ACC Simulering

- Analys av komponent för områdeskontroll

LUNDS UNIVERSITET
Lunds Tekniska Högskola

LTH Ingenjörshögskolan vid Campus Helsingborg
Computer engineering

Examensarbete:
Daniel Thunberg
Victor Sandberg

LTH Ingenjörshögskolan vid Campus Helsingborg
Lunds universitet
Box 882
251 08 Helsingborg

LTH School of Engineering
Lund University
Box 882
SE-251 08 Helsingborg
Sweden

Tryckt i Sverige
E-Huset
Tryckeriet
Lunds universitet
Lund 2013

Sammanfattning

SAAB Training & Simulation AB utvecklar träningssimulatorer för både militärt och civilt bruk.
En av dessa simulatorer, ATC Training, är planerad att vara en helhetslösning där alla delar inom flygledning finns representerade. Simulatorn är i nuläget inte helt färdigutvecklad och därför inte redo att levereras till kund. I simulatorn saknas det bland annat funktionalitet för simulering av områdeskontroll. Därför finns det behov att undersöka hur simulering av områdeskontroll skulle kunna implementeras. För att implementera funktionalitet för detta finns det planer på att återanvända kod ifrån den äldre simulatorn DATS.

Det långsiktiga målet med examensarbetet är att alla komponenter för områdeskontroll ska identifieras och lyftas ut ur DATS för att sedan skapa en WISE Service som kopplas till ATC Training.
Servicen hanterar endast logiken som finns på serversidan, d.v.s. användargränssnitt är helt ointressant för examensarbetet och kommer därför inte analyseras.

Resultatet av examensarbetet är en förberedelse inför skapandet av en WISE Service som sköter simulering av områdeskontroll. I förberedelsen ingår analys av vilken funktionalitet som finns i komponenten, vilken data som behövs för att genomföra en simulering samt hur kommunikation med resten av systemet går till.

Den komponent i DATS som innehåller funktionalitet för områdeskontroll är DatsFlight. Komponenten har funktionalitet för att beräkna färdplaner utifrån DATS simplaner, se ifall två eller fler flygplan krockar under deras rutter och behandla hand over.
DatsFlight kan också ta emot de event som programmet använder för att kommunicera. Detta kan till exempel vara event som ”HOVAccept” vilket accepterar en överlämning av en flight från en sektor till en annan.

Nyckelord: Flygledningss simulator, områdeskontroll, WISE, SAAB

Abstract

SAAB Training & Simulation AB develops training solutions for military and civil use.
One of these solutions, ATC Training, is intended to be a comprehensive solution where all parts in Air Traffic Control are represented. The simulator is currently not fully developed, and therefore not ready to be delivered, but there is no functionality for simulation of Area Control Center. Therefore there is a need to examine how the simulation of Area Control Center could be implemented.
To implement this functionality, there are plans the re-use code from an older simulator, DATS, which is owned by Saab.

The long term goal of this thesis is that all components for Area Control should be identified and taken out of the DATS and then be used to create a WISE Service which is connected with ATC Training.
The service handles only the logic behind the simulation, i.e. user interface is completely irrelevant to the thesis and will therefore not be analyzed

The result of this thesis is mainly preparation for creating a WISE Service that handles the simulation of Area Control. The preparation includes analysis of the functionality available in the component, the data needed to perform a simulation as well as how communication with the rest of the system works.

The component of DATS that contains functionality for Area Control is Dats Flight. The component has the functionality to calculate flight plans from DATS simplans, determine if two or more aircraft will collide during their routes and performs a handover.
DatsFlight can also receive the events that the application uses to communicate. This may include events such as "HOVAccept" which accepts a handover of a flight from one sector to another.

Keywords: Flight Simulator Management, Area Control Center, WISE, SAAB

Förord

Denna rapport är resultatet av det examensarbete som under vårterminen 2013 gjordes på Saab Training and Simulation i Helsingborg.

Arbetet har varit mycket givande och gett försmak på det kommande ingenjörsyrket.

Vi skulle vilja tacka Jacob Blomberg vår handledare och samtliga anställda på Saab i Helsingborg för gott stöd under arbetets gång, en positiv atmosfär för oss studenter och givande diskussioner över kaffet.
Vi vill också tacka Christin Lindholm för bra och givande input på vårt arbete och snabba svar vid frågor.
Sist men inte minst vill vi tacka Anna Di Julio och Mikael Karlsson för intressanta vinklar på problem och trevligt sällskap under vistelsen på Saab.

Helsingborg maj 2013

Daniel Thunberg och Victor Sandberg.

Innehållsförteckning

1 Bakgrund

Denna rapport kommer att behandla ett examensarbete gjort för Saab Training Simulation AB våren 2013.
Saab AB utvecklar utbildningslösningar, för både militär och civil utbildning. Nyligen har en produkt för träning av flygledare, ATC Training, lanserats av Saab. ATC Training är planerad att vara en helhetslösning där alla delar inom flygledning finns representerade. Simulatorn är i nuläget inte helt färdigutvecklad, och därmed inte redo att levereras till kund, utan det saknas funktionalitet för simulering av områdeskontroll. Därför finns det behov att undersöka hur simulering av områdeskontroll skulle kunna implementeras. Områdeskontroll omfattar all kontroll av luftrummet som finns mellan de detaljerade sektorerna såsom flygplatser och andra hårt reglerade sektorer.

Att en simulator innehåller funktionalitet för områdeskontroll är inget nytt koncept utan Saab har idag en äldre simulator som innehåller funktionalitet som liknar den sökta, DATS, vilken är utvecklad av ett annat företag och inköpt av SAAB.
Då det saknas kunskap om de delar som berör områdeskontroll hos Saab skall examensarbetet undersöka dessa. Det är också osäkert om de delar som söks fungerar som de skall.

1.1 Teknisk bakgrund

För att förstå vilken funktionalitet som behövs i en ACC simulator behövs förståelse för flygledningens grundläggande principer.
I det här kapitlet presenteras de delar om områdeskontroll, DATS och ATC Training som behövs för att förstå vad examensarbetetet handlar om.

1.1.1 Områdeskontroll

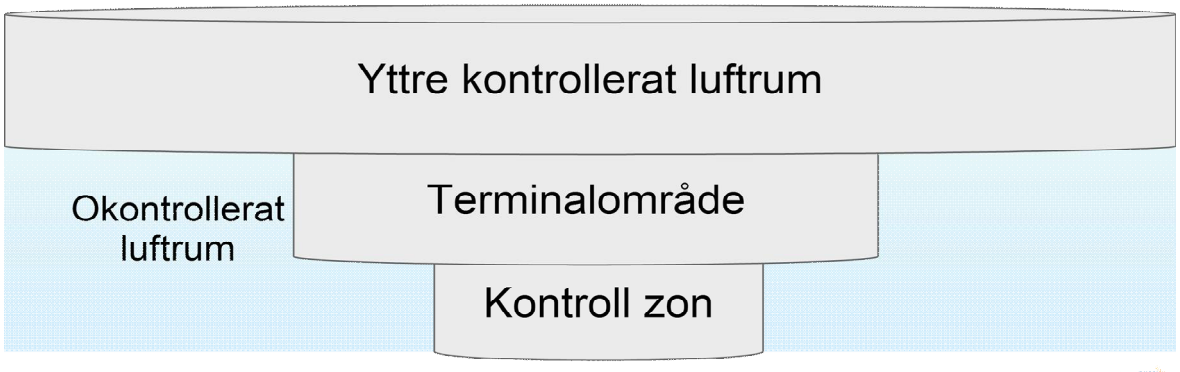


Bild 1-1. De olika kontrollzonerna luftrummet är uppdelat i.

I bild 1-1visas de tre nivåer i luftrummet som flygledning delas in i. Ledning av trafik på flygplats (flygplatskontroll) sker i Kontroll zonen, ledning av luftrummet runt flygplatsen (terminalkontroll) sker i

terminalområdet samt kontroll av all trafik i luften mellan flygplatser (områdeskontroll) sker i det yttre kontrollerade luftrummet.
Områdeskontroll kan kortfattat beskrivas som ett sätt att använda den kortaste rutten för varje flygning samtidigt som man ser till att inga kollisioner i luftrummet sker. För att effektivisera områdeskontroll har man delat in luftrummet i sektorer. En sektor är ett geografiskt område mellan 2,9 km och 14 km höjd, all trafik under 2,9 km höjd anses vara hobbytrafik och regleras inte.
I Sverige finns 30 sektorer som sköts av två olika områdeskontroller. Södra Sverige sköts av Malmö Sturup, mellersta och norra delen sköts av Stockholm Arlanda. Områdeskontrollens uppdrag är att leda trafik genom sektorerna på ett så effektivt sätt som möjligt. Detta samtidigt som de skall undvika kollisioner mellan flygplan och andra luftburna fordon. En grupp om två eller fler personer är ansvariga för varje sektor. Arbetsuppgifterna innehåller moment som att koordinera inkommande flygplan, kommunicera med piloten om eventuella ändringar i kursen, kommunicera med de omkringliggande sektorerna för att se om ändringar har skett. Idag sker uppdateringar av flygplanets kurs digitalt men även på papper ifall datorsystemen skulle gå ner.[1, 2]

1.1.2 Färdplan

För att kunna reglera trafiken så måste varje flygplan lämna en färdplan innan de lyfter. Oftast lämnar flygbolaget in flera färdplaner åt gången när de vet att det ska flyga, det kan vara år före avfärd. Färdplanen beskriver varifrån de åker, destination, tid för avfärd, önskad flygrutt med mera.
Den önskade hastigheten, höjd samt rutt bestäms av det som skrivs in i fält 15 på färdplanen. T.ex. kan det se ut som ”N0300F200 DCT KUM DCT STA”, detta innebär att en hastighet på 300 knop, en höjd på 20000 fot samt en rutt genom de förbestämda punkterna KUM och STA önskas.[3]
För all trafik genom Europa är det Eurocontrol som beräknar flygrutter utifrån inskickade färdplaner. Det kan hända att den önskade rutten ändras för att undvikas kollisioner eller för att leda om trafik från hårt belastade sektorer.
För att undvika kollisioner i luftrummet justeras i första hand flygplanens höjd och beroende på flygplanets utrustning för höjdmätning använder man olika stor höjdseparation när man skiljer på flygplan. De moderna flygplanen har instrument med tillräcklig noggrannhet för att kunna använda en så liten höjdseparation som 1000 fot vilket är ca 300 meter. [2]

1.1.3 WISE Connectivity

Det traditionella tillvägagångssättet för att koppla samman flera applikationer är att skapa ett gemensamt gränssnitt för kommunikation. Detta är en väldigt tidskrävande och kostsam metod, därför har Saab utvecklat WISE Connectivity. Huvudidén med WISE Connectivity är att utvecklaren inte ska behöva göra några ändringar i applikationerna som ska kopplas ihop utan istället använda ett grafiskt grässnitt för hela integrationen. Den enda programmering som behövs är för att skapa drivrutinen som applikationen använder för att kommunicera med systemet.

