LiU

TDDI02 Programmeringsprojekt

Knattarna

Erfarenhetsrapport

Knattar:
Jonathan Möller
Patrick Klimek
Anton Sundkvist
Jacob Hedén

Handledare:
Rebecka Geijer Michaeli

December 22, 2014

Vecka -46

När man skriver en designspecifikation vill man ha ett väldigt ordnat, välstrukturerat dokument, en ritning. Det kan därför vara bra att försöka få en översiktsbild av hur dokumentet ska se ut innan man börjar skriva på det. Det var väldigt svårt att skapa en bra designspec med tanke på att ingen av oss i gruppen tidigare arbetat med Java eller Android Studio. Då vi kanske kastade oss in på att börja skriva känns resultatet något oordnat och inte helt klart.
I vårat projekt finns det mycket som måste studeras innan en bra implementation kan påbörjas. Vi måste lära oss en IDE, ett nytt programmeringsspråk samt ett API. Vi visste innan att all undersökning och informationsökning skulle ta mycket tid och försena kodningen, men såklart underskattade vi mängden vi var tvungna att lära oss, vilket leder till förseningar i planeringen. I framtida projekt kommer vi lägga mycket krut i de tidiga stadierna av projektet för att ta reda på vilka verktyg som kommer användas och hur mycket tid som måste spenderas för att lära sig dessa.
Efter Jonas föreläsningar har vi lärt oss hur en professionell Software engineering-process ser ut. Ett realistiskt programmeringsprojekt har sin grund i kundens behov. I vårt fall är vi själva kunden så vi fick ställa kraven på produkten som vi skulle utveckla. Man skulle tro att det inte behövs en kravspecifikation om vi själva vet vad vi vill göra. Men det är snarare tvärtom. Tack vare kravspecifikationen får man inramning av projektet. Egentligen är det inget nytt för oss, de flesta programmeringslabbar vi gjort har någon form av specificerade krav. Kravspecifikationen måste alltid vara verifierbar, d.v.s. man bör inte använda adjektiv som ”produkten ska vara rolig” eller ”produkten ska har en snygg design”. Istället ska man använda tekniskt avgränsade krav, som t.ex. ”Sökningen i databasen ska inte ta mer än 10 sekunder”.

Vecka 47

När man ska göra ett större system är det oundvikligt att man behöver flera team-medlemmar. Detta ställer höga krav på kommunikation och samarbete. Det första som bör göras är att synkronisera alla verktyg och att alla resurser är lättillgängliga. Vi har löst detta genom Redmine och Github. Trots detta uppstår det ibland tidskostande missförstånd. För att minska förvirringen kan man bestämma tider då gruppmedlemmarna programmerar i par. Detta halverar ju antalet inputs i systemet samtidigt som missförstånd kan redas ut direkt. Grupparbeten påverkas även av bortresta medlemmar, sjukdom osv, vilket vi drabbats av under veckan. Arbetsfördelningen kan bli lite skev i sådana lägen och om den också inte varit helt spikad kan det leda till ett förvirrat tillstånd och handlingsförlamning. Vi lyckades ändå få arbete utfört och alla medlemmar kunde bidra till projektet. Det vi lärt oss är att ett konsekvent strukturerat team gör att sådana situationer kan hanteras på ett bättre sätt och minskar förvirringen av vad som behövs göras och av vem.

Vecka 49

Eftersom vi inte hade några förkunskaper i Java/Android så har mycket tid denna vecka gått åt att lära oss mer om dessa två områden. Vi har lärt oss hur mycket tid som måste spenderas för att lära sig ett nytt språk och en ny IDE. I början av projektet så underskattade vi tiden som behövdes för detta. Vi kommer ta med oss våra nya erfarenheter till framtida projekt för att kunna göra en mer realistisk tidsbedömning. Som en följd av detta så har vi insett vi måste prioritera ska-kraven och att alla bör och eventuellt-krav inte kommer hinnas med innan deadline för vår produkt. Eftersom ungefär 5% (D. Bell 2005) av all mjukvara som utvecklas sätts i bruk kan vi förstå att generellt så behöver man mer tid på sig att utveckla en produkt känns användbar. Det är likadant i vårat fall med vår produkt.

Vecka 50

Vi har insett att vi inte kommer få med alla krav eller snarare att de kanske måste omarbetas lite i designen för att de ska fungera med android utan stora komplikationer. Det stämmer överens med det överhängande problemet vi har dragits med under projektets gång, att vi inte hade kunskap om vår utvecklingsmiljö då vi utarbetade designen. Detta har också lett till svårigheter då vi vill utöka vår funktionalitet med t.ex. databas, eftersom designen inte tagit i akt hur olika objekt (t.ex Kalender objektet som beräknar tider) fungerar och kan återskapas med databasinformation. Vad vi lärt oss är hur svårt det är att skriva en bra designspecifikation med lite förkunskaper och vilka problem det kan leda till. Det är något vi verkligen kommer ha i åtanke för kommande projekt.