Martijn Beenes en Marcel de Vries

# Wat is er nieuw in Visual Studio Team System 2008

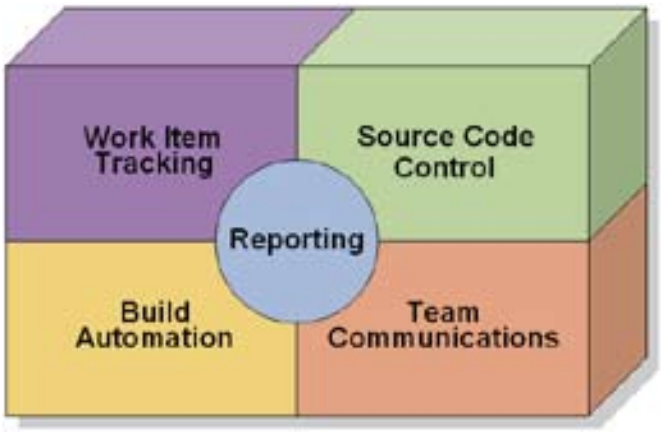
## TEAMEDITIES EN TEAMSERVER UITGERUST MET NIEUWE FEATURES

Met de release van Visual Studio 2008 is er al veel gesproken over alle mooie nieuwe features die geboden gaan worden in het .NET 3.5-platform, zoals LINQ, C# 3.0 en Visual Basic 9. Wat misschien veel mensen nog niet weten is dat er in de tussentijd ook door het developmentteam dat verantwoordelijk is voor Team System is gewerkt aan een set nieuwe features voor de verschillende Teamedities van Visual Studio en de Team Foundation Server. In dit artikel besteden de auteurs aandacht aan de belangrijkste zaken om zo een goed beeld te geven van wat binnenkort mogelijk is.

Binnen Team System is er een duidelijk onderscheid tussen het Team Foundation Server-gedeelte en de client-edities van Team System, onder de naam Visual Studio Team Architect, Developer, Test en Database Professional. De Team Foundation Server is op te delen in vijf onderdelen, zoals in afbeelding 1 is te zien. De belangrijkste nieuwe features en wijzigingen aan de server zijn doorgevoerd in Version Control en Build Automation. Aan de client-kant van het product zijn de Developer- en Test-edities de onderdelen waarin de meeste veranderingen hebben plaatsgevonden.

