Eric Denekamp

Monitoring

Iedere beheerder kent ze wel, de applicaties die liever niemand in beheer wil nemen omdat de applicatie niet beheerbaar is, maar toch gebruikt moeten worden. Iedere ontwikkelaar kent ze wel, die beheerders die komen klagen dat de applicatie die net opgeleverd is, niet voldoet aan de standaarden, niet beheerbaar is of om een andere reden terug over de schutting gegooid wordt. In dit artikel hoop ik deze mensen bij elkaar te brengen en in ieder geval vanuit het gezichtspunt van een beheerder aan te geven wat de verwachting is van deze beheerder. Eerst zal het probleem uitgewerkt worden vanuit het gezichtspunt van een beheerder en daarna zal ik proberen een aantal punten te bespreken die bijdragen aan een oplossing.

Het probleem

Een beheerder is de persoon waar de gebruikers komen klagen (al dan niet via een helpdesk), dat een applicatie voor hun gevoel niet goed functioneert. In zulke gevallen zal een beheerder willen gaan onderzoeken wat er aan de hand is. Dit onderzoek moet dan handen en voeten krijgen: evenementen in een log, performance counters, foutmeldingen die hout snijden. Deze informatie kan gebruikt worden om een goede inschatting te maken waar het gedrag aan zou kunnen liggen. Indien deze informatie geheel of gedeeltelijk ontbreekt, is de applicatie een zwarte doos waar de beheerder weinig mee kan. Een snel gekozen oplossing zal in dat geval ook zijn: herstart de machine maar en hoop dat het opgelost zal zijn. In een heleboel gevallen zal dit ook het geval zijn voor de kortere termijn, maar een structurele oplossing zal er in een dergelijk geval niet komen. De beheerder zal in een dergelijk geval gaan praten over een slechte applicatie en er zal weerzin ontstaan om meer van dergelijke applicaties in beheer te nemen.

Beheerders nemen geen slecht beheerbare applicaties in beheer

Jammer genoeg wordt bij het specificeren van de eisen voor een applicatie vaak wel nagedacht over de functionele specificaties en de punten vanuit een invalshoek die te maken hebben met de bedrijfsinzet van de applicatie. Een set van specificaties met betrekking tot het beheer wordt echter zelden opgenomen in de eisen voor een maatwerkapplicatie. Vaak leidt dit tot een discussie of een applicatie wel of niet (goed) in beheer kan worden genomen. Het achteraf aanpassen van een applicatie aan de eisen van de beheerders is meestal meer werk dan het in een keer tijdens het ontwikkelen van de applicatie goed beschrijven van de punten waar de beheerder om vraagt.

Oplossingen

In het algemeen zijn er drie categorieën te bedenken waarin de oplossingen te verdelen zijn. Deze drie categorieën zijn uiteindelijk een aanvulling op elkaar. Deze categorieën zijn:

- Logboeken met duidelijke events, het liefst Windows eventlogs en geen platte tekstbestanden;
- Performance-counters die applicatiespecifieke informatie geven en die een aanvulling zijn op de reeds bestaande Windows performance counters;
- Goede, 'verbose' foutmeldingen.

Logboeken

Aangezien een beheerder niet de hele dag de tijd heeft om te blijven kijken wat er allemaal gebeurt op een server, is het voor hem noodzakelijk dat de uitzonderlijke dingen die plaatsvinden op die server, vastgelegd worden in een logboek. Dit logboek kan dan in geval van een melding van vreemd gedrag geraadpleegd worden. De meldingen in dit log kunnen variëren van b.v. het geslaagd starten van een service tot een foutmelding omdat een gebruiker niet goed inlogt op de applicatie. Voor de meeste beheerders is een dergelijk logbestand genoeg om fouten binnen het systeem op te sporen en te elimineren.

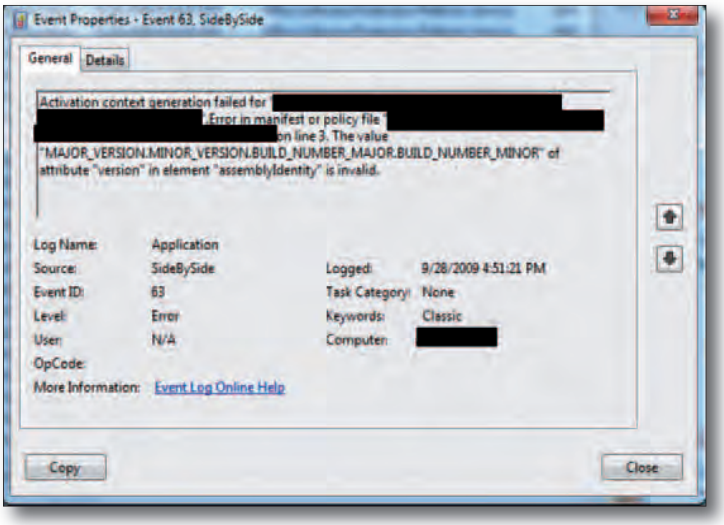

Fig. 1: Voorbeeld van een logentry die voor een beheerder zinloze informatie bevat, maar wel als foutmelding geregistreerd staat.

