RUWave - Client

Design- en implementatiekeuzes

Cathalijne van Wettum
Koray Yanik
Tom de Ruijter

1 juli 2010

Samenvatting

Een abstract is een verkorte versie van het hele verslag en moet kort, de volgende dingen aan de lezer duidelijk maken:

- Onderzoeksdoelen
- Onderzoeksmiddelen
- Onderzoeksresultaten
- Conclusies

Deze schrijven we pas als laatst.

Inhoudsopgave

1 Inleiding

1.1 Aanleiding

Om het uiteindelijke doel, een Wave-Client, te verwezenlijken is het nodig om een duidelijk communicatieprotocol te hebben tussen onze client in wording en een server die de Wave service aabied. In de Pilot-fase van R&D1 is deze gespecificeerd en nu kan de aandacht gericht worden op het ontwikkelen van een toepassing die deze implementeert. Om een programma zoals een client te bouwen, is een ontwerp nodig. Wat dus nog rest, is het maken van ontwerpkeuzes zodat implementatie zo soepel mogelijk verloopt en dat het product conform is met de gestelde doelen.

Ook de methode van uitwerking is nog onbepaald en dus zijn een ontwikkelvorm en programmeertaal die voldoen aan de gestelde doelen nodig.

1.2 Onderzoeksvragen

Het gespecificeerde Client-Server protocol zegt met opzet niks over de vorm die de client heeft voor de eindgebruiker, maar alleen over de informatie die de client moet sturen wanneer hij informatie wil of heeft ontvangen van de server. De huidige implementatie van Google's Wave heeft de vorm aangenomen van een webapplicatie, die een specifieke webbrowser behoeft. Meteen treed de vraag op of dit ook voor dit project een gewenste applicatievorm is.

Wat samenhangt met dit probleem, is de keuze van een programmeertaal waarin de client gebouwd gaat worden. De programmeertaal maakt bepaalde zaken makkelijker om te ontwikkelen en andere dingen erg moeilijk. Sommige talen zijn gemaakt voor het ontwikkelen van webapplicaties, terwijl anderen daar niet voor bedoeld zijn. Ook de mate van ervaring van de groepsleden met de taal speelt een rol, omdat dit het ontwikkelproces voorspoedigd. Vandaar dat de keuze van een programmeertaal veel invloed heeft op het uiteindelijke resultaat.

De Wavelet, de eenheid waarin alle informatie wordt uitgewisseld, heeft ook een representatie nodig die geschikt is voor opslag en bewerkingen. Het is nog niet bekend of de representatie van een Wavelet zoals in het protocol ook geschikt is als representatie in het programma. Vandaar dat de vraag, wat een geschikte datarepresentatie zou zijn, vooralsnog onbeantwoord.

De representatie van Wavelets voor de gebruiker, in de vorm van een Wave, is wellicht niet dezelfde als de ruwe datarepresentatie. Een geschikte visuele representatie voor de eindgebruiker is dan nodig. Zo'n representatie moet simpel en begrijpelijk zijn, zodat de gebruiker niet het overzicht verliest. Hoe deze visualisatie gerealiseerd word, is nog vrij. De enige randvoorwaarde is dat er geen informatie verloren gaat voor de eindgebruiker.

Als de client onderverdeeld wordt in stukken die allemaal hun eigen verantwoordelijkheid hebben voor het functioneren van Wave, dan is delegatie binnen de groep goed mogelijk en kan er tijd bespaard worden. Onderdelen die vertegenwoordigd moeten worden, zijn een onderdeel dat XMPP praat met de server, een extra onderdeel daarbij dat ook PubSub kan praten, een onderdeel dat de Wavelet definieert, een onderdeel dat Operational Transformation toepast en een onderdeel dat al deze dingen aan elkaar knoopt.

.... Welk onderdeel krijgt welke verantwoordelijkheid?

Naast de representatie van Waves voor de gebruiker, is de rest van het uiterlijk van de client ook nog vrij. De uiteindelijke client, die GUI-based zal zijn, moet er rekening gehouden worden in hoeverre deze interface aanpasbaar en uitbreidbaar is. Zaken als windowmanagement en toolbars kunnen zowel flexibel als star zijn, maar moeten sowieso de basisfunctionaliteiten bieden. Tot op welke hoogte is het slim om met deze dingen rekening te houden?

De opsomming van de hierboven geintroduceerde vragen is als volgt:

- Welke applicatievorm is het geschiktst voor de client gegeven onze doelen, dependent of stand-alone?
- Welke programmeertaal is voor de gekozen applicatievorm het beste, gegeven onze doelen?
- Welke representatie van Wavelet-data is het geschikst, gebasseerd op opslagefficiëntie en bewerkbaarheid?
- Welk clientonderdeel heeft verantwoordelijkheid voor..................?
- Tot op welk niveau moet de GUI aanpasbaar zijn, gegeven onze doelen?

1.3 Hypothesen

Volgens onze doelen is het wenselijk om een zo groot mogelijk draagvlak te hebben voor de client. Als het nodig is om een specifieke webbrowser te hebben om de client te laten werken, dan perken we onze gebruikersgroep in. De voorkeur gaat nu dus uit naar een standalone programma dat geen installatie behoeft.

Voor het schrijven van een standalone applicatie, zijn heel veel programmeertalen geschikt. Omdat de ervaring binnen de projectgroep beperkt is, ligt het voor de hand om voor een taal te kiezen die standaardbibliotheken bezit om zo tijdefficiënter te werken. De enige taal die daar nu voor in aanmerking komt, is Java. Daar is voldoende ervaring voor aanwezig.

Omdat alle informatie die van de server ontvangen wordt XML data is, moet er een eenheid komen die de XML data opslaat om er verder dingen mee te doen. Het makkelijkst lijkt vooralsnog iets te bouwen/gebruiken dat XML ‘begrijpt’ en van daaruit een weergave kan tonen. Dit scheelt extra onderdelen, dus tijd.

Nou, het clientdeel dat.................

De GUI moet bestaan uit een aantal vensters en werkbalken, die ieder in en uit een hoofdvenster te slepen zijn, zodat flexibiliteit aanwezig is en de eindgebruiker zelf kan beslissen hoe hij zijn werkomgeving indeelt.

2 Theoretisch kader

Het Wave protocol is een uitbreiding op het XMPP protocol en maakt daarbovenop gebruik van PubSub om alle partijen op de hoogte te houden van ontwikkelingen. Kort toegelicht: XMPP maakt gebruik van XML-stanza's om informatie tussen partijen over te brengen. XMPP vereist encryptie en authorisering, voordat vrij informatie uitgewisseld kan worden. PubSub is een extensie op XMPP, die uitgaat van Nodes waarop informatie gepubliceerd kan worden, waarop alle gebruikers die aangemeld zijn bij die node een notificatie ontvangen.

Betreffende verschillende clientvormen. Een mogelijke clientvorm zou zijn de vorm die Google heeft aangenomen bij hun Wave client, namelijk een webapplicatie die in een webbrowser draait. Het probleem met webbrowsers is dat ze erg veel van elkaar verschillen en lang niet altijd interpreteren ze code op dezelfde manier en dat je veel moeite moet doen om aan al die platforms ondersteuning te bieden. Gelukkig worden webbrowsers steeds beter. Voordeel is ook dat bijna iedereen er een heeft. Met browsers als Mozilla Firefox en Internet Explorer heb je meerendeel van de markt gedekt. Ook veel emailclients werken via een webbrowser en Wave is qua vorm enigszins verwant aan emailen.

Een andere vorm is een standalone applicatie, die na installatie met een enkel commando is opgestart en op computers met hetzelfde besturingssysteem ook hetzelfde werkt. De enige extra moeite, is dat de applicatie dan gedownload en geinstalleerd moet worden waardoor deze oplossing voor mobiele gebruikers niet direct voor de hand ligt. Wave is ook verwant te noemen aan instant messaging, waar je in veel vensters tegelijk werkt en een webbrowserapplicatie daar vaak onhandig is.