### Team Build

Laten we beginnen met Build Automation, beter bekend als Team Build. Wat direct in het oog springt, is dat Microsoft heel duidelijk naar de community heeft geluisterd. Daar wordt al sinds de release van Team Foundation Server 2005 geroepen dat er een optie zou moeten zijn om met Team Build continuous integration te ondersteunen. Dit is ook precies wat er is toegevoegd aan het product. Eigenlijk kun je stellen dat het Team Build-onderdeel van de Team Foundation Server een volledige refactoring-slag heeft doorgemaakt, waarbij er onder andere continuous integration is toegevoegd. Men heeft een volledig nieuw objectmodel beschikbaar gemaakt voor de integratie vanuit 3rd-party of eigen geschreven software. De nieuwe Team Build-omgeving introduceert ook een aantal nieuwe concepten, waaronder die van een build-agent. Een build-agent is een programma dat op een poort van de build-machine staat te luisteren naar build-commando's. De agent is verantwoordelijk voor de bootstrap van het build-proces, waaronder het downloaden van de juiste build-scripts naar de build-machine en het opstarten van MSbuild op basis van dit script. De build-agentsoftware dient op de build-machine te worden geïnstalleerd, voordat je deze kunt gebruiken in een build-definitie. Dit komt omdat er bij het selecteren van een agent in de build-definitie wordt gecontroleerd of deze te bereiken is op het netwerk. Bij het aanmaken van een build-definitie valt direct op dat er een volledig nieuwe wizard is gekomen waarmee je deze samenstelt. De schermen van de wizard bieden feitelijk een schil om de 'oude' wizard-schermen en bovendien is er een aantal nieuwe instellingen bij gekomen. Een opvallende en veel gevraagde aanpassing is wel dat er nu zelf een locatie geselecteerd kan worden waar de build-definitie (TFSBuild.Proj) binnen Version Control wordt opgeslagen. In de 2005-versie van TFS moet dit altijd een vaste locatie zijn binnen de version-control-boom. Nadat je een locatie hebt opgegeven, kun je opgeven van welke build-agent je gebruik wilt maken. Als je de agent hebt geselecteerd, heb je de optie gekregen het Drop Management te gaan inregelen. Met Drop Management wordt het mogelijk automatisch de steeds maar groeiende hoeveelheid build-data op te schonen. Dit wordt gedaan door het opgeven van één of meer Retention Policies voor de nieuwe build, Met een Retention Policy kan worden aangegeven hoelang de build-history bewaard moet blijven. (Met Team Foundation Server 2005 was dit per definitie oneindig en kon je alleen met server command-line-tools build-resultaten verwijderen). Door het opgeven van een retentie-policy wordt gezorgd dat 'oude' build-data automatisch wordt opgeruimd. Bij het instellen van de Retention Policy kan onderscheid gemaakt worden tussen builds die de status failed, stopped, partially succeeded of succeeded hebben. Overigens kun je nadat een build is uitgevoerd, alsnog aangeven dat de data van die build helemaal niet verwijderd moeten worden. De Retention Policy zal die build-data dan niet verwijderen. Hiervoor moet de build gemarkeerd worden als 'Retain Indefinitly'; zie afbeelding 2. Wanneer de markering weer verwijderd wordt, zal de data weer onder de Retention Policy vallen en verwijderd worden. Als laatste optie kan een keuze worden gemaakt wanneer er automatisch een build moet worden uitgevoerd. Dit is de plaats waar de Con-




Afbeelding 1. Onderdelen Team Foundation Server

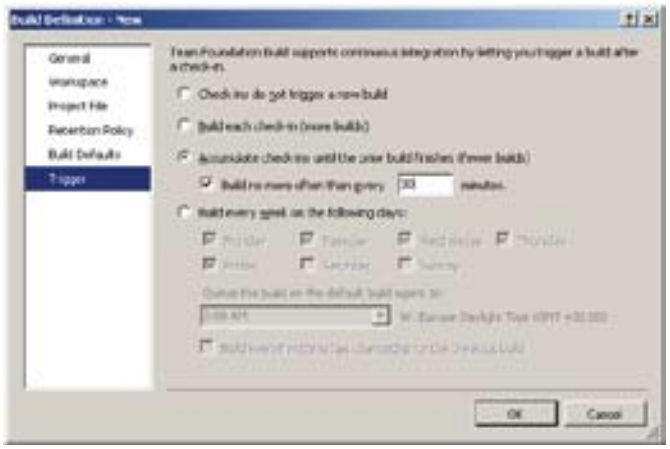
Afbeelding 2. Retain Indefinitely

tinuous integration-features zijn terug te vinden.
Door het opgeven van een trigger kan er gespecificeerd worden wanneer de build gestart wordt. Je kunt voor een van de volgende opties kiezen: No Trigger, Each Checkin, Accumulated en Scheduled; zie afbeelding 3. In het geval van No Trigger, geef je aan dat de build handmatig gestart zal worden. Als je kiest voor Each Checkin, dan zal iedere keer een build worden gestart als een change-set wordt ingecheckt in Version Control. Accumulated betekent dat er niet voor iedere check-in een build wordt gestart. Pas op het moment dat er changes zijn en een eerdere gestarte build al klaar is, wordt een nieuwe build gestart. Daarbij kan eventueel ook nog opgegeven worden dat er maar 1 build per x minuten uitgevoerd mag worden. Dit zorgt ervoor dat er minder frequent builds worden uitgevoerd. De laatste optie is het ingeven van een schedule. Hierbij kun je opgeven op welke dagen en op welk tijdstip een build moet worden uitgevoerd. Daarbij is er de keuze om toch een build uit te voeren ook al zijn er geen change-sets ingecheckt in Version Control. Overigens is het zo dat je in Team Foundation Server 2008 geen build meer start, maar dat je een build zogenaamd queuet. Iedere agent kan een aantal builds in een queue hebben staan die worden uitgevoerd zodra er tijd is. Hierbij wordt de queue afgelopen op basis van FIFO (First In First Out). Het is mogelijk een build die in de queue staat te annuleren of uit te stellen.

