Nikander & Margriet Bruggeman zijn werkzaam als zelfstandige .NET- en SharePoint-consultants

Portals en webservices for remote portlets

ONTDEK DE MOGELIJKHEDEN VAN WSRP IN COMBINATIE MET SHAREPOINT 2003 EN 2007

Er zijn meer verschillende soorten webserviceprotocollen dan er smaken ijs van Ben & Jerry's zijn, en Web Services for Remote Portlets (WSRP) is daar één van. WSRP is een protocol dat een belangrijke rol in portalomgevingen kan vervullen, en als je in Microsoft-termen aan portals denkt, denk je aan SharePoint.

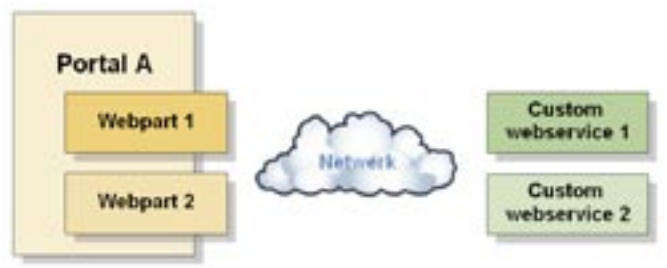
Bij SharePoint-projecten zul je waarschijnlijk nog niet veel te maken krijgen met WSRP en er zijn diverse leden van het SharePoint-productteam geweest die zich in negatieve zin over dit protocol hebben uitgelaten. Toch denken we dat WSRP een belangrijk protocol is, omdat het de mogelijkheid biedt applicaties te hergebruiken op presentatieniveau. Van Amerikaanse partners hebben we gemerkt dat er de nodige belangstelling voor dit protocol is, hoewel we zelf in Nederland nog nooit een klant er over hebben horen beginnen. In dit artikel bespreken we wat WSRP is en wat de voordelen ervan zijn. We gaan ook dieper in op de redenen waarom er bij Microsoft regelmatig enigszins gereserveerd op wordt gereageerd. Ten slotte bekijken we welke ondersteuning er binnen SharePoint 2003 en 2007 voor WSRP bestaat.

Portals proberen samenwerking en efficiëntie te stimuleren door mensen toegang te bieden tot (gestructureerde en ongestructureerde) content en applicatieservices. Gebruikers van een portal krijgen via één enkele interface toegang tot informatie, applicaties, processen en mensen: via de portal-interface. Portals zullen informatie verkrijgen uit diverse (hetzij lokaal, hetzij remote) informatiebronnen zoals databases, line of businessapplicaties, webservices, websites en contentproviders. Een portal zal deze informatie verzamelen en op pagina's tonen in de portal. Om de terminologie van WSRP te kunnen begrijpen, is het nodig de term portalpagina nader te definiëren. Een portalpagina bestaat uit een verzameling van diverse soorten informatie. Tegenwoordig bieden alle portalproducten wel de mogelijkheid individuele componenten toe te voegen aan een portalpagina. In SharePoint-termen worden dergelijke componenten webparts genoemd, in WSRP worden deze componenten portlets genoemd.
WSRP is een OASIS (Organization for the Advancement of Structured Information Standards) standaard die het gemakkelijker maakt content en interactieve webapplicaties in andere applicaties te hergebruiken. Als we 'andere applicaties' zeggen, bedoelen we eigenlijk 'portals', omdat het gecentraliseerd aanbieden van informatie typisch tot het oplossingsdomein van een portal behoort. WSRP biedt de mogelijkheid applicaties te hergebruiken op presentatieniveau door te beschrijven hoe presentatiegerichte, interactieve webservices geconsumeerd kunnen worden.

Hoe ziet een WSRP-architectuur er uit?

Om te zien hoe WSRP ingezet wordt in portals, zullen we eerst kijken naar een vaak voorkomend patroon in portals, waarbij portlets (webparts) zijn opgenomen in een portal. Deze portlets tonen remote content die wordt aangeleverd door webservices. De portlets zelf zijn verantwoordelijk voor het implementeren van de implementatielaag. Dit is te zien in afbeelding 1.

