LoggiLue

Grupp 2

Projektplan

V. 1.0

2017-04-21

Dokumenthistorik

Datum	Version	Beskrivning	Författare
170314	0.1	Skapandet av första versionen.	Kajsa, David, Hannes
170315	0.2	Fortsättning på första versionen.	Kajsa, David, Hannes
170315	0.3	Införandet av KronoX, API, Inkrementell Klient-Server samt Arkitektur i ordlistan och förklaring av dessa. Lagt till två nya risker och beskrivning av dessa av risk C och D.	Oscar
170316	0.4	Fortsättning på första versionen.	Kajsa, David, Hannes
170321	1.0	Fortsättning på första versionen.	Kajsa, David, Hannes
170331	1.1	Gjort ändringar efter retrospektmötet: la till domänkunskap, Scope creep & FAK i ordlistan, ändrade ansvarsområde för Kajsa och Oscar, tog bort osäkert/säkert i planeringen, la till beskrivning för grovplanen, versionsnummer i dokumenthistoriken, specificerat produktbeskrivningen, beskrivning av låg/mellan/hög i riskanalysen, skapat tydligare riskdiagram.	Kajsa
170411	1.2	Förbättring av projektplanen efter granskning. Ändring av timmar i grovplaneringen. Förklarning av Bootstrap, VoV-dokument i ordlistan, borttagning av unittestning, vecka 17 borttagning av unittestning i planeringen,	David, Hannes, Oscar, Kajsa
170412	1.3	Borttagning av "vi" genom hela dokumentet.	Kajsa
170421	2.0	Små ändringar efter retrospektmöte 2 och utifrån Kristinas kommentarer. Lagt till referens för klient-server.	Kajsa

170424	2.01	Små ändringar av stavfel och formuleringar från feedback av RS2.	David

Innehåll

Projektplan

Syfte

Syftet med detta dokument är skapa en helhetsuppfattning av projektet. Dokumentet skall ge en övergripande planering för projektet.

Ordlista

API - En förkortning av Application Programming Interface, enkelt förmedlad så hjälper det olika företag att dela med sig av data på ett hanterbart sätt.

Arkitektur - inom mjukvaruutveckling syftas detta på en övergripande modell av hur systemet ska fungera och se ut.

Inkrementell - är en benämning på en sorts metod att jobba inom mjukvaruutveckling, man delar upp arbetet i inkrement. Varje inkrement är en del av en liten portion av systemet som man jobbar på.

Klient-server - en sorts arkitektur inom mjukvaruutveckling, anses vara den mest dominanta då den används i dem flesta av dagens system. Består av en server som tillhandahåller tjänster till klienten när dessa efterfrågas av klienten.

KronoX - Är ett övergripande system som används för schemaläggning såväl som bokningar för lokaler och dylikt, anpassat för användning av högskolor och universitet.

It's learning - Webbplats för studenter och lärare.

MySQL - Databasplattform som används för att skapa databaser.

YouTrack - Webbplats för att logga tid i projekt.

Procedurmanual - Stilguide och verktygs förklaring. Vilka procedurer som kommer ingå i projektet.

Domänkunskap – kunskap som hämtas av specialister och experter för att utveckla domänkunskap.

FAK – Ett krav för systemet. Se kravspecifikation för detaljer.

Scope creep – De ändringarna och nya idéer som kan uppstå efterhand som ett projekt växer. Uppkommer lätt om kraven inte är ordentligt specificerade.

Bootstrap – Ett CSS-ramverk för design av webbsidor.

Vov-dokument - Verifiering och valideringsdokument

Referenser

[1] IT-ord, klient-server (https://it-ord.idg.se/ord/klient-server/) (hämtad 2017-04-21)

Översikt av projekt

Syfte

Syftet med projektet är att skapa en produkt som underlättar för studenter när de vill dela anteckningar från föreläsningar eller liknande eftersom det inte finns något bra alternativ till detta just nu. T.ex. om en student skulle vara sjuk och missa en föreläsning kan denna gå in och ta del av anteckningar från kurskamrater.

Omfattning

Projektet kommer fokusera på att skapa ett system som är användarvänligt och kommer därför att lägga mycket tid på design. Det kommer inte att designas egna komponenter på sidan utan gruppen kommer använda olika ramverk för att skapa utformningen och designen av sidan. Ramverk som Bootstrap, där mycket CSS redan är förinställt och klart, kommer att användas. Produkten kommer främst att anpassas till större skärmar som datorer och små skärmar i andra hand. Koden kommer främst skrivas från grunden av oss själva, men det kan förekomma bitar av inkopierad kod. Projektet kommer slutligen att testas och driftsättas på en webbserverit.