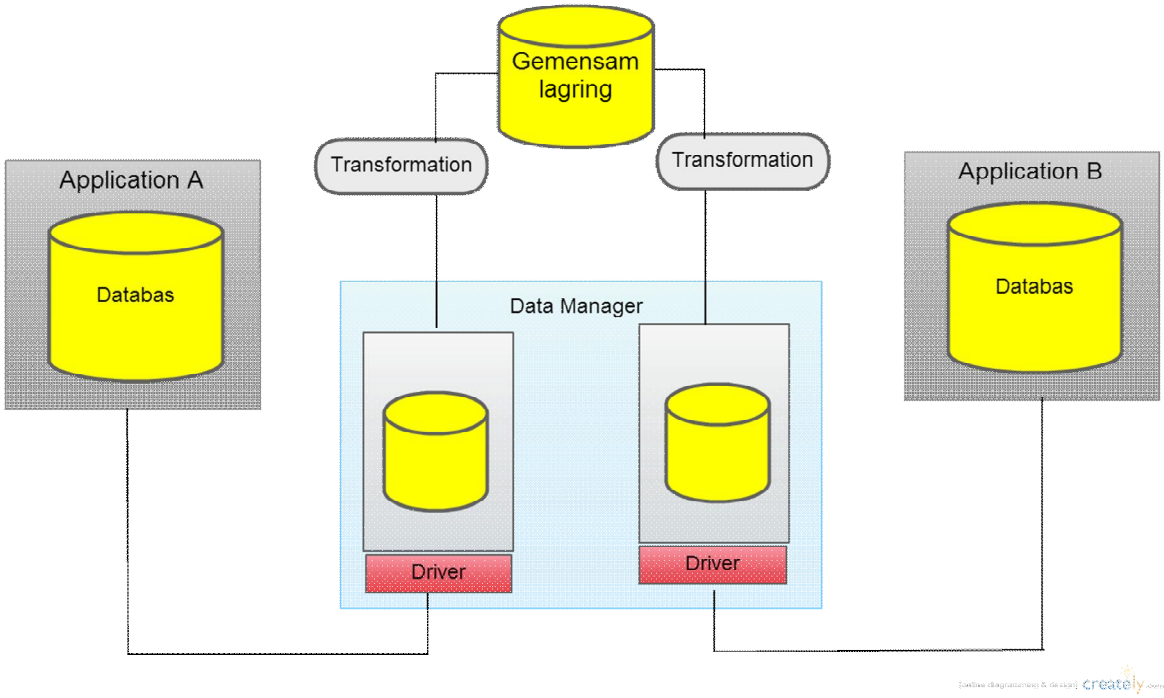


Bild 1- 2. En överblick hur de olika delarna i WISE Connectivity hänger ihop.

De applikationer som används för att utvidga funktionaliteten i ett WISE system kallas för WISE Service. I bild 1-2 visas hur två applikationer kopplas till varsin Data Manager. Data Managerns uppgift är att synkronisera den sammanlänkade applikationens databas med systemets lokala databas och att uppdatera applikationen när någon ändring har skett på en databaspost som påverkar denne. WISE Servicens output skrivs direkt till den sammanlänkade Data Managers databas, vilket gör den tillgänglig för resten WISE systemet för vidare omvandling till andra databaser.[6, 7]

1.1.1 DATS

DATS är en applikation för att simulera flygtrafikledning inom både det civila och militära. Applikationen är skapad av BAE Systems C-ITS men numera ägd av Saab.

DATS består av ett antal komponenter vilka körs parallellt för att kunna ta hand om all data som behövs för att genomföra en simulation. En komponent

kan ses som ett fristående program som har en eller ett par uppgifter att utföra. Uppgifterna utförs genom att lägga till, ta bort och ändra i systemets databaser och genom att skicka meddelanden till resten av systemet. [4, 5]

1.1.2 CATS

För att koppla samman komponenterna i DATS har bassystemet CATS använts. CATS uppgift är att tillhandahålla ett gemensamt kommunikationsgränssnitt.

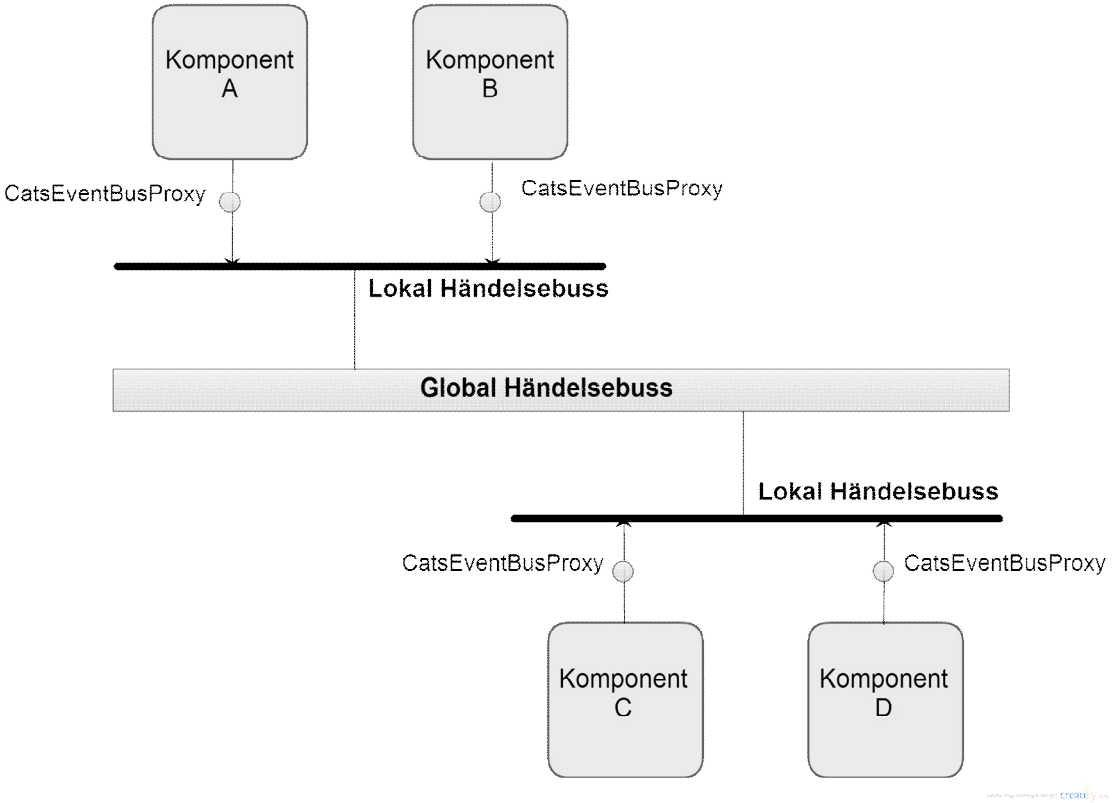


Bild 1-3. Enkel beskrivning över eventhantering i CATS

Bild 1-3 förklarar hur CATS används för att koppla samman komponenter med hjälp av händelsebussar, även kallat eventbussar. Eventbussar finns i två nivåer, en global vilken finns på systemets huvudserver och en lokal buss som körs på varje dator som ingår i systemet.
Den globala bussens huvuduppgift är att sammankoppla alla lokala bussar och distribuera events mellan dem.
En lokal eventbuss kopplas till systemets globala buss och meddelar denne om vilka events komponenten vill blir notifierad om.
Events som ska skickas ifrån en komponent till en annan gör detta genom CatsEventBusProxy vilket är ett gränssnitt mellan komponenten och den lokala eventbussen.

1.2 Problemformulering

Målet med examensarbetet är att analysera hur simulering av områdeskontroll kan gå till och ta fram förslag på hur denna funktionalitet kan implementeras i ATC Training.
Arbetet går ut på att lokalisera den komponent i DATS som behandlar områdeskontroll för att sedan undersöka vad den gör och hur den fungerar. Möjligheten att lyfta ur och kapsla koden så att den går att använda med ATC Training skall undersökas.

Frågorna som kommer besvaras i rapporten är följande:

- Vilka uppgifter sköts av områdeskotrolltjänsten?
- Vilka komponenter i DATS berör Områdeskontroll?
    - Vilken funktionalitet finns i komponenterna?
    - Vilka objekt används i DATSs gemensamma databas?
    - Hur kommunicerar komponenterna?
- Är det möjligt att skapa en WISE Service utifrån områdeskontroll komponenterna i DATS?
- Vilka nya objekt kommer att behöva skapas i ATC Trainings databas?

