Hogia Notification

Notifieringsmodul för Hogia Redovisning&Revision

LUNDS
UNIVERSITET
Lunds Tekniska Högskola

LTH Ingenjörshögskolan vid Campus Helsingborg
Datateknik

Examensarbete:
P-A Eriksson

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
Media-Tryck
Biblioteksdirektionen
Lunds universitet
Lund 2010

Sammanfattning

Hogia är ett utvecklingsföretag inom mjukvaruindustrin som bland annat levererar en webbaserad portal för redovisnings- och revisionsbranschen. Denna portal är en plattform där byråer knyter samman kommunikationen med sina kunder.

Portalen saknar i dagsläget en tjänst, som via exempelvis mail, notifierar dess användare om nya händelser som skett, såsom inkomna dokument. En sådan tjänst har varit ett önskemål från Hogias befintliga kunder.

Under examensarbetet som beskrivs i denna rapport har en modul som utför den här typen av notifieringar utvecklats i ramverket ASP.NET. Modulen består bland annat av en Windowsservice som med jämna tidsintervall anropar ett affärslager, där största delen av all logik finns. Affärslagret undersöker om någon användare ska notifieras.
Notifieringsmodulen består även av en webbservice som är modulens gränssnitt utåt, och nås av utomstående program för att lägga till händelser.
Ett webbaserat grafiskt gränssnitt har implementerats där användare, baserat på rättigheter, kan administrera olika inställningar, bland annat rörande notifieringstider. Gränssnittet erbjuder även användare olika typer av information i form av statistik.
Notifieringsmodulen har en egen databas, samt ett datalager, och är i stort sett oberoende av utomstående applikationer.

Notifieringsmodulen har utvecklats steg för steg, där utvecklandet av databasen gjordes i ett tidigt skede. Prototyper av olika delar av notifieringsmodulen, som har kunnat diskuteras med kollegor, har under projektets gång tagits fram. Dessa prototyper har ofta bestått av små Windowsapplikationer som agerat förenklade versioner av den slutgiltiga produkten.

Resultatet av detta projekt är en prototyp av en notifieringsmodul som kunden får utvärdera, och som sedan kan vidareutvecklas till en fullt fungerande produkt.

Nyckelord: Webbportal, notifiering, ASP.NET.

Abstract

Hogia is a development company in the software industry. One of the products they deliver is a web-based portal for the accounting and auditing industry. This portal is a platform for agencies linking communication with their customers.

The portal today lacks the ability of notifying customers, for instance via email, when a new event that concerns the customer has been added. This could for example be when somebody has uploaded a new document in Hogia's product Documents. Agencies have been asking for this kind of service from Hogia.

A module that delivers this kind of service has been developed in the ASP.NET framework, during the bachelor thesis which is described in this report.
This module consists of a Windows service which with a given time-interval calls methods in a business layer. The business layer contains most of the logic in the notification module and will examine the database to see if any user should be notified.
The module also contains a web service which acts as an interface for other applications to call when adding new events.
Since the notification module has its own database and data layer, it is mostly unaffected by changes made in Hogia's other applications.

The notification module has built in functionality for, among other things, scheduling the notification intervals. A web-based graphical interface has been implemented for managing these settings. The graphical interface also offers users different kinds of statistics.

Development of the notification module has been conducted using a step by step approach, where small prototypes of parts of the system have been created over time. These prototypes have been used mainly for testing purposes, but some of them have also been used as a basis for discussion with colleagues.

The result of this project is a prototype of a notification module which customers can evaluate and give feedback on. After the customer evaluation the prototype needs further development before it can be considered a fully functional product.

Keywords: Web portal, notification, ASP.NET.

Förord

Jag vill rikta ett stort tack till Martin Elvheim som har varit min handledare på Hogia, Lise Jensen som har varit min examinator på skolan och min chef Peter Flemsjö.

Jag vill även tacka Per Hultqvist, Anders Rydell och Torbjörn Rosendahl för stor hjälp i alla tekniska frågor, Olena Back, Ann-Sofie Simonsson Haasz och Anna Klavsäter Doll som har kommit med bra synpunkter och designförslag.

Sist men inte minst vill jag tacka Andreas Lundgren för goda råd under rapportskrivandet.

Innehållsförteckning

Inledning

Denna rapport beskriver examensarbetet utfört på Hogia Redovisning & Revision, där en notifieringsmodul för en befintlig webbportal har utvecklats.

Rapporten förklarar bakgrunden till arbetet och ger en introduktion till en del av Hogias verksamhet och relevanta produkter som företaget levererar. Den innehåller även en beskrivning av de definierade problem som under examensarbetet ska lösas, samt vilka tillvägagångssätt som har vidtagits för att lösa dessa.
Implementationen av den färdiga lösningen förklaras, och exempel ges på framtida vidareutvecklingsmöjligheter.

1 Bakgrund

Examensarbetet tog sin början när bolagschefen på Hogia Redovisning & Revision tog kontakt via telefon efter att ansökan om ett heltidsjobb som systemutvecklare skickats in via internet.

Två veckor efter detta skedde en arbetsintervju på ett av företagets kontor i Göteborg, och efter några dagar kom besked om att anställningen beviljats. Ytterligare några dagar efter detta, närmare bestämt den 27 oktober 2009, var det dags för första dagen på jobbet.

Hogia innefattar flertalet affärsområden, och de första veckorna ägnades mycket tid åt att ta del av information om Hogias verksamhet, få en överblick av flera av de produkter företaget levererar, och en insikt i företagets arbetssätt.

Efter dessa veckor kunde projektet rörande notifieringsmodulen påbörjas.

1.1 Hogia-gruppen

Hogia grundades 1980 av Bert-Inge Hogsved och företaget har idag totalt ca 500 anställda med kontor i Sverige, Norge och Finland. Det är fortfarande familjeägt och huvudkontoret ligger i Stenungsund.
Hogia-gruppen är uppdelad i flera mindre bolag där varje bolag är komplett och har det fulla ansvaret för allt från utveckling och försäljning till kundservicen inom sitt verksamhetsområde.

Några av de produkter som bolagen inom Hogia-gruppen utvecklar och säljer är:

- Lönesystem, Tidsredovisningssystem, Personalsystem
- Ekonomi- och affärssystem, fastighetssystem, kassasystem
- Bokslut- och analysprogram
- Logistikprogram för godstransporter
- Bokningsprogram för färjetrafik
- System för kollektivtrafik och terminalhantering

1.2 Hogia Redovisning & Revision

Bolaget som examensarbetet utförs på heter Hogia Redovisning & Revision och har i dagsläget 19 anställda. De erbjuder produkter och tjänster till redovisnings- och revisionsbranschen och har idag ca 35 % av den svenska marknaden inom området.

De produkter och tjänster som bolaget erbjuder är program för löpande redovisning, löneadministration, skatt, bokslut och revisionsuppdrag. Hogia Redovisning & Revision erbjuder alltså allt som hör till en redovisnings- och revisionsbyrås vardag. De har nära samarbete med branschorganisationerna och målsättningen är att underlätta och säkra upp en byrås dagliga arbete så att de på ett enkelt och effektivt sätt skall kunna effektivisera sitt arbete. [1]

1.3 Vad gör en redovisnings- och revisionsbyrå?

En redovisnings- och revisionsbyrå hjälper företag, och i viss mån privatpersoner, med löpande administration, deklaration och bokslut.

Många mindre bolag som inte vill, eller har bemanning att sköta sin ekonomi, kan överlåta den ekonomiska administrationen till ett företag som sköter detta åt dem. Ofta får de då en kontaktperson i form av en konsult på en redovisnings- och revisionsbyrå som sköter deras ekonomi.

De kunder som byråerna har är främst småföretagare. Småföretagarna anlitar en konsult från en redovisnings- och revisionsbyrå för att t.ex. sköta sin ekonomi, deklaration och lönehantering.

1.4 Hogia Redovisning & Revision VO KA

Hogia Redovisning & Revision VO KA är en egen avdelning av Hogia Redovisning & Revision som riktar sig mot stora byråer. VO i avdelningsnamnet står för VerksamhetsOmråde, och KA står för Key Account. Avdelningen utvecklar, säljer och lämnar support på bland annat en produkt som heter Hogia Business Service Platform. Det är en portal som beskrivs närmare i nästa kapitel.

1.5 Hogia Business Service Platform (Portalen)

Istället för att en konsult på en redovisnings- och revisionsbyrå ska maila eller posta ut t.ex. lönespecifikationer till alla anställda på ett företag så har Hogia utvecklat en portal som heter Hogia Business Service Platform, där de publicerar detta material. De anställda på småföretagen har i sin tur inloggningar för att logga in på portalen och ta del av sin information där. På så sätt blir hanteringen av bl.a. lönespecifikationer mycket säkrare, smidigare och mer strukturerad.

Hogia Business Service Platform är webbaserad och har som standard en inloggning med användarnamn och lösenord. Detta gör det möjligt för alla användare att alltid kunna komma åt och ta del av sin information från en dator med internetuppkoppling. Det skapas en kommunikationskanal mellan redovisnings- och revisionsbyråer och deras kunder.

Portalen är till största delen utvecklad i VB.NET och dess grafiska webbgränssnitt är utvecklat i ASP.NET Web Forms. De databaser som portalen använder sig av är Microsoft SQL-Server 2005 - 2008.

1.5.1 Tredjepartsapplikationer

Portalen är ett ramverk som går att utöka med andra applikationer. Dessa applikationer kan vara program som Hogia eller andra företag levererar.

Några av de applikationer som är integrerade med portalen idag är program för:

- Dokumenthantering
- Personalhantering
- Tidsrapportering
- Kassasystem
- Bokföring
- Reseräkningar
- Lönerapporter

Kunder som abonnerar på portalen köper till dessa applikationer efter eget behov.

1.5.2 Single-Sign-On
Portalen är gjord för att kunna integreras med tredjepartsapplikationer. Det finns t.ex. företag som levererar webbaserade löneberedningssystem och dessa vill man kunna integrera med portalen på ett smidigt sätt.

Single-Sign-On innebär som namnet antyder att man bara behöver logga in en gång. Har man en gång loggat in i portalen behöver man inte ange användaruppgifter igen även om man ska besöka tredjepartsapplikationer som inte är Hogias applikationer. Detta förbättrar arbetsflödet för användaren eftersom denne inte längre behöver hålla reda på flera inloggningsuppgifter, och därför lättare kan navigera mellan olika delar av systemet.

1.5.3 Säkerhet
Systemet är förberett för att kunna utöka säkerheten med alternativa inloggningar som t.ex. e-legitimation, inloggning med dosa eller SMS.
I dagsläget levererar Hogia inte någon egen alternativ inloggning, utan det är utomstående företags lösningar som får integreras som en tredjepartsapplikation med portalen. Ofta levererar sådana företag sina lösningar som ett gränssnitt som är enkelt att integreras med. Trots detta, och att portalen är förberedd för alternativa inloggningar, krävs komplettering i koden för att det ska fungera mot det gränssnitt som något utomstående företag levererar. Detta görs enbart på beställning från kund.

1.5.4 Webbparts
Portalen har en arbetsyta med så kallade webbparts. En webbpart är en komponent som kan vara integrerad med, och direkt visa information från en tredjepartsapplikation.
Med möjligheten för användaren att själv välja vilka webbparts som ska visas och fritt kunna placera dessa på arbetsytan, ges en skräddarsydd och överskådlig vy med direktåtkomst till tredjepartsapplikationerna. T.ex. kan man ha en webbpart med sina fem senaste lönespecifikationer. Ifrån denna webbpart får man direktåtkomst till specifikationerna, istället för att behöva logga in i löneprogrammet och hämta dem där.

På bilden B3 i Bilagor ser man två webbparts. Den ena har Dokument som titel och är kopplad till Documents för presentation av uppladdade dokument. Om man klickar på ett dokument får man frågan om man vill spara eller öppna det. Den andra webbparten på bilden presenterar nyheter i portalen.

2 Problembeskrivning

Hogia har mottagit önskemål från kunder om en funktion som gör att de kan bli notifierade via t.ex. mail eller SMS när någonting nytt har kommit in i portalen. Detta kan exempelvis vara en ny lönespecifikation eller en räkning som ska attesteras. Tack vare en notifieringstjänst skulle kunden direkt bli uppmärksammad på att någonting nytt har inkommit, och behöver därför inte logga in i portalen enbart i syfte att kontrollera detta. Kommunikationen mellan byrå och kund, eller kunder emellan, blir snabbare och risken minskar att kunden missar att i rätt tid ta del av viktig information.

Ett exempel på en notifieringsfunktion är på den sociala mötesplatsen Facebook på internet (www.facebook.com). Där har man möjligheten att bli notifierad via mail om en annan användare har skickat en förfrågan om att bli vän på mötesplatsen. Tack vare detta behöver man inte logga in på Facebook i syfte att se om man fått förfrågningar.

Ett annat exempel är ett tandläkarbesök. Ofta får man ett SMS dagen före med en påminnelse om tid och plats för det bokade besöket.

Hogia använder sig av VB.NET som huvudspråk och kräver att även notifieringsmodulen ska vara skriven i VB.NET. Det har därför inte varit aktuellt att undersöka om något annat programmeringsspråk skulle lämpa sig bättre för denna typ av projekt. Den databas som finns tillgänglig är Microsoft SQL Server 2008, och utvecklingsmiljön är Microsoft Visual Studio 2008. Forskning skall göras i vad kunderna vill ha och vad de skulle kunna tänkas behöva. Hogia kräver inte en fullt fungerande produkt, utan de efterfrågar en modul som kan skicka mail och som är planerad för att enkelt kunna byggas ut med flera kommunikationskanaler, såsom SMS.
Modulen skall kunna testas och utvärderas dels av anställda på Hogia, och dels av en kund. Därefter skall den kunna vidareutvecklas till en fullt fungerande produkt.

Det har inför examensarbetet inte varit aktuellt att undersöka och eventuellt licensiera befintliga lösningar av en notifieringsfunktion. Hogia vill ha en lösning som är skräddarsydd efter deras tjänster och behov, som de fritt kan vidareutveckla och sälja utan påtvingade begränsningar.

3 Produktmål

Det huvudsakliga målet med en notifieringstjänst är som tidigare nämnts att användarna ska uppmärksammas på viktiga händelser. I detta fall är också målet att öka bekvämligheten hos kunden som inte längre behöver logga in i portalen för att se om någonting viktigt har skett. Detta sparar kundens tid eftersom det minskar de väntetider som kan uppstå i arbetet.

Notifieringstjänsten som levereras i slutet av detta examensarbete skall vara utbyggbar med flera kommunikationskanaler utifrån användarens behov. Den skall också vara förberedd för att kunna presentera mer statistik.

4 Begränsningar

För att ta fram en fullt fungerande produkt krävs mycket testning och feedback från kunder. På grund av begränsad tid bestämdes tidigt att detta inte skulle hinnas med under examensarbetet. Därför prioriterades vissa funktioner bort, och produkten begränsades till en fungerande prototyp som kunderna kan använda för testning. Efter examensarbetet kommer produkten att vidareutvecklas, så målet har varit att under examensarbetet ta fram en första version av en notifieringsmodul.

Denna första version är begränsad till att bara notifiera via mail, och bara för händelser från en enda applikation. Den kommer heller inte att innehålla ett fullt fungerande system för felhantering eller en komplett implementation av statistik.
Fördelarna med en begränsad version är att grundidén är densamma, men kunden får snabbare en övergripande förståelse för hur notifieringstjänsten kommer att fungera. I samband med detta blir det lättare för dem att testa, och de ger feedback på det som är mer relevant inför vidareutveckling. Felsökningen underlättas också då felkällorna är färre, samt att eventuella ändringar av funktionalitet inte blir lika omfattande.

5 Arbetsmetod

Notifieringsmodulen har planerats och byggts upp steg för steg sedan projektets start. Prototyper för visning, testning och diskussion har tagits fram med jämna mellanrum för att successivt bredda förståelsen för hur slutprodukten kommer att se ut. Dessa prototyper har ofta bestått av små Windowsapplikationer. En stor fördel med dessa har varit att testning av funktionalitet kunnat utföras utan att behöva ta hänsyn till design. En annan fördel har varit att dessa applikationer kunnat simulera t.ex. inkomna händelser från andra program.
Uppdelningen i sådana iterationer finns som en viktig del i många projektmodeller, exempelvis Scrum där iterationer tillämpas strikt över en bestämd tidsperiod, och kan innefatta ett helt team av medarbetare. [10] Hogia har dock inte krävt att arbetet ska följa en specifik modell, utan ett mer fritt utvecklande har tillämpats.

Databasen ansågs som en viktig del i början av projektet. Mycket tid och diskussioner lades på denna, tills en godtagbar lösning tagits fram, innan utveckling av övriga delar i projektet påbörjades.

Den grafiska designen kom mycket sent i projektet. Detta för att det i projektets tidigare skede var lättare att se hur logiken bakom notifieringsmodulen skulle fungera. Dock har diskussioner under utvecklandet av det grafiska gränssnittet resulterat i ändringar av viss funktionalitet.

6 Implementation

Arbetet inleddes med att framställa en skiss tillsammans med handledare på Hogia, ett flödesdiagram över hur de olika delarna i projektet skulle hänga ihop. Den ursprungliga lösningen som började implementeras visade sig dock senare inte optimal då den lösningen förutsatte att applikationen själv läste av databaserna från utomstående applikationer. Detta skulle innebära att notifieringsmodulen hade blivit mycket känslig för ändringar i databaser hos utomstående program, vilket i värsta fall skulle kunna leda till att notifieringsmodulen kraschar. För att undvika detta skulle omskrivning av notifieringsmodulen bli nödvändig för varje ändring av ett utomstående program.

Efter diskussioner med en kollega konstruerades en mer optimal lösning där utgångspunkten är att notifieringsmodulen skall vara så fristående som möjligt. Detta innebär istället att utomstående program får anpassas för att kunna kommunicera med notifieringsmodulen. Fördelen med detta är att notifieringsmodulen inte kommer att behöva skrivas om i syfte att vara kompatibel med nya eller förändrade utomstående applikationer.

