Methodology

De grootste uitdaging voor een web 2.0 oplossing is dat de denktrend afwijkt van een traditionele website. De oplossing bestaat niet langer uit alleen informatie, pagina's of functionaliteit waar je doorheen kunt klikken, maar de weboplossing is een vervanging voor een desktopapplicatie.

Unified proces en web 2.0

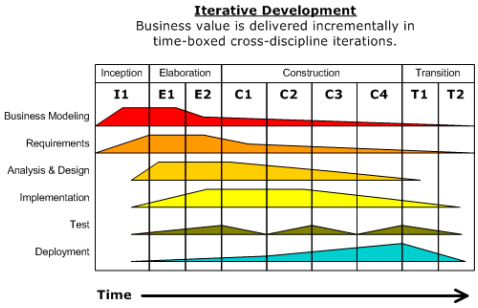
Meer gestructureerd en efficiënter werken

De toename van complexiteit bij een web 2.0 (Java) oplossing wordt veroorzaakt door de eisen betreffende de rijkere gebruikersinterface. Mijn ervaring is dat traditionele Java web frameworks en stacks onvoldoende flexibiliteit bieden om de rijkere gebruikersinterface te realiseren. Voldoen de huidige ontwikkelmethoden nog wel?

Naast zaken die eigenlijk voor alle typen applicaties gelden zoals iteratief werken, requirements vastleggen, risico's minimaliseren en technische oplossingen uitwerken zijn er voor specifieke web 2.0 toepassingen ook een aantal best practices te onderkennen.
Dit artikel gaat in op de best practices vanuit unified proces die op maat gemaakt zijn voor de efficiënte ontwikkeling van een web 2.0 applicatie die aansluit op de wensen van de opdrachtgever en de gebruikers. De best practices zijn verzameld uit een aantal web 2.0 projecten die onder mijn leiding zijn uitgevoerd.

Unified proces

Unified proces is een ontwikkelmethodiek die maakt dat projecten gestructureerd en efficiënter kunnen worden uitgevoerd. Unified proces is ontstaan uit analyse van problemen uit mislukte projecten en het verzamelen van de best practices uit projecten die zijn geslaagd.



Deze methodiek reikt een aantal handvaten aan, zoals: fase indeling van een project, workflows voor alle betrokken rollen, samenwerking tussen disciplines en een gemeenschappelijk kader en beschrijvingen van de gebruikte begrippen. Unified proces biedt oplossingen aan om requirements duidelijk en ondubbelzinnig vast te leggen, complexiteit te beperken en voldoende te testen.

Web 2.0

Rijke interactie met de gebruiker maakt dat de gebruiker het gevoel krijgt van het werken met een single screen desktopprogramma - een rich internet application (RIA) - in plaats van een traditionele paging website. Voor het ontwikkelen van een RIA zijn een aantal technologieën in opkomst. Enerzijds zijn deze op plugins gebaseerd, zoals Flash en Flex van Adobe, Silverlight van Microsoft en natuurlijk JavaFX van Sun. Anderzijds is door de komst van moderne browsers met HTML 5, CSS 3 en snelle javascript engines zonder deze plugins al een zeer rijke ervaring te bereiken.
Web 2.0 maakt dat de gebruiker veel interactiever bezig is met een website. De single screen interface kan gebruikt worden voor het makkelijk vindbaar en toegankelijk maken van informatie, bijvoorbeeld in Funda, maar ook als desktopvervanger in GMail. Daarnaast is het sociale aspect van web 2.0 belangrijk. Gebruikers worden naast consument ook producent: ze reageren op bestaande content, op elkaar en delen foto's, video's en ervaringen.

Michel van Egmond
is projectmanager bij IPROFS.

Voor bedrijven is web 2.0 een nieuwe uitdaging, omdat nu het moment is om een grote groep consumenten aan zich te binden als een community. Bij voldoende gebruikers ontstaat een momentum dat ervoor kan zorgen dat nog meer klanten de weg naar de website zullen vinden en dus gebruik gaan maken van de geboden diensten. Bij het aanbieden van vernieuwende diensten is het dus zaak om de eerste te zijn, omdat anders de community al eerder bij een andere aanbieder is ontstaan.