## Build Extensibillity

Zoals al eerder is aangegeven, is er een compleet nieuw objectmodel beschikbaar. Via dit objectmodel zijn alle 'oude' en nieuwe Team Build-features beschikbaar. Zo kan bijvoorbeeld een nieuwe build-definitie worden aangemaakt, een build worden gestart of informatie aan de build worden toegevoegd. Het objectmodel is te uitgebreid om hier in zijn geheel te behandelen, maar een klein voorbeeld maakt al snel duidelijk dat het objectmodel eenvoudig is te gebruiken. In codevoorbeeld 1 wordt een nieuwe build in de build-queue geplaatst. Als eerste wordt een referentie opgehaald naar de Team Foundation-server. Dit gebeurt op basis van een url zoals die ook in Visual Studio wordt opgegeven. Vervolgens wordt de buildserver-service opgevraagd. Deze service is het startpunt als er iets met TeamBuild moet worden gedaan. Met de buildserver-service kan vervolgens voor een bepaald TeamProject een build-definitie worden opgehaald. Een build heeft altijd een eigen build-definitie die gescoped is per TeamProject. De build-definitie bevat de gegevens zoals deze tijdens het doorlopen van de wizard voor het aanmaken van een nieuwe build zijn opgegeven. Er kan op dit moment gekozen worden om daar wijzigingen in aan te brengen. Als laatste wordt de build-definitie gebruikt om een build-request in de build-queue te plaatsen.

Naast het nieuwe objectmodel is ook de implementatie van de build-scripts rigoureus aangepast en biedt daardoor veel betere mogelijkheden de out-of-the-box-functionaliteit uit te breiden en aan te passen aan de specifieke wensen van het project. Net als met Visual Studio Team System 2005 wordt bij het aanmaken van een build een TFSBuild.Proj-bestand aangemaakt die alle MSBuild-properties bevat die voor de build nodig zijn. Vanuit dit bestand wordt naar een generieke targets-file verwezen in waarin de daadwerkelijk logica van de build staat

