# Software architecture analysis of usability

Folmer, Eelke

# Chapter 12

# Samenvatting

Een van de eigenschappen van software die de laatste tien jaar meer aandacht heeft gekregen is gebruiksvriendelijkheid, beter bekend als usability. Een softwareproduct met een slechte usability is gedoemd om te falen in een concurrerende markt. Organisaties die software ontwikkelen spenderen daarom steeds meer aandacht aan om er voor te zorgen dat hun software usable is. Uit de praktijk blijkt echter dat de kwaliteit van de software (waar ook usability een onderdeel van is) niet zo hoog is als het kan zijn. Organisaties besteden een relatief groot deel van hun geld en tijd aan het repareren van usabilityproblemen tijdens de latere fases van de software-ontwikkeling. Er zijn diverse redenen waarom usabilityproblemen pas vaak laat ontdekt worden:

Het probleem met het reparereren van bepaalde usabilityproblemen tijdens de laatste fases van de software-ontwikkeling is dat sommige van de oplossingen die bedacht worden om de usability te verbeteren veranderingen aan de software architectuur vereisen. Onder een software architectuur wordt verstaan: "de fundamentele organisatie van een systeem, uitgedrukt in zijn componenten, de relaties van deze componenten tot elkaar en de omgeving, en de principes die zijn ontwerp en evolutie beheren". Een software architectuur is te vergelijken met de fundering van een huis. Het wijzigen van de software architectuur tijdens de latere fases van de software-ontwikkeling is dan ook erg duur omdat dit invloed heeft op bijna alle delen van de tot dan toe ontwikkelde software, die hierdoor vaak voor een groot deel herschreven moet worden. Deze usability verbeterende oplossingen noemen we dan architectuurgevoelig. Vanuit deze restricties komt dan ook het algemene besef voort dat de kwaliteit van software (waaronder usability) tot een zekere hoogte beperkt en bepaald worden door het ontwerp van de software architectuur.

Software architecten zijn vaak niet op de hoogte van van deze beperking en er zijn ook geen technieken voorhanden die expliciet focussen op het analyseren van software architecturen voor de ondersteuning van dit soort architectuurgevoelige kwaliteitsverbeterende oplossingen. Het meten van softwarekwaliteit gebeurt daarom vaak pas wanneer het systeem klaar is. Zeker voor usability is dit vaak het geval aangezien dit erg lastig te meten is wanneer er geen werkend systeem is. Interface prototypes kunnen maar een beperkt inzicht geven in toekomstige usability van een systeem omdat usabiltiy ook van snelheid, betrouwbaarheid afhangt. Als men tijdens deze fase tot de ontdekking komt dat bepaalde usability verbeterende oplossingen moeten worden toegevoegd heeft dit als gevolg dat sommige delen van de software opnieuw ontwikkeld moeten worden. Het herontwikkelen van software gaat gepaard met hoge kosten en omdat ook tijdens de ontwikkeling afwegingen moeten worden gemaakt, bijvoorbeeld tussen kosten en kwaliteit, leidt dit tot software systemen met een lagere usability dan mogelijk is.

Om toch op een kosteneffectieve manier systemen te ontwikkelen met hoge usability hebben wij de relatie tussen usability en software architectuur bestudeerd.

De drie hoofdresultaten van dit onderzoek zijn:

- **Architectuur-gevoelige usability patterns (ASUP)**: een usability pattern beschrijft een oplossing voor een bepaald usability probleem. Wij hebben een collectie van usability patterns geidentificeerd zoals b.v. undo (het ongedaan maken van bepaalde acties) die usability verbeteren maar die vanwege hun invloed op de architectuur erg moeilijk toe te voegen zijn tijdens de laatste fases van het ontwikkelproces. ASUP's zijn een extensie van bestaande usability patterns en we hebben voor deze patterns informatie toegevoegd over de generieke implementatie- en architectuurgevoeligheid. ASUP's kunnen worden gebruikt voor het analyseren van architecturen; wanneer de generieke implementatie bekend is kunnen software architecten bepalen wat dit betekent in hun specifieke context en of hun architectuur aangepast moet worden om deze patterns te ondersteunen.

- **Software Architectuur Usability Framework (SAUF)**: om de ASUP's te gebruiken voor het ontwerpen en analyseren van software architecturen hebben we ze beschreven in een framework die een relatie tussen software architectuur en usability beschrijft. Door middel van deze relaties kan een ontwerper op basis van specifieke usability requirements analyseren welke ASUP's mogelijk kunnen worden geïmplementeerd om te voldoen aan deze requirements. Aan de andere kant kan op basis van een aantal geïmplementeerde ASUP's worden bepaald of er voldoende ondersteuning is voor specifieke aspecten van usability.

- **Scenario gebaseerde Architectuur Analyse Methode voor Usability (SALUTA)**: Om een architect te assisteren in het analyseren van een architectuur voor usability hebben we een aantal stappen gedefinïeerd die een structuur bieden voor het begrijpen en redeneren over hoe bepaalde ontwerpbeslissingen de usability beinvloeden. SALUTA gebruikt het SAU framework voor het analyseren van de ondersteuning van usability.

Al het onderzoek in mijn proefschrift is gefinancierd door het Europese Unie gefinancierde STATUS (Software Architecture That Supports Usability) project. Het STATUS project was een samenwerking tussen verschillende industriële en academische partners in Europa (Verenigd Koninkrijk, Nederland, Griekenland en Spanje) en had als doel het bestuderen van de relatie tussen software architectuur en usability en het identificeren van karakteristieken van software architecturen die de usability van software kunnen verbeteren. Dit project heeft zich gericht op case studies van e-commerce systemen en is begonnen op 1 december 2001 en heeft 36 maanden geduurd.

Om de ontwikkelde methoden (oa. SALUTA) te valideren zijn er 3 case studies uitgevoerd in het domein van web gebaseerde applicaties.

- **Webplatform**: Een systeem wat gebouwd is voor de Rijksuniversiteit Groningen door het ECCOO (Expertise Centrum Computer Ondersteunend Onderwijs). Het Webplatform is een Content Management Systeem (CMS) dat volledig gebaseerd is op XML (extended markup language). Het bewerken van

de inhoud en de structuur van informatie verloopt via de webbrowser, onafhankelijk van tijd en plaats. Het webplatform is ontwikkeld om de kwaliteit van de informatievoorziening binnen de RUG op een hoog peil te kunnen houden.

- **eSuite**: Esuite is ontwikkelt door LogicDIS (partner STATUS) en is een system wat toegang bidet tot verschillende ERP (Enterprise Resource Planning) systemen door middle van een web interface. ERP systemen draaien in een algemeen op grote mainframe computers en bieden users vaak alleen een terminal gebaseerde interface. eSuite biedt een web interface bovenop deze verschillende ERP systemen aan zodat men deze b.v. via internet of een intranet kan benaderen.

- **Compressor**: De Compressor catalogus applicatie is ontwikkeld door de Imperial Highway Group (IHG) (partner STATUS) voor een grote client in de koel industrie. Het is een e-commerce applicatie die toegang biedt tot een grote database met informatie over compressors. Potentiele klanten kunnen b.v. door middel van dit system zoeken naar gedetaileerde informatie over verschillende compressors en compressor onderdelen.

Bij alle drie case studies is de software architectuur geanalyseerd voor de ondersteuning van usability. Er zijn verschillende interviews afgenomen met software architecten en personen verantwoordelijk voor de usability specificaties. Ook technicsche specficaties zoals architectuur beschrijvingen en documentatie van ontwerp beslissingen zijn bestudeerd. Bij iedere case study is een advies uitgebracht Iedere analyse heeft een advies uitgebracht:

- **Webplatform**: Inzicht in tot hoe ver de architectuur bepaalde usability oplossingen ondersteund.

- **eSuite**: Verbeteringen aan de software architectuur.

- **Compressor**: Een gemotiveerde keus voor een bepaalde architectuur met een hoge support voor usability terwijl met de keuze uit meerdere architecturen voorhanden had.