Under projektets gång har flera testprogram skapats. Dessa har varit till stor hjälp under utvecklandet av notifieringsmodulen, bland annat för att testa funktionaliteten hos olika delar av modulen. Några av dessa testprogram är:

- En webbapplikation som visar en överskådlig vy över alla tabeller i databasen och dess innehåll.
- En Windowsapplikation som simulerar att händelser kommit in i Hogias Documents.
- En Windowsapplikation som direkt notifierar alla som väntar i kö på att bli notifierade oavsett tid.
- En Windowsapplikation för att administrera mallen för hur ett mail ska se ut.
- En Windowsapplikation för att administrera tidsinställningar.

En stor fördel med att konstruera sådana testprogram är att de kan utgöra en liten del av ett stort projekt, som kan testas oberoende av andra delar i projektet. Eftersom det är ett testprogram behöver inte hänsyn tas till estetiska detaljer, detta sparar många gånger mycket tid.

Den nuvarande lösningen består av nedanstående komponenter som beskrivs i efterföljande kapitel:

- Databas
- Datalager
- Affärslager
- Windowsservice
- Webbservice
- Webbklient

Se bild 1 för en grafisk beskrivning om hur komponenterna kommunicerar med varandra.

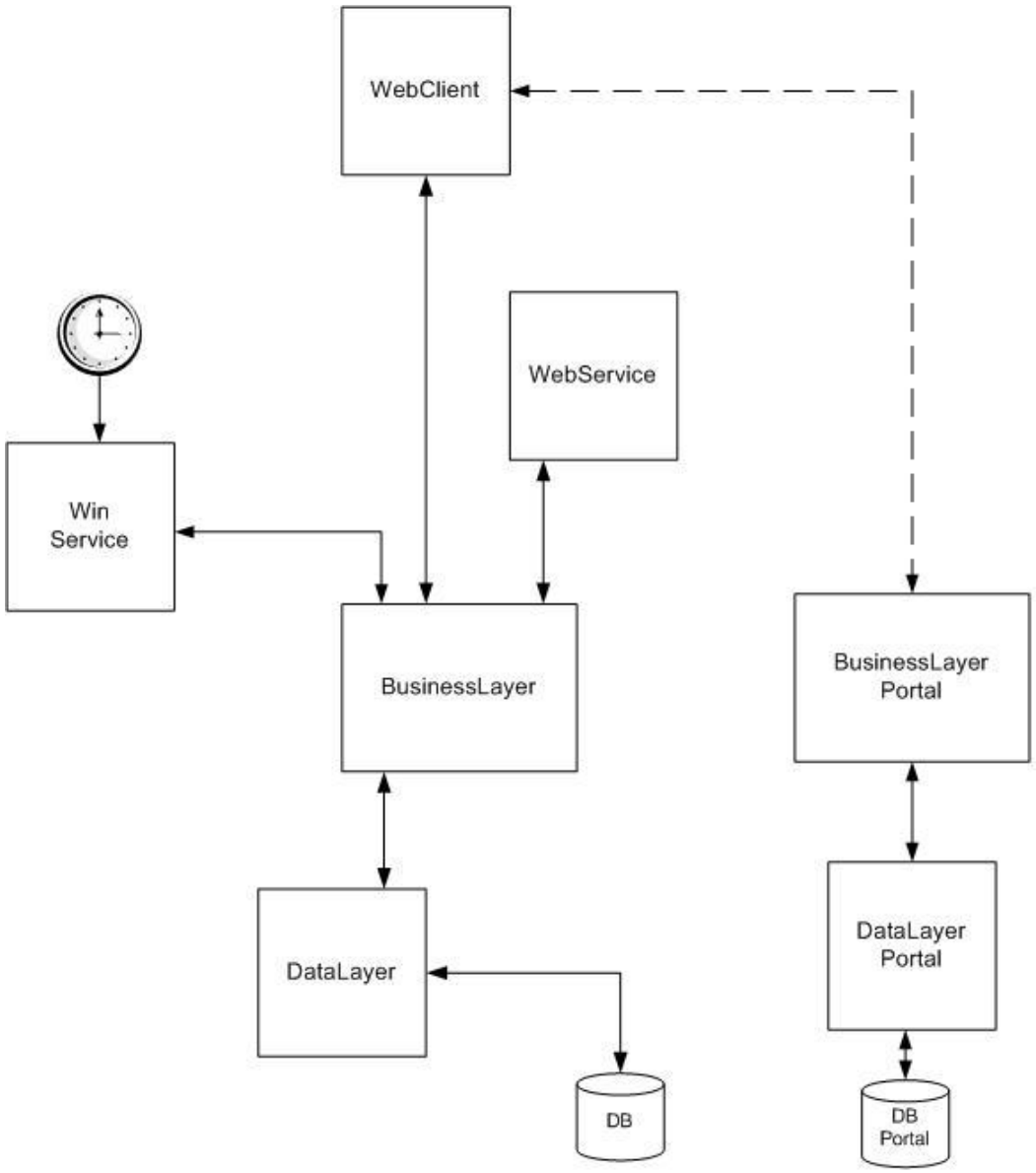

Bild 1: Flödesdiagrammet beskriver hur de olika delarna i notifieringsmodulen kommunicerar med varandra.

Portalens databas håller alla portalens användare, och det är även med dessa användare man loggar in i notifieringsmodulens grafiska gränssnitt för att administrera inställningar. Det grafiska gränssnittet beskrivs i kapitel 7 Grafiskt gränssnitt.

6.1 Komponenter

6.1.1 Databas

Microsoft Visio har använts som verktyg för att skissa på databasdesigner. Visio är ett utmärkt verktyg för att snabbt ta fram skisser, och är en del av Office-paketet, vilket Hogia innehar licenser för. Dessa skisser har diskuterats med kollegor och steg för steg har förbättringar gjorts och nya skisser tagits fram. Inte förrän en godtycklig lösning funnits, har utvecklandet av en riktig Microsoft SQL-databas påbörjats.

Databasen består i dagsläget av 14 tabeller, se bild B10 i Bilagor.

Många tabeller i databasen är normaliserade. Normalisering innebär att data i vissa kolumner bryts ut till egna tabeller och ersätts med referenser till data i dessa fristående tabeller. Ett exempel är StatisticsReceiverID i tabellen StatisticsEvent. Denna kolumn innehåller ett id som är en referens till en rad i tabellen StatisticsReceiver. På detta sätt lagras själva e-postadressen som en sträng bara på ett ställe och på alla andra ställen lagras en referens i form av ett siffertal som tar avsevärt mycket mindre plats än att lagra hela e-postadressen flera gånger.
Detta har störst effekt i tabellen StatisticsEvent som är den enda tabell där stora mängder data kommer att lagras, som inte automatiskt rensas bort av notifieringsmodulen.
Normalisering har också en annan stor fördel. Om e-postadressen skulle komma att ändras behöver den bara ändras på ett ställe.

6.1.2 Datalager

Datalagret är kopplingen mellan databasen och affärslagret. Det innehåller kod som kan anropas från affärslagret för att enkelt kunna läsa, skriva och uppdatera innehållet i databasen.

Datalagret som används i notifieringsmodulen är autogenerat med hjälp av CodeSmith och NetTiers som mall. Det innebär att det utifrån databasen genereras lagrade procedurer och funktioner som kan anropas från ASP.NET.

6.1.3 Affärslager

Den största delen av notifieringsmodulens logik finns i affärslagret. Affärslagret innehåller klasser och metoder som anropas av notifieringsmodulens övriga komponenter, exempelvis webbservice, Windowsservice och webbklient.
Nedan beskrivs, som ett exempel, metoden AddNotification som anropas utifrån av webbservicen för att lägga till en notifiering.

```
AddNotification(String sourceApplicationName, String eventTypeName,
String receiverEmail, String companyIdentityNumber, String message)
```

- sourceApplicationName: Namnet på applikationen händelsen kommer ifrån, exempelvis ”Documents”
- eventTypeName: Typen av händelse, exempel ”Nytt dokument” eller ”Status uppdaterad”
- receiverEmail: Mailadress till den användare som ska bli notifierad
- companyIdentityNumber: Organisationsnumret på företaget som notifieringen gäller
- message: Meddelande som kommer med i notifieringsmailet, exempel från Documents skulle kunna vara rubriken på dokumentet som lagts till.

När denna metod anropas utförs operationer i affärslagret för att utifrån de inställningar en användare har räkna ut vilken tid notifieringen ska skickas, och lägga detta i en kölista för notifieringar.

6.1.4 Windowsservice

Windowsservicen är en process som körs i bakgrunden på servern där notifieringsmodulen är installerad, och med ett bestämt tidsintervall anropar komponenten i notifieringsmodulen som kollar om det är någon som behöver notifieras just nu. Komponenten i detta fall är en metod i affärslagret. För en mer detaljerad beskrivning av affärslagret, se kapitel 6.1.3 Affärslager.

6.1.5 Webbservice

Webbservicen är det gränssnitt som kan nås utifrån, av andra applikationer, exempelvis Documents, för att lägga in en händelse som en användare ska bli notifierad om.
En webbservice används i detta fall eftersom det är en standard som är plattformsoberoende, och är därför enkelt tillgänglig för de flesta utomstående klienter. I nuläget innehåller webbservicen bara den metod som anropas för att lägga till en notifiering.

6.1.6 Webbklient

Webbklienten är notifieringsmodulens grafiska gränssnitt och är det enda en vanlig användare kommer att se av notifieringsmodulen. Denna beskrivs närmare i kapitel 7 Grafiskt gränssnitt.

6.2 Statistik

Att implementera fullt fungerande statistik har inte varit en del av examensarbetet, men notifieringsmodulen lagrar en del av de data som behövs för att presentera statistik i framtiden. Dessa data består av en tidsstämpel för när meddelandet skulle skickas, till vilken användare och företag och vilket kommunikationssätt som användes. Se tabellen StatisticsEvent i bilden över databasdesignen, B10 i Bilagor.
Se även kapitel 9.1 Statistik, för vidare diskussion om framtida utveckling rörande statistik.

7 Grafiskt gränssnitt

Notifieringsmodulens grafiska gränssnitt är byggt med bland andra följande teknologier:

- Web Forms
- HTML
- CSS
- AJAX

För en vanlig användare erbjuder gränssnittet inställningar för om och när notifieringar ska skickas. En användare med administratörsrättigheter har tillgång till ytterligare inställningar, såsom att administrera standardvärden, och att redigera hur ett notifieringsmail ska se ut. Administratörer kan även ta del av olika typer av statistik rörande notifieringsmodulen.

7.1 Inloggning

Notifieringsmodulen har en egen inloggningssida, se bild 2, där man loggar in för att få åtkomst till dess grafiska gränssnitt.

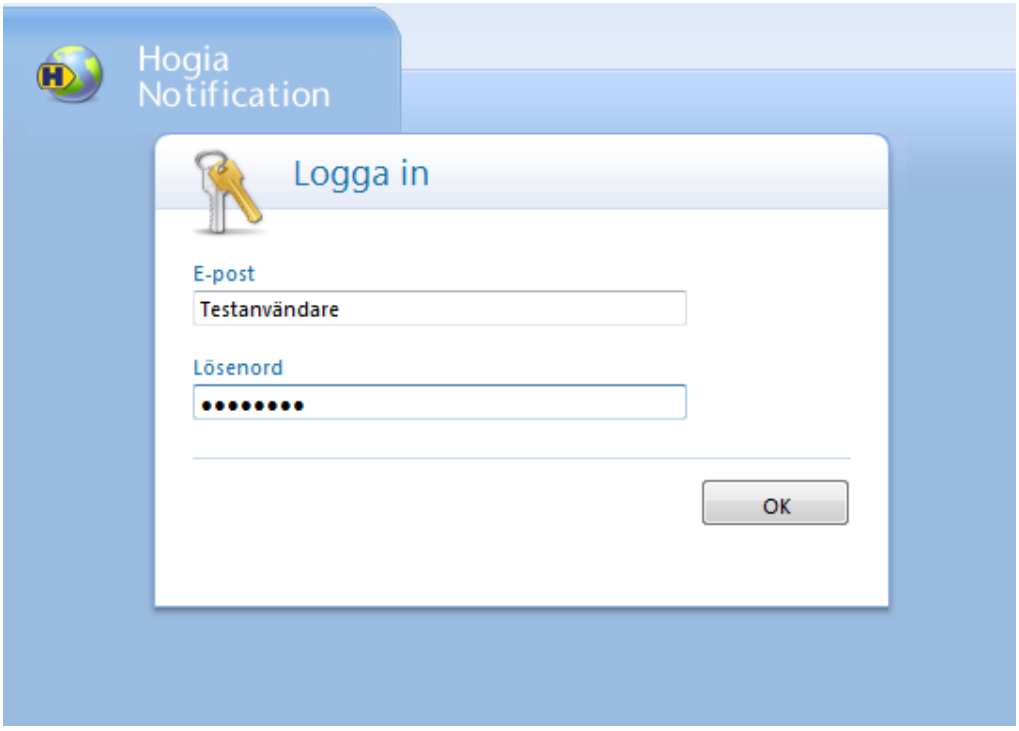

Bild 2: Bilden visar en skärmdump med inloggningssidan till notifieringsmodulen.

Användaruppgifterna är de samma som används för att logga in i Hogia Business Service Platform (Portalen).
Portalen har dock som tidigare nämnts ett Single-Sign-On-system, vilket innebär att när användaren väl har loggat in i portalen, så kommer denne inte att behöva ange sina användaruppgifter igen för att nå notifieringsmodulens grafiska gränssnitt. Dessa nås istället via en direktlänk i portalen.

7.2 Inställningsmöjligheter

7.2.1 Mall för meddelande

En administratör kan via det grafiska gränssnittet anpassa hur notifieringsmailen ska se ut genom att redigera en mall, se bild B8 i Bilagor.

För att underlätta ändringar är gränssnittet utformat som ett mail, och nedbrutet i sektionerna Ämne, Rubrik, Händelse och Sidfot.
Ämne är det som hamnar i ämnesraden på mailen, dvs. det första användaren ser innan denne har öppnat ett notifieringsmail.
Rubrik och Sidfot är det som står längst upp, respektive längst ner i ett notifieringsmail. Händelse kan upprepas flera gånger däremellan, baserat på antalet händelser som mailet ska notifiera om.

I fälten Ämne, Rubrik och Sidfot kan kommandot [Count] skrivas, detta byts i mailet ut mot antalet händelser som berörs.
I fältet Händelse kan kommandona [Application], [EventType] och [Message] skrivas. [Application] byts ut mot den applikation händelsen kommer ifrån, t.ex. Documents som är en av Hogias applikationer. [EventType] byts ut mot vilken typ av händelse det gäller, ex. Nytt dokument. [Message] byts ut mot ett meddelande som kommer från applikationen, detta kan t.ex. vara vilken användare som har lagt upp dokumentet och rubriken på detta.
Längst ner på inställningssidan illustreras ett verklighetsbaserat exempel på hur mailet kommer att se ut där alla kommandon är utbytta mot exempeldata.

Se även bild B1 i Bilagor, för ett illustrerande exempel på hur ett riktigt mail från notifieringsmodulen, öppnat i mailklienten Microsoft Outlook, kan se ut.

7.2.2 Tidsinställningar

Notifieringsmodulens huvudsyfte är att fungera som ett hjälpmedel. För att undvika att notifieringar blir ett störande moment för kunden, finns möjligheten att begränsa antalet notifieringar, och tider på dygnet de ska skickas ut.
De inställningar som finns berör mellan vilka klockslag och dagar i veckan notifieringar får komma, samt intervallet mellan utskicken.

Ett exempel skulle kunna vara att man bara vill ha notifieringar på arbetstid. Via gränssnittet kan man då ställa in att notifieringar enbart får komma måndag till fredag mellan kl. 07:00 och 16:00. De händelser som inträffar utanför denna tidsram skickas vid nästa tillåtna tid.
För att undvika att många notifieringar kommer på rad finns möjligheten att sätta ett intervall med en gräns för hur ofta notifieringar får komma. Istället för att en notifiering kommer för varje enskild händelse kan då flera händelser samlas ihop och skickas i ett och samma mail.

De globala inställningarna sätts av en administratör och agerar som standardvärden för alla nya användare som ansluter sig till tjänsten. Varje användare har i sin tur möjligheten att avvika från standardinställningarna och ha sina personliga inställningar istället. Se bilderna B4 till och med B7 i Bilagor.

7.3 Layout

Notifieringsmodulens grafiska gränssnitt levereras med ett standardutseende som presenteras på bilderna i detta dokument, men har liksom portalen stora möjligheter att layoutas om efter kundens önskemål. Vissa kunder vill ha ett skräddarsytt utseendet på portalen för att den ska följa deras tema med färger och loggor osv. Detta görs då på beställning från kunden och kan på samma sätt göras för notifieringsmodulen.

8 Slutsatser och personliga reflektioner

Den levererade notifieringsmodulen uppfyller de mål som satts upp, och erbjuder den funktionalitet som önskas, inom ramen av examensarbetet.

Förhoppningen är att Hogias kunder blir nöjda med produkten som levereras för testning, och att de vidareutvecklingsmöjligheter som finns uppmärksammas så att produkten i framtiden effektiviserar deras arbete.

8.1 Personliga mål

Det har inte bara varit kundens önskemål som skulle tillgodoses med detta examensarbete, utan det har även funnits personliga vinningar att göra. De främsta personliga målen som har varit aktuella är att få ökade kunskaper om programmering i allmänhet, och Visual Basic och .NET i synnerhet. Att komma in i arbetslivet och skaffa arbetslivserfarenhet har också varit viktiga personliga mål, samt att avsluta utbildningen och få ut en examen som färdigutbildad högskoleingenjör.

Examensarbetet har inneburit en brant utvecklingskurva, som medfört mycket ny kunskap samt insikt i arbetet på ett större utvecklingsföretag inom teknikbranschen. De personliga målen har således uppfyllts med råge, och om samma projekt skulle utföras på nytt, skulle det kunna slutföras på en bråkdel av tiden.

Ett bra sätt att lära sig VB.NET har varit att följa utbildningsfilmer som bland annat kan hittas på http://msdn.microsoft.com/ .
Det har också varit bra att sätta upp mål för enkla program och stegvis öka svårighetsgraden på dessa. Ett första mål i VB.NET kan vara så enkelt som att skapa en hemsida som läser och skriver till en databas. Nästa steg kan vara att implementera AJAX.

8.2 Programmering

Att kontinuerligt göra enhetstester under utvecklandet kan spara mycket felsökning och frustrationer mot slutet av ett projekt. Användandet av små testprogram har varit en viktig del under utvecklandet av notifieringsmodulen. Att låta utomstående personer användartesta och komma med synpunkter har också varit avgörande för kvaliteten, både på funktionalitet och grafisk design, på den i slutet av examensarbetet levererade prototypen.

Att ha långa metoder som gör många saker, och som behöver kommenteras mycket för att man ska förstå vad de gör, är inte att rekommendera. Ett bättre sätt att programmera är att försöka hålla nere metoderna så att de helst bara gör en sak var. På så vis kan ett metodnamn väljas som förklarar vad metoden gör, och kommentarer kommer inte att behövas i samma utsträckning. Koden blir renare, lättare att förstå och underhålla. Metoder som bara gör en sak blir också lättare att återanvända på flera ställen.

8.3 Tidplan

Det togs fram en ursprunglig tidplan för examensarbetet, se bild B12 i Bilagor. Detta var en grov uppskattning eftersom den gjordes i ett tidigt skede av examensarbetet. På grund av att fler uppdrag och arbetsuppgifter än väntat, som inte var relaterade till examensarbetet dök upp, drog examensarbetet ut på tiden och denna tidplan kunde inte följas. Upplägget har dock i stort sett varit detsamma och en ny tidplan, som representerar hur arbetet verkligen bedrevs, skapades i slutet av arbetet, se bild B13 i Bilagor.

Den mest uppenbara skillnaden mellan tidplanerna är att den slutgiltiga tidplanen spänner över fler veckor. Största orsaken till detta är att många andra arbetsuppgifter har utförts på företaget.
Något som också kan uppmärksammas är att den ursprungliga tidplanen saknar det juluppehåll som togs.

Att som i den ursprungliga tidplanen lägga en längre period mot slutet på testning, blev inte aktuellt efter att det beslutats att modulen skulle begränsas till att levereras som en prototyp istället för en fullt fungerande produkt.

Skrivandet av rapporten fortlöpte inte med jämna intervall som det från början var tänkt, utan större delen av rapportskrivandet skedde mot slutet. Dock har statusrapporter, som skrivits varje vecka, varit ett bra underlag för den slutgiltiga rapporten.

Bortsett från ovan nämnda skillnader följdes den ursprungliga tidplanen relativt bra.

8.4 Problem

Under examensarbetet har diverse problem stötts på, och många utav dessa har fördröjt examensarbetets flöde.
Den största bidragande faktorn till att examensarbetet tagit längre tid än normalt, förutom att övriga arbetsuppgifter har utförts på företaget, har varit bristande kunskap. Det faktum att Visual Basic, hela .NET-ramverket och utvecklingsmiljön Visual Studio var helt främmande från början, innebar att mycket tid lades ned på att skaffa kunskap och erfarenhet i dessa teknologier.

Det har under projektets gång stundtals tagit lång tid att få svar på tekniska frågor på företaget. Detta har inte bara varit negativt utan har i många fall resulterat i att egen forskning har gjorts, och lett till att mycket ny kunskap har inhämtats.

Problem stöttes också på när grundidén bakom första databasdesignen, som tidigare nämnts, visade sig opassande för denna typ av projekt. Detta resulterade i att databasdesignen fick göras om från grunden.

Det har varit svårt att skapa en grafisk design som är estetiskt tilltalande, och som samtidigt inte ger användaren en missvisande bild av notifieringsmodulens funktionalitet. Designen har diskuterats mycket med kollegor och användartester har gjorts. Dessa användartester har utförts på personer som sedan tidigare inte haft insikt i hur notifieringsmodulen fungerar. Stor vikt har lagts på personernas första intryck. Den grafiska designen har, baserat på testpersonernas synpunkter, omarbetats ett antal gånger för att uppnå en så användarvänlig miljö som möjligt.

9 Framtida vidareutveckling

Trots att en fungerande prototyp av en notifieringsmodul leveras i slutet av examensarbetet, så lämnas stora möjligheter till framtida utveckling.

Att kunna skräddarsy mailen så att de blir mer estetiskt tilltalande med hjälp av HTML-formaterat innehåll, är en av dessa framtida möjligheter, och skulle ge ett mer professionellt intryck hos slutanvändaren.

Att bli notifierad via flera olika kommunikationskanaler, såsom SMS, är också en framtida möjlighet. SMS kan komma att bli en stor fördel, speciellt för kunder där datoranvändande inte är en del av vardagen. Kunden kanske även vill bli notifierad via flera kanaler samtidigt, eller olika kanaler beroende på tid på dygnet. Eftersom detta varit känt redan från början av utvecklandet, så har notifieringsmodulen byggts på ett sådant sätt att utökandet av flera kanaler ska bli så enkelt som möjligt.
Fler kommunikationskanaler innebär ett utökande av det grafiska gränssnittet som möjliggör inställningar för dessa. Logik för att hantera inställningar, samt stöd för utskick via den nya kommunikationskanalen behöver läggas till, men ändringar i kärna och databas skall inte behövas.

När flera program utvecklas för att kunna integreras med notifieringsmodulen kommer förmodligen ett gränssnitt önskas, där användaren kan administrera från vilka program och vilka händelser i dessa program denne vill bli notifierad om.

Felhantering är viktigt, både för att minska användarfel och för att visa relevanta felmeddelanden om någonting går fel. Eftersom målet har varit att ta fram en prototyp som kunder kan testa, har detta inte prioriterats, och ett bättre felhanteringssystem står därför på listan över framtida vidareutvecklingsmöjligheter.

9.1 Statistik

Implementation av en komplett statistikdel har inte ingått i examensarbetet, men statistik är viktigt, sett från flera aspekter. En viktig del kan vara att få information om hur många och vilka som använder tjänsten, hur många mailutskick som görs i snitt varje timme eller vecka etc.

En annan viktig del kan vara att få information om det finns mail som inte kunnat skickas, hur lång fördröjning det i värsta fall har varit innan mail som legat i kö kunnat skickas iväg.

För att kunna förbättra det grafiska gränssnittet kan man också tänka sig att eventuella användarfel loggas och visas i form av statistik. Många användarfel på samma ställe kan ge en hint om att gränssnittet inte är logiskt för användaren, och utifrån detta kan förbättringar göras.

I framtiden kommer möjligtvis också rapporter med olika betalningsplaner för kunder att skapas med notifieringsmodulens statistik som grund. Ett exempel kan vara där kunden betalar för varje SMS som skickas av notifieringsmodulen. Statistiken över skickade SMS används då för att beräkna vad kunden skall betala för denna tjänst.
Just nu diskuteras på företaget vilka punkter som är mest relevanta att presentera för kunden vid visning av notifieringsmodulen. Några av dessa tänkbara punkter är:

- Hur länge tjänsten har varit aktiv för kunden
- Totalt antal skickade meddelanden sedan tjänsten togs i bruk
- Totalt antal skickade meddelanden det senaste dygnet
- Antal meddelanden som väntar på att bli skickade just nu
- Antal skickade meddelanden per tidsenhet i snitt
- Antal användare av tjänsten
- Antal användare med personliga inställningar

10 Terminologi

Term	Förklaring
SQL	Står för Structured Query Language, och är en ANSI-standard som låter dig komma åt och manipulera databaser. [5]
VB .NET	Står för Visual Basic .NET och är ett programmeringsspråk utvecklat av Microsoft, och är en av huvudbeståndsdelarna i .NET-ramverket. [6]
ASP.NET	ASP.NET är utvecklat av Microsoft och används för att skapa webbsidor. ASP står för Active Server Pages och ASP.NET utgör den största delen av .NET-ramverket. [6]
.NET	Microsofts .NET-ramverk är en miljö för att bygga och köra webbapplikationer och webbservicar. [6]
HTML	Står för HyperText Markup Language och är det mest använda språket för att skapa webbsidor. [3]
CSS	Står för Cascading Style Sheets, på svenska stilmall, och är ett språk för att beskriva hur ett dokument, exempelvis ett HTML-dokument, ska presenteras grafiskt. [2]
CodeSmith	Ett verktyg som kan användas vid mjukvaruutveckling. Verktyget genererar automatiskt, genom att använda sig av olika mallar, programkod och har stöd för att generera kod för ett flertal olika programmeringsspråk, däribland VB .NET. [7]
NetTiers	Är en uppsättnings mallar som används av CodeSmith för att automatiskt generera programkod. [8]
Web Forms	Är en del av ASP.NET. Det används för att skapa hemsidor. Web Forms är mycket likt HTML, men är uppdelat i två delar, en del som beskriver grafiken, och en del som utgör logiken. [9]
DB	Databas
Lagrad procedur	SQL-kod som är lagrad som en funktion i databasen. Istället för att SQL-kod skickas ner till databasen så anropas en funktion i databasen där SQL-koden redan finns. Kan öka prestandan. [4]
Documents	Rättighetsbaserad produkt från Hogia där användare kan spara och hämta dokument. Documents finns både som en Windows- och en webbapplikation.
Microsoft Outlook	Mailklient för Windows, utvecklad av Microsoft.
PHP	Står för PHP: Hypertext Preprocessor och är ett skriptspråk och kan användas för att skapa dynamiska

	webbsidor. [11]
AJAX	Står för Asynchronous JavaScript and XML och används för att dynamiskt kunna uppdatera delar av en webbsida utan att ladda om hela sidan. [12]