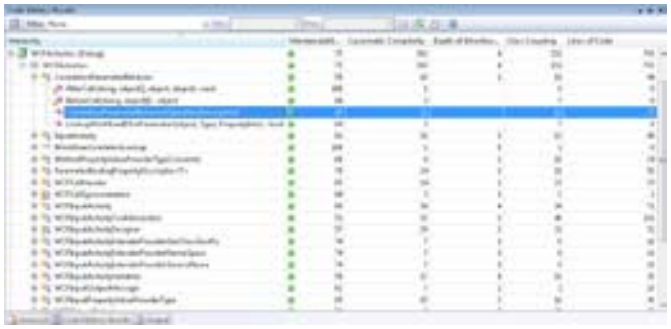


Afbeelding 3. Build-triggers

beschreven. Hoewel je deze file volledig kunt herschrijven, is het soms gemakkelijker om gebruik te maken van de aanwezige 'extension points'. Dit is eenvoudig te realiseren doordat er voor bijna elke target die door de build wordt uitgevoerd een pre-target en post-target is gedefinieerd. Deze targets bevatten standaard geen logica, maar kunnen opnieuw gedefinieerd worden in de TFSBuild.proj om extra logica aan de build toe te voegen. Met Visual Studio Team System 2005 was het niet mogelijk per solution extra stappen uit te voeren. De hele lijst met solutions werd in zijn geheel verwerkt en de BeforeCompile- and AfterCompile-targets werden maar één keer aangeroepen. Met Visual Studio Team System 2008 is dit veranderd. De BeforeCompile- en AfterCompile-targets zijn er nog steeds, maar er zijn extra targets toegevoegd. Voor en na het compileren van één enkele solution worden de BeforeCompileSolution- en AfterCompileSolution-targets aangeroepen. Dit biedt nog meer mogelijkheden om de build aan te passen en per solution extra logica toe te voegen. Denk hierbij bijvoorbeeld aan het genereren van technische documentatie met SandCastle of het maken van een MSI-package voor de solution.

```
using System;
using Microsoft.TeamFoundation.Client;
using Microsoft.TeamFoundation.Build.Client;

namespace QueueBuild
{
  class Program
  {
    static void Main(string[] args)
    {
      // Get the team foundation server.
      TeamFoundationServer tfs = TeamFoundationServerFactory.
      GetServer("http://tfs:8080");

      // Get the IBuildServer - the main point of entry to the new
      // Team Build OM.
      IBuildServer buildServer = (IBuildServer)tfs.GetService(typeof(
      IBuildServer));

      // Get the build definition for which a build is to be queued.
      IBuildDefinition definition =
      buildServer.GetBuildDefinition("TeamProj", "MyBuild");

      // Queue a build.
      buildServer.QueueBuild(definition);
    }
  }
}
```

