Datavetenskap

Opponenter:

Sebastian Skoglund

Per-Erik Svensson

Respondenter:

Anders Brännström

Richard Nilsson

Investigating and Implementing a DNS Administration System

Oppositionsrapport, D-nivå

2007:xx

1 Sammanfattat omdöme av examensarbetet

Uppsatsen är välformulerad rent språkligt och innehåller en bra beskrivning över det arbete som utförts och resultatet av detta arbete. På vissa ställen kan den dock uppfattas som självrättfärdigande då det saknas argumentation kring varför en viss lösning valts. Å ena sida är rapporten välgjord och gedigen då den tydligt förklarar de problem som skulle lösas och hur dessa löstes. Å andra sidan saknas vetenskapligheten då det är svårt att hitta svar på varför problemen löstes som de gjorde. Ett utredande kapitel beskriver varför en egen implementation valdes men inga delar förklarar varför den egna implementationen ser ut som den gör. Tidvis gömmer sig argumenten bakom vaga hänvisningar till kravspecifikationen.

2 Synpunkter på uppsatsen knuten till examensarbetet

Uppsatsen är till sin utformning mycket teoretisk i bemärkelsen kunskapsintensiv. Uppgiften, som den beskrivs i uppsatsen är dock rent praktisk. Därav finns en viss saknad kring implementationsdetaljer. Det ska dock påpekas att rapporten klart och tydligt redogör att delar av arbetet innehållit affärshemligheter och därför inte kan avslöjas. De delar som tar upp implementation redogör med bakgrund av detta mycket väl för vilket arbete som utförts.

2.1 Titel

Titeln fångar innehållet i rapporten bra. Kanske skulle det tydliggöras att rapporten inte undersöker hur ett DNS-administrationssystem ska implementeras utan att ”investigation” syftar på en undersökning av existerande system.

2.2 Uppsatsens disposition

Mycket bra. Enkel, tydlig och lätt att följa. Delar av kapitel 2 ses dock gärna som bakgrund till implementationskapitlet eftersom det till en början är något oklart huruvida en egen implementation har genomförts eller inte. Rapporten syftar till att undersöka vilken lösning som passar Ninetech bäst och agera utifrån resultatet. Att resultatet blev en egen implementation står inte klart förrän i kapitel 3 varför delar av kapitel 2 känns felplacerade.

2.3 Begreppsapparat

Uppsatsen följer vår uppfattning om innebörden bakom datavetenskapliga termer och jargong. Författarna ger intryck av att veta vad de talar om och endast vid ett tillfälle förekommer större frågor kring använda begrepp. Det som åsyftas här är vad som avses med klient-server-modellen, vilket förklaras ytterligare i avsnitt 3.2. Mindre frågor finns kring SSL, där det är oklart om författarna är överens med det datavetenskapliga samhället kring termen. Uppsatsen påpekar bland annat att SSL inte tillåter filöverföring eller kommandopromptsanslutning vilket kan uppfattas som en missuppfattning av vad SSL är. Mer om detta i avsnitt 3.4.

2.4 Argumentering och slutsatsdragning

Kapitel 3 undersöker existerande system vilket görs på ett bra sätt med tydliga, om än diskutabla[1], kriterier. Det kan dock ifrågasättas huruvida alla möjligheter verkligen uttömts. Kapitlet sållar på ett ganska kortfattat sätt bort sex av nio alternativ utan att undersöka om delar av dessa system kan användas för att bygga ett system helt i enlighet med önskemålen. Metoden har möjligtvis varit bra men är i så fall något kortfattat beskriven. Det som kan kännas kort är hur sökningen som resulterade i de första nio system gick till. Utan en sådan beskrivning känns urvalet ganska godtyckligt och läsaren tvingas lite på att rapportförfattarna genomfört sökningen på ett bra sätt.

Rapporten ställer också upp ramarna för projektet i form av en kravspecifikation. Här redogörs flera krav som ur ett vetenskapligt perspektiv kan ifrågasättas. Exempelvis är det svårt att i rapporten hitta någon undersökning som leder fram till användandet av exempelvis C#, ASP.NET, SQL Server och Active Directory. Ett annat krav som hänvisas till är att ett existerande system inte får kosta mer än 200 dollar för att vara ett alternativ till egen utveckling. Detta krav känns orimligt och lite gripet ur luften. Vad ställs denna kostnad mot? Argumentationen bakom är inte redogjord för. Det är mycket möjligt att uppdragsgivaren satt detta som en gräns men i det läget borde författarna reagera och informera om orimligheten i kravet. Kvarstår det så måste det finnas en anledning som kan redogöras för.