11 Referenser

[1] Hogia Redovisning & Revision: Hogia Redovisning & Revision - program för att underlätta branschens arbete. Hämtad den 28 april 2010 från: http://hogia.se/ror/

[2] w3schools.com: (1999 - 2009). CSS Introduction. Hämtad den 2 maj 2010 från: http://www.w3schools.com/css/css_intro.asp

[3] w3schools.com: (1999 - 2009). Introduction to HTML. Hämtad den 2 maj 2010 från: http://www.w3schools.com/htmL/html_intro.asp

[4] Microsoft Corporation: (2010). SQL Stored Procedures. Hämtad den 15 maj 2010 från: http://msdn.microsoft.com/en-us/library/aa174792%28SQL.80%29.aspx

[5] w3schools.com: (1999 - 2010). Introduction to SQL. Hämtad den 18 maj 2010 från: http://www.w3schools.com/sql/sql_intro.asp

[6] w3schools.com: (1999 - 2010). ASP.NET Introduction. Hämtad den 18 maj 2010 från: http://www.w3schools.com/aspnet/aspnet_intro.asp

[7] CodeSmith Tools, LLC: (2010). The best .NET, C#, VB, SQL and more template based source code generator tool. Hämtad den 18 maj 2010 från: http://www.codesmithtools.com/

[8] CodeSmith Tools, LLC: (2010). CodeSmith Framework Template Sets. Hämtad den 18 maj 2010 från: http://www.codesmithtools.com/features/frameworks.aspx#nettiers

[9] Microsoft Corporation: (2010). Introduction to ASP.NET and Web Forms. Hämtad den 18 maj 2010 från: http://msdn.microsoft.com/en-us/library/ms973868.aspx

[10] Scrum Alliance: (2009). Scrum Alliance -What Is Scrum? Hämtad den 23 maj 2010 från: http://www.scrumalliance.org/learn_about_scrum

[11] The PHP Group: (2001 - 2010). PHP: Hypertext Preprocessor
Hämtad den 26 maj 2010 från: http://php.net/

[12] w3schools.com: (1999 - 2010). AJAX Tutorial. Hämtad den 26 maj 2010 från: http://www.w3schools.com/ajax/default.asp

Bilagor

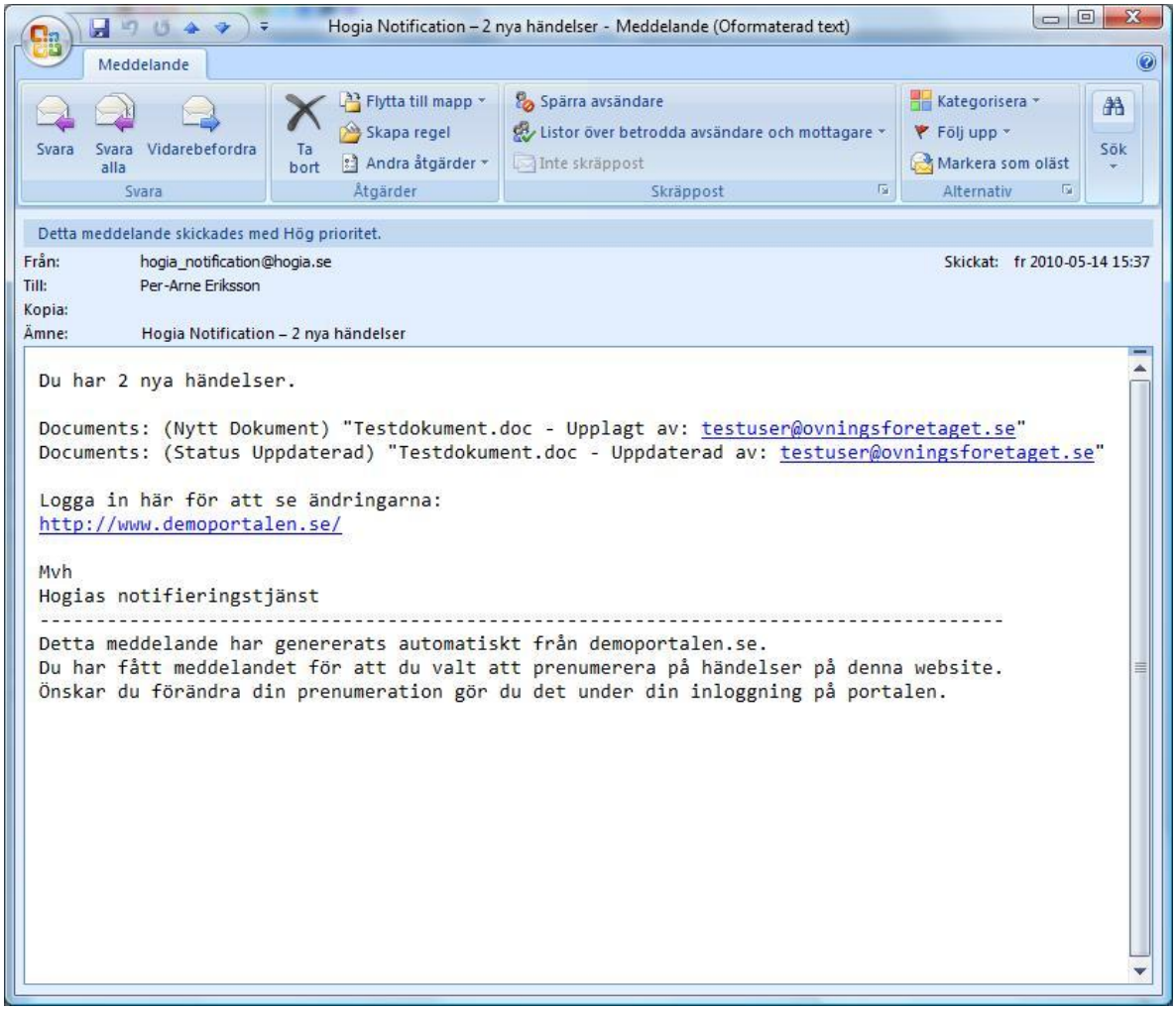

Bild B1: Exempel på hur ett notifieringsmail kan se ut, öppnat i mailklienten Microsoft Outlook.

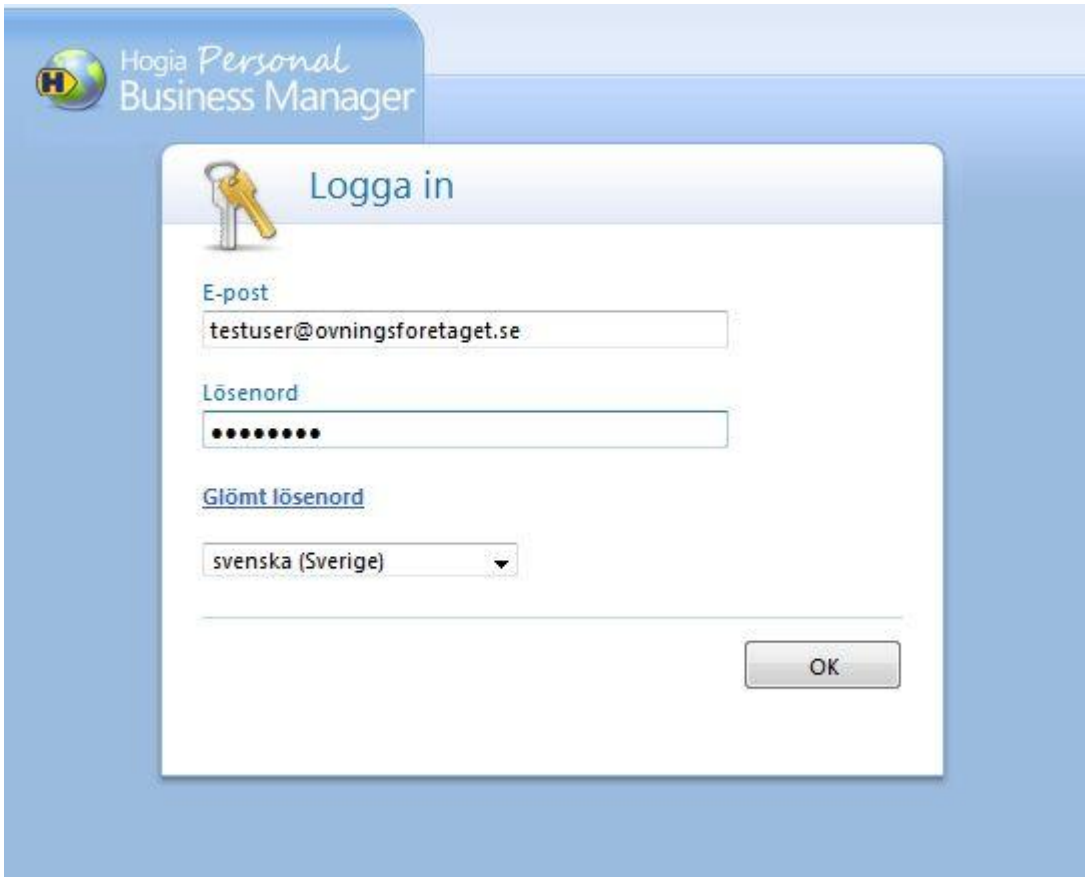

Bild B2: Portalens inloggningsdialog.

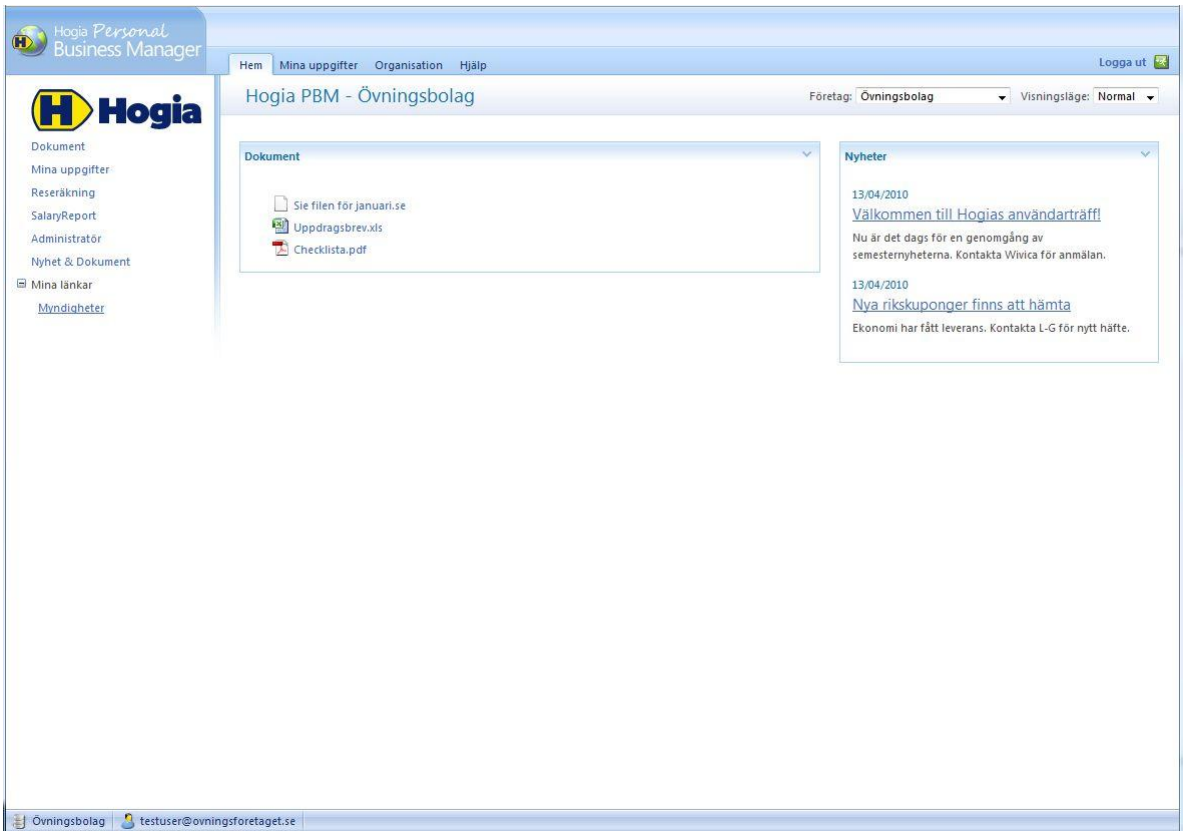

Bild B3: Portalen i dess standardutförande, dvs. utan skräddarsydd layout för ett företag.

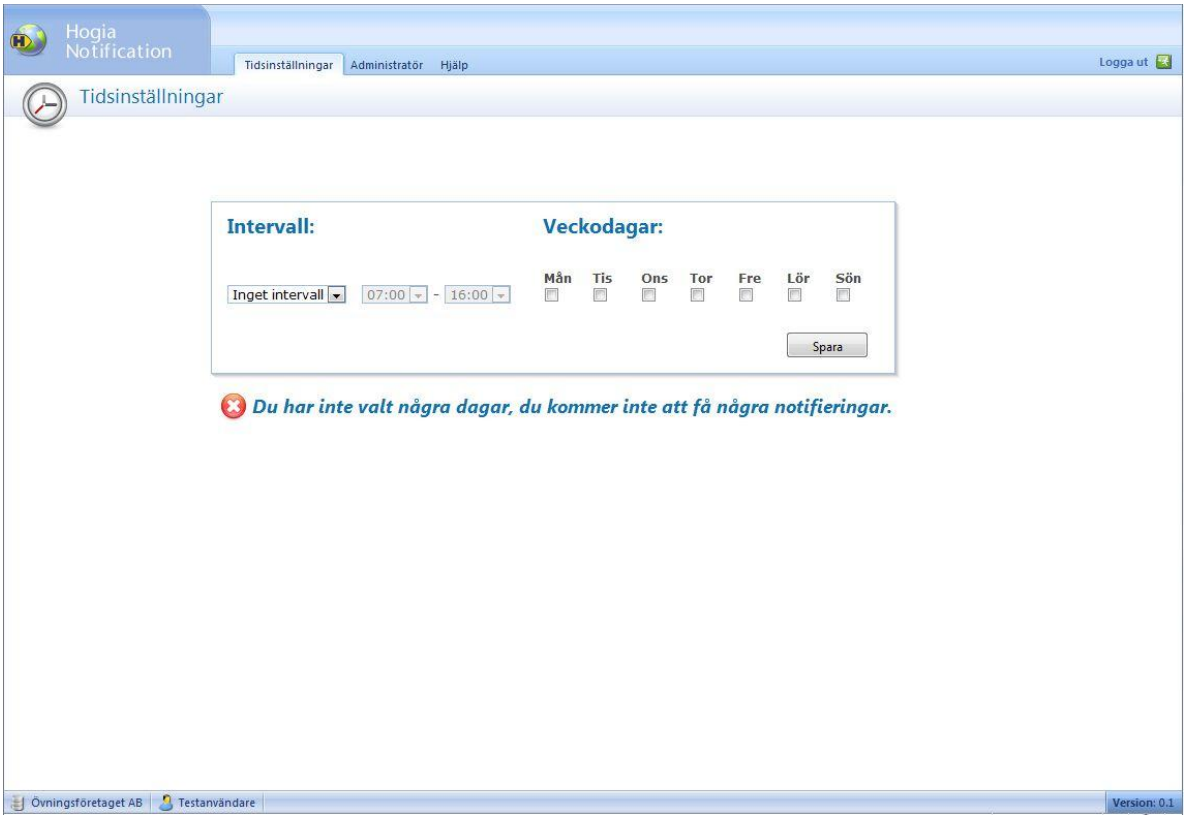

Bild B4: Fliken tidsinställningar för en användare, i notifieringsmodulens grafiska gränssnitt.

Bild B5: Exempel på val gjort i tidsinställningar.

Bild B6: Exempel på val gjort i tidsinställningar.

Bild B7: Exempel på val gjort i tidsinställningar.

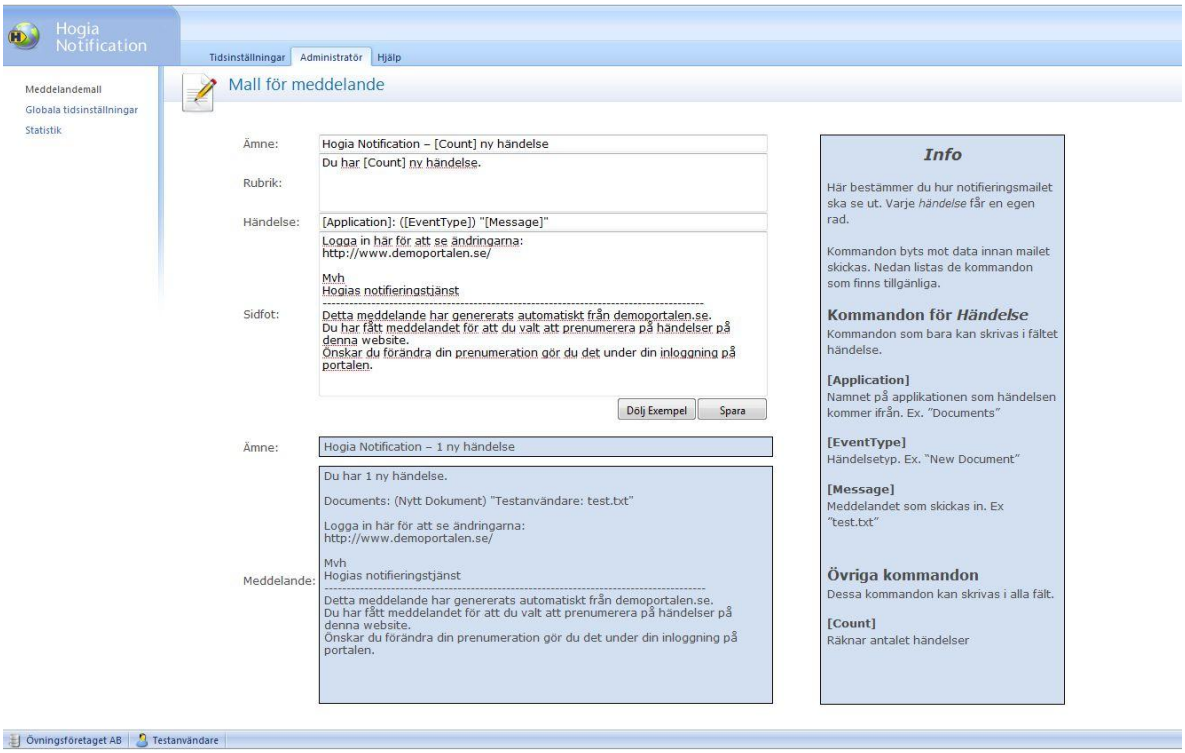

Bild B8: Fliken Meddelandemall, i notifieringsmodulens grafiska gränssnitt.

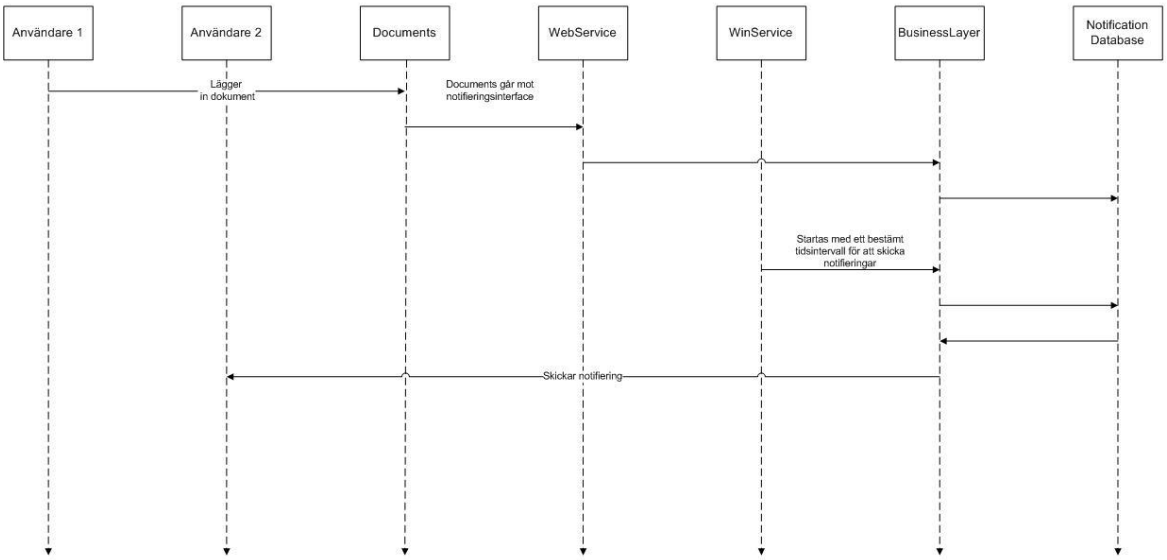

Bild B9: Sekvensdiagrammet beskriver ett flöde från att Användare 1 lägger in ett dokument i applikationen Documents, tills dess att Användare 2 får ett mail med en notifiering om detta.

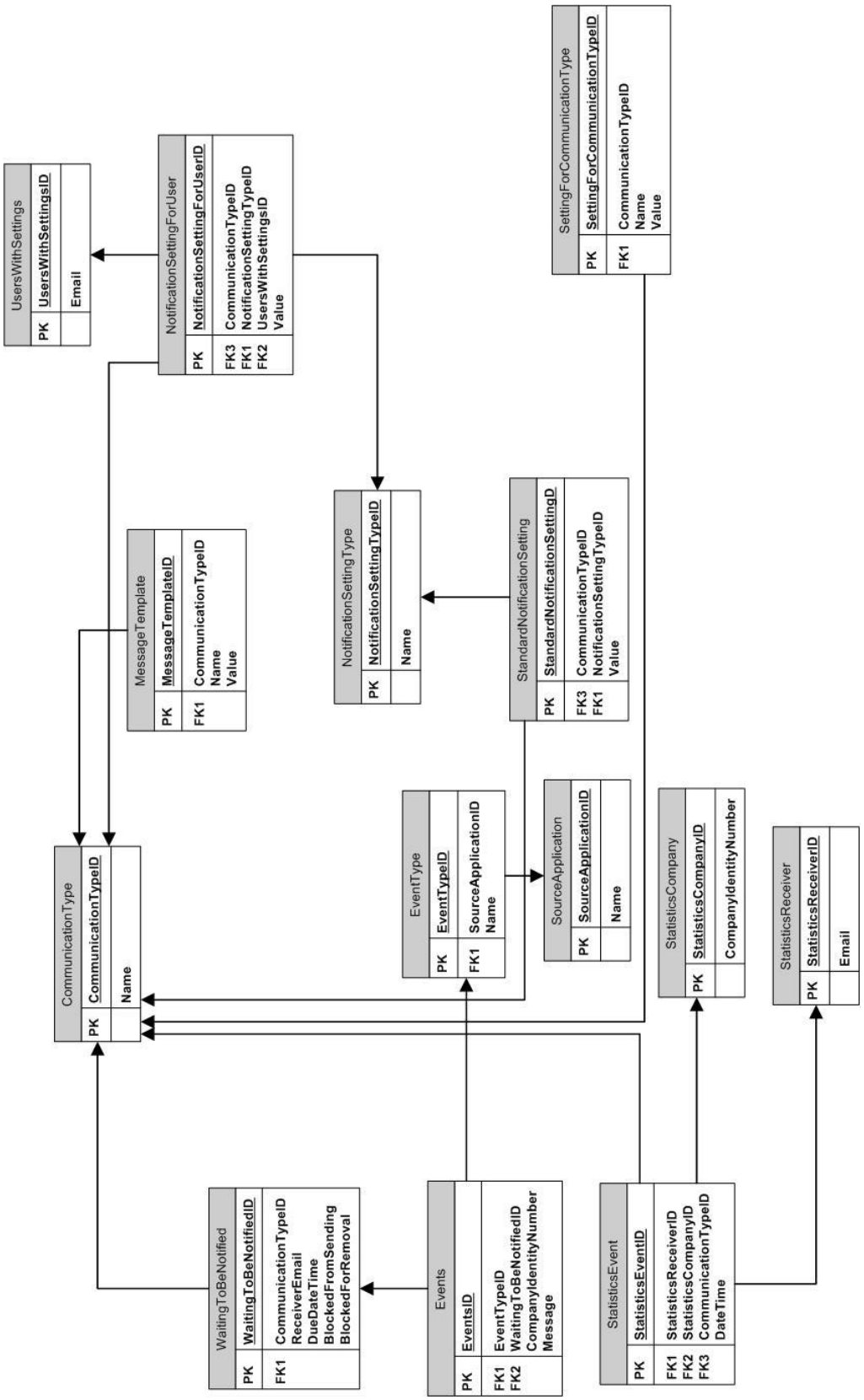

Bild B10: Modell över tabeller och relationer i databasen.

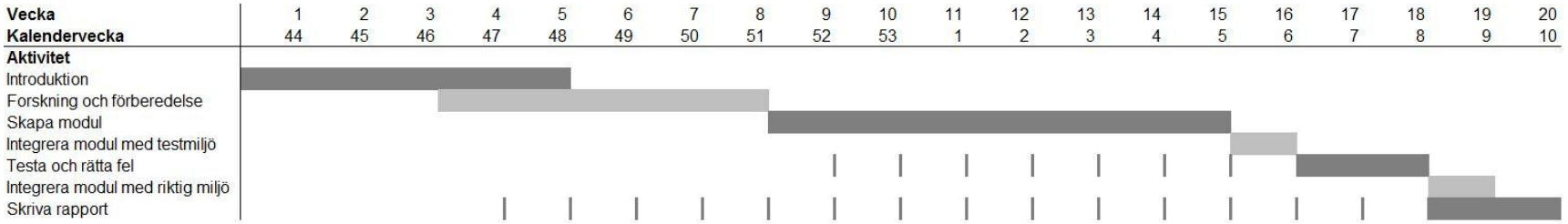

Bild B12: Ursprunglig tidplan.

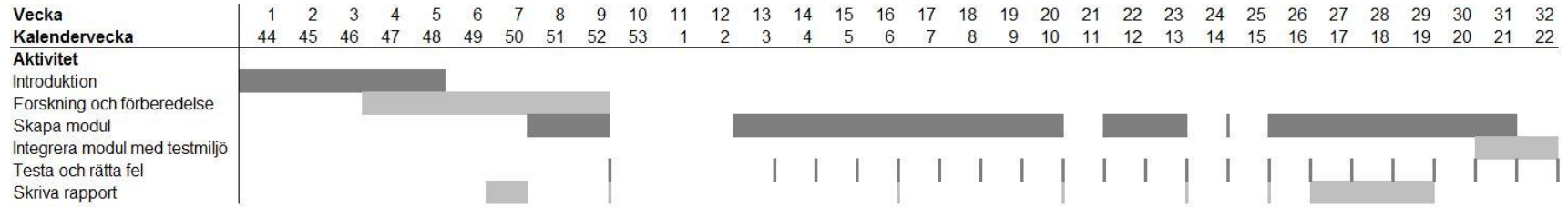

Bild B13: Slutgiltig tidplan.