Een logentry hoeft natuurlijk niet een volledig boekwerk te zijn, maar informatie over de reden van de entry en de randvoorwaarden waarom deze entry bestaat is het minimum aan informatie dat erin zou moeten staan. Deze logentry wordt namelijk ergens in de applicatie gegenereerd, omdat er bepaalde randvoorwaarden bestaan. Een

Een logentry moet minimaal de reden en randvoorwaarden bevatten

melding van deze randvoorwaarden en de impact ervan maakt het voor een beheerder een stuk makkelijker om ze op te heffen.
Ook zou het voor een geautomatiseerde beheertool handig zijn als iedere foutmelding een eigen foutcode meekrijgt. Dit maakt het makkelijker om te filteren in de eventlogs, maar ook om in een vervolgstadium met geautomatiseerde tooling zoals System Center Operations Manager te monitoren op foutcodes die optreden en op basis daarvan op te treden. Dit kan dan zowel reactief als proactief.
Voor het maken van goede eventlogs, met name de Windows eventlogs, is binnen Visual Studio de namespace System.Diagnostics en System.Diagnostics.Eventing aanwezig waarmee het mogelijk is om gestandaardiseerde logs te bouwen.

Performance counters

Iedere applicatie is anders, iedere applicatie heeft andere punten waarop de performance gemeten kan worden. Threads worden gestart, queries worden gestart, threads worden afgesloten, geheugen wordt gealloceerd en weer vrijgegeven, en zo is er nog veel meer te verzinnen. Natuurlijk is het mogelijk om de algemene zaken te meten, zoals processorgebruik, algemene geheugenallocatie en netwerkgebruik, maar voor het goed beheren en inregelen van een applicatie is ook de applicatiespecifieke informatie nodig. Middels deze

Beheer verschuift dan van reactief beheer naar proactief beheer

counters kan een beheerder kijken in hoeverre bijvoorbeeld de schaling van de server goed is of dat er opgeschaald moet worden.
Ook kunnen performance counters een indicatie zijn dat er iets mis is met de server of applicatie. Hiervoor is een logische inzet van performance counters nodig op een plek waar processen, threads of andere zaken gestart en gestopt worden. Met name de invulling van een applicatie waar componenten van de applicatie met elkaar communiceren zijn plekken waar de beheerder grip op wil hebben. Zeker als het gaat om applicaties met meerdere lagen en componenten die op verschillende machines draaien, is het kunnen meten in welke component het probleem of de bottleneck zit, enorm van belang. Aangezien de beheerder niet de mogelijkheid heeft om in de applicatie te kijken wat waar wanneer hoe gebeurt, moet de applicatie deze informatie op de een of andere manier aanbieden.
Ook kan een beheerder op deze manier zijn werkterrein deels verschuiven van reactief beheer naar proactief beheer. Op het moment dat een organisatie de stap naar deze vorm van beheer kan gaan maken is er voor de gebruiker een heel groot voordeel te behalen. Symptomen van problemen kunnen in een vroeg stadium, soms nog voor de problemen zich openbaren, gesignaleerd en bestreden worden. Hierdoor is het heel goed mogelijk dat de problemen zich helemaal niet meer voordoen.
Voor het aanmaken van deze counters in Visual Studio kan ook weer gebruik gemaakt worden van System.Diagnostics en System.Diagnostics.Eventing.

Goede Foutmeldingen

In het geval van een client-applicatie is het voor een beheerder noodzakelijk om te kunnen werken met de informatie van de gebruiker. Hiervoor is het dan ook essentieel om die gebruiker goed in te lichten als er iets mis gaat en ook een mededeling te geven die de gebruiker kan doorgeven aan de beheerder. De foutmelding “An unknown error has occurred” is dan niet erg informatief. Natuurlijk is niet iedere fout te ondervangen met een uitgebreide mededeling van wat er mis is, maar enige duidelijkheid en een richting waarin de beheerder van de applicatie kan gaan zoeken zou handig zijn. In de code van de applicatie zit iets wat de foutmelding triggert en deze trigger zou omschreven kunnen worden. Een belangrijke reden dat gebruikers foutmeldingen wegklikken is dat de beheerder van hun omgeving of applicatie meestal niets met de foutmelding doet of kan.

Conclusie

Een van de dingen die een beheerder moet doen en uiteindelijk ook graag doet, is het helpen van zijn gebruikers. Hiervoor is hij natuurlijk afhankelijk van de gereedschappen die hij daarvoor beschikbaar heeft. Deze gereedschappen zijn voor een groot gedeelte de beheertools de hij krijgt bij applicaties en de besturingssystemen. Hij heeft echter ook handvaten nodig om de applicaties die hij beheert goed te kunnen behandelen. Hiervoor is hij voor een groot gedeelte aangewezen op de handvaten die de applicatie hem biedt: events die duidelijk zijn, performance counters waar hij relaties uit kan halen en goede foutmeldingen.
Een tweede stuk gereedschap dat een beheerder zou kunnen en willen inzetten, is System Center Operations Manager (SCOM), waarvoor de in dit artikel genoemde zaken als bron kunnen dienen om nog sneller en eventueel geautomatiseerd problemen te traceren en op te lossen. Ook hiervoor wordt informatie van u als ontwikkelaar gevraagd. In een volgende editie van dit blad zal uitgelegd worden hoe u de beheerders die daarom vragen, kunt bedienen met de benodigde informatie en gereedschappen. •

Eric Denekamp

Eric Denekamp is trainer/consultant bij InfoSupport. Hier is hij verantwoordelijk voor het infrastructuur curriculum en geeft hij technische trainingen op het gebied van de besturingssystemen Windows 7, windows Server 2008 R2, de System Center product groep en de Forefront productgroep. Daarnaast werkt hij ook aan de hosted omgevingen die Info Support voor zijn klanten beheert. Ook ontwerpt hij omgevingen voor klanten van Microsoft en Info Support in binnen en buitenland. Eric is een gevraagd spreker op evenementen van Microsoft.

Visual Studio 2010 TIP:

Visual Studio 2010: Call Hierarchy

Als je een methode selecteert, dan kun je de Call Hierachy opvragen ("Ctrl+K, T" of via het contextmenu). Het resultaat laat dan alle Calls To, Calls From en Overrides zien.