Chalmers tekniska högskola

Utvecklingsmetoder

- i ett småskaligt projekt

Martin Calleberg,
Vidar Eriksson,
Linus Hagvall,
Jonatan Magnusson

Läsperiod 4, vårterminen 2013
Ingenjörskompetens och kommunikation (LSP310)

Sammandrag

Flera metoder har framtagits för att underlätta skapandet av digitala applikationer. Bland andra finns analysmodell, användningsfall och sekvensdiagram. Under projektets gång har dessa tre metoderna prövats. Användningsfall är ett sätt att tydliggöra krav, analysmodellen visar strukturen och relationerna mellan klasserna och sekvensdiagram visar i detalj hur olika användningsfall ska implementeras

De metoder projektet haft störst nytta av är användningsfall samt analysmodellen. Användningsfall har hjälpt till att effektivisera tidsåtgången och analysmodellen har bidragit till en strukturerad kod. Sekvensdiagrammet tillförde inte särskilt mycket då de inte var tillräckligt bearbetade innan implementationen började

Innehållsförteckning

1. Inledning

Sedan det första datorspelet skapades har samhället exploderat av den snabbt växande TV- och datorspelsindustrin. Allt fler företag, så väl som privatpersoner, försöker etablera sig i spelbranschen. Dock är det inte lika enkelt som den allmänna uppfattningen fått det att framstå, vid komplexa program kan mindre fel snabbt växa sig stora och leda till kod som är omöjlig att korrigera. Det har därför framtagits metoder för att på ett systematiskt och effektivt sätt få fram vad som behövs för att applikationen ska fungera precis som det är tänkt, samt stödja framtida förändringar.

1.1 Bakgrund

Spelet som utvecklades under projektet har tagits fram genom att använda tre vedertagna utvecklingsmetoder. Metoderna är *användningsfall*, *analysmodell* och *sekvensdiagram*, och tas upp i de kommande kapitlen. Själva spelet är relativt litet, men koden och applikationens struktur har skapats genom att efterfölja dessa metoder.

1.2 Syfte

Syftet med denna rapport är att beskriva och redogöra för arbetsprocessen med avseende på den metodik som användes. Rapporten ska visa huruvida de metoder som använts bidragit till ett välbyggt program, samt hur användbara de varit. Detta kommer att göras genom att diskutera både hur metoderna har hjälpt till under processens och vad som kunde ha hänt utan dessa metoder.

2. Teori

Som hjälpmedel för realiseringen finns det ett antal etablerade metoder som man kan använda sig av. Bland dessa metoder finns exempelvis användningsfall som beskriver handlingar användaren kan tänkas utföra, analysmodellen som visar korrelationen mellan klasser samt sekvensdiagram vilket visar systemets arbetsflöde.

2.1 Användningsfall

Användningsfall (eng. use case) är ett enkelt sätt att beskriva krav på ett program innan det ska implementeras. Dessa ska beskriva varje handling som, utefter kraven, ska kunna genomföras i en applikation (Von Hacht, 2013). Varje användningsfall ska beskriva exakt vad som händer, vem som utför det samt vad som kan ske om en exceptionell händelse inträffar.

Representationen kan göras med hjälp av två kolumner där den första representerar *Användarens* handling och den andra *Systemets* svar på detta. Exempel på handlingar som användaren kan göra är interaktion med systemet genom tangenttryckningar, musklick eller liknande handlingar som kommer att förändra systemet. Dessa handlingar skall sedan tolkas av systemet, och beroende på dess tillstånd kommer olika svar på interaktionen ske. Se figur 1 nedan för ett exempel på hur ett användningsfall kan se ut.