Codevoorbeeld 1. Queue een build



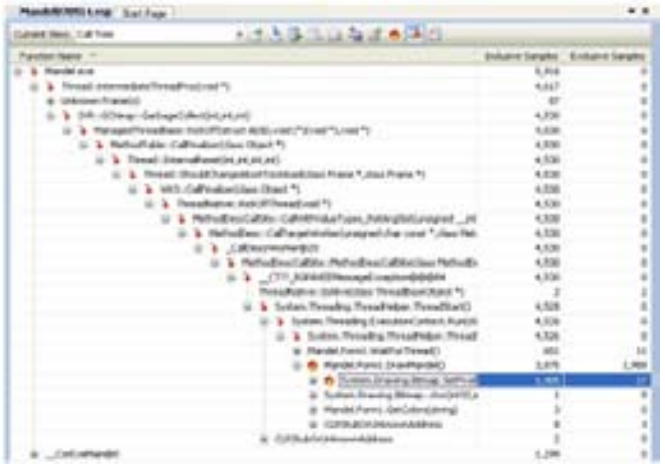
Afbeelding 4. Code metrics report

## Version Control

Naast de Automated Build kent ook Version Control een aantal interessante nieuwe features. Ook hier geldt dat Microsoft goed gekeken en geluisterd heeft naar de community. Dit blijkt als eerste doordat de Annotate-optie (in 2005 beschikbaar als losse powertool-download) nu standaard in Team Foundation Server 2008 is te integreren. Met Annotate kun je in de Source-editor in de linkerkantlijn zichtbaar laten maken welke code door wie aan de file is bijgedragen. In de volksmond is deze tool beter bekend als de 'blame tool', omdat je op deze manier kunt zien wie een bepaalde fout in code heeft geïntroduceerd. Een andere direct in de IDE geïntegreerde feature van de powertools is FolderDiff; voorheen bekend als TFS Tree Diff. FolderDiff kan gebruikt worden om te bekijken of de files die je lokaal op disk hebt staan eventueel veranderd zijn, terwijl je geen verbinding met de Team Foundation Server hebt gehad. In het difference-window krijg je dan de optie om alsnog de bestanden te synchroniseren. Gelukkig zul je hoogstwaarschijnlijk deze feature minder vaak nodig hebben dan in de 2005-versie, omdat nu ook het offline-gebruik sterk is verbeterd. Als geen verbinding met de server kan worden gemaakt, kun je blijven werken in een offline-mode. Daarbij wordt lokaal bijgehouden welke files worden veranderd en zodra je weer online bent, kunnen de veranderingen worden doorgevoerd op de server. Verder was het in Team Foundation Server 2005 niet mogelijk bestanden permanent te verwijderen uit version-control. Bestanden die verwijderd werden, werden alleen maar gemarkeerd als 'verwijderd' en waren daarmee standaard niet meer zichtbaar in de version control Explorer. Je kon echter kiezen verwijderde bestanden zichtbaar te maken, zodat deze teruggehaald konden worden. In Team Foundation Server 2008 is er nu de extra optie bijgekomen om bestanden of hele directory-structuren permanent te verwijderen uit version control. Hierbij wordt dan ook alle data definitief uit de version control-database gehaald, en is restore alleen nog mogelijk door het terugzetten van een SQL-back-up. Als laatste feature is het gedrag 'Get latest version on Check-out' als optie toegevoegd. In Team Foundation Server 2005 wordt een check-out uitgevoerd op de versie die op dat moment in de workspace zit. Dit hoeft dus niet de laatste versie te zijn. In 2008 kun je nu op de server (per Team Project) configureren dat bij een check-out altijd eerst de laatste versie wordt opgehaald. Dit is vooral een verbetering voor mensen die gewend waren met Visual Source Safe te werken, omdat ze hiermee datzelfde gedrag weer terugkrijgen.

## Team Developer Edition

De belangrijkste toevoeging die je terug kunt vinden in de Developer Edition van Team System is het kunnen laten uitvoeren van een metrics-report op een project of solution. Deze metrics hebben betrekking op de onderhoudbaarheid van de code. Het gaat hierbij om een rapport dat onder andere de statistieken berekent rondom cyclomatische complexiteit, depth of inheritance, class coupling, lines of code en een maintainabillity-index (gebaseerd op de Halstead maintainabillity); zie afbeelding 4.



Afbeelding 5. Hoth path-indicatie

Deze statistieken geven een indicatie of de code die geschreven is, goed onderhoudbaar is. De Halstead maintainability-index wordt afgeleid uit de andere statistieken. Dit getal drukt de onderhoudbaarheid van de code uit, waarbij geldt dat hoe dichter dit getal bij de 100 ligt, hoe beter onderhoudbaar de code is. Verder zijn in de Developer Edition de Profiler-tools verbeterd. Hierbij zijn voornamelijk verbeteringen aangebracht in het doorzoeken van de rapportages. Het is nu mogelijk queries te maken die je zelf kunt samenstellen op basis van performance-criteria. Verder geeft men nu visueel in het resultatenvenster weer welke code een zogenaamd 'hotpath' bevat; zie afbeelding 5. Dit is een locatie waar mogelijkerwijs (at runtime) de meeste tijd in de programmatuur wordt doorgebracht en dient als indicatie om mogelijke optimalisaties door te voeren.

Ook kun je nu een vergelijking laten uitvoeren tussen twee performancerapportages. Hierbij zie je dan wat de verbeteringen (of verslechteringen) zijn na het doorvoeren van een codeoptimalisatie. Wat helemaal geweldig is aan deze optie, is dat deze ook vanaf de command-line beschikbaar is gemaakt. De volgende command-line levert een vergelijkingsrapportage op tussen de rapportage van gisteren en die van vandaag.

*Vsperfreport /diff perfsessionYesterday.vsp perfsessionToday.vsp*

Op deze manier kun je nu voortaan een vergelijkingsrapport opnemen in de dagelijkse build en op die manier eenvoudiger detecteren wanneer het werk van de voorgaande dag een significante performance-impact heeft gehad op je product. Doordat je dan snel, nadat de code is ingecheckt, kunt achterhalen welke change-sets mogelijk dit gevolg hebben gehad, kun je ook snel er voor zorgen dat wordt gevonden welke code dit heeft veroorzaakt. De laatste verandering die ik wil noemen voor de Developer Editie is het feit dat de Unit-Test-features nu beschikbaar zijn gemaakt vanaf de Professional Editie. De Developer-editie heeft nog wel de profiling- en coverage-features die unit-testen compleet maakt, maar je kunt nu al MSTest gebruiken voor unit-testen als je een Professional Editie van Visual Studio hebt aangeschaft.

## Team Test Editie

In de Team Test Editie is ook een aantal verbeteringen doorgevoerd. Allereerst heeft men de recording-functies verbeterd, zodat nu http-verzoeken - veroorzaakt door AJAX-scripts - ook worden opgepikt. Op deze manier is de toolset veel beter geschikt om WEB 2.0-websites te kunnen testen. Verder is het nu mogelijk om van een opgenomen script een component te maken dat je in een andere test wilt hergebruiken. Dit is een aardige techniek om op basis van een aantal losse scripts een specifiek scenario te testen,

waarbij de losse testen in een vaste volgorde worden doorlopen. Denk daarbij bijvoorbeeld aan het alleen opnemen van de inlogprocedure en deze als component voortaan opnieuw te gebruiken in uitgebreidere scenario's. Een belangrijk onderdeel van Team Test is de mogelijkheid een Load-test uit te voeren. Bij het aanmaken van een Load-test heb je de beschikking over een extra optie en dat is de keuze voor een zogenaamd testmixmodel. Je kunt instellen of je de testmix wilt laten afhangen van het aantal testen dat je hebt (% testen per run), van het aantal virtual users (% users dat een bepaalde test draaien) of wil je de testmix laten bepalen door de snelheid waarmee de user interactie heeft met het systeem (% testen per tijdseenheid).

## Goede verwerking feedback

In dit artikel zijn maar een paar belangrijke features van de volgende versie van Visual Studio Team System besproken. Naast deze features is er nog een aardige waslijst als het gaat om de onderhoudbaarheid van de Team Foundation Server, de integratie met Office 2007 en Windows SharePoint Services, performanceverbeteringen, enzovoort. Ik denk dat de conclusie op zijn plaats is dat het Visual Studio Team System-team erg veel energie heeft gestopt in het verwerken van feedback op hun product en dat dit tot een aantal nieuwe set features heeft geleid. Al met al zal dit het dagelijkse werk met het product aanzienlijk versoepelen.

**Martijn Beenes** is lead developer bij Info Support voor de businessunit Finance. Martijn is betrokken bij het Technical Adoption Programma(TAP) voor Visual Studio Team System ORCAS. Naast de werkzaamheden als lead developer op het PDC van Info Support, waar de softwareontwikkelstraat Endeavour wordt ontwikkeld, geeft Martijn trainingen bij Info Support.
Martijn is te bereiken via martijnb@infosupport.com

**Marcel de Vries** is IT-Architect bij Info Support voor de businessunit Finance (www.infosupport.com). Marcel is de trekker van het Technical Adoption Programma(TAP) voor Visual Studio Team System ORCAS waar Info Support aan meewerkt. Naast het schrijven van artikelen is hij een veelgevraagde spreker op seminars en conferenties waaronder Microsoft Developer Days en de SDC. Naast zijn werkzaamheden als IT Architect bij diverse klanten geeft Marcel ook trainingen bij Info Support. Verder is hij intern Info Support-architect van de softwareontwikkelstraat Endeavour. Marcel is door Microsoft in 2006 verkozen tot MVP Team System en heeft deze award ook dit jaar kunnen prolongeren.
Marcel is te bereiken via marcelv@infosupport.com.

**Referenties**

http://blogs.infosupport.com/marcelv
http://blogs.infosupport.com/martijnb