Best practices

Best practices en daaruit komende verfijningen die specifiek voor de ontwikkeling van een web 2.0 applicatie zijn toe te voegen aan het unified proces zijn ondermeer:

- Interactie ontwerper
- Frontend ontwikkelaar
- Usability onderzoek
- Perpetual bèta

Interactie ontwerper

De interactie ontwerper zorgt ervoor dat het schermverloop en de scherminrichting dusdanig is dat de functionaliteiten gebruiksvriendelijk te gebruiken zijn. Hij legt het schermverloop vast in de sitemap en de screenflow artefacts, en de schermindeling wordt vastgelegd in de wireframes. De use cases, interactieontwerp en het grafisch design hangen nauw samen, zodat het niet verwonderlijk is dat de requirements analist, grafisch designer en interactie ontwerper dienen samen te werken binnen de requirements management discipline.
Door de inzet van de juiste tools zoals bijvoorbeeld Swipr is er veel tijdwinst te boeken. Swipr bundelt het schermverloop, sitemap en wireframes tot een navigeerbare set van HTML-documenten en een eenvoudig prototype dat prima te gebruiken is voor communicatie met de gebruikers/opdrachtgever en het implementatieteam. Door de schermen te verdelen in componenten en van deze componenten het gedrag te documenteren, kan de gebruikersinterface compacter en dus beter onderhoudbaar worden gedocumenteerd.

Frontend ontwikkelaar

Binnen de groep van ontwikkelaars is dit de specialist die alles weet van RIA-technologieën en de eigenaardigheden van de verschillende browsers, opdat de correcte werking en opmaak bij zoveel mogelijk klanten is gewaarborgd. Daarnaast moet deze specialist weten hoe goede vindbaarheid in search engines en het verzamelen van e-commerce informatie betreffende websiteverkeer kan worden geïmplementeerd. Dit maakt het frontend specialisme een veelzijdige tak van sport.
We zien dat browsers en hun plugins zo krachtig worden dat ze van een dunne presentatielaag veranderen in een volwaardige tier van de oplossing. Best practices die serverside in Java gelden, blijken ook op te gaan in de frontend. Goede frontend ontwikkelaars zien we dus objectgeoriënteerd werken, patterns, zoals het MVC pattern toepassen en hun code regelmatig refactoren. Het gebruik van standaarden, design patterns en een stijlgids bevorderen herbruikbaarheid en leiden tot lagere ontwikkelingskosten.

Het gebruik van web 2.0 technologieën binnen Java webapplicaties is niet altijd even triviaal, omdat niet binnen alle frameworks en stacks de ondersteuning en integratie even goed geregeld is.
Er dienen door de software architect keuzes gemaakt te worden over de verantwoordelijkheid van frontend en backend, met als ene uiterste het genereren van frontend code door de backend – zoals in GWT – en aan de andere kant een volledige scheiding van frontend en backend, met het gebruik van XML of JSON services als koppelvlak.
Bij de keuze voor een specifieke stack of framework dient ook de kennis van het team bekeken te worden. Kennis van nieuwe RIA's zoals JavaFX of Flex zal nog dienen te worden opgebouwd bij de rest van het team en de frontend ontwikkelaar.

Vanuit mijn persoonlijke ervaring blijkt dat een expert inschakelen of een frontend ontwikkelaar met de specifieke kennis hiervoor zeker tijdwinst zal opleveren.
Het dient dan ook de aanbeveling om in een vroeg stadium van een project, zoals de inceptie- of eleboratiefase, een proof of concept (poc) uit te voeren. Bij de poc zal dan moeten worden aangetoond dat een vertical slide uit de architectuur oftewel de applicatie van frontend tot backend technisch goed werkt en er voldoende kennis aanwezig is bij het team.

Door de met de poc aangetoonde technische oplossing te documenteren in het software architectuur document (sad) zijn de technische oplossingen te gebruiken door het implementatieteam. Een goede architect weet de complexiteit te reduceren door de verantwoordelijkheden van de frontend en de backend te scheiden en te verduidelijken. Hierdoor zal het team productiever zijn in het realiseren van de oplossing, omdat dan vooraf duidelijk is in welke laag welke functionaliteiten thuis horen en deze dan niet in meerdere lagen dubbel of verschillend worden geïmplementeerd.

Java-ontwikkelaars zijn over het algemeen erg blij met de frontend specialist, omdat zij zich dan kunnen blijven focussen op ondermeer de backend in Java en minder op alle details die het ontwikkelen van een gebruiksvriendelijke frontend met zich meebrengt.

Bedrijven kunnen nu met web 2.0 consumenten aan zich binden als community.

Usability onderzoek
Een usability onderzoek geeft inzicht in de bruikbaarheid van een weboplossing. Een usability onderzoek is gebaseerd op het observeren van een doelgroep representatieve respondenten die een klikmodel van de webapplicatie gebruiken door daar taakopdrachten op uit te voeren. Het observeren hiervan wordt ondersteund met behulp van eyetracking. Eyetracking is een methode om het gebruikersgedrag te analyseren door middel van het volgen van de ogen.

Uit the Marketing Sherpa report excerpt: Het rode gebied is waar de gebruikers als eerste kijken en bij veranderende kleuren neemt de aandacht af. De X-en laten het clickgedrag zien en de rode lijn laat zien tot waar gebruikers scrollen.