1.3 Avgränsning

Det är bara områdeskontroll som tas upp i examensarbetetet, ingen annan del av flygledning. Examensarbetetet fokuserar på analys av DATS och dess komponenter. Klasser i DATSs källkod som berör användargränssnitt behandlas inte då dessa gränssnitt kommer att ersättas av nya när komponenten överförs till ATC Training. Inget GUI kommer heller att skrivas till ATC Training.
Kommunikation mellan komponenter i CATS behandlas inte på någon djupare nivå än att vilka komponenter som kommunicerar med vilka dokumenteras.

2 Metod

Större delen av arbetet i det här examensarbetet är att undersöka kod samt göra en mindre litteraturstudie om områdeskontroll och flygtrafikledning. Litteraturstudien görs i samband med undersökningen för att få en djupare förståelse inom ämnet.

Området examensarbetet berörde var nytt för medlemmarna i projektgruppen och därför bröts uppgiften ner i mindre deluppgifter för att underlätta planering av arbete och utvärdering av resultat.
Uppgifter som skulle utföras punktades upp och prioriterades i en veckoplan som sattes upp i början av projektet och sedan uppdaterades med nya poster allt eftersom. Den slutgiltiga veckoplanen finns i appendix 8.1. För att bestämma vilken prioritet varje uppgift hade ställdes den uppskattade tidsåtgången mot hur stor nytta resultatet av uppgiften ansågs ha.
Hittades ett sidospår i en uppgift som ansågs ha större vikt än de tidigare listade punkterna omprioriterades veckoplanen med den nya uppgiften. På detta sätt kunde arbetet hela tiden hålla hög fart i rätt riktning istället för att gång på gång ta snedsteg som kunde kosta mycket tid.

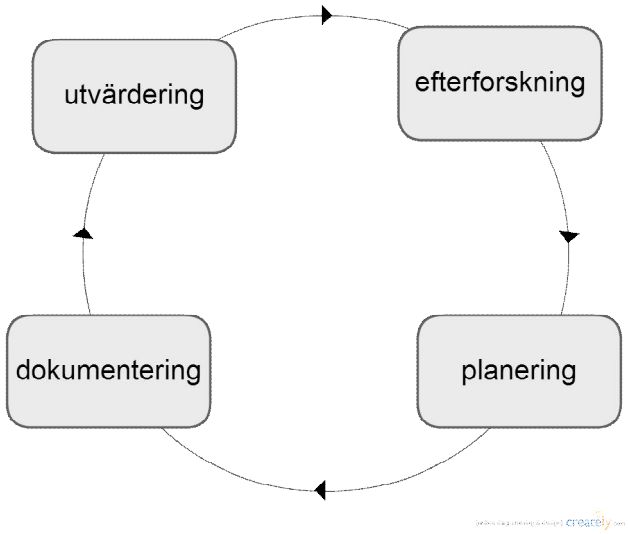


Bild 2-1. Arbetsflöde.

Då merparten av examensarbetet går ut på att undersöka källkod har en undersökande metod används. Arbetssättet är kvalitativt där uppsatta sökord används för att hitta det eftersökta faktum. Arbetet sker cirkulärt där uppgiften bryts ner till en konkret utgångspunkt som sedan går igenom ett antal steg där det görs efterforskning, planering, dokumentering samt i slutet på varje cykel en mindre utvärdering. Bild 2-1 visar en grafisk representation av arbetssätet.

De uppsatta sökorden och metoden kan komma att ändras under arbetets gång allt efter arbetet fortskrider.
Utifrån examensarbetets frågeställning kunde tre huvuduppgifter sättas upp:

- Förberedande undersökning av områdeskontroll och flygsimulering.
- Undersökning av källkod för DATS.
- Implementering av kod i ATC Training med hjälp av WISE.

Varje delmoment krävde i någon mån att nya områden undersöktes eller att dokumentation för nya system lästes igenom och analyserades.
Den första och andra uppgiften är till viss del fristående från varandra och är inte absolut nödvändig att slutföra innan de andra delarna påbörjas därför utfördes dessa parallellt.
Analys av källkoden i DATS är examensarbetets största fokus och rapporten kommer därför att till största del behandla detta.
Implementering i ATC Training har lägst prioritet av de uppsatta punkterna och kommer därför endast utföras i mån av tid. Denna punkt är trots sin låga prioritet viktig att ha med då det är mot detta undersökningarna kommer att vara vinklad.

2.1 Förberedande undersökning

Eftersom ingen av projektmedlemmarna hade någon tidigare kunskap om flygledning eller flygsimulatorer behövdes en inledande litteraturstudie göras. Informationen hämtades genom en föreläsning från Luftfartsverket samt genom att läsa de källor som finns presenterade i kapitel 7.
Föreläsningen tog upp grunderna om områdeskontroll och det var den som litteraturstudien bygger på. För att få ytterligare information och fylla i de delar som missades under föreläsningen lästes interna utbildningsmaterial. Det var till exempel exakta siffror som mellan vilka höjder en sektor är definierad som hämtades ur de interna materialen.
Sökmotorer för publikationer som *libHub* användes också men sökorden, se kapitel 3, som används vid granskningen av DATS gav inga användbara resultat. Sökmotorn *google* gav inte bättre resultat än artiklar på *wikipedia* som tyvärr inte innehöll något nytt och saknade pålitliga källor. Endast *luftfartsverket* hade ett kort stycke om områdeskontroll som kunde användas. De fakta som fanns att tillgå dokumenterades i en löpande text med avseende att skapa en grundförståelse för läsaren i vår rapport. Informationen hölls på en grundläggande nivå då texten inte skall göra läsaren till expert i ämnet och för att rapporten skall fokusera på den tekniska biten och inte flygledning. Utökningen av kunskap inom ämnet fortlöpte genom hela projektet då nya problem och termer hela tiden stöttes på och handledare gav ny input som behövdes ta i beaktning.

2.2 Granskning av DATS

DATS innehåller funktionalitet för simulering av områdeskontroll. För att undersöka hur simulering av områdeskontroll kan utföras analyserades DATS genom att granska källkoden och undersöka innehållet i simulatorn.

2.2.1 Sökande efter områdeskontrollkomponent

För att hitta de delar i DATS som berörde områdeskontroll studerades programmets källkod. Till hjälp användes de sökord som tas upp i analys, kapitel 3, som hämtats ur den förberedande undersökningen och den föreläsning som gavs av Luftfartsverket.
Resultatet från sökningarna listades och prioriterades för att sålla bort så mycket onödig information som möjligt. Prioriteringen var väldigt grov och behandlade endast filnamn, klassnamn och kommentarer. Filer vars filnamn innehåller något av sökorden prioriterades högst och undersöktes därmed först.
För att hitta de intressanta delarna bland resultaten söktes de högt prioriterade filerna igenom. Först undersöktes det om filen hade några klasspecifika kommentarer, sedan om det fanns kommentarer till metoderna och attributen. Fanns inga kommentarer var koden oftast självförklarande.

De filer som innehöll något intressant dokumenterades och nästa fil undersöktes.
Metoden för att söka efter ledord sågs till en början som ett steg i rätt riktning snarare än ett slutgiltigt resultat då det bara är filer som innehåller ledorden som valts ut som hittas. Den viktigaste funktionaliteten kan ligga i filer som inte innehåller sökorden vilket gör denna metod väldigt osäker och snarare ses som en vägvisare.
Sökningarna är dock väldigt tidseffektiva och kan ge väldigt bra resultat ifall bra sökord har valts.
Trots den uppenbara osäkerheten i sökmetoden sågs ett mönster bland sökresultaten där en specifik komponent förekom under samtliga sökningar. Denna komponent var Datsflight, eftersom komponenten innehöll en stor mängd områdeskontroll relaterade termer beslutades det att denna skulle undersökas närmre.

2.2.2 Undersökning av DatsFlight

När sökning och gallring har lett fram till att komponenten DatsFlight i DATS behandlar områdeskontroll startar största och mest tidskrävande delen i examensarbetet.
Huvuduppgiften är att skapa en överblick av vilken funktionalitet, data som används och hur komponenten kommunicerar utåt. För att komma fram till detta gjordes kodgranskning där DatsFlights källkod gicks igenom.
För att granska källkoden och skapa en uppfattning om vad varje klass gjorde lästes i första hand kommentarer i koden.
Först lästes klasskommentarer och ett dokument som beskrev klassernas övergripande funktionalitet skapades. Dokumentet utökades successivt när en allt noggrannare genomgång av funktioner och kommunikation gjordes.
Fanns det inga kommentarer i filen hade funktionerna självförklarande namn och funktionaliteten var oftast given.
Att dokumentera ett program genom att granska källkoden är en väldigt tidskrävande uppgift, därför utfördes även ett antal andra undersökningar på simulatorn där menyer jämfördes och simulationer kördes. Detta gjordes för att få fram fler termer som används för att beskriva uppgifter inom områdeskontroll, dessa termer kunde senare användas för att söka upp specifik funktionalitet i komponenten.

2.3 WISE Service

Då tiden var en viktig faktor för om de uppsatta målen för examensarbetet skulle hinnas göras klart eller inte delades uppgiften med att skapa en WISE Service upp i ett par mindre delar.

- Identifiera vilka delar i DatsFlight som kommunicerar med övriga komponenter i DATS.
- Göra DatsFlight kompilerbar och fristående från övriga komponenter i DATS.
- Skapa WISE Service

Den första delen utförs genom att samla ihop alla externa headerfiler som områdeskontrollkomponeten använder sig av. De headerfiler som inkluderas från externa projekt ger en fingervisning om vilka projekt som påverkar DatsFlight, dessa möjliga kommunikationskanaler undersöks vidare.
För att kompilera koden krävs flera externa filer som inte finns byggda, projektet filen tillhör letades upp och filerna skapades. Flera av de projekt som behövde byggas krävde i sin tur ett antal externa filer för att kunna kompileras, detta skapade en lång kedja av headerfiler som behövde identifieras och inkluderas i projekten.
De filer som skapades vid kompilering av externa projekt inkluderades med hjälp av Visual Studios 2010 ”Additional include directories”.
Därefter flyttades projektets rootmapp till en annan plats på datorns hårddisk och alla externt inkluderade filer flyttades till en egen mapp. Detta för att på ett enkelt sätt kunna se vilka filer som behövs.

2.4 Källkritik

Information om områdeskontroll är i sig inte hemlig men då det är få som efterfrågar den så finns väldigt lite om ämnet på till exempel luftfartsverkets hemsida. Som källa användes i stället en intern utbildning på Saab som gavs av Luftfartsverket. Ytterligare kursmaterial från Saabs interna utbildningar användes också. Dessa källor ses som trovärdiga då informationen kommer ifrån ett statligt affärsverk som bedriver utbildning inom ämnet. [9] Luftfartsverket eller utbildaren har heller inget att vinna på att sprida falsk information. Att använda en tio år gammal källa är inga problem då den grundinformation som används i detta arbete inte ändrats på lång tid.

Skybrary.aero är ett initiativ av *Eurocontrol* och *International Civil Aviation Organization* [ICAO] med det enda syftet att skapa ett forum för att snabbt kunna nå ut med information till allmänheten. Artiklarna utgår från information hämtat från de största organisationerna inom flygtrafik, till exempel ICAO som är ett FN organ. [10,11]

När källkoden undersöktes förutsattes det att kommentarerna var korrekta och att koden fungerande. Kommentarerna i koden antogs vara korrekta då koden är skriven av ett seriöst företag som bara förlorar på felaktigt kommenterad kod. Om kommentarerna varit felaktiga hade dessa litats på då tiden för projektet var begränsad och att undersöka samtliga kommentarer hade varit för tidskrävande.

För att hämta information om hur scenarion skapades användes DATS användarmanual *operators handbook* samt information av expert Saab. Detta ses som en trovärdig källa då handboken är skriven av företaget som utvecklat och sålt programvaran. [8] Handboken är skriven för att utbilda kunden och är daterad februari 2011 så informationen är aktuell.

När WISE undersöktes användes *WISE Connectivity SDK - Developer's Guide* vilket ses som den bästa källan att söka i då den är skriven av företaget som skapat programvaran.[7] Guiden är skriven för att underlätta vid utveckling mot plattformen WISE. Den är uppdaterad feb 2013 så det ses som aktuell information. Därför ses SDK som en källa som inte behöver ifrågasättas. Ifall fel skulle förekomma i SDK kommer det inte att påverka examensarbetets resultat något större. Detta eftersom ingen direkt implementation görs i WISE med denna som utgångspunkt utan den används mer för att skapa grundläggande förståelse för hur WISE Connectivity är uppbyggt för rapporten.

3 Analys

Detta kapitel innehåller analysen av de data som hittades då källkoden till DATS genomsöktes. Analysen bör ses som ett sätt att diskutera de resultat som kommer att redovisas i nästa kapitel.

3.1 Undersökning av DATS

3.1.1 Val av sökord

De sökord som valdes för att söka i källkoden till DATS diskuterades fram i samarbete med handledare på Saab efter att förarbetet om områdeskontroll gjorts. De orden som valdes är vanligt förekommande termer inom områdeskontroll.
Sökorden som användes var area control center, sector, approach och flight plan. Varianter på sökorden främst svenska översättningar och förkortningar användes också med varierande resultat.
För att på ett så effektivt sätt som möjligt söka i koden användes windows xp inbyggda sökfunktion. Alternativet ”A word or phrase in the file” användes då den söker igenom hela filen och inte bara dess filnamn vilket ger betydligt större sannolikhet att få bra resultat.
Sökfunktionen visade ibland resultat som sökningen innan inte hade visat och ibland inga resultat alls. För att kunna lita på sökresultaten gjordes de två gånger för att minimera den felkällan.
Sökorden gav olika bra resultat. Det självklara valet att söka efter ACC gav väldigt många svar, drygt 2700. De flesta resultaten var dock när acc var del av engelska ord som access, accessor osv. Att gå igenom alla resultat ansågs vara för tidskrävande och därför togs detta sökord bort helt.
Att söka efter ”area control center” eller ”area control” som helhet gav inga resultat alls vilket däremot ”area” gav. Detta berodde på att områdeskontroll aldrig nämndes vid namn i kommentarer till koden, istället beskrevs de olika moment som områdeskontroll är uppbyggt av.
”Area” gav drygt tusen resultat vilka förekom i majoriteten av programmets komponenter då det är ett väldigt brett begrepp och används av många komponenter.
Varianter på orden ”sektor” och ”Färdplan” gav väldigt intressanta resultat då de inte var spridda över flera olika komponenter utan var koncentrerade till ett projekt, ”DatsFlight”. Att just dessa ord gav bra resultat beror på att de är fundamentala begrepp för områdeskontroll och mycket av den funktionalitet som finns i den eftersökta komponenten utgår därifrån.

3.1.2 Genomgång av resultat

Bland de resultat som hittades vid sökning gjordes först en grov gallring för att reducera mängden till en mer lätthanterlig nivå. Sökresultat som inte hade filändelserna ”.h” eller ”.cpp” ignorerades vid sökningarna då detta är databasfiler, xmlfiler eller resursfiler som används av flera olika komponenter och därför innehåller väldigt varierad information.
De filer som hade ”test” i sin sökväg eller i sitt filnamn togs bort från resultaten då de antogs att ha med test att göra och inte tillförde något till projektet. Sökresultat som behandlar användargränssnitt plockades också bort eftersom detta är utanför examensarbetets avgränsningar. Detta innebar att samtliga projekt med ändelsen ”client” ignorerades.

Med de filer som fanns kvar efter den första gallringen påbörjades arbetet att sortera och prioritera sökresultaten. Den första prioriteringen utfördes i två steg. Först grupperades alla filer som hade sitt ursprung i samma projekt ihop. I varje grupp prioriterades filer efter var sökordet påträffades, där förekomst i filnamnet prioriteras högst, i klassnamn näst högst och kommentarer lägst.
De högst prioriterade filerna lästes igenom extra noga där främst kommentarer till klasser och funktioner undersöktes.
Saknades dokumentation helt i filen undersöktes koden och dess projekt för att skapa en överblick av klassens funktionalitet. Många av de filer som undersöktes saknade dokumentation av något slag och fanns det dokumentation kunde svenska och engelska språket blandas mellan de kommentarer som fanns.
Ifall inga intressanta resultat förekom gjordes processen om med nästa sökord.

”FPLData” var en av de filer som hittades tidigt och fångade stort intresse då den enligt en klasskommentar innehöll ”Data för en flights passage in i en sektor”, vilket var mycket likt det som eftersöktes. ”FPLData” tillhör komponenten ”DatsFlight” som är en av de mest förekommande bland sökresultaten. Till exempel innehåller den även ”SectorData.cpp” som också är högt prioriterad.

3.2 Genomgång av DatsFlight

När det fastställts av handledare på Saab att DatsFlight var den sökta komponenten påbörjades arbetet med att dokumentera dess funktionalitet, vilken data som används och hur den kommunicerade med det övriga programmet.
Att ha kunskap om vilka delar som tillhör ”CATS” och vilka som inte gör det är en del i att porta programmet så att det går att använda ihop med WISE. De delar som är exklusiva för ”CATS” kommer senare att tas bort eller skrivas om så att de anpassas till WISE uppbyggnad. Den kod som inte tillhör DATSs grundstomme kommer att kapslas och använda rakt av tillsammans med WISE.

För att förstå hur komponenterna i DATS kommunicerar beslutades det tillsammans med handledare på Saab att de event som skickades inom programmet skulle studeras. För att på ett effektivt sätt se vilka delar som kommunicerade användes DATSs inbygga program ”EventTrace”. Eventen kommer att vara intressanta efter att modulen kapslats för att användas med WISE.

3.2.1 Jämförelse av menyalternativ

För att undersöka vilken funktionalitet i simulatorn som berör områdeskontroll på ett mer strukturerat sätt än att slumpmässigt klicka igenom varenda meny valdes istället att jämföra menyer när funktioner för områdeskontroll är av och påslagna.
Det finns två sätt att aktivera funktionaliteten för områdeskontroll i simulatorn, antingen genom att lägga till raden ”<useSectorControl/>” i konfigurationsfilen ”dats.config.xml” eller genom att aktivera ”Use Sector Functions” i menyn ”Scenario preparation”. Avaktivering av funktionaliteten görs genom att ta bort raden man skrev i ”dats.config.xml” eller genom att bocka ur rutan ”Use Sector Functions”.
För att få med alla inställningar måste samtliga menyer kontrolleras när simulatorn är i sina olika tillstånd, skapning av scenario, skapning av övning samt under körning.
Vilka alternativ som syns i menyerna är även beroende av vilken roll användaren har, därför gavs datorn som testet utfördes på alla roller och kan därmed se allt. Problemet med detta är att man inte får med information om vilka inställningar som specifikt riktas till övningsledaren och vilka som riktas till den som skall utföra övningen.

Bild 3-1. Scenario Preparation utan useSectorControl

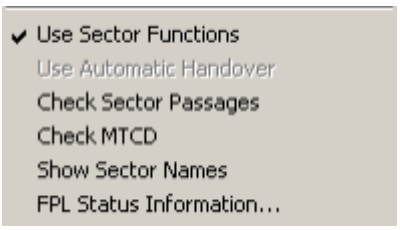


Bild 3-2. Scenario Preparation med useSectorControl

Bild 3-1 visar menyn ”Scenario preparation” i tillståndet ”create Scenario” när ”useSectorControl” är avslagen. Bild 3-2 visar motsvarande meny när ”useSectorControl” är igång. I menyn ”Excersise” är det endast ”Hide All FLEGs” som skiljer åt.
Skillnader mellan menyerna är lätta att urskilja och samtliga menyer jämförs i programmets alla tillstånd på samma sätt som beskrivits ovan.

Meny inställningar	Beskrivning
Automatic HOV	Gör automatiskt en överlämning när en flight äntrar en sektor
Check Sector Passage	Kontrollerar ifall flighters rutter går igenom någon sektor
Check MTCD	Kontrollerar ifall det förekommer risk för kollision inom 20 minuter
Show Sector Name	Visar sektorernas namn i kartan
FPL Status Information	Ger en övergripande bild över samtliga flighter i spelet. Starttid, rutt, tid när sektorgränser nås, nuvarande ägare samt vilka roller flighten är koordinerad till
Approach Sector List	Information om vilka flighter som är koordinerade till den sektor rollen äger visas
Hide All FLEGs	Döljer alla rutter på kartan.
HOV Roles	Verktyg för att definiera förinställda roller för handover.

Tabell 3-3. Beskriver de menyval som identifierades vid menyjämförelse.

De olika menyalternativen kunde med lätthet spåras i koden genom att söka igenom DATS med ”checkSectorPassage” och ”AutomaticHOV” som sökord då det visade sig att dessa var funktioner i DatsFlight, samtliga menyalternativ finns beskrivna i tabell 3-3.
Att DatsFlight var den komponent som behandlade områdeskontroll var fastställt sedan tidigare men denna undersökning gav ytterligare information om vilka funktioner som slutanvändaren stöter på.

3.3 Skapande av scenario

Ett scenario i DATS är en planerad sekvens av händelser som simulerar ett händelseförlopp på ett verklighetstroget sätt.
Målet med att köra scenarion är att få en större förståelse för hur simulering av områdeskontroll fungerar samt att övervaka vilka events som skickas.
För att skapa ett scenario som beskriver händelser vilka är relaterade till områdeskontroll måste först uppgifterna som utförs definieras.
Kortfattat är det att

- Leda flygplan genom och mellan sektorer.
- Förhindra kollisioner mellan flygplan.

Baserat på den funktionalitet som DATS har att erbjuda genom menyer och vad som görs rent praktiskt i simulatorn byggdes scenariot med handover upp.
Simulering av kollisioner fungerade inte tillräckligt bra för att något resultat av värde skulle kunna hämtas härifrån**.** Detta berodde på att inga notifikationer om att en kollision var på väg skickades till de roller som kontrollerade planen.

3.3.1 Handover

För att simulera en handover behövs viss grunddata skapas i scenario skaparen: sektorer, ägare till sektorerna och minst en flight som flyger genom mer än en sektor.
Sektorer bygger på ett objekt som beskriver områden, ”area”, därför måste dessa definieras först. När ”area”-objekten finns på plats är det möjligt att para ihop dessa med ”sector”-objekt.
Roller skapas utifrån de datorer som är uppkopplade mot samma serverkonfiguration i DATS, det är möjligt att tilldela flera roller till samma dator. Rollerna kan innehålla en objektreferens till ett ”sector”-objekt, det är denna sektor som rollen styr över.
I ”scenario skaparen” definieras simplaner för varje flight med rutter som går igenom de tidigare skapade sektorerna.
För att se vad som händer och vilka meddelanden som skickas finns ett par verktyg tillgängliga för de som är involverade i övningen.
Övningsledarrollen har tillgång till ”FPL Status Information” och ”Approach Sector List” medan ”traineerollen” endast har tillgång till den sistnämnda.
För att minimera den mänskliga felkällan överläts all hantering av handover till simulatorn genom att aktivera ”AutomaticHOV”.

3.4 EventTrace

Ett steg i undersökningen av DATS var att förstå hur kommunikation mellan och inom komponenterna fungerade. För att göra detta användes ”EventTrace” vilket är ett inbyggt verktyg i DATS för övervakning av Events. ”EventTrace” skriver i realtid ut de events som skickas av DATS under körning av simulatorn.
Större delen av de events som hanteras av DatsFlight skickas inte under ett felfritt scenario utan de används för att uppmärksamma vid ovanliga händelser t.ex. vid ändring av simplan eller ruttdata. För att simulatorn ska generera de events som var av intresse behövdes ett scenario som hanterade dessa ovanliga händelser. Scenariot som är beskrivet i kapitel 3.3.1 kombinerat med manuell ändring av simplaner och rutter under körning täcker in många av dessa ovanliga händelser.

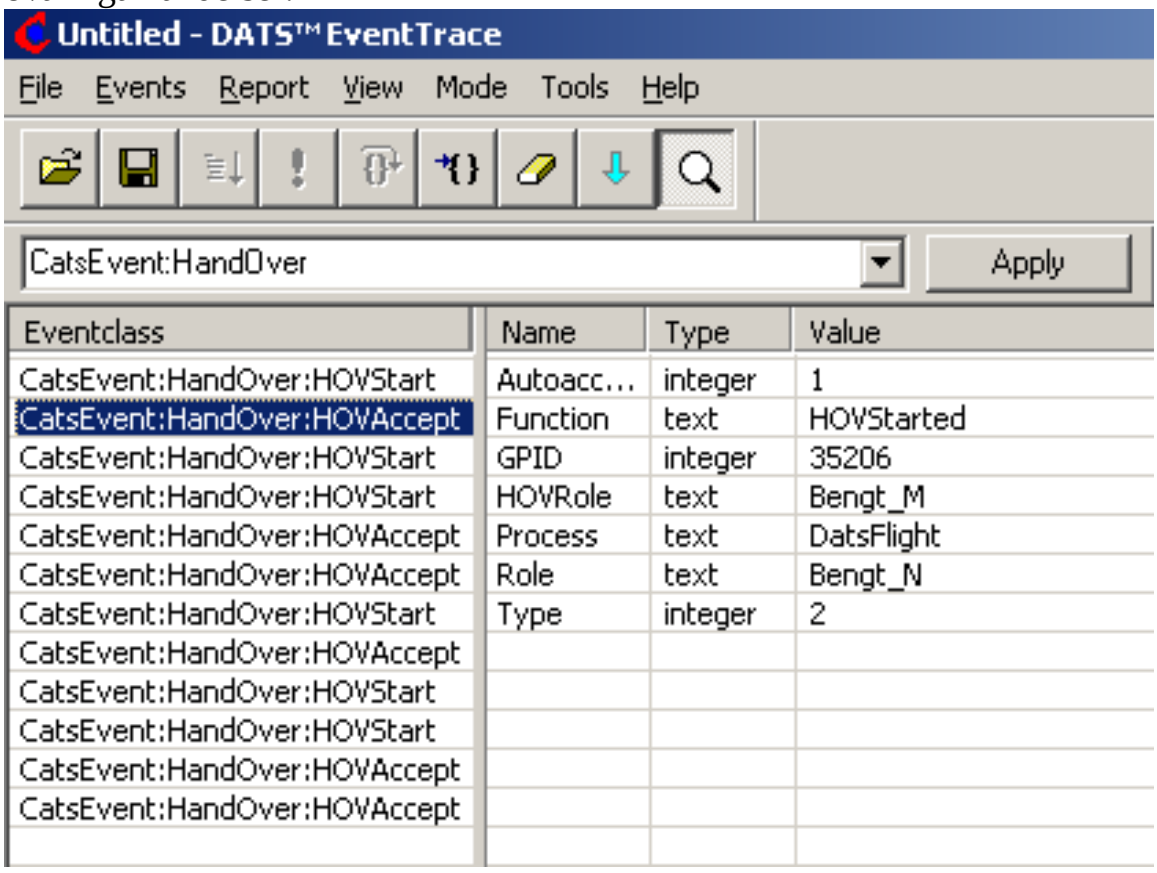


Bild 3-4. EventTrace under körning av scenario.

Till vänster i bild 3-4 är några av de events som skickas under körning, Till höger i samma bild syns parametrarna i eventet som är markerat, ”CatsEvent:HandOver:HOVAccept”. Samtliga events som hanteras av DatsFlight presenteras i kapitel 4.3.

3.5 Byggning av Dats

För att undersöka vilka headerfiler som krävs för att bygga och köra DatsFlight så påbörjades byggning av projektet, till detta användes kompilatorn i Visual Studio 10.

När projektet först skulle byggas saknades många av de filer som behövde inkluderas för att bygga projektet. Då projektet tidigare byggts mot en nätverksplats saknades de output kataloger som var angivna för projektet. Första steget blev då att ändra samtliga kataloger för output i Visual Studio 10 samt sökvägar till de inkluderade filerna vars sökväg hade hårdkodats.

När filen som skulle inkluderas saknades söktes det först efter den bland DATSs komponenter. Om den inte hittades var det oftast en headerfil som skapades vid kompilering av ett annat projekt.
Om försök att bygga projekten i DATS i en annan ordning än den tänkta kommer många av de filer som behöver inkluderas saknas. Därför blev andra steget i byggprocessen att hitta samtliga filer som måste inkluderas och deras projekt för att skapa dessa filer om det visade sig att de saknades.

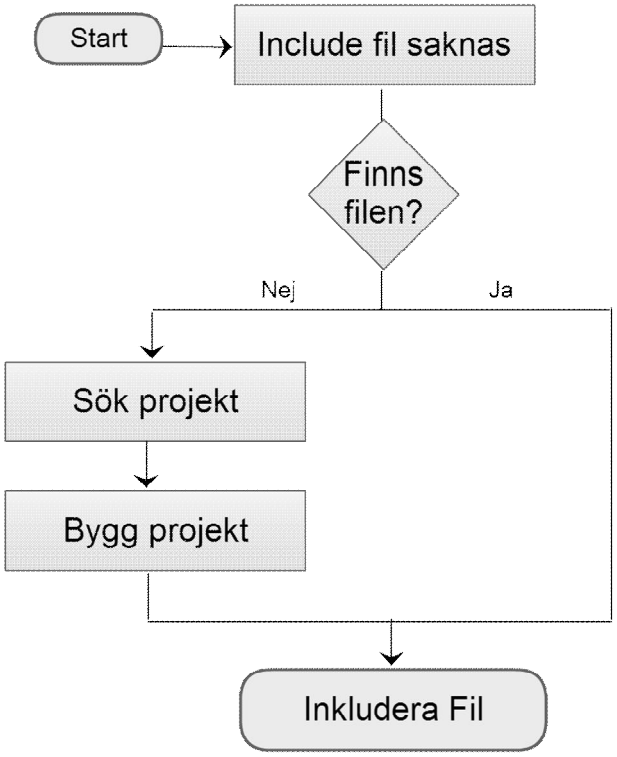


Bild 3-5. händelseförlopp vid byggning av projekt

Projekten som skulle byggas kunde i sin tur sakna nödvändiga filer vilket gjorde arbetets gång mycket likt att gå igenom ett dataträd med en rekursiv metod. Se figur 3-5 för förklaring.

De filer som byggts inkluderades först med hjälp av VS10 ”Additional include directories” för att på ett enkelt sätt kunna inkludera alla filer.
När alla externa delar var byggda och projektet saknade några ”.lib” filer så lyfts projektet till en annan plats på hårddisken och alla inkluderade filer i ”Additional include directories” togs bort. Sedan lades alla filer som skulle inkluderas i en egen mapp. Filerna i mappen finns listade i appendix 8.3. Mappen pekades sedan på med hjälp av ”Additional include directories” så att alla filer inkluderades. På så sätt kunde man se exakt vilka filer som behövs för att bygga projektet och i fortsättningen ta bort de filer som WISE kommer att tillhandahålla funktionaliteten för.

4 Resultat

Då undersökningen av DATS kom fram till att DatsFlight var den enda komponent som innehöll funktionalitet för områdeskontroll beslutades det att denna komponent skulle undersökas vidare.
I detta kapitel finns en sammanställning över funktionalitet, data samt kommunikation i DatsFlight.

4.1 Funktionalitet

Här presenteras all praktisk funktionalitet i DatsFlight. Funktionaliteten är ordnad efter vilken klass i DatsFlight den är implementerad i.

4.1.1 AirportFlightList

AirportFlightList sköter flygplatsers avgångar och ankomster.
Förutom att skapa en flygplats finns även funktionalitet för att ändra och returnera ankomsttider samt avgångstider. Den innehåller bland annat:

- Från vilken gate ankomst eller avgång kommer ske
- Vilken tid avgång eller ankomst kommer att ske
- Om ankomst eller avgång är godkända.

4.1.2 CatsDatsFlightDefines

Innehåller information om vilket tillstånd en flight har. Tillstånden är till exempel om flighten är initierad, om flighten är koordinerad till någon sektorägare, till vilken sektor flighten överförs och från vilken sektor flighten överförs.

4.1.3 CatsDatsFlightRegistry

Innehåller medlemsvariabler för alla registervärden till CatsDatsFlight. Läser information från registret vid uppstart för att sedan starta simulatorn med dessa inställningar. Innehåller inställningar som:

- Om alla objekt kollas när de passerar en gräns till en sektor
- Om en HOV alltid startar när ett objekt närmar sig sektorgränsen
- Hur många sekunder innan sektorns gräns som programmet upptäcker en passerande flight
- Hur lång tid innan programmet själv accepterar en HOV från okontrollerat luftrum eller från sektor som saknar ägare.
- Hur lång tid, i sekunder, innan en varning kommer att utfärdas från MTCD systemet
- Hur lång sträcka innan en varning eller ett larm kommer att utfärdas från MTCD systemet.

4.1.4 CatsDatsFlightServer
Implementerar COM- gränssnittet mellan klienterna i DatsFlight. Installerar eller avinstallerar kommunikation med klienterna samt reglerar deras rättigheter utifrån vilken roll de har.

4.1.5 CatsDatsFlightServerExe
Är den klass som initierar servicen, skapar tickgeneratorn och event loggas. Den initierar servicens run() metod vilket fungerar som servicens main-metoden.

4.1.6 CatsDatsFlightServerImpl
Projektets huvudklass som anropas av run() i CatsDatsFlightServerExe vilket medför att huvudklassen och alla dess underordnade klasser utför sina uppgifter. Innehåller det mesta av logiken för områdeskontrollsimulering.

4.1.7 CatsDatsFlightServerPlugin
Har hand om att skapa, spara, öppna, stänge, starta, stoppa och förbereda spel.

4.1.8 SectorData
Fungerar som en lagringsklass för data om sektorer.
Klassen innehåller:

- Namn på sektorn
- Beskrivning av sektorn
- Vilket geografiskt område som sektorn avgränsar.
- Vilka roller som äger sektorn.

4.1.9 FPLData
Innehåller färdplanen för en flight. Se appendix 8.2 för tabell.
Klassen har funktioner för att ändra i färdplanen. Konflikter beräknas, handover och koordination av flighter hanteras.

4.1.10 Conflict
Fungerar som en lagringsklass för data om en flights konflikt med andra flighter. Har variabler för vilken typ av larm som skickas, vilken tid och plats de beräknade konflikterna kommer ske på.

4.1.11 SectorPassage
Behandlar data för en flights passage in i en sektor. Har attribut som beskriver vilken tid flighten äntrar och lämnar en sektor.

4.1.12 CatsEvent
I DatsFlight finns det 12 klasser för att hantera events. Dessa events beskrivs i kapitel 4.3.

4.1.13 DatsDataGameObserver
Kollar ifall databasen har blivit uppdaterad och ifall någon av de uppdaterade posterna berör DatsFlight. Anropar DatsGameObserver’s konstruktor.
Klassen är ej färdigskriven och används därför inte.

4.2 Datamodell

I det här kapitlet presenteras de objekt DatsFlight använder sig av i DATSs gemensamma databas.
Data för DatsFlight sparas i två nivåer, dels på en lokal nivå som är knuten till det scenario som är igång och dels i en SQL-databas som är gemensam för hela simulatorn.
När ett scenario skapas eller sparas i DATS synkroniseras den aktiva ”simulation area” med den gemensamma databasen, BasicData.

4.2.1 BasicData

BasicData innehåller den information som alla scenarion utgår ifrån när de skapas. I samband med att scenariot sparas synkroniseras denna med databasen.
BasicData kan delas upp i två kategorier: gemensam data för hela systemet, t.ex. information om spelpjäser, samt data som är bunden till en viss ”simulation area”.
All data som på något sätt påverkar områdeskontroll påträffades i den ”simulation area” specifika delen av BasicData.

4.2.1.1 Sector

”Sector” definierar sektorer.
Sector har en eller flera objektreferenser till ”Area”-objekt för att beskriva vilket geografiskt område sektorn avgränsar.

Fält - SECTOR	**Beskrivning**
ObjectDescription	Bestämmer vilken kategori objektet tillhör i databasen. I det här fallet = ”Sector”
Description	Beskrivning av objekt i fritext.
Sector:Frequency	Attributreferens till ett ”frequency”-objekt. Beskriver frekvens för sektorns radiokommunikation.
Objektreferens:Area	Referens till ett eller flera ”Area”-objekt

4.2.1.2 Simplan och Färdplan

Simplanen innehåller all information som simulatorn behöver om en flight för att genomföra en simulation med denna. Utifrån simplanen bygger simulatorn en färdplan. Färdplanen innehåller beräknat fält 15 data.
En fullständig tabell över färdplanen hittas i Appendix 8.2.

4.2.1.3 Area

”Area” definierar ett geografiskt område i kartan vilket används för att beskriva sektorer.

Fält - AREA	**Beskrivning**
ObjectDescription	Bestämmer vilken kategori objektet tillhör i databasen. I det här fallet = ”Area”
Area:Type	Bestämmer vilken typ av område objektet är. I det här fallet = ”SECTOR CONTROL AREA”
Area:UpperLimit	Områdets övre kontrollgräns
Area:LowerLimit	Områdets undre kontrollgräns
Area:Active	Bestämmer om området är aktiverat eller ej.
Area:Coordinates	Innehåller koordinater för att bestämma områdets avgränsning.

4.2.1.4 Character

”Character” eller ”roll” beskriver de användare som utför en övning.
”Character”-objekt skapas utifrån de datorer som är uppkopplade mot samma serverkonfiguration i DATS, det är möjligt att tilldela flera roller till samma dator.
”Character” kan ha attributreferens till en annan roll. Flera roller kan därför representeras av en grupp och tillsammans referera till en sektor som de gemensamt är ”ägare” av samtidigt som de är unika i övriga avseenden, menyinställningar osv.

Fält - CHARACTER	**Beskrivning**
ObjectDescription	Bestämmer vilken kategori objektet tillhör i databasen. I det här fallet = ”Role”
Character:UserName	Rollens användarnamn.
Character:Signature	Används endast för Trainee rollen. Signaturen presenteras i etiketter för objekt som rollen äger.
Character:Description	Fritext, beskrivning av rollen
Character:CharacterType	Beskriver vilken typ av roll objektet är. TRAINEE eller PILOT. PILOT används av övningsledare.
Character:Group	Beskriver vilken grupp rollen tillhör.
Character:Instruction	Text som beskriver rollens uppgifter för övningen.

4.3 Kommunikation

För att kunna skapa en WISE Service utifrån DatsFlight behöver de kommunikationskanaler som används identifieras och undersökas.
DATS bygger på CATS och är därför ett eventdrivet program, d.v.s. programmets tillstånd förs framåt genom att komponenterna skickar events som triggar funktioner för t.ex. beräkning av rutter eller handover.

4.3.1 Event

Samtliga events som Datsflight blir notifierad om finns listade i det här kapitlet.

4.3.1.1 Flight:Departure

Ett flygplan har lyft ifrån en flygplats. Avfärdslista uppdateras.

Attribut	Beskrivning
GPID	Unik identifierare
AirportName	Namn på flygplats
FlightCode	Namn på Flight
Gate	Namn på gate flyget avgår ifrån
Time	Tid för avgång

4.3.1.2 Flight:Arrival

Ett flygplan har landat på en flygplats. Ankomstlistor uppdateras.

Attribut	Beskrivning
GPID	Unik identifierare
FlightCode	Namn på Flight
AirportName	Namn på flygplats
Gate	Namn på gate flyget anländer till.
Time	Tid för landning

4.3.1.3 Simplan:UpdateStartime

Starttid för en aktiv simplan har ändrats. Beräkningar för simplanen måste göras på nytt.
Beräkningarna är: tid för sektorpassage, FLEG och MTCD.

Attribut	Beskrivning
GPID	Unik identifierare
EventId	ID för simplanen som är ändrad
StartTime	Ny starttid

4.3.1.4 ComputeRoute:Request

Simplan har ändrats och färdplansdata behöver beräknas på nytt.

Attribut	Beskrivning
GPID	Unik identifierare
ComputeProposed15	1 ifall alternativet ”Compute Field 15 from simplan routes” valts annars 0
SimplanEvent	Innehåller den simplan som ska beräknas
StartTime	Ny starttid för färdplan
UpdateClients	1 ifall klienter har uppdaterats annars 0

4.3.1.5 ComputeRoute:Result

Skickas som svar på ett ComputeRoute:Request, innehåller beräknad fält 15 data.

Attribut	Beskrivning
GPID	Unik identifierare
Proposed15	Beräknad Fält 15 data.
Size	Antalet noder flighten flyger igenom.
StartTime	Starttid för färdplanen
UpdateClients	1 ifall klienter har uppdaterats annars 0

4.3.1.6 ComputeRoute:Positions

Beräknar FLEG utifrån ny fält 15 data.

Attribut	Beskrivning
GPID	Unik identifierare
EventId	ID för simplanen som startar flighten
Positions	Innehåller information om rutt. Presenterad som koordinater
Size	Antalet noder i rutten

4.3.1.7 Order:RuleSet:SectorHandover

Handover påbörjas, spelpjäs överförs till en ny ägare.

Attribut	Beskrivning
Class	Namn på objektets klass. CatsEvent:Order:RuleSet: SectorHandover
eventName	Namn på eventet SectorHandover
InboundSector	Attributreferens till ”sector”. Sektorn flighten ska överföras till.
CoordinationPoint	Attributreferens till ”beacon”. Punkten där flight ska bli koordinerad. Används ej.

4.3.1.8 HandOver:HOVStart

Handover påbörjas, spelpjäs överförs till en ny ägare.

Attribut	Beskrivning
GPID	Unik identifierare
Function	Vilken funktion som har skapat eventet.
Process	Vilken service som har skapat eventet. I det här fallet: DatsFlight
Role	Rollen som äger/skickar spelpjäsen
HOVRole	Rollen som ska ta emot spelpjäsen
Type	Vilken typ av roll som tar emot spelpjäsen. I det här fallet: 2 för Trainee

Det finns två möjliga svar på ett HandOver:HOVStart, acceptera eller avböja.

4.3.1.9 HandOver:HOVAccept

Handover accepteras av den roll som spelpjäsen blir överförd till.

Attribut	Beskrivning
GPID	Unik identifierare
Autoaccepted	Har värdet 1 om överföring skedde med UseAutomaticHov, annars 0.
Function	Vilken funktion som har skapat eventet.
Process	Vilken service som har skapat eventet. I det här fallet: DatsFlight
Role	Rollen som äger/skickar spelpjäsen
HOVRole	Rollen som ska ta emot spelpjäsen
Type	Vilken typ av roll som mottar överlämningen. I det här fallet: 2 för Trainee

4.3.1.10 HandOver:HOVReject

Handover nekas, antingen av en användare eller på grund av ett fel.
Fel kan t.ex. uppstå när en handover försöker göras med en spelpjäs som redan genomgår en handover.

Attribut	Beskrivning
GPID	Unik identifierare
function	Vilken funktion som har skapat eventet. StartHOVToController
Process	Vilken service som skapar eventet. I det här fallet: DatsFlight
Role	Rollen som äger/skickar spelpjäsen
HOVRole	Rollen som ska ta emot spelpjäsen
Type	Vilken typ av roll som mottar överlämningen. I det här fallet: 2 för Trainee

4.3.1.11 Simplan:Start

En simplan startar. Ankomst och avgångslistor uppdateras och en färdplan för flighten skapas utifrån simplanen.

4.3.1.12 Report:Fpl

En ny färdplan har skapats, flighten initieras och rutt, sektorpassage, MTCD, FLEG beräknas.

5 Slutsats

Arbetssättet som beskrivs i kapitel 2 följdes under vissa perioder väldigt bra och under andra inte riktigt lika strikt. Detta varierade berodde på vilken typ av uppgift som var igång. När DatsFlight eftersöktes följdes metoden väldigt bra, veckoplanen uppdaterades och prioriterades kontinuerligt när något nytt kom upp, alla fynd som eventuellt kunde vara av nytta dokumenterades och utvärderades.
När komponenten hade hittats och funktionaliteten skulle undersökas var det svårt att veta från vilket håll problemet skulle angripas. Antingen en djupdykning ner i källkoden där samtliga klasser i DatsFlight systematiskt dokumenteras eller genom att utföra tester på simulatorn och på det sättet försöka bilda en uppfattning om var det kan vara lämpligt att börja leta.
De två momenten utfördes parallellt och i slutänden visade det sig att testning av simulatorn gav mer användbara resultat då de belyste en stor del av den praktiska funktionalitet som finns i DatsFlight.

DATS är en något ålderdomlig programvara och användargränssnittet är inte lättjobbat då det finns en stor mängd olika menyer och i menyerna finns det ytterligare listor med olika inställningsalternativ.
Den låga användarvänlighet i simulatorn är inget större problem då logiken bakom områdeskontrollsimulering finns där och det är detta som i slutändan är intressant för examensarbetet.

5.1 Resultat

Vilka uppgifter sköts av områdeskotrolltjänsten?
Den förberedande undersökningen kom fram till att områdeskontrolltjänstens uppgift är att leda trafik genom sektorerna på ett så effektivt sätt som möjligt. Detta innebär att kollisioner mellan flygplan och andra luftburna fordon ska förhindras genom att leda om trafiken på ett lämpligt sätt. Koordinera flygplan på väg en sektor genom att kommunicera med piloten om eventuella ändringar i kursen samt att kommunicera med omkringliggande sektorer.

Vilka komponenter i DATS berör Områdeskontroll?
DatsFlight är den enda komponent i DATS som behandlar områdeskontroll.

Vilken funktionalitet finns i komponenterna?
Den funktionalitet som hittades i undersökningen av DatsFlight uppfyller de delar i områdeskontroll som förundersökningen kom fram till. Där funktionalitet för att hantera hand over, kollisionsdetektering, sektorpassager, ändringar i färdplan och beräkning av FLEG och fält 15 utifrån simplaner

upptäcktes. All funktionalitet i DatsFlight finns presenterad i kapitel 4.1 där samtliga klasser och dess huvuduppgift gås igenom.

Vilka objekt används i DATSs gemensamma databas?
För att göra en simulering av områdeskontroll behövs vissa grunddata vilken finns beskriven i kapitel 4.2. Här presenteras endast de objekt som är specifika eller väldigt viktiga för DatsFlight, t.ex. Sector och Character. Flygplanspositioner och liknande data används av DatsFlight när beräkning av sektorpassager ska göras men då detta är fundamentalt för alla komponenter i simulatorn togs det inte upp.

Hur kommunicerar komponenterna?
Kommunikation mellan komponenter i DATS kartlades för att få en förståelse om vilka delar i DatsFlight som kunde tas bort inför implementeringen i ATC Training. Kommunikation mellan komponenterna i DATS sker med hjälp av de events som finns presenterade i kapitel 4.3.1.

Är det möjligt att skapa en WISE Service utifrån områdeskontroll komponenterna i DATS?
Det är fullt möjligt att göra om DATS komponenten DatsFlight till en WISE Service genom att ersätta de klasser som implementerar CATS gränssnittet med motsvarande klasser ifrån WISE. All logik för att utföra en simulering finns på plats och kommer därför inte behöva kompletteras något ytterligare. Därför är det fullt möjligt för Saab att påbörja skapandet av en WISE Service om man fortfarande anser sig ha behov av denna del.

Vilka nya objekt kommer att behöva skapas i ATC Trainings databas?
De databasposter som beskrivs i kapitel 4.2 kommer att behöva skapas i ATC Training för att kunna definiera sektorer, areor, roller, simplan och färdplan. Objekttyperna i DATSs databas kan behöva ändras för att passa den nya strukturen i ATC Training.

5.2 Framtida utveckling

För framtida utveckling skall DatsFlight implementeras i ATC Training. För att kunna porta DatsFlight och därmed slutföra arbetet behövs det först skapas skelettfiler för att ersätta de .lib filer som fattas. Skelettfilerna innehåller de tomma klasser och funktionsnamn som fattas för att kunna kompilera projektet.

Det kommer att behöva undersökas vilka klasser och funktioner som inte längre behövs eller behöver skrivas om när modulen skall användas ihop med WISE. Den funktionalitet som finns i CatsDatsFlightServer och

CatsDatsFlightServerExe implementerar gränssnittet CatsEventBusProxy för kommunikation med eventbussarna i CATS. Dessa två klasser kommer därför att behöva ersättas med motsvarande klasser för kommunikation mot Data Managern i WISE Connectivity.

Mallar för events i DatsFlight kommer att behöva skapas i ATC Trainings databas. Dessa mallar finns färdiga i BasicData och kommer därför bara att behöva föras över därifrån, en eventuell modifiering av objekttyper kan dock behöva göras. En vidare undersökning av var de event som beskrivs i kapitel 4.3.1 kommer ifrån behöver göras. Detta för att ATC Training ska kunna skapa dem och trigga funktionaliteten i DatsFlight.

Att lägga till GUI i ATC Training är det sista som behöver göras men för att kunna använda funktionaliteten i DatsFlight måste det finnas möjlighet att nå den genom programmet.

6 Terminologi

.lib fil	En biblioteksfil
Ankomstlista	En lista i DATS som visar alla flygplatsankomster som kommer göras under simulationen.
ATC - Flygkontrollenhet	Sammanfattande benämning på områdeskontrolltjänst, inflygningskontrolltjänst och flygplatskontroll-tjänst
Avgångslista	En lista i DATS som visar alla flygplatsavgångar som kommer göras under simulationen.
Beacon	Förutbestämd koordinat i kartan som anges med ett namn, t.ex. WILBUR.
Event	Ett objekt som skickas av DATS för att uppmärksamma resten av systemet på att något har hänt.
Exercise	Bygger på information av ett scenario. Innehåller samma information som scenariot tillsammans med en unik logg för de events som uppkommer under en körning av övningen
FLEG	Flight leg. Ett ben är avståndet mellan avgång och ankomst. På en resa mellan London och Chicago är London-Frankfurt ett av dessa ben.
Handover	När en flight blir överlämnad från en sektor till en annan sker en handover.
Koordinera flight	Flighten är samordnad. Flygledare har kommit överens om till vilken sektor flighten är på väg.
MTCD	Medium Term Conflict Detection. Ett system för att upptäcka konflikter mellan flighter upp till 20 minuter framåt i tiden.
Roll (i DATS)	En användares roll bestämmer vilka privilegier denne har. I DATS finnstvå olika roller, PILOT och TRAINEE. PILOT används av övningsledaren och TRAINEE av studenten.
Scenario	Mall för en exercise, innehåller all information som behövs för att kunna köra en simulation.
Simplan	All information som behövs för att beskriva en flight samlas i en simplan. Innehåller bl.a.

	resväg, höjd, marsfart.
Simulation area	Används för att gruppera grundläggande information i databasen. Nyskapade scenarion skapas utifrån den simulation area den tillhör. Viss data, såsom spelpjäser är gemensamma för hela databasen.
Skelettfil	Fil som innehåller tomma klasser.
Spelpjäs	Ett objekt som representerar flygplan, fordon, flygfält, startbanor och radarstationer.
Tickgenerator	En komponent i DATS som ser till att de olika komponenterna utför sina uppgifter.
WISE Service	Syftet med en service är att utvidga funktionaliteten i ett system genom att koppla denna till systemets databas där den kan tillföra sina ändringar och beräkningar.
Ägare	Den roll som äger ett visst objekt i en simulering.

7 Referenser

[1] Info om områdeskontroll
http://www.lfv.se/sv/Om-oss/Produktion-En-Route/Luftrum/
(09:49 11-02-2013)

[2] Föreläsning om flygtrafikledning och flygsimulering.
(27-02-2013 Stephan Svensson, Utbildare Luftfartsverket)

[3] Information om Fält 15
http://www.skybrary.aero/index.php/Flight_Plan_Filling#Item_15
(16:41 21-05-2013)

[4] Information om DATS
DATS™ System tools handbook

[5] Expertmöte, diskussion om DATS
05-04-2013 Lars Gryt, systemutvecklare och Mathias Lubera, systemutvecklare

[6] Föreläsning WISE.
(04-04-2013 Mathias Lubera systemutvecklare)

[7] WISE Connectivity SDK
03-02-2013 Saab Training and Simulation)

[8] Information om Scenario skapande i DATS
DATS™ Operators Handbook (LAKS)

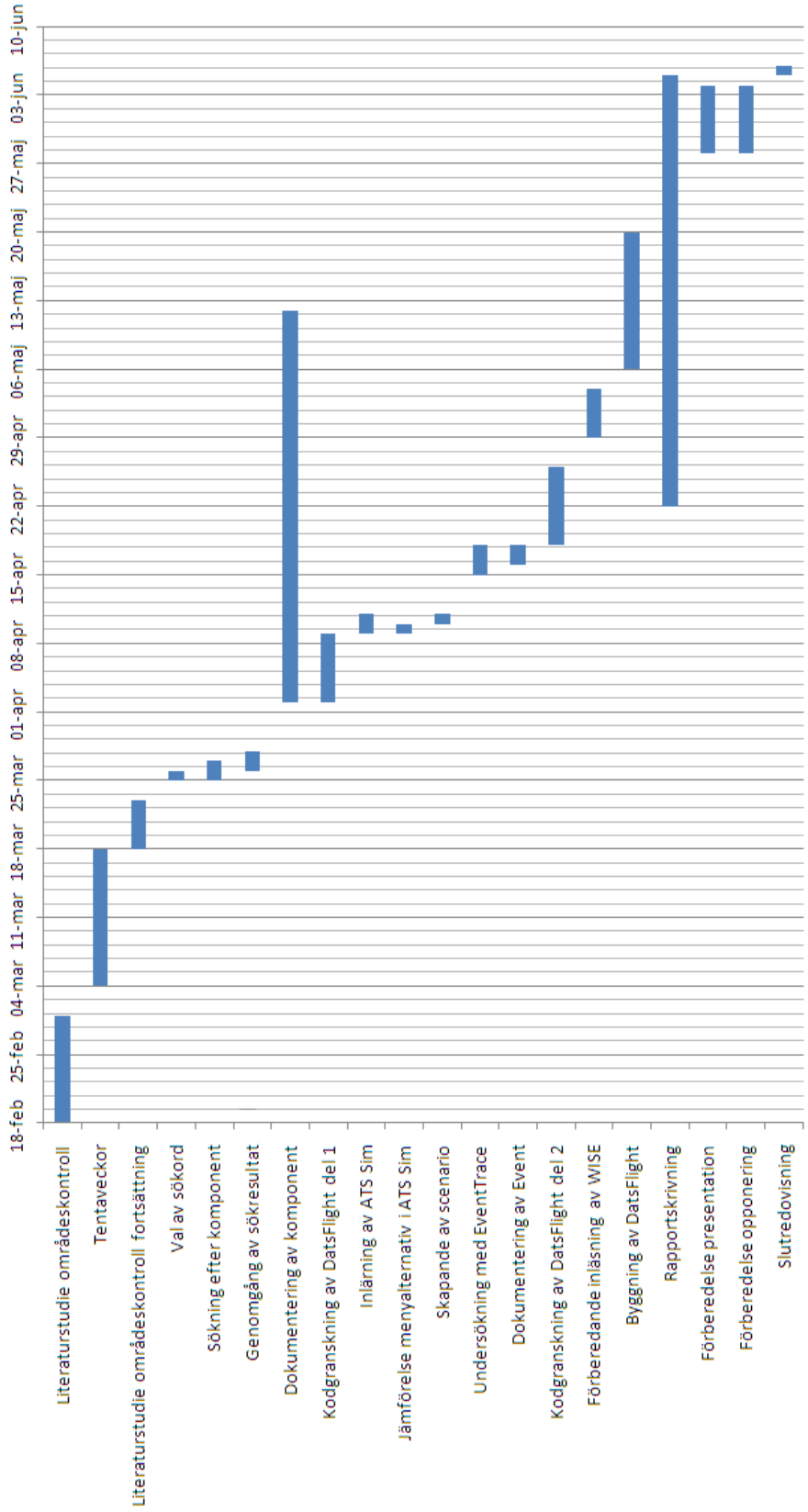
[9] Utbildning på Luftfartsverket.
http://www.lfv.se/sv/Tjanster/Utbildning
(11:25 14-05-2013)

[10] Om Skybrary
http://www.skybrary.aero/index.php/About_SKYbrary
(*16:18 25-05-2013*)

[11] Om ICAO
http://www.icao.int/
(*16:22 25-05-2013*)

8 Appendix

8.1 Tidplan



8.2 Färdplan

Objekt för färdplan i databas.

ObjectDescription	Beskrivning	Färdplan FPL
Class	Class name	CatsEvent:Report:Fpl
Role	Receiver	
Sender	DATS Sender	
Priority	DATS Priority	
Prio	Priority	
Adress	Address	
TimeNo	Sending time	
MessageSender	Sender	
MessageClass	Message type	
CallSignal	Aircraft identification	
Rules	Flight rules	
Type	Type of flight	
NoObject	Number	
ObjectName	Type of aircraft	
Turbulence	Wake turbulence category	
Equipment	Equipment	
DepAirport	Departure aerodrome	
DepTime	Time	
Descr	Cruising speed, level and route	
DestAirport	Destination aerodrome	
Eet	Total EET	
AltAirport	Alternate aerodrome	
Information	Other information	
EOBT	EOBT (rel exe start)	