Bij deze aanpak is het noodzakelijk dat alle portlets fysiek op de portalserver geïnstalleerd worden, of, als het gaat om een serverfarm, op elke portal-web-front end. Elke portlet die ooit gebruik zal maken van de custom webservices 1 en 2 zal zijn eigen presentatielogica moeten implementeren. Als je naar kosten en tijd kijkt, is dit scenario niet optimaal.
Afbeelding 2 is vrijwel identiek aan afbeelding 1. In dit scenario wordt de businesslogica ontsloten via de custom webservices 1 en 2. Het verschil is dat er in deze architectuur twee WSRP-producers zijn toegevoegd die een algemene presentatielogicalaag bieden.
De portlets (hier WSRP consumer 1 en 2 genoemd) bevatten geen eigen presentatielogica meer, maar zijn vervangen door generieke portlets die in staat zijn om WSRP-services te consumeren. Het enige dat nu nog moet worden gedaan om de aangeboden diensten van de custom webservices 1 en 2 te consumeren, is het configureren van deze WSRP-consumer webparts.
WSRP-interfaces worden op eenduidige wijze gedefinieerd door het WSRP-protocol. Het resultaat daarvan is dat alle webservices



Afbeelding 1. Basis portlet-architectuur

Afbeelding 2. Webparts en WSRP

Afbeelding 3.Een typische WSRP-architectuur in een WSRP-compliant portal

Afbeelding 4. Een typische SharePoint WSRP-architectuur

die de WSRP-standaard implementeren naadloos aansluiten en aangeboden kunnen worden in een portalproduct dat WSRP ondersteunt. Een WSRP-compliant portal dient allereerst een generieke WSRP-consumer portlet te bevatten die in staat is om met WSRP-services te communiceren. Daarnaast kunnen portals ook bevatten. Dit zou kunnen betekenen dat bepaalde diensten in de portal via WSRP-producers aangeboden worden, maar veel meer betekent dit dat alle portlets die aan een portal toegevoegd worden ook via WSRP gepubliceerd kunnen worden, zonder dat de portlet- ontwikkelaar hier (idealiter) op enigerlei wijze rekening mee hoeft te houden. Deze WSRP-services kunnen vervolgens weer geconsumeerd worden in andere WSRP-compliant portals.

Dit levert een plaatje op dat vergelijkbaar is met afbeelding 3. Dit plaatje ziet er anders (en eenvoudiger) uit in SharePoint-omgevingen, maar daar komen we zo dadelijk op. Afbeelding 3 toont dat portal A een WSRP-producer is. Een aantal van de WSRP-services dat door portal A wordt aangeboden, wordt geconsumeerd in portal B, een WSRP-compliant portal die een generieke WSRP-consumer portlet bevat. Portal B bevat op haar beurt weer een aantal portlets dat door de WSRP-producer van portal B beschikbaar gesteld wordt aan andere portals. WSRP-consumers, zoals portal C, zijn in staat om de portlets in portal B te consumeren via WSRP-services van portal B. Portal C gebruikt weer een generieke WSRP-consumer portlet om deze diensten af te nemen. Ten slotte is het mogelijk dat user-agents zoals een browser de mark-up consumeren die door portal B wordt geleverd.

Afbeelding 3 laat duidelijk zien waarom de term 'portlet' in de titel van de specificatie zit in plaats van een algemenere naam als 'web services for presentation logic reuse' of iets dergelijks. WSRP is gemaakt met het oog op het hergebruiken van portlets in een portal op presentatielaagniveau. Hoewel het niet onmogelijk is dat WSRP in andere omgevingen gebruikt wordt, zal WSRP toch vooral toegepast worden in portalomgevingen (en dat heeft weer alles te maken met de eerder besproken eigenschap van portals om informatie te verzamelen en weer te geven).
De visie van Microsoft over portals is anders vergeleken met vendors die WSRP-producers in hun portalproducten implementeren. In de Microsoft-optiek consolideert een SharePoint-portal alle informatie op één grote centrale portalserver. Het is vanuit deze visie niet noodzakelijk om de user-interface van SharePoint-portlets via WSRP-services te hergebruiken in andere portals. Als een eindgebruiker de content of application-services wil afnemen die door deze portlets geboden wordt, dient deze naar de centrale SharePoint-portal te gaan. Je hoeft er daarom ook niet vanuit te gaan dat het concept van WSRP-producers binnen afzienbare tijd in SharePoint-technologie opgenomen zal worden.
Dit betekent overigens niet dat WSRP geen enkele rol kan spelen in SharePoint-portals. WSRP-consumers kunnen wel degelijk helpen bij het consolideren van content en applicaties in een centrale locatie. Dit resulteert in een architectuur zoals beschreven in afbeelding 4. Hierin wordt een traditionele webservice getoond die de businesslogica bevat. De WSRP-service encapsuleert de presentatielaag. WSRP-consumer-webparts in verschillende portals consumeren deze WSRP-services.