Hiermee wordt inzicht verkregen in het kijk- en klikgedrag en daarmee uiteindelijk ook in het afwegings- en beslisgedrag van de respondenten. Hardop denken door de respondenten tijdens de uitvoering van de taakopdrachten levert ondersteunende data op die inzicht geeft in hoe zij met de mogelijkheden en geboden informatie omgaan. In combinatie met de data van de eyetracker geeft dit een goed beeld van de gebruiksvriendelijkheid en daarmee de verbeterpunten van de webapplicatie. Nadat de respondenten alle taken hebben voltooid, worden zij gevraagd een vragenlijst in te vullen waaruit een waardering voor de gebruiksvriendelijkheid van de website volgt.

Er zijn verschillende momenten in de UP fasering wanneer het handig is een usability onderzoek uit te voeren. Het eerste moment is einde eleboratie, omdat dan een eerste deel van de use cases, interactie ontwerp en grafisch design zijn uitgewerkt (iteratief) en er eventueel een clickable prototype is gerealiseerd. Het heeft de voorkeur om aan de hand van het clickable prototype een usability onderzoek te houden, omdat dit het onderzoek vereenvoudigt. Tevens heeft dit als groot voordeel dat de klant eerder een zichtbaar en concreet product getoond kan worden. Na een aantal iteraties in de constructiefase kan er ook een usability onderzoek gehouden worden op de deels werkende applicatie. Het verwerken van de dan uit het usability onderzoek voortvloeiende verbeteringen zal aanzienlijk duurder uitvallen. Het aanpassen van specificaties kost nu eenmaal minder tijd dan het aanpassen van een deels werkende applicatie.

Perpetual bèta
Om de time to market zo kort mogelijk te houden kan er besloten worden om na elke iteratie de oplossing voordat deze volledig is doorgetest 'live' te zetten, zodat deze gebruikt kan worden. De applicatie komt dan eigenlijk nooit uit het zogeheten 'bèta-stadium'. Dat de 'live' kwaliteit onder druk komt te staan door de wens zoveel mogelijk functionaliteiten in een zo kort mogelijke tijd te kunnen aanbieden en deze niet volledig door te testen is begrijpelijk. Doordat er toch frequent releases plaatsvinden, zijn kleine schoonheidsfoutjes over het algemeen snel weer weggewerkt. En in zeer kritische omgevingen blijkt uit mijn ervaring dat vaak nog steeds geldt, goed is goed genoeg.

Vaak zal de initieel gekozen of de voor dan betaalbare architectuur niet direct voldoen aan schaalbaarheids- of performance eisen bij stijgend gebruik van de webapplicatie. Dit soort uitdagingen zie je ook bij meer Agile-achtige methoden zoals Scrum en XP.
Mochten de benodigde architectuurwijzigingen erg groot worden dan is het verstandig om weer een elaboratiefase in te gaan en opnieuw een poc uit te voeren, voordat in nieuwe constructie iteraties use cases worden geïmplementeerd. Dit is geheel in lijn met het iteratieve karakter en is een wijze van risicomanagement van het unified proces, namelijk grootste risico's eerst oplossen.

Noodoplossingen zoals het bijplaatsen van veel servers of het accepteren dat de webapplicatie langzaam is als gevolg van een grote groep gebruikers hebben meestal geen lange levensduur, echter tenondergaan aan je eigen succes is een luxe probleem.
Perpetual bèta en unified proces maken een korte time to market op een goede manier mogelijk.

En zeg nou zelf wat is er leuker voor zowel klant, opdrachtgever, projectmanager en bouwer dan regelmatig aan je vrienden en familieleden te kunnen laten zien waarmee je bezig bent of wat er verbeterd is?

Conclusie
Toevoegen van best practices op het gebied van interactie ontwerp, frontend ontwikkeling en een usability onderzoek aan unified proces 2.0 maakt dat de realisatie van een web 2.0 applicatie efficiënter verloopt, de oplossing beter aansluit en als perpetual bèta mee kan groeien bij de wensen van de opdrachtgever. «

Perpetual bèta en unified proces maken een korte time to market goed mogelijk.

Links
http://en.wikipedia.org/wiki/Perpetual_beta
http://meta.wikimedia.org/wiki/Why_wikipedia_runs_slow
http://technologie.hyves.net/forum/2407031/Bl6q/Waarom_zoveel_webservers/?pageid=10HVOBLIZJPWS0OW0
http://www.marketingsherpa.com/exs/SMBGExcerpt_07.pdf
http://www.ambysoft.com/unifiedprocess/agileUP.html
http://www.swipr.com/