WHITE PAPER

MOBIL WEBBAPPLIKATIONSUTVECKLING
- EN FALLSTUDIE

Open End AB ägs av Strakt Holdings Inc. och grundades i Januari 2001. Vårt mål är att med utgångspunkt i modern organisationsteori, utveckla effektivare och mer flexibla system för informationshantering. Våra system är anpassade till de allt högre krav som ställs på informationshantering i moderna organisationer.

Open End AB
Norra Ågatan 10 A, 416 64 Göteborg
Telefon 031-749 08 80, Fax 031-749 08 81
info@openend.se, www.openend.se

BAKGRUNDEN

Webben som medium har i praktiken delat upp sig i två användningsområden. Dels de traditionella webbplatserna som funnits med från början och det som allt mer börjat kallas applikationer. Medan traditionella webbplatser närmast är interaktiva dokument med syftet att presentera information är applikationerna snarare program som utför en uppgift som traditionellt varit reserverade för de program man kör i sin skrivbordsmiljö, ett mycket typiskt exempel en e-postklient.

Tidiga webbaserade e-postklienter började lanseras för ganska länge sedan nu, men det var inte förrän webbläsarna började stödja metoder för att skicka och ta emot data från webbservrarna i bakgrunden (vad som populärt har kommit att gå under beteckningen 'AJAX') som de webbaserade applikationerna började komma ens i närheten av användarkomforten hos skrivbordsapplikationerna. Det populära genombrottet kom när Google gav sig in i branschen och lanserade sin tjänst GMail och senare också sin tjänst Google Docs. Sedan dess är Googles GMail och andra produkter måttstocken för hur en bra webbaserad applikation bör se ut.

Idag finns en uppsjö av mycket kraftfulla applikationer, och om man vill se imponerande exempel på vad som är möjligt att göra är t.ex. diagramredigeringsprogrammet Gliffy (www.gliffy.com) eller teckningsprogrammet Muro från Deviant Art (muro.deviantart.com) slående exempel.

Vad som gör detta skeende ännu intressantare är att mobiltelefonerna har blivit såpass kraftfulla att deras webbläsare nu börjar närma sig kapaciteten hos skrivbordsmiljöernas webbläsare. Vad som verkligen kom att ändra spelreglerna var i någon mån Apples iPhone, men framför allt deras iPhone av tredje generationen, som med sin 3G-kapacitet gjorde att deras användare plötsligt hade en webbläsare med full funktionalitet i fickan som mer eller mindre ständigt var ansluten till internet. Mycket snart efter detta lanserade Google sitt operativsystem för bla mobiltelefoner (Android). Sedan har också både Research in Motion med sin produktlinje Blackberry och Palm's relativt nylanserade WebOS anslutit sig till trenden.

Alla dessa leverantörer har en sak gemensamt (*): man baserar sin webbläsare på den öppenkällkodsbaserade rendreringsmotorn WebKit. Detta får en intressant implikation: webbaserade applikationer kan också bli applikationer i den nya generationen av smarta mobiltelefoner. För att återgå till exemplet Muro som nämndes ovan kan man i skrivande stund konstatera att det faktiskt fungerar på både den Nexus One och den iPhone 3GS (med iOS 4.0.2) som vi har till hands som testmaskiner. I och för sig inte med någon anpassning till deras mindre skärmstorlekar och lite i långsammaste laget på en iPhone 3GS, men att något sådant alls är möjligt säger en hel del om potentialen hos de modernaste mobila webbläsarna.

Det finns numera också en del webbapplikationer som är anpassade för de nya mobiltelefonerna i avseende på deras mycket mer begränsade skärmyta och relativt sett svagare datakraft. Som vanligt under senare tid har Google lett utvecklingen och publicerar mobilanpassade versioner av de flesta av sina reguljära webbapplikationer. Det bör inflikas att dessa webbapplikationer ofta är minst lika, i vissa fall mer kapabla än deras leverantörsspecifika (vad som oöversatt brukar kallas 'native') motsvarigheter.