Rent allmänt kan sägas att uppsatsen är argumentationsfattig. När argumentation förs så är den ofta välformulerad och precis, men i vissa lägen saknas argument och rapporten övergår mer i läroboksform. Fakta om vilka lösningar som valts redogörs för men information om varför saknas. Inga jämförelser mellan olika lösningar syns och viss varning för

[1] Läs mer under 3.3.

auktoritetstänkande utförs. Det är inte vetenskapligt att blint luta sig mot vad uppdragsgivaren vill – det är möjligt att de inte vet bäst. Vidare finns det avsnitt innehållandes påståenden som inte finner stöd i någon referens eller egen argumentation. Bland annat så nämns vid något tillfälle att en rad experter utformat standarden ITIL baserat på välkända ”best practices” utan att källa anges. Detta innebär ytterligare en auktoritetsvarning – läsaren litar inte på er för att ni använder ord som ”expert” och ”best known”.

2.5 Sammanfattningen

Denna del redogör väl för rapportens innehåll och uppgiftens omfattning. Både de problem som skulle lösas och inom vilka ramar lösningen skulle ligga redogörs för på ett kortfattat sätt. Möjligen kan meningen: ”This thesis describes the process of simplifying the DNS administration procedures of NinetechGruppen AB”, skrivas om så att fokus förskjuts från att uppsatsen beskriver en process mot att uppsatsen beskriver ett system och dess komponenter.

2.6 Språkbehandling

Överlag är det mycket god engelska. Få stavfel och ett lagom formellt språk borgar både för hög läsförståelse och för precisa beskrivningar. En allmän anmärkning är att förkortningen ”e.g.” används väl frikostigt, en annan att vissa stycken syftar tillbaka på tidigare stycken samtidigt som de är mycket korta. Dessa borde läggas samman med sina föregående stycken. Vissa begrepp hanteras mycket bra och det är intressant att det finns svenskar som använder uttrycket ”nowadays” framför ”now days” och ”alluring” framför ”attractive”. I enstaka fall blir språket dock lite missvisande – ”obligatory” används vid ett tillfälle då det kan misstänkas att det mindre moraliskt/religiöst förknippade ”mandatory” är avsett.

2.7 Referat och källförteckning

Gedigen, omfattande källförteckning med en mängd kvalitativa källor. Citationstecken används korrekt vid direkta referat.

3 Genomgång av uppsatsen kapitelvis

3.1 Kapitel 1

Bra introduktion till arbetet. Kapitlet lägger grunden för rapporten och gör att läsaren förstår vilket arbete som ligger till grund för uppsatsen. Avsnittet om ”Reading guidelines” lägger också upp dispositionen på ett bra sätt. Några mindre saker att kommentera är att förkortningen ITIL inte skrivs ut eller förklaras förrän i kapitel 2, men det används redan i detta kapitel. Det fjärde stycket är vidare lite väl kortfattat och syftar dessutom tillbaka på stycket innan varför det kan vara idé att slå ihop de båda. Det tredje sista stycket talar om lösningar och ger som exempel ett problem vilket är förvirrande.

Under ”Reading guidelines” föreslås att viss förkunskap inom datavetenskap ska finnas hos läsaren, vilket kan tyckas underförstått av att det är en D-uppsats i datavetenskap.

3.2 Kapitel 2

Kapitlet är välskrivet och informationsrikt. Det känns genomarbetat och beskriver de olika områdena på ett lättbegripligt sätt. Språkets form är klanderfritt men vissa styckefel och en överanvändning av förkortningen ”e.g.” finns.

Kapitlet är visserligen informationsrikt men i vissa avseenden för omfattande. Stora delar av kapitlet är mer terminologi än teori för en datavetare. Det som är med bör också ha relevans för resten av rapporten. Som exempel kan CMDB nämnas; begreppet förekommer senare endast på ett ställe. Vidare, C# är ett programmeringsspråk och ni har använt det. Vad dess för- och nackdