Modularisatie in Sakai

Jos Bonsink (10172920) en Mustafa Karaalioglu (10217665), Informatica

20 januari 2013

Architectuur van Sakai

De Sakai omgeving moet aan verschillende eisen voldoen. Zo moet het systeem verschillende soorten applicaties kunnen draaien binnen de Sakai omgeving. De architectuur van Sakai is zodanig ontworpen dat het aan deze eisen voldoet. De architectuur bestaat uit verschillende lagen, deze worden kort besproken.

1. Cliënt: op de cliënt laag bevindt zich de internet browser van een gebruiker. De meerderheid van de Sakai applicaties zal hun output representeren in HTML. De browser kan dit vervolgens aan de gebruiker tonen.

2. Verzamelaar: de output van een of meerdere Sakai applicaties worden in deze laag verzameld.

3. Presenteerder: de presentatie laag creëert een fragment van de user interface waarin data van een Sakai tool wordt gerepresenteerd. Er wordt gebruik gemaakt van standaard grafische elementen zodat er een consistente Sakai beleving kan worden gegarandeerd.

4. Services: een service is een verzameling van klassen welke data beheren met vooraf bepaalde gedragsregels. Het gedrag van een service is vastgelegd in een API. Services zijn zo gemaakt dat ze modulair, herbruikbaar en beweeglijk zijn over verschillende Sakai omgevingen.

5. Systeem: de systeem laag is de server omgeving waarop Sakai draait. Denk aan web servers en databases.

De ontwikkelteams

De Sakai gemeenschap die contributies levert is grofweg op te delen in de volgende vier groepen: technische cordinatie comit(TCC), Sakai Fellows, onderhoudsteam en een stuurgroep voor de Sakai Open Academic Environment(OAE).

De TCC is een open groep waarbij alle communicaties publiekelijk toegankelijk zijn en waarbij iedereen uitgenodigd is om mee te werken. De groep bestaat grotendeels uit gevorderde leden die processen in de gemeenschap sturen en ondersteunen.

Sakai Fellows is een programma waarbij contributies van belangrijke vrijwilligers erkend en beloond worden. Het doel is om leiderschap en contributies binnen de gemeenschap te bevorderen. Elk jaar worden er zes Sakai

Fellows geselecteerd door een kleine comité bestaande uit Sakai gemeenschapsleden. De beloning is in de vorm van een kleine financiële bijdrage voor Sakai gerelateerde activiteiten.

Het onderhoudsteam waakt over de architectuur, bugs, patches en helpt bij het managen van de issue tracker en bestaat uit leden van de gemeenschap. Nieuwkomers komen meestal als eerst in contact met een lid uit dit team voor vragen of trouble-shooting.

De OAE stuurgroep is verantwoordelijk voor overzicht van het OAE project en balanceert behoeftes van meewerkende instanties met de vereisten voor een succesvolle release van de gemeenschap. De stuurgroep bestaat uit organisaties die significant resources hebben geïnvesteerd.

Gelijktijdig features toevoegen

Het ontwerp van Sakai laat makkelijk toe om verschillende tools onder één dak te brengen. Het framework staat ook toe dat meerdere tools of services binnen een web applicatie, een service implementatie in een andere web applicatie kunnen aanroepen. Dit alles maakt het eenvoudiger om met verschillende programmeurs gelijktijdig features toe te voegen.

Dit is vergelijkbaar met het ontwerp van Eclipse. De tools zijn vergelijkbaar met plugins in Eclipse en services zijn vergelijkbaar in beide producten.

Afwegingen

Modulaire architecturen minimaliseren afhankelijkheden tussen verschillende componenten en maximaliseren de interne cohesie binnen een component. Hierdoor wordt het mogelijk om individuele componenten onafhankelijk van elkaar te kunnen ontwikkelen. Het is echter niet gemakkelijk om dit voor elkaar te krijgen. Softwaretechnisch wordt het erg complex. Uiteindelijk heb je wel het voordeel dat de verdere ontwikkeling van het product makkelijker verloopt. Ook kunnen de kosten van het ondersteunen van het product worden gereduceerd. Men moet echter wel opletten een product met een te grote mate aan modulariteit te ontwikkelen. Hierdoor kan het overzicht verloren raken.

Tegenover een modulair architectuur staat een geïntegreerd architectuur. Er is veel afhankelijkheid tussen verschillende componenten waardoor ze efficiënt met elkaar kunnen samenwerken. Het nadeel is dat als je een aanpassing maakt aan een component er ook aanpassingen moeten worden gedaan

in anderen. Hierdoor verloopt het ontwikkelproces moeizamer en zal de ondersteuning van het product meer tijd kosten. Het grote voordeel is dat er een veel grotere performance kan worden behaald doordat de componenten heel hecht met elkaar samen werken. Een modulair architectuur offert performance op voor flexibiliteit.