Normal flow of events:

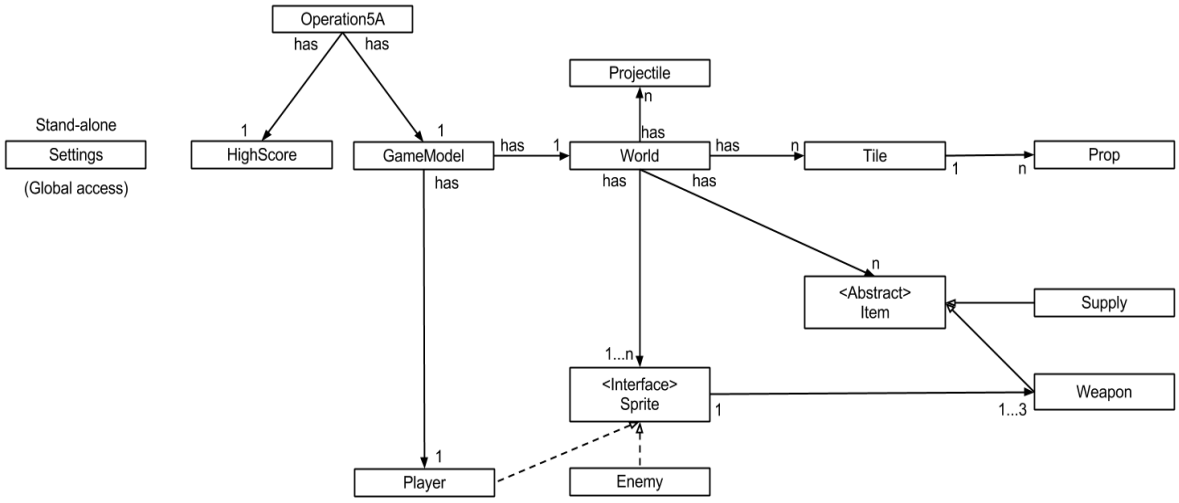
	Actor	System
1	Clicks left mouse button	
2		Create a bullet traveling the direction the player is facing. Continuously moves the bullet until it hits an enemy or a solid object. (See use case: “Enemy hit by player”).

Figur 1: Exempel på hur ett användningsfall kan se ut

Användningsfall ska vara enkla att förstå både för programmerare och personer utan kunskaper i området (Kuhn, 2005). Tack vare detta är det lätt för kunden att förstå hur applikationen kommer att se ut, och om denne har samma uppfattning av programmets funktionalitet och utseende som utvecklarna. Det blir därför lättare att skapa en dialog där utvecklarna hela tiden kan kolla verifiera kraven och se om kunden fortfarande är nöjd med hur systemet fungerar.

2.2 Analysmodellen

En analysmodell är en modell över de klasser applikationens modell är tänkt att innehålla och hur deras relation sinsemellan ser ut (Scaled Agile Framework, 2013). Detta visualiseras genom att rita ut alla klasser som är avsedda att användas och sedan dra linjer mellan dessa. Detta är ett UML-diagram (Unified Modeling Language) där linjerna symboliserar relationer mellan klasserna. Varje klass får bara förekomma en gång på diagrammet och man vill helst göra modellen så löst kopplad som möjligt genom att minimera antalet linjer mellan de olika klasserna. Se figur 2 nedan för ett exempel på hur en analys modell kan se ut.