Omdat wavelet documenten doorgaans tekst met opmaak bevatten liggen een aantal representaties voor de hand. Een geschikte representatie voor tekst met opmaak is HTML. Dit zou in onze wave client vooral makkelijk te begrijpen en te implementeren zijn (HTML is simpel, en er zijn al vele mogelijke implementaties voor), ware het niet dat na nader onderzoek HTML niet geschikt voor het wave concept is. Bij HTML maakt men tekst op door het tussen tags te zetten, waarbij een tekst als ¡b¿dit¡/b¿ dik gedrukt is. Het grote probleem met dit concept is dat de opmaakregels in de tekst zelf zitten. Google heeft speciaal voor het annotation systeem gekozen, waarbij opmaakregels buiten de tekst zelf staan, om het deltasysteem te ondersteunen, waarbij het karakternummer uitmaakt. Als de opmaakregels onderdeel van de tekst zelf zijn, benvloeden zij deze karakternummers. Afwegend de makkelijkheid van het implementeren en gebruik tegen de ongeschiktheid voor opslag maakt HTML echter een heel erg geschikte tussentaal. Zo zouden we wavelet documenten uit een andere, geschiktere taal eerst kunnen omzetten naar HTML alvorens we deze op het scherm weergeven.

Applicaties kunnen op verschillende manieren ingericht worden. Denk aan een volledig scherm, waarbij vensters, tabbladen en werkbalken volledig aan te passen zijn en vrij geplaatst kunnen worden. Deze oplossing zie je bijvoorbeeld in de programmeeromgeving Eclipse en wordt ook wel aangeduidt als 'mainwindow' oplossing. Een andere vorm is alle berichten en modules in losse vensters stoppen. Dat is geschikt als je aan meerdere dingen tegelijk werkt en je voor ieder van die dingen een setje gereedschap nodig hebt, dan is dit een geschikte oplossing. Het is een minder geschikte oplossing als je aan minder dingen tegelijk werkt, dan zijn losse vensters vaak onhandig.

3 Methoden

Methoden

De keuze is gevallen op een aantal knoppen die losse schermen openen. Deze keuze is voortgekomen uit persoonlijke ervaring over overzicht en dat men zich beter kan concentreren opn het belangrijkste als er niet teveel extra dingen rond staan.

Met XMLstrings is het makkelijk opslaan, maar wordt het wel geheugeninefficiënt. Ook zou je de data nog op kunnen slaan in strings, ongeacht wat het is. De keuze is gevallen op XML omdat het gemakkelijk is in ons project en het geheugengebruik niet de eerste zorg is.

- Inventariseren vormen die client kan hebben. - Voor- en nadelen van eigenschappen van iedere vorm ten opzichte van onze doelen. - Redeneren

- Talen inventariseren die geschikt zijn voor de hierboven geschikte vorm. - Waar hebben we ervaring mee? - Voor- en nadelen van eigenschappen van iedere vorm ten opzichte van onze doelen. - Redeneren

- Verschillende vormen van Wave data representatie opsommen en voor-en nadelen noemen.

- Vormen van representatie inventariseren - Voor- en nadelen opsommen...
- Verdere aspecten van de client inventariseren.
- Verantwoordelijkheid van alle onderdelen.

. . . . . .

4 Resultaten

In 'Resultaten' presenteer je de resultaten van experiment, maar doe je er nog geen uitspraken ver. Je moet dus je best doen om bepaalde dingen weg te laten en andere dingen juist wel op te schrijven. Je geeft hier eigenlijk geen rauwe data die direct uit het experiment kwam, maar een beschrijving hiervan met tekst. Je kunt de lezer later nog verwijzen naar tabellen en figuren met resultaten waar ze zelf de data kunnen zien. Het is echter weer overbodig om beide data in zowel een tabel als een figuur te zetten, als dit geen toegevoegde waarde heeft..

5 Discussie

Voor een beschrijving, zie hieronder. Ik vind het wel gek dat ze in deze volgorde staan en niet andersom. . . Misschien nog eens teruglezen of vragen.