Mål

Målet efter projektets slut är att ha en hemsida där man kan logga in, ladda upp anteckningar och se anteckningar med hjälp av kategorisering via ett schema. Detta genom att använda en användarvänlig design.

Produkt

Produktbeskrivning

På webbsidan ska användaren med namn och lösenord kunna logga in på sitt konto. Kontot ska kopplas till det program användaren studerar. Därifrån ska man kunna välja vilken kurs- och i sin tur vilken föreläsning eller liknande händelse på schemat som är av intresse. Till en börja kommer fokuset att ligga på att endast få systemet att fungera för studenter som läser till informationsarkitekter. Om budgeten tillåter kommer fler program att implementeras.
Användaren ska där kunna se tillhörande inlägg från händelsen. Inläggen är i form av anteckningar, bilder och annat studiematerial man fått med sig från föreläsningen.
Schemat ska vara kopplat till KronoX API och uppdateras efter det. Dagar som varit kommer sparas på webbsidan så att man kan gå tillbaka i schemat för gamla föreläsningar och dessa motsvarande anteckningar.

Målgrupp

Målgruppen för LoggiLue kommer att vara studenter på Malmö högskola som läser programmet informationsarkitekt. Studenten ser att man delar med sig av sina anteckningar från föreläsningar och liknande samt har något att vinna på att läsa andras anteckningar. Studenten har bra teknisk kunskap men hemsidan kommer vara fokuserad på användbarhet ändå för att tilltala så många användare som möjligt.

Process

Utvecklingsprocess

Gruppen kommer arbeta inkrementellt och iterativt. Detta innebär att gruppen kommer att genomföra flera olika versioner av produkten, den första med minsta möjliga funktionalitet och sen byggs det på mer funktionalitet med varje version.

Även kravinsamlingen kommer att ske iterativ och största del via domänkunskap, även genom att observera en del på liknande system som Facebook och It's Learning.

Genom att genomföra en kravanalys kommer kraven att prioriteras och på så sätt se till så att det inte finns några krav som krockar med varandra.

Arkitekturen som har valts är Klient-server eftersom webbsidan kommer behöva en databas för att spara och hämta information [1].

Koden kommer att skrivas med bestämda standarder så att alla skriver på ett enhetligt sätt och så att alla förstår koden. Koden skrivs individuellt men alla gruppmedlemmar måste förstå all kod, därav standarderna. Kod-filer ska organiseras logiskt i en struktur som underlättar att förstå hur delar av koden relaterar till varandra. Detaljer över hur koden skall skrivas finns i dokumentet "Coding_Guidelines_LoggiLue"

Eftersom den största källan till inhämtning av data för design och användargränssnitt kommer ske via domänkunskap, kommer gruppen främst att vända sig till sig själva. Därefter analyseras resultatet av detta och därmed skapas design av användargränssnittet.

Implementationen kommer att ske stegvis efterhand som nya funktioner skapas. Testning kommer att ske under hela tiden som koden skrivs av författaren. Innan alla delar implementeras kommer medlematna att testa varandras funktioner för att upptäcka kodblindhet. Sedan implementeras alla delar och ett integrationstest genomförs för alla delar tillsammans. När systemet anses vara klart kommer ett systemtest att utföras för testa hela systemet. På så sätt valideras produkten.

Bemanning och ansvarsområden

Kajsa Araskoug: Grundläggande kunskaper inom HTML 5 & CSS 3 och Python 3. Ansvarar för dokumentation.

Hannes Linnér: Grundläggande kunskaper i HTML 5 och CSS 3. Har även jobbat med databaser och kopplingar till en back-end. Grundläggande kunskap i Python 3. Ansvarar över databasen.

David Hurtig: Grundläggande kunskap inom HTML 5, CSS 3, databaser och Python 3. Tar ansvar för programmeringen.

Oscar Kaczmarek: Grundläggande kunskaper i HTML 5, CSS 3 och Python 3. Ansvarig för design.

Planering

Budgeten för antalet timmar är 220 timmar per person. Då kraven är prioriterade kommer gruppen börja med att fokusera på att bygga ett system som gör det mest grundläggande - logga in, ladda upp och se anteckningar. Därefter byggs funktionerna på baserat på tiden och utifrån kraven. Timmarna är angivna i per/person.