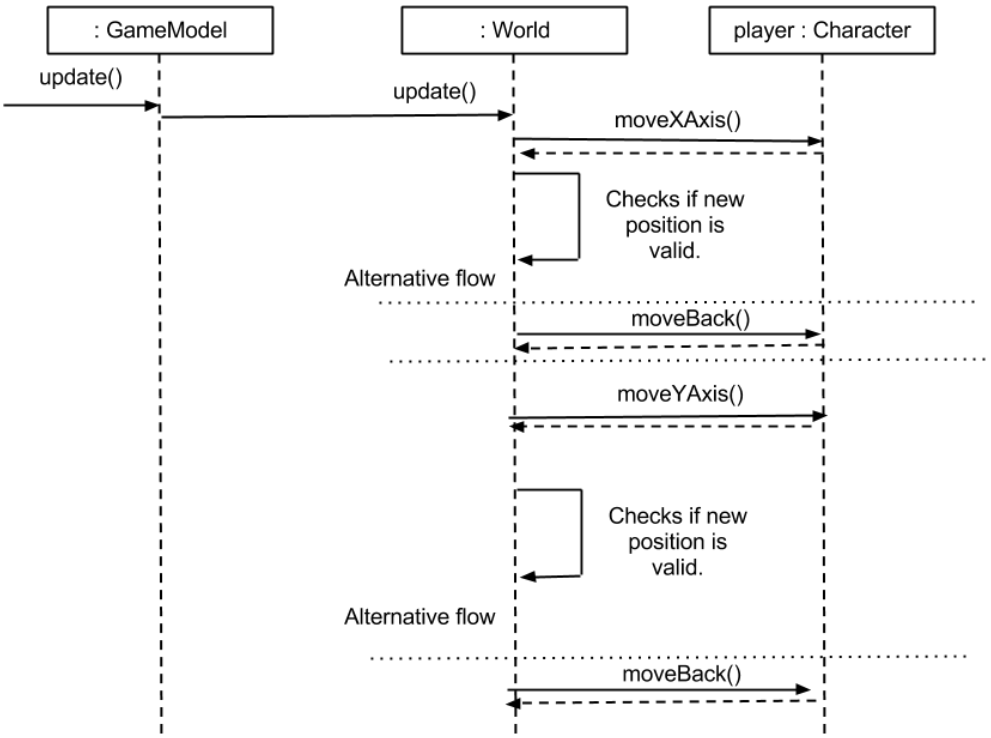


Figur 2: Ett exempel på hur en analys modell kan se ut

En analysmodell hjälper till att få struktur i ett program. Detta genom att det tydligt syns exakt hur många kopplingar (linjer) som går mellan olika delar av programmet. Genom att arbeta med att ta bort linjer, och fortfarande ha en fungerande modell, kan antalet onödiga kopplingar dras ner och göra programmet bättre ur ett objektorienterat synsätt (Scaled Agile Framework 2013). Med andra ord hjälper en väl bearbetad analysmodell till att göra ett program som är lättöverskådligt och flexibelt för framtida förändringar.

2.3 Sekvensdiagram

Ett sekvensdiagram är, till skillnad från användningsfall, inte riktad mot kunden utan fokuserar på att underlätta för utvecklarna. Ett sekvensdiagram visar hur de olika klasserna och metoderna kallar på varandra och vilket objekt som jobbar (Bell 2004). Detta visas genom att sätta ut de olika klasser som är tänkt att användas och sedan visa med pilar hur anropen och resultaten skickas mellan objekten. Se figur 3 för för ett exempel på hur ett sekvensdiagram kan se ut.



Figur 3: Exempel på hur ett sekvensdiagram kan se ut

Sekvensdiagram används för att visa interaktion mellan olika objekt i den korrekta ordningen de kallas på (Bell 2004). Sekvensdiagram skapas för att underlätta vid utarbetandet av vilka metoder och klasser som faktiskt behövs och på vilket sätt metodanrop kan kortas ner eller förenklas. Detta för att applikationen inte ska bli alltför komplex och svår att ändra i.

De kan också vara till hjälp i de fall en utvecklare behöver sätta sig in i ett program. Detta eftersom utvecklaren i fråga har diagram över de viktigaste funktionerna, och kan då snabbare sätta sig in i koden och se hur det är tänkt att fungera (Bell 2004).

Då sekvensdiagram är en fördjupning inom ett användningsfall, och även visar relationer mellan klasser så är det viktigt att både användningfallet och analysmodellen är välbearbetade innan sekvensdiagrammet skapas. Om analysmodellen förändras under utvecklingen kommer även de tillhörande sekvensdiagrammen att behöva förändras.

3. Metod

Under projektet användes de tidigare nämnda metoderna för att bidra till en välgjord applikation. Detta kapitel tar upp hur det gick när metoderna skulle användas för att börja implementera en grundläggande modell i projektet. Det kommer också tas upp hur projektet har dokumenterats under arbetsprocessen.

3.1 Implementation

Under implementationen av applikationen har teorin följts till stor del. De användningsfall som projektet använder sig av har skapats som de bör och uppdaterats under projektets gång vid behov. Både analysmodellen och sekvensdiagrammen har likaså skapats i enlighet med dess teori.

3.1.1 Användningsfall

Tidigt i projektet skapades ett antal användningsfall som gjorde kraven tydliga och välbeskrivna. Användningsfallen användes under hela projektet för att skapa en iterativ och väldefinierad process. Kontinuerliga tester har genomförts för att se till att samtliga implementerade användningsfall fungerar som tänkt.

