrale machine) gecombineerd met een rijke programmeeromgeving voor de client-applicatie. Wel dient de .NET Runtime geïnstalleerd te zijn op elke client machine. Ook speelt .NET security een belangrijke rol bij Smart-Clients.

Ing. Xander Buffart is IT-architect bij Info Support te Veenendaal (e-mail: xanderb@infosupport.com).