6 Conclusie

Bij Discussie & Conclusie trek je de conclusies van je onderzoeksvragen met betrekking tot je resultaten, je hypothesen en eventueel de hypothesen van anderen. Beargumenteer waarom de hypothesen juist/niet juist zijn. De resultaten vergelijken met geciteerde resultaten van andere onderzoeken is ook goed. Als het kan, doe dan ook uitspraken over dit onderzoek. Zeg alleen dingen die je direct kunt afleiden uit eigen onderzoeksresultaten en de onderzoeksresultaten van anderen, dus niet te hard van stapel lopen en iets algemeens zeggen, omdat hierdoor speling optreedt. Geef suggesties voor vervolgonderzoek, aan de hand van de vragen die nu in beeld komen.

Bijlagen

Projectdoelen

Bij aanvang van het project, zijn een aantal einddoelen opgesteld die gehaald moeten worden voor het slagen van het project. Deze zijn in de loop van de tijd bijgesteld en hebben nu een vorm bereikt die voorlopig niet zal veranderen:

- Google Wave client-server protocol specificaties aanvullen tot een concreet en goed implementeerbaar protocol.
- Het maken van een programma dat, in overeenstemming met het Wave protocol, geencrypte berichten naar de server kan verzenden en zich dan kan authenticeren.
- Het uitbreiden van dit programma met functionaliteiten om Waves op te halen van de server.
- Het uitbreiden van dit programma dat Waves kan weergeven.
- Het uitbreiden van dit programma zodat Waves ook bewerkt kunnen worden.

Daarbij is het bij het ontwerpen en implementeren van de client belangrijk dat de toegankelijkheid hoog blijft, zodat we een grotere doelgroep behouden. Dit houdt ook in dat we zo dicht mogelijk bij de door Google gepubliceerde specificaties willen blijven. Het is ook van belang dat er niet systeem-gebonden gewerkt wordt, omdat dit de gebruikersgroep beperkt en zo een basis leggen voor iets waar men wellicht niet op voort wil bouwen.

Open-Source te werk gaan is een doel, omdat zo alle ideeën non-commercieel blijven, innovatie gestimuleerd wordt doordat iedereen die zelf een goed idee heeft mee kan werken en zo wordt Wave verspreid over een groter publiek.

Begrippenlijst

- Blip — Een document met daarin gespreksinformatie.
- childWavelet — Er bestaat een Wavelet n en een Wavelet m, waarvoor geldt: m is een kind van n.
- Client — Representeert een participant in de participantList, die een service bij de server aanvraagt.

- Conversation Manifest — Een document met daarin de structuur van een gesprek; de structuur van blips.
- Delta — Een lijstje met n of meer operaties.
- Document — Een document bestaat uit XML elementen, die zijn opgemaakt met ranged key-value stijlelementen.
- History Hash — Een hash van een Wavelet, die in combinatie met een versienummer gebruikt wordt om synchronisatie mogelijk te maken.
- Master Server — Een server is Master Server van een Wave, als een client deze Wave bij deze Server heeft aangemaakt.
- operations — Bewerkingen die een participant op een Wavelet kan uitvoeren.
- parentWavelet — Er bestaat een Wavelet n en een Wavelet m, waarvoor geldt: n is een kind van m. Hierbij zegt de relatie ' m kind n' dat Wavelet n, Wavelet m bevat.
- participant — En gebruiker OF n groep van gebruikers, die gebruik maakt van de Wave service die een Server de participant via een client aanbiedt.
- participantList — Een lijst met daarin unieke participanten, die allen de Wavelet van de participantList mogen bekijken en eventueel bewerken.
- Server — De server biedt de Wave service aan zijn clients aan.
- Wave — Een verzameling van minstens n Wavelet.
- Wavelet — Een verzameling van documenten en n participantList. Een Wavelet is het domein van operations.
- Wave view — De verzameling van Wavelets in n Wave, waar n gebruiker toegang tot heeft.

Referenties