Vissa användningsfall har ändrats under projektets gång som resultat av nya krav. Detta har också resulterat i flera ändringar på olika ställen. Eftersom allting bygger på krav och användningsfall så har dessa ändringar spridit sig i projektet. Bland annat har analysmodellen tvingats omarbetas.

3.1.2 Analysmodell

Ett antal bearbetade användningsfall över de grundläggande kraven för applikationen har använts för att skapa en analysmodell av den. Denna modell har sedan blivit basen för hur programmets data ska hanteras och på vilka sätt de olika delarna av modellen ska känna till varandra. Är användningsfallen för få eller ej tillräckligt bearbetade kan denna modell bli missvisande då delar troligtvis måste läggas till senare för att applikationen ska fungera korrekt. Därför har analysmodellen tittats över om ett användningsfall har ändrats drastiskt för att hitta eventuella problem eller nya lösningar. Se APPENDIX för en jämförelse sida vid sida av den första och nuvarande versionen av analysmodellen.

3.1.3 Sekvensdiagram

Användningsfall utvecklas till sekvensdiagram. När projektet har både användningsfall och analysmodell kan sekvensdiagram skapas av ett par, eller alla, användningsfall. Dessa har

sedan uppdaterats samtidigt som de användningsfall de är baserade på har ändrats. Under implementationen har de varit hjälpsamma då de visat några av de grundläggande metodanropen för att applikationen skulle fungera på ett korrekt sätt. Dock ändrades många av dessa anrop då nya bättre lösningar kunde hittas.

3.2 Dokumentation

Under projektets gång har det hållits regelbundna möten där det har bestämts vad som skall göras kommande vecka. Då både användningsfall och analysmodellen är metoder som på ett praktiskt sätt delar upp och strukturerar applikationen så har dessa varit viktiga vid beslut om vad som skulle implementeras till nästa möte.

Under arbetet skrevs också en RAD (Requirements and Analysis Document) och en SDD (System Design Document). Dessa dokument hanterar kraven och analysen av hela applikationen. RAD:en är skriven framförallt för kunden och personer som ej är insatta i programmering medan SDD:en är riktad till andra utvecklare. RAD:en tar upp vad som bör vara implementerat för att applikationen ska få kallas färdig och är de krav som har arbetats efter. Detta dokument kan hittas i APPENDIX.

4. Resultat

Applikation ansågs vara klar då de grundläggande kraven uppfylldes, dessa krav kan hittas i kapitel 1.3 i RAD:en, se APPENDIX. Då alla grundkrav för applikationen var implementerade kunde metoderna utvärderas. I detta kapitel kommer de tidigare nämnda metoderna tas upp för att se hur användbara de var under skapandet av applikationen.

4.1 Användningsfall

Användningen av användningsfall har skapat tydlig uppdelning och tydliga riktlinjer under implementation av applikationen. Med hjälp av användningsfall har problem kunnat konkretiseras och det blir tydligt vad som krävs för att lösa problemet. Den iterativa process som använts under framtagning av applikationen bygger till stor del på dessa användningsfall. Ett användningsfall innebär en tydligt avgränsad del av applikationen som därför på ett enkelt sätt gör det möjligt att implementera specifika delar av ett program hela vägen från modell till vy och även dela upp implementationen mellan utvecklarna. Då alla användningsfall har baserats på kraven för applikationen så har även användningsfall till stor del kunnat ersätta kraven under implementationen.

4.2 Analysmodellen

Analysmodellen har bidragit till att tidigt skapa en bild av hur det fullständiga programmet skall vara uppbyggt. Med hjälp av denna modell gick implementationen smidigare då själva basen av applikationen redan var uttänk. Detta bidrog också till att det blev enklare att få koden mer strukturerad.

4.3 Sekvensdiagram

Att rita sekvensdiagram för ett par användningsfall var till stor hjälp under utformningen av modellen. Det är dock svårt att i ett så tidigt skede bestämma hur hela modellen ska fungera

och verkligen sätta sig in i hur objekt ska kalla på varandra. I detta projekt märktes det att sekvensdiagrammen var till föga nytta då dessa inte bearbetats tillräckligt. De fungerade därför mer som en bas som kunde förändras till det bättre då förbättringar kunde göras under implementationens gång.