Under våren 2010 blev trenden dessutom extra tydlig när ett antal ramverk med hög profil släpptes som var specifikt inriktade på att skapa gränssnitt speciellt inriktade mot moderna mobiltelefoner: jQTouch (www.jqtouch.com) och Sencha Touch (www.sencha.com) är två kända exempel, och i slutet av våren annonserade också gruppen kring det extremt populära Javascript-ramverket jQuery utvecklingen av jQuery Mobile (jquerymobile.com) sina intentioner att ge sig in i branschen.

(*): Microsoft är i skrivande stund precis på gång att försöka lansera sitt system Windows Phone 7 som inte innehåller WebKit utan deras egen rendreringsmotor, men de är alldeles för tidigt att säga vad som kommer att komma ut av det.

© Open End 2010

BAKGRUNDEN

Att stödja den nya generationen av mobiltelefoner är inte helt lätt för oss inom mjukvaru-utvecklingsbranschen. Apple's utvecklingsmodell är kvävande för utvecklare på flera sätt och under våren 2010 gjorde de med all önskvärd tydlighet klart att utvecklingsmetoder som spänner över många plattformar är något de inte vill ha i sin plattform. Deras språk (Objective-C) och deras egna ramverk är egenartade jämfört med de som varit i majoritet hittills så utveckling specifikt för denna plattform riskerar att bli väldigt inlåst i just iOS som målmiljö. Och trenden är mycket tydlig här: iPhone var först och är i dagsläget utan konkurrens störst, men det finns flera kraftfulla konkurrenter som nu tydligt vinner mark i relation till iOS-världen. För stark bindning mot iOS är alltså inte önskvärt.

Vi kom nästan omedelbart fram till beslutet att implementera det mobila stödet i Eutaxia som en mobilanpassad webbapplikation. Den moderna webbläsaren är en tillräckligt kraftfull plattform som dessutom spänner över alla mobila plattformar som vi vill stödja (och några till). Eutaxia var dessutom redan utvecklad som en webbapplikation så vi insåg också att om vi höll oss till att anpassa gränssnittet till mindre skärmar och koncentrera det till det viktigaste skulle vi kunna ha en första mobil version ute på en tillräckligt kort tid.

ATT ÖVERBRYGGA GAPET

Vi har tillverkat ett gränssnitt som ligger mycket nära gränssnittet hos Apple's iPhone. Resonemanget bakom detta är att iPhone ännu så länge är den absolut mest dominerande mobila plattformen, i alla fall i Sverige. Meningen med detta är att skapa igenkänning hos så många som möjligt. Men om den dagen kommer då mängden telefoner baserade på konkurrerande operativsystem på allvar börjar utmana mängden iOS-baserade kan det mycket väl hända att vi börjar vrida gränssnittets utseende mer åt någon form av minsta gemensam nämnare.

En sak som en webbapplikation inte tillhandahåller enkelt (ännu) är möjligheten att utnyttja telefonernas speciella hårdvara som t.ex. kamera, GPS eller accelerometer. Nu har inte detta blivit ett problem för just Eutaxia ännu, men inte heller här är en webbapplikation ett absolut hinder. Det finns nämligen system som öppnar för möjligheten att utnyttja också detta. Ett antal leverantörer tillhandahåller en form av "behållare" för webbapplikationer där applikationen ges utökad åtkomst till detta. Den mest kända av dessa är PhoneGap (www.phonegap.com). Vi har alltså inte behövt ta till detta ännu, och om vi kommer att göra det är fortfarande mycket oklart. Men PhoneGap har redan kommit att bli en ytterst viktig komponent, av helt andra anledningar som vi kommer att beskriva i nästa sektion.

SYNA DEN I SÖMMARNA

På OpenEnd har vi under de senaste åren varit noggranna med att arbeta under en strikt kvalitetskontroll vars viktigaste del är en stor mängd automatiserade tester. När vi utvecklar kör vi relevanta delar av testsviten mot det vi jobbar med, men framför allt görs det en jättekörning varje natt av samtliga tester mot samtliga plattformar vi har ett uttalat stöd för.