Vecka 10-12 (sprint 1)

- Projektplan: 4-5h. Sammanställa planering över projektet.
- Kravdokument: 4h. Dokument med projektets krav.
- Gruppkontrakt: 1h. Kontrakt med förhållningsregler mellan projektmedlemmarna.
- Skapa en demoversion med fungerande kod: 5h

Vecka 13-15 (sprint 2)

- Produkt med delmängd krav specificerade: 20h
- Första version av verifiering- & valideringsdokument: 15h
- Genomföra tester enligt verifiering- & valideringsdokumentet + dokumentation: 10h
- Kodgranskning och en dokumentgranskning: 15h
- Designdokument: 4h. Projektets övergripande arkitektur.

Vecka 16-18 (sprint 3)

- Första version av produkten som helhet: 20h
- Alla krav prioriterade som ”ska”/”must” är implementerade och testade enligt verifiering- och validerings-dokument: 10h
- Tester ska ha genomförts för de implementerade kraven enligt testfall i verifiering- och validerings-dokument och dessa tester ska ha dokumenterats: 20h
- Användbarhetsanalyser, användbarhetstester och white-box-testning ska ha genomförts minst en gång: 20h

Vecka 19-22 (sprint 4)

- Problem som noterats tidigare ska ha åtgärdats: 20h
- Vissa av de krav som prioriterats som ”bör”/”should” ska ha implementerats: 30h
- Regressionstester ska ha genomförts för förändringar som skett: 25h
- Ny implementation ska ha testats minst en gång: 20h
- Som minst ska en andra omgång av granskningar och tester som faller under individuell fördjupning ha genomförts: 30h
- För analyser så ska en kompletterande rapport ha gjort som adresserar hur problem som pekats ut i den första analysen åtgärdats: 20h

Grovplan

Timmarna är angivna i per/person och är något i underkanten då planen är i grova drag. Detta eftersom oväntade saker kan uppstå och för att inte spräcka budgeten på 220h timmar är det bra att ha utrymme att jobba med.

Vecka 10

- Möte 10 timmar: Jobba med dokumentation
- Möte med handledare Kristian 2 timmar: Introduktion

Vecka 11

- Möte 10 timmar: Jobba med dokumentation
- Möte med handledare Kristian 0.5 timmar: Introduktion

Vecka 12

Aktiviteter

- Styrelsemöte 2 timmar: Gå igenom agenda på vad som har gjorts tidigare och vad som ska göras i framtiden.
- Möte med handledare Kristian 0.5 timmar
- Struktur av databas: 4 timmar.
- Få upp en laborationsmiljö 8 timmar: Få upp en miljö som gruppen kan börja arbeta i.
- Skapa en hemsida 1 timmar: Skapas lokalt.

Vecka 13

Aktiviteter

- Styrelsemöte 2 timmar: Gå igenom agenda på vad som har gjorts tidigare och vad som ska göras i framtiden.
- Möte med handledare Kristian 0.5 timmar
- Påbörja krav FAK-1 (se kravdokument) 4 timmar
- Påbörja krav FAK-2 (se kravdokument) 4 timmar
- Designdokument klart 5 timmar: Dokumentets första version ska vara i sin färdiga form.

Vecka 14

Aktiviteter

- Styrelsemöte 2 timmar: Gå igenom agenda på vad som har gjorts tidigare och vad som ska göras i framtiden.
- Möte med handledare Kristian 0.5 timmar
- Fortsättning med krav FAK-1 (se kravdokument) 5 timmar
- Fortsättning med krav FAK-2 (se kravdokument) 5 timmar
- Börja med VoV-dokumentet 5 timmar: Skrivs efter hand som funktionerna byggs.

Vecka 15

Aktiviteter

- Styrelsemöte 2 timmar: Gå igenom agenda på vad som har gjorts tidigare och vad som ska göras i framtiden.
- Möte med handledare Kristian 0.5 timmar
- Avsluta & testa krav FAK-1 (se kravdokument) 5 timmar: Funktionen ska vara klar för testning
- Avsluta & testa krav FAK-2 (se kravdokument) 5 timmar: Funktionen ska vara klar för testning
- Avsluta VoV-dokumentet 2 timmar: Ska var i sin färdiga form
- Kodgranskning & dokumentgranskning 4 timmar: Delar av individuell fördjupning.