5. Diskussion

Då detta har varit ett förhållandevis litet projekt så har inga av metoder fått en lika framträdande roll som de skulle ha under ett större projekt. Detta projekt har till stor del varit möjligt att överblicka utan hjälp av till exempel analysmodellen, däremot i ett större projektet hade till exempel analysmodellen fått en allt viktigare roll.

Det är svårt att säga hur det hade blivit utan användningsfall eftersom de är så tätt förknippat med krav som alltid kommer att finnas i ett projekt. Då användningsfallen har använts i en iterativ process där man implementerar ett i taget så hade troligtvist implementationen utan dessa inte varit till lika stor del iterativ. Det hade blivit motsvarande flera användningsfall som skulle implementeras på samma gång vilket innebär att det blir svårare och tar längre tid innan man kan genomföra bra tester av det som implementeras. Detta hade i sin tur kommit att generera fler fel och att applikationen i sin helhet tagit längre tid att implementera.

Utan en ordentlig analysmodell innan implementationens början så hade det blivit mycket svårare att få en väl strukturerad kod. Det hade i många fall inneburit att det blev så kallad “spaghettikod” vilket innebär att förhållanden blir väldigt röriga och olika delar av applikationenen blir hårt ihopsatta. Sådan kod blir också väldigt svår att sätta sig in i och hantera.

Sekvensdiagram är den enda metoden projektet hade klarat sig lika bra utan. Då dessa diagram inte bearbetades tillräckligt mycket blev de senare under implementationen svåra att följa. Istället för att strikt följa dem kunde det hittas mycket bättre och snyggare lösningar än de otillräckliga som fanns i diagrammen. Därav hade det inte varit någon större vinst i att skriva utefter dessa, dock hade de troligtvis varit till mycket större hjälp om de hade blivit bearbetade mer från början.

6. Slutsats

Metoderna som tas upp i denna rapport har för detta projekt varit till stor nytta. Analysmodellen har varit till stor hjälp och hjälpt till att göra applikationen mer välstrukturerad och lättare att implementera. Denna modell har varit ett väldigt användbart verktyg. Utan denna är det svårt att skapa användningsfall och ännu svårare att komma igång med en grundläggande implementerad modell som man kan börja bygga ut. Analysmodellen har helt klart gjort att koden blivit mycket mer strukturerad än om ingen liknande metod använts alls.

Användningsfallen har också varit till stor hjälp, utan dessa hade målen ej varit lika tydliga och det hade blivit svårt att planera hur implementationen skulle utföras. Otydliga mål gör att delar av programmet kommer att skrivas om och tillsammans med sämre planering hade det resulterat i att applikationen hade tagit betydligt längre tid att skriva.

De sekvensdiagram som användes i det här projektet har inte varit till stor nytta. Till stor del beror det på att de inte var ordentligt genomtänkta och således inte hade en bra lösning. Det innebar att implementationen av metodanropen istället bestämdes då det var dags för själva klassen att implementeras. Dock fanns själva klasstrukturen klar vilket ändå gjorde att implementationen fortlöp förhållandevis bra under detta projektet.

Sammanfattningsvis kan man säga att användningen av analysmodell, användningsfall och sekvensdiagram kan vara till stor hjälp under utvecklingen av en applikation och blir exponentiellt viktigare när applikationen växer i storlek. Dock är det viktigt att bearbeta de olika metoderna även om applikation är liten för att de fortfarande ska vara hjälpsamma och inte göra den då lilla applikation till ett stort problem.

Källförteckning

Kuhn, B. (2005). *Why Use Use Cases?.*
http://www.carnegiequality.com/2005/11/15/why-use-use-cases/,
hämtad 2013-05-06.

Bell, D. (2004). *UML basics: The sequence diagram.* IBM 2004-02-16,
http://www.ibm.com/developerworks/rational/library/3101.html,
hämtad 2013-05-06.

Scaled Agile Framework (2013). *Domain Modeling Abstract.*
http://scaledagileframework.com/domain-modeling/,
hämtad 2013-05-06.

von Hacht, J. (2013). *Requirement Elicitation* och *Analysis.*
http://www.cse.chalmers.se/edu/course/TDA367/#lect,
hämtade 2013-05-06

Appendix

- Appendix 1 - UML klasser
- Appendix 2 - RAD