Webservice api

Andreas De Lille

February 25, 2014

1 Inhoud

In dit document zal ik een aantal mogelijke uitwerkingen voor de webservice met elkaar vergelijken.

2 Eerste uitwerking

Deze eerste uitwerking heeft voor elke test een apart functie. Zo is bijvoorbeeld voor de resultaten van ping op te vragen een functie getPing() die de ping van een testbed terug geeft.

2.1 Functies

Hierbij zijn er veel verschillende functies nodig.

- getPing
- getDuration
- getTestbedName
- getTestDescription
- ..

2.2 Besluit

Hierbij heeft men per functie een pagina die specifiek voor dat deel informatie kan teruggeven. Het nadeel is wel dat er veel verschillende functies zijn die bijna hetzelfde doen. Verder hebben veel van die functies dezelfde parameters. het zou handiger zijn om deze functie meer generisch op te stellen. Een gebruiker vraagt Ook moeten uiteindelijke gebruikers altijd een andere functie aanroepen. Dit is dus allicht niet het beste idee. Bijgevolg zal ik hier niet verder op ingaan.

3 Tweede uitwerking

De tweede mogelijkheid is om meer generisch te werken. Met een functie getResult() om het resultaat van een test op te halen. We moeten deze functie natuurlijk wel meegeven van welke test we een resultaat willen. Daarnaast zou ik nog 2 functies voor zien getTest en getTestbed die info geven over respectievelijke een test en een testbed.

3.1 Functies

Deze uitwerking heeft maar 3 functies

- get data van een test opvragen.
- getTest Test opvragen.
- getTestbed Testbed opvragen.

3.1.1 Besluit

We zien hier nog altijd een verderling van functies die eigenlijk niet noodzakelijk is. Waarom zou het opvragen van een commando van een test anders moeten verlopen dan het opvragen van een resultaat? Uiteindelijk is dat ook data die verbonden is met een test. De derde methode werkt dit verder uit.

4 Derde uitwerking

Een derde mogelijkheid is een generische functie. Daaraan moeten we meegeven wat we willen krijgen.

4.1 Functies

Er zijn 2 functies een get en een put. De eerste dient om info op te vragen. De tweede functie zorgt ervoor dat we de informatie kunnen vernieuwen. De tweede functie vereist uiteraard een vorm van authenticatie.

- get
- put

4.2 Get

4.2.1 Doel

Deze functie heeft als doel om eenvoudig de databank te raadplegen. via bijvoorbeeld get/test/ping/value/1/2013-02-23 18:23:25/2013-02-23 18:35:01 Wordt de gemiddelde ping van testbed1 tussen de opgegeven periode.

4.2.2 Parameters

Hier worden de parameters besproken parameternamen tussen () zijn optioneel.

- objectType (test — testbed) Deze parameter geeft aan of men informatie wil opvragen over een test of over een testbed.
- objectName : ('testnaam' — 'testbednaam') Dit geeft aan om welk test/testbed het gaat.
- property : (value — status — name — urn — history — frequency — ..) De property die men wil terug krijgen. Als men history opgeeft zal een lijst terug gegeven worden van de waarden van die test op dat testbed.
- testbedid : ('testbedid') Dit kan een string of een getal zijn om het testbed te identificeren. Dit is niet altijd nodig, als men bijvoorbeeld de beschrijving van een test wil opvragen moet men geen testbedid meegeven omdat deze verondersteld wordt van dezelfde te zijn op verschillende testbeds. Hetzelfde kan gezegd worden voor de frequentie, tenzij we willen dat een ping test niet op elk testbed met dezelfde frequentie uitgevoerd wordt.
- (from) : Als from of till is opgegeven wordt er normaliter een gemiddelde teruggegeven. Als enkel from opgegeven is zal het gemiddelde berekend worden vanaf de fromwaarde tot het huidige waarde. Behalve als we een history opvragen dan wordt er geen gemiddelde berekend. In plaats daarvan wordt een array van waardes terug gegeven zodat men snel kan zien wanneer een fout nog is opgetreden.
- (till) : gelijkaardig aan hierboven, maar nu tot een bepaalde waarde. Als beide opgegeven zijn zal het gemiddelde van die periode gegeven worden.

4.2.3 Opmerkingen

- Wordt een type test (bijvoorbeeld een ping test) altijd met dezelfde frequentie uitgevoerd, of moet deze kunnen verschillend van testbed tot testbed? Indien deze niet moet verschillend kunnen we de frequentie bij de test opslaan. Als dit wel moet kunnen moeten we de frequentie per testbed bijhouden wat aanpassingen vereist.

- De eerste parameter objectType (test — testbed) kan eigenlijk weggelaten worden op voorwaarde dat er geen testbed en test dezelfde naam/id hebben.

4.3 Besluit

De laatste uitwerking lijkt mij de meest logische. Het is zeer intuïtief om de ping van de virtualwall op te halen met het commando: test/ping/value/virtualWall/ of als we de eerste parameter weglaten ping/value/virtualWall.

5 Conclusie

Er zijn 3 uitwerkingen. De eerste heeft per test een aparte functie. De tweede is een hybride uitwerking waarbij het opvragen van testresultaten al gegroepeerd wordt. De derde is volledig generisch. De derde lijkt mij het eenvoudiste omdat gebruikers dan gene onderscheid van getTest, getTestbed en getResult moeten onthouden. Verder kan er dan eenvoudig gelinkt worden ping/frequency geeft dan de frequentie weer terwijl ping/value de waardes bevat. Op deze manier kunnen we onze webservice beschouwen als een verzameling objecten waarvan we waarden opvragen. De eerste parameter objectType kan behouden worden om te vermijden dat de webservice onoverzichtelijk wordt door de generische opbouw.