Vecka 16
Aktiviteter

- Styrelsemöte 2 timmar: Gå igenom agenda på vad som har gjorts tidigare och vad som ska göras i framtiden.
- Möte med handledare Kristian 0.5 timmar
- Implementera nya krav 15 timmar.

Vecka 17
Aktiviteter

- Styrelsemöte 2 timmar: Gå igenom agenda på vad som har gjorts tidigare och vad som ska göras i framtiden.
- Möte med handledare Kristian 0.5 timmar
- Tester, användartester ska genomföras på färdiga funktioner: 8 timmar.
- Implementera krav ca 10 timmar.

Vecka 18
Aktiviteter

- Styrelsemöte 2 timmar: Gå igenom agenda på vad som har gjorts tidigare och vad som ska göras i framtiden.
- Möte med handledare Kristian 0.5 timmar
- Åtgärda fel 10 timmar: Åtgärda fel som uppkom under testerna
- Användbarhetsanalys 5 timmar

Vecka 19
Aktiviteter

- Styrelsemöte 2 timmar: Gå igenom agenda på vad som har gjorts tidigare och vad som ska göras i framtiden.
- Möte med handledare Kristian 0.5 timmar
- Implementera krav ca 15 timmar.

Vecka 20
Aktiviteter

- Styrelsemöte 2 timmar: Gå igenom agenda på vad som har gjorts tidigare och vad som ska göras i framtiden.
- Möte med handledare Kristian 0.5 timmar
- Fortsätta implementera krav 15 timmar.

Vecka 21
Aktiviteter

- Styrelsemöte 2 timmar: Gå igenom agenda på vad som har gjorts tidigare och vad som ska göras i framtiden.
- Möte med handledare Kristian 0.5 timmar
- Testa implementation 5 timmar: Testa så att allt fungerar efter implementationen.

- Regressionstester 10 timmar: Testar nya funktionaliteten av systemet och även systemet som helhet.
- Projektet färdigställs 5 timmar: Kontrollerar att allt är klart

Vecka 22
Aktiviteter

- Styrelsemöte 2 timmar: Gå igenom agenda på vad som har gjorts tidigare och vad som ska göras i framtiden.
- Möte med handledare Kristian 0.5 timmar
- Individuell fördjupningsrapport 10 timmar: Färdig för inlämning på fredag

Milstolpar

Demo av produkten 2017-03-24
En statisk enkel webbsida att kunna visa upp.
Leverabler:

- Presentation av projektet.
- Ramverk implementerat.

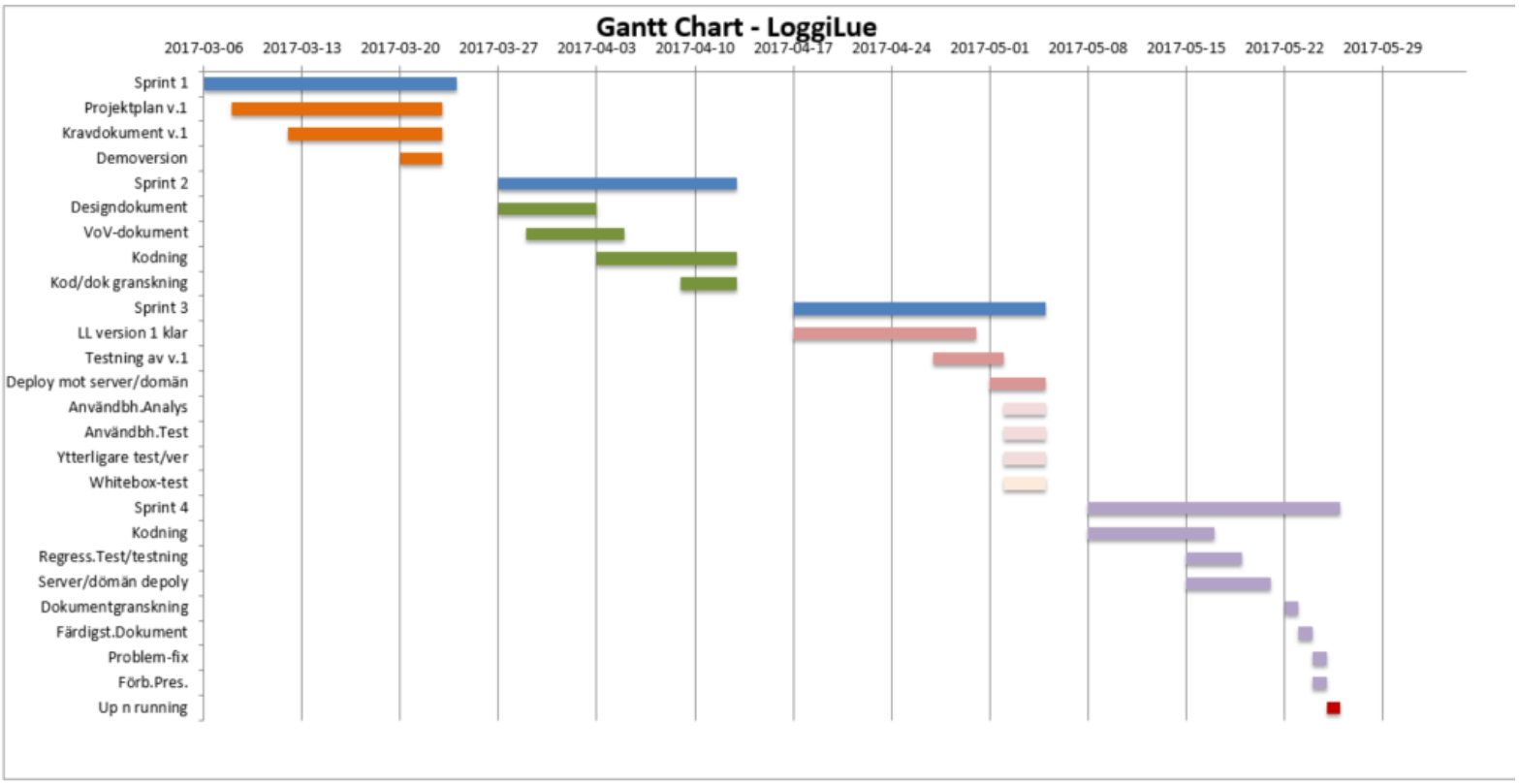
Databas 2017-03-26
En databas som är tillgänglig.
Leverabler:

- Kopplad databas till webbsidan.
- Upplagt på pythonanywhere/webbserver

Första version av produkten 2017-04-30
Det skall finnas en första version av produkten som helhet.
Leverabler:

- Grundläggande prioriterade krav skall finnas på plats och funka.

Gantt-schema



Sprint
Individuella uppgifter

Task Name	Start	End	Duration (days)
Sprint 1	2017-03-06	2017-03-24	18
Projektplan v.1	2017-03-08	2017-03-23	15
Kravdokument v.1	2017-03-12	2017-03-23	11
Demoversion	2017-03-20	2017-03-23	3
Sprint 2	2017-03-27	2017-04-13	17
Designdokument	2017-03-27	2017-04-03	7
VoV-dokument	2017-03-29	2017-04-05	7
Kodning	2017-04-03	2017-04-13	10
Kod/dok granskning	2017-04-09	2017-04-13	4
Sprint 3	2017-04-17	2017-05-05	18
LL version 1 klar	2017-04-17	2017-04-30	13
Testning av v.1	2017-04-27	2017-05-02	5
Deploy mot server/domän	2017-05-01	2017-05-05	4
Användbh.Analys	2017-05-02	2017-05-05	3
Användbh.Test	2017-05-02	2017-05-05	3
Ytterligare test/ver	2017-05-02	2017-05-05	3
Whitebox-test	2017-05-02	2017-05-05	3
Sprint 4	2017-05-08	2017-05-26	18
Kodning	2017-05-08	2017-05-17	9
Regress.Test/testning	2017-05-15	2017-05-19	4
Server/dömän depoly	2017-05-15	2017-05-21	6
Dokumentgranskning	2017-05-22	2017-05-23	1
Färdigst.Dokument	2017-05-23	2017-05-24	1
Problem-fix	2017-05-24	2017-05-25	1
Förb.Pres.	2017-05-24	2017-05-25	1
Up n running	2017-05-25	2017-05-26	1

Riskanalys

Syftet med att utföra en riskanalys är att man redan i projektets start kan identifiera projektet och på så sätt förebygga att de inträffar samt hur man gör om en risk trots allt skulle inträffa. Genom att identifiera både risker och konsekvenser blir gruppen medveten om riskerna och har dem i åtanke under projektets gång för att undvika att de inträffar.

Identifierade risker

Riskernas sannolikhet och konsekvens kommer att anges i form av låg, mellan och hög. En låg konsekvens innebär att detta inte kommer ha någon större betydelse för projektet. Om konsekvensen är mellan innebär det att projektet kan komma att påverkas till en viss gräns. En hög konsekvens kommer innebära att det har en stor påverkan på projektet och bör därför ha en ordentligt utformad handlingsplan för att förebygga att de inträffar.

Är sannolikheten att något ska inträffa angivet som låg innebär detta att det inte förväntas inne inträffa alls. Är det angivet som mellan så tros det kunna hända och är den angiven som hög tros risken med stor sannolikhet kunna inträffa.

A. Scope creep

Det finns risk att gruppen är så ambitiösa att gruppmedlemmar kommer på nya funktioner hela tiden löpande under projektet istället för att fokusera på de funktioner som definierades i början av projektet.
Sannolikhet: Hög, då man alltid strävar efter att skapa en så bra produkt som möjligt kan det kanske bli svårt att hålla sig på banan.
Konsekvens: Hög, gruppen kan glömma vilka krav som är mest prioriterade och gör de som de kanske tycker är roligast eller liknande.
Handlingsplan: Gruppen behöver fokusera på kravspecifikationen och planeringen. Kommunikationen i gruppen är viktig så att man berättar vad man gjort för att undvika funktioner som inte är prioriterade i kravspecen. Prioritera kraven och gör dessa i den ordningen för att undvika att missa viktig funktionalitet. Om detta mot förmodan ändå skulle inträffa så får personen i fråga ta tiden som det tog från fritid.

B. Fördela uppgifter

En utmaning kan vara att dela ut lagom mycket arbetsuppgifter till alla gruppmedlemmar och att jobba tillsammans med att skriva kod.
Sannolikhet: Mellan. Sannolikheten att vissa gruppmedlemmar jobbar mer än andra kan inträffa då kunskap, kompetens och ambitioner varierar.
Konsekvens: Hög, kan bli att vissa gruppmedlemmar gör väldigt mycket medans andra gör väldigt lite.
Handlingsplan: För att undvika detta är det viktigt att samtliga gruppmedlemmar är tydliga under varje möte med vad dem gör, om man har mycket att göra eller för lite i förhållande till dem andra. Detta är inte alltid tydligt direkt när uppgifterna delas upp då det kan visa sig att dem är mer eller mindre krävande än trott. Därför är kommunikationen viktigt.

C. Frånvaro av gruppmedlemmar i en kritisk punkt av projektet

Att människor kan vara frånvarande är en potentiell risk som inte ska bortses, anledningar bakom detta kan vara allt möjligt som påverkar människan psykiskt såväl som fysiskt.

Sannolikhet: Låg, sannolikheten att någon gruppmedlem försvinner under projektets gång är hyfsat stor dock att någon försvinner under en kritisk punkt är potentiellt inte lika stort.
Konsekvens: Hög, projektet kan i värsta fall sammanfalla om gruppen inte lyckas ta ansvar för den frånvarandes del. Detta kan vara att personen som är mest erfaren inom programmering försvinner och resten av gruppen inte lyckas hantera detta.
Handlingsplan: Går till en viss del hand i hand med punkt B införandet av en sorts beredskapsplan, för att hantera detta bör man ha en alternativ person som kan ta över varje uppgift som varje respektive gruppmedlem får. Ett exempel är om Person A får i uppgift att ha hand om programmeringen så bör också Person B vara beredd på att ta över detta utifall Person A försvinner precis som att Person A ska vara beredd att ta över Person B.

D. Förändringar av kraven som kräver stor omarbetning av designen läggs fram
Ju längre projektet går vidare desto större chans att man vill omarbeta något man enats om tidigare i projektet, i vissa fall kan dessa ändringar kräva stora förändringar av systemet
.
Sannolikhet: Mellan, människan är en obestämd varelse. I vissa fall kan man vara bestämd för en sak medan längre fram i projektet ha ändrat sin åsikt och vilja på så sätt ändra kraven i systemet.
Konsekvens: Hög, förändringar av kraven som kräver stor omarbetning av designen är väldigt tidskonsumerande och kan halta projektet väldigt enkelt.
Handlingsplan: Hålla sig till kraven som lagts fram och försöka att inte ändra på dessa, endast ändringar som påverkar det positivt bör framläggas. Det vill säga förändringar som både gör systemet bättre men också kan implementeras i mån av tid.

Riskdiagram