Att vi redan hade en rutin för testandet och en så stor mängd tester påverkade också vårt beslut att välja att implementera det mobila stödet i Eutaxia som en mobil webbapplikation. Mål nummer ett var att köra så mycket av de redan existerande testerna mot iPhone-simulatorn. Det är inte helt lätt, men vi lyckades åstadkomma en tillräckligt bra kombination. För den som är intresserad följer här en mycket nerkortad beskrivning - och lite mer tekniskt krävande - av hur det går till.

© Open End 2010

Eftersom OpenEnd har varit övervägande inriktat mot Python som programmeringsspråk/plattform körs är all serverside-kod gjord i Python, närmare bestämt med hjälp av ett testningsverktyg som tagits fram för testningen av ett annat system som OpenEnd varit mycket involverat i utvecklingen av (PyPy och dess testningsverktyg py.test, se codespeak.net/pypy och pytest.org). Vi hade alltså mycket nära kännedom om detta verktyg.

Men medan utvecklingen av skrivbordsversionen av Eutaxia pågick var testningstraditionen inom Javascript-världen fortfarande mycket svag, nästan obefintlig. Vi ville implementera automatisk testning på flera nivåer (enhets-, integrations- och funktionell) och helst av allt ville vi att det skulle ske via vårt existerande testverktyg. Verktyget py.test är gjort för att kunna utöka via insticksmoduler, så en sådan modul utvecklades som en del i arbetet med att utveckla Eutaxia, och den har sedan också kommit att renodlats som ett projekt med öppen källkod (se pypi.python.org/pypi/oejskit). Systemet startar och driver en webläsare att automatiskt ansluter till en utvecklingsserver som tillhandahåller testsviter.

Testplanen för mobilversionen var att iPhone-simulatorn skulle läggas till i de reguljära testkörningarna, på precis samma sätt som vanliga skrivbordswebbläsare gör testkörningar. Detta visade sig däremot lättare sagt än gjort, och ett ganska omfattande arbete krävdes för att detta skulle bli verklighet.

Basen för testkörningarna mot iPhone-simulatorn kom istället att bli PhoneGap (se 'Att överbrygga gapet'), en brygga mellan 'native' iPhone-applikationer och webbapplikationer. Från vår Python-kod kunde vi:

1. Modifiera ett förpreparerat PhoneGap-projekt att skicka vidaretelefonens webbläsare mot en serveraddress som kommer att tillhandahålla testsviten som skall köras.

2. Skapa ett per tillfälle modifierat AppleScript som i sin tur instruerar Apple's eget XCode att bygga om PhoneGap-projektet från steg 1 i processen och sedan starta det.

Efter dessa steg körs testsviterna på samma sätt som om det hade varit en lokal webbläsare som startats. Initierade läsare kommer att se att det här är ett långsökt sätt att göra något som borde vara ganska enkelt, men iPhone-simulatorn är på inget sätt gjord för att drivas av oövervakade automatiska processer så därför fick vi ta till den här metoden (*).

I efterhand kan vi se att möjligheten att köra testsviterna från utvecklingen av skrivbordsversionen av Eutaxia var avgörande för att vi skulle framgångsrikt färdigställa den mobila varianten i tid och det är såvitt vi ser det ytterligare en indikation på att det var lämpligt att besluta sig för att implementera mobilstödet i form av en mobil webbapplikation.

(*): Not till tekniskt extremt insatta läsare: ja, vi känner till iphonesim (github.com/jhaynie/iphonesim), men på alla våra testmaskiner är vi drabbade av 'Issue 3'/'Session could not be started' som är beskriven i deras avvikelselista ('Issues' på GitHub).

Se vidare en fråga vi ställt kring ämnet på StackOverflow: http://stackoverflow.com/questions/3658813/clean-way-to-programatically-control-safari-on-iphone.

Den ganska tunna mängden svar efter de första tre dagarna antyder att det här inte är ett särskilt lätt problem att lösa. Vi kommer definitivt gå vidare med det här systemet och ambitionen är att någon gång få bort minst XCode från kedjan.

Jacob Oscarson

Utvecklare, Teknisk skribent
Open End AB

© Open End 2010