Voor- en nadelen WSRP

Grote vendors als BEA, IBM, Oracle, Sun en ook Microsoft ondersteunen de WSRP-specificatie, hoewel de populariteit van WSRP zich vooralsnog beperkt tot niet-Microsoft-producten. Enkele malen zijn vanuit het SharePoint-kamp kritische geluiden te horen geweest over WSRP. In de volgende sectie parafraseren we de belangrijkste exponent van deze kritiek.

Kritiek op WSRP door Doc Holladay

Doc Holladay werkt voor Microsoft, zat bij het SharePoint Portal Server-team en was als voting member lid van de OASIS technology council voor WSRP 1.0. Doc Holladay heeft een blogpost gewijd aan WSRP. De titel van de blogpost, "WSRP vs RSS? Game over" (http://isvchalktalk.com/archive/2005/10/12/96.aspx), laat niets aan duidelijkheid te wensen over. Doc Holladay heeft verschillende redenen waarom hij niet al te dol is op WSRP, in de rest van deze sectie zullen we ze kort bespreken:

- **Zakelijk oninteressant**
  Vanuit een zakelijk perspectief kan het voor een vendor zeker aantrekkelijk zijn een WSRP-consumer te bouwen, maar zijn er ook vendors die interesse hebben in het bouwen van een WSRP-producer? Hoe aantrekkelijk is het voor een vendor om het heel gemakkelijk te maken om applicatie-informatie te hergebruiken (op geavanceerd presentatieniveau) los van het oorspronkelijke product? Valt hier geld mee te verdienen of juist veel geld mee te verliezen?
- **Beperkte UI-ervaring**
  WSRP biedt een beperkte en gedateerde user-interactie-ervaring. WSRP 1.0 mist bijvoorbeeld het vermogen om portlet-to-portlet interactie op de client-side te ondersteunen.
- **Complexiteit**
  De WSRP-specificatie is te complex. Bij het bestuderen van het protocol zal al snel blijken dat er bij WSRP meer om de hoek komt kijken dan het teruggeven van html via een webservice.
- **Security-issues**
  Het gebruik van WSRP brengt security-issues met zich mee. Denk hierbij aan het embedden van dynamische, onbekende code afkomstig van meerdere locaties in je portalpagina.
- **Concurrentie van RSS**
  RSS is een protocol dat gemakkelijk is te gebruiken en goed is in het verzamelen en tonen van (statische, niet-interactieve) informatie. Een deel van het probleem (het hergebruiken van informatie) dat ook door WSRP opgelost kan worden, is daarmee weggenomen.

Reactie op kritiek

Het WSRP-protocol heeft heel wat meer te bieden dan het doorgeven van html via een webservice en is heel wat rijker qua functionaliteit dan RSS. WSRP maakt het gemakkelijk presentatielogica te hergebruiken en biedt een duidelijke interface voor de interactie met deze presentatielogica. Dit maakt het gemakkelijk content en application-services te hergebruiken in portals. Daarnaast biedt WSRP een oplossing voor het publiceren, vinden en binden met zulke services, voortbouwend op bestaande standaarden als XML, SOAP en WSDL. Hoewel WSRP momenteel vooral gebruikt wordt

om html via SOAP te verzenden en ontvangen, is het protocol agnostisch aangaande de gebruikte mark-up-language. In de rest van deze sectie gaan we nader in op de kritiek van Doc Holladay:

- **Zakelijk oninteressant**
  Vendors aarzelen met het aanbieden van WSRP-diensten. Wij geloven zelf ook wel dat het voor vendors minder aantrekkelijk is om WSRP-producers te bouwen dan WSRP-consumers, maar toch kun je op de OASIS-website (http://www.oasis-open.org/) een lijst met grote vendors vinden die hier mee bezig zijn. Het is voorstelbaar dat leveranciers van WSRP-services geld zullen verdienen via abonnementmodellen, maar wat je in ieder geval zult zien (zie sectie 'Hoe ziet een WSRP-architectuur er uit?') is het hergebruiken van WSRP-services in een organisatie. Portalproducten die WSRP-producers ondersteunen, maken dit zeer eenvoudig.
- **Beperkte UI-ervaring**
  De WSRP 1.0-specificatie biedt een beperkte user-interactie-ervaring. Dit heeft ook met de volwassenheid van de specificatie te maken. Het is goed om te beseffen dat de WSRP 1.0-specificatie uit augustus 2003 stamt en het mag dan ook niet verrassend heten dat de user-interactie wat gedateerd overkomt. De nieuwe versie van WSRP, versie 2.0, die begin 2007 uitkomt, adresseert bijvoorbeeld wel portlet-to-portlet-interactie.
- **Complexiteit**
  De WSRP-specificatie is complex. WSRP ondersteunt een breed scala aan scenario's wat (net zoals bij elke andere willekeurige technologie) leidt tot verhoogde complexiteit. Dat wil niet zeggen dat je als ontwikkelaar hier rechtstreeks mee te maken krijgt. De complexiteit zal typisch door een of ander framework geabstraheerd worden. Verderop in dit artikel wordt besproken hoe een WSRP-service geconsumeerd kan worden in SharePoint. Deze procedure bestaat feitelijk uit niets meer dan het specificeren van de url van de WSRP-service. Het is lastig vol te houden dat dit complex is. Het maken van een WSRP-producer is ook al niet moeilijk, diverse portalproducten (maar niet SharePoint) maken dit heel eenvoudig. Voor het Microsoft-platform is er een commercieel framework beschikbaar dat aangeboden wordt door NetUnity (zie ook sectie 'In SharePoint 2003'). Dit framework is gebaseerd op .NET (er is ook al een .NET 2.0-framework beschikbaar) en maakt het mogelijk met een paar regels code een WSRP-producer te maken. Ook in dit geval is de complexiteit ver te zoeken, dit wordt afgehandeld door het onderliggende framework.
- **Security-issues**
  Het gebruik van WSRP brengt aanvullende security-issues met zich mee. WSRP-consumers hebben weinig of geen controle over de inhoud van mark-up-fragmenten die aangeleverd worden door WSRP-producers. Onbekende code zoals client-side JavaScript kan embedded worden in WSRP-responses waardoor WSRP-consumers vatbaar zijn voor script injection attacks. Dat betekent dat een consumer veel vertrouwen moet hebben in een WSRP-producer.
  De bedenkers van de WSRP-specificatie zagen dit niet als een groot probleem, omdat de inherente beperkingen van JavaScript zelf de risico's al kleiner maken. Daarnaast is het ook mogelijk dat WSRP-producers binaire data via WSRP-responses teruggeven, maar in deze scenario's wordt het aan de consumer overgelaten om hier al of niet iets mee te doen.
  Kort samengevat is een WSRP-service blootgesteld aan dezelfde security-risico's als andere webservices. In de toekomst zal WSRP gebruik kunnen maken van webservicestandaarden als WS-Security om deze problemen op te lossen. Momenteel zal een WSRP-service die veilig moet zijn, gebruikmaken van HTTPS in combinatie met client- en servercertificaten.
- **Concurrentie van RSS**
  RSS is goed in het tonen van statische, niet-interactieve informatie. WSRP biedt daarnaast ook de mogelijkheid tot het aanbieden van zeer interactieve diensten. Helaas, en dat zullen we ook zien in de sectie 'WSRP ondersteuning in SharePoint', is de Microsoft-implementatie van een WSRP-consumer juist niet sterk in interactie. In dat geval ligt het gebruik van RSS inderdaad meer voor de hand, maar we willen wel benadrukken dat dit veroorzaakt wordt door een beperkte implementatie van een WSRP-consumer, niet door het WSRP-protocol zelf.

WSRP-interfaces

Het WSRP-protocol is te complex om in het bestek van dit artikel volledig te bespreken. We slaan daarom geavanceerdere onderwerpen in dit protocol (zoals state, portlet-modes, window-states, mark-up url's, cloning en namespace encoding) over en beperken ons tot het minimum dat nodig is om de configuratie van WSRP-consumers in SharePoint te begrijpen. De WSRP-specificatie definieert vier interfaces waar je (direct of indirect) mee te maken krijgt als je wilt communiceren met WSRP-services, te weten:

- Service Description. Deze interface biedt een consument de mogelijkheid vast te stellen welke functionaliteit wordt ondersteund door een WSRP-producer en de portlets die worden aangeboden.
- Registration. Deze interface wordt gebruikt om een WSRP-consumer te registreren of te deregistreren bij een WSRP-producer. Tijdens de registratie geeft een WSRP-producer een handle terug, die vervolgens bij elke consumer call meegegeven wordt. Het staat consumenten en producers vrij om deze relatie op elk moment te beëindigen, de consument doet dat door te deregistreren, de producer doet dat door de consument via een foutmelding op de hoogte te stellen van dit feit.
- Mark-up. Deze interface bevat operaties voor het genereren van mark-up-informatie (bijvoorbeeld html) en het interacteren met de mark-up.
- Portlet Management. Deze interface wordt gebruikt om portlets te beheren. Voorbeelden hiervan zijn het clonen en vernietigen van portlets, of het ophalen van portlet-definities.

Een WSRP-producer kan gelokaliseerd worden door discovery-mechanismen als UDDI (Universal Description, Discovery and Integration) en WSIL (Web Services Inspection Language). Codevoorbeeld 1 laat de WSDL zien van een voorbeeld WSRP-service, waarin de locaties van de services gespecificeerd worden die de vier eerder genoemde interfaces implementeren.

WSRP-ondersteuning in SharePoint

In dit gedeelte bespreken we de WSRP-ondersteuning in SharePoint die alleen bestaat uit de implementatie van WSRP-consumenten. Allereerst gaan we nader in op de WSRP-ondersteuning in SharePoint 2003, vervolgens laten we zien hoe je WSRP-services consumeert in SharePoint server 2007.

In SharePoint 2003

In deze sectie bespreken we hoe een WSRP Consumer Web Part in een SharePoint 2003-omgeving geconfigureerd en gebruikt kan worden. We maken hiervoor gebruik van de WSRP Web Part Toolkit for SharePoint Products and Technologies die op de GotDotNet-website (http://www.gotdotnet.com) gevonden kan worden. Dit webpart is geschreven in .NET 1.1, is gemaakt door het SharePoint-productteam zelf en diende als proof of concept voor het WSRP Consumer Web Part dat is opgenomen in SharePoint 2007 (deze wordt in de volgende sectie besproken).
Let op: dit webpart heeft moeite met het omgaan met interactieve WSRP-producers. We hebben niet uitgebreid getest of dit ook voor de SharePoint 2007-versie geldt, maar de resultaten die je kunt behalen met de 2003-versie van het webpart zijn beperkt tot het consumeren van diensten die statisch en informatiegericht zijn. Dit levert een ervaring op die behoorlijk vergelijkbaar is met RSS-oplossingen. Om het WSRP Consumer Web Part te kunnen gebruiken, moet de

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions targetNamespace="urn:oasis:names:tc:wsrp:v1:wsdl"
        xmlns:bind="urn:oasis:names:tc:wsrp:v1:bind"
        xmlns="http://schemas.xmlsoap.org/wsdl/
        xmlns:wsdl=http://schemas.xmlsoap.org/wsdl/
        xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">

   <import namespace="urn:oasis:names:tc:wsrp:v1:bind"
           location="wsrp_v1_bindings.wsdl"/>

   <wsdl:service name="WSRPService">
     <wsdl:port binding="bind:WSRP_v1_Markup_Binding_SOAP"
                name="WSRPBaseService">
       <soap:address location="http://my.service:8080/WSRPService"/>
     </wsdl:port>
     <wsdl:port binding="bind:WSRP_v1_ServiceDescription_Binding_SOAP"
                name="WSRPServiceDescriptionService">
       <soap:address location="http://my.service:8080/WSRPService"/>
     </wsdl:port>
     <wsdl:port binding="bind:WSRP_v1_Registration_Binding_SOAP"
                name="WSRPRegistrationService">
       <soap:address location="http://my.service:8080/WSRPService"/>
     </wsdl:port>
     <wsdl:port binding="bind:WSRP_v1_PortletManagement_Binding_SOAP"
                name="WSRPPortletManagementService">
       <soap:address location="http://my.service:8080/WSRPService"/>
     </wsdl:port>
   </wsdl:service>
</wsdl:definitions>
```

Codevoorbeeld 1.

web.config-file van SharePoint 2003 worden aangepast. In de volgende procedure wordt uitgelegd wat je moet doen:

1. Open de web.config-file van de SharePoint virtual server (die van ons is te vinden op de default-plek; [drive letter]:\inetpub\wwwroot).
2. Lokaliseer de *<SafeControls>* sectie en voeg een nieuwe safe-control toe voor de webpart consumer library:

```
<SafeControl Assembly="WSRPConsumerWebPartLibrary"
Namespace="WSRPConsumerWebPartLibrary" TypeName="*" Safe="True" />
```

3. Lokaliseer de *<httpModules>* sectie en verwijder het commentaar dat geplaatst is rond het volgende element:

```
<add name="Session" type="System.Web.SessionState.Sessi
onStateModule"/>
```

4. Lokaliseer het *<pages>* element en zet de *enableSessionState*-property op true:

```
<pages enableSessionState="true" enableViewState="true"
enableViewStateMac="true" validateRequest="false" />
```

Nu de web.config-file van SharePoint geconfigureerd is, moet het WSRP-consumer-webpart nog geïnstalleerd en toegevoegd worden. We zullen er hier van uitgaan dat je de WSRP Web Part Toolkit van de GotDotNet-website succesvol gedownload hebt. We zullen het webpart zodanig configureren dat het gebruikmaakt van een WSRP-testservice die wordt aangeboden door NetUnity (http://www.netunitysoftware.com). Deze WSRP-testservice is te vinden op de volgende locatie: http://wsrp.netunitysoftware.com:80/WSRPTestService/WSRPTestService.asmx. In de volgende procedure wordt de installatie van het WSRP-consumer-webpart uitgelegd:

1. Ga naar de WebPartLibrary/deployment-folder van de WSRP Web Part Toolkit en kopieer WSRPConsumerWebPartLibrary.dll naar de Global Assembly Cache (GAC) door deze te slepen in de [drive letter]:\WINDOWS\assembly-folder.
2. Ga naar de WebPartLibrary/deployment-folder van de WSRP Web Part Toolkit en kopieer WSRPConsumerWebPartLibrary.config in de volgende folder: [drive letter]:\inetpub\wwwroot\sites\wsrptest\wpresources\wsrpconsumerwebpartlibrary. WSRPTest is de naam van onze test SharePoint-site, dus als je een andere naam gebruikt, moet je deze naam veranderen door de naam van een eigen test SharePoint-site.
3. Open WSRPConsumerWebPartLibrary.config.
4. Lokaliseer het *<add>* element met de *MarkUpURL* key-attribute en zorg dat deze verwijst naar de test NetUnity WSRP-producer:

```
<add key="MarkUpURL" value="http://wsrp.netunitysoftware.
com:80/WSRPTestService/WSRPTestService.asmx" />
```

5. Lokaliseer het *<add>* element met de *PortletManagementURL* key-attribute en zorg dat deze verwijst naar de test NetUnity WSRP-producer:

```
<add key="PortletManagementURL" value="http://wsrp.netunitysoftware.
com:80/WSRPTestService/WSRPTestService.asmx" />
```

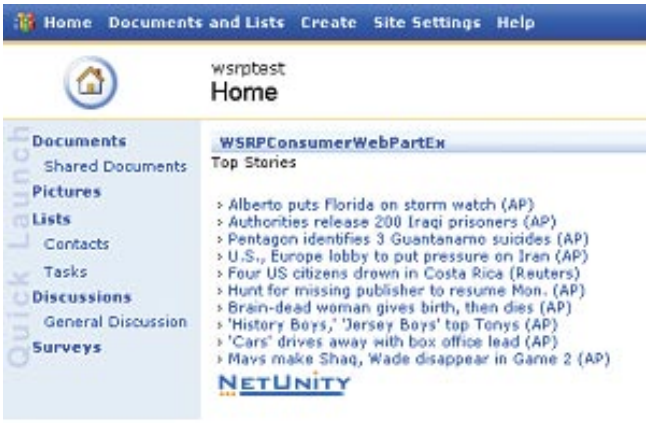
6. Lokaliseer het *<add>* element met de *RegistrationURL* key-attribute en zorg dat deze verwijst naar de test NetUnity WSRP-producer:

```
<add key="RegistrationURL" value="http://wsrp.netunity
software.com:80/WSRPTestService/WSRPTestService.asmx" />
```

7. Lokaliseer het *<add>* element met de *ServiceDescriptionURL* key-attribute en zorg dat deze verwijst naar de test NetUnity WSRP-producer:

```
<add key="ServiceDescriptionURL" value="http://wsrp.netunitysoftware.
com:80/WSRPTestService/ WSRPTestService.asmx" />
```

8. Open een command prompt en typ: iisreset.
9. Ga naar een SharePoint-site. In ons voorbeeld heet deze WSRPTest.
10. Klik Modify Shared Page > Add Web Parts > Import.
11. Ga naar de WebPartLibrary/deployment-folder van de WSRP Web Part Toolkit en selecteer WSRPConsumerWebPartEx.dwp. Klik Open.



Afbeelding 5. WSRP-consumer-webpart in SharePoint 2003.

12. Klik Upload.
13. Sleep het WSRPConsumerWebPart naar een webpart-zone.

Open nu vanuit de browser deze SharePoint-site, via *http://[server]/sites/[site name]*. Als je de SharePoint-site benadert via *http://localhost/sites/[site name]* werkt het WSRP-consumer-webpart niet. Dit is een kleine eigenaardigheid in deze implementatie van het WSRP-consumer-webpart. Omdat de source-code beschikbaar is via de GotDotNet-website kun je dit probleempje zelf oplossen als je die behoefte voelt. Het WSRP-consumer-webpart zoekt contact met de NetUnity test WSRP-producer en toont de html die door de producer aangeleverd wordt. Het resultaat hiervan is te zien in afbeelding 5.

In SharePoint 2007

SharePoint 2007 bevat out-of-the-box een WSRP-consumer die gebruikt kan worden om WSRP 1.1-services te consumeren. Jammer genoeg wordt WSRP 2.0 nog niet ondersteund. Om deze WSRP-consumer te gebruiken, moet eerst op de server ten minste één trusted WSRP-producer gedefinieerd worden. De volgende procedure legt uit hoe dat moet:

1. Ga naar de volgende locatie: [drive letter]:\Program Files\Microsoft Office Servers\12.0\Config.
2. Open het bestand TrustedWSRPProducers.xml.sample.
3. Hernoem dit bestand naar TrustedWSRPProducers.config.
4. Verander de inhoud van TrustedWSRPProducers.config zoals in codevoorbeeld 2.

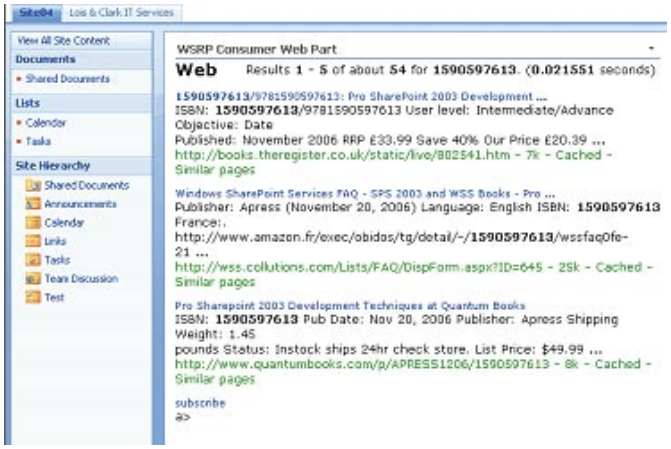
Nu is er een trusted WSRP-producer aangemaakt die verwijst naar een test-WSRP-producer van NetUnity. Vervolgens moet de WSRP-consumer toegevoegd worden aan een pagina. Dit wordt in de volgende procedure uitgelegd:

1. Ga naar een willekeurige SharePoint-pagina.
2. Kies (rechtsboven) Site Actions > Edit Page.
3. Kies een willekeurige webpart-zone en kies Add a Web Part.
4. Ga in het scherm Add Web Parts -- Web Page Dialog naar de All Web Parts-sectie en selecteer het WSRP Consumer Web Part.
5. Klik Add.

Het WSRP-consumer-webpart is nu toegevoegd aan de pagina, maar moet nog geconfigureerd worden. Dit wordt uitgelegd in de volgende procedure:

1. Kies edit > Modify shared WebPart.
2. Maak bij de Producer dropdown-list een keuze uit één van de eerder geconfigureerde producers, bijvoorbeeld Sample.
3. Kies vervolgens uit een van de aanwezige portlets die door de WSRP-producer worden aangeboden.

Afbeelding 6 laat een voorbeeld zien van een geconfigureerde WSRP-consumer-webpart in een portalpagina.



Afbeelding 6. WSRP-consumer-webpart in SharePoint 2007

```
<Configuration>
  <Producer Name="Sample" AllowScripts="true">
    <ServiceDescriptionURL>
      http://wsrp.netunitysoftware.com:80/WSRPTestService/WSRPTestSer
vice.asmx
    </ServiceDescriptionURL>
    <RegistrationURL>
      http://wsrp.netunitysoftware.com:80/WSRPTestService/WSRPTestSer
vice.asmx
    </RegistrationURL>
    <MarkupURL>
      http://wsrp.netunitysoftware.com:80/WSRPTestService/WSRPTestSer
vice.asmx
    </MarkupURL>
    <PortletManagementURL>
      http://wsrp.netunitysoftware.com:80/WSRPTestService/WSRPTestSer
vice.asmx
    </PortletManagementURL>
  </Producer>
</Configuration>
```

Codevoorbeeld 2.

Wat is nieuw in WSRP 2.0?

De huidige versie van WSRP is versie 1.1. Dit is ook de versie die in SharePoint 2007 wordt ondersteund. Ondertussen is men bezig met het definiëren van WSRP-versie 2.0. Deze wordt begin 2007 verwacht. Een document waarin de draft-versie van WSRP 2.0 beschreven wordt, is te vinden op http://www.oasis-open.org/committees/download.php/14948/wsrp-specification-2.0-draft-12.doc.

De WSRP 2.0-specificatie richt zich met name op het verbeteren van de huidige specificatie, het verbeteren van coördinatie tussen portlets, zodat WSRP-producers en -consumers ervoor zorgen dat portlets een context delen (dit is conceptueel vergelijkbaar met webpart-connecti-ons in SharePoint) en verbeteringen in portabiliteit en het beheer van resource-lifetime.

Meer informatie

Als je meer wilt lezen over de WSRP-specificatie en het maken van WSRP-producers raden we het boek 'Pro SharePoint 2003 Development Techniques' (http://www.amazon.com/Pro-SharePoint-2003-Development-Techniques/dp/1590597613/sr=11-1/qid=1164265974/ref=sr_11_1/104-8256237-7778301) aan. In dit boek worden hedendaagse ontwikkeltechnieken in combinatie met SharePoint 2003 besproken, zoals het gebruik van WSRP in portals.

Voor of tegen?

We hebben gezien wat Web Services for Remote Portlets is en in wat voor scenario's het gebruikt kan worden. We hebben bestaande kritiek op het protocol besproken en gezien hoe je WSRP-services in SharePoint 2003 en 2007 kunt consumeren. Ten slotte hebben we aandacht besteed aan de nieuwe features in WSRP 2.0. Zelf denken we dat WSRP een nuttig en krachtig protocol is, hoewel je het gebruik ervan in Microsoft-omgevingen niet vaak zult aantreffen. En of je nu een voor- of tegenstander bent van WSRP, we hopen in ieder geval dat je bij de beoordeling een duidelijk onderscheid zult maken tussen de kracht van het WSRP-protocol zelf en de beperkte Microsoft-implementatie ervan.

Nikander Bruggeman en Margriet Bruggeman

(Lois & Clark IT Services, http://www.lcbridge.nl) zijn werkzaam als zelfstandige .NET en SharePoint consultants, ontwikkelaars en architecten. Zij zijn onder andere auteurs van het boek Pro SharePoint 2003 Development Techniques.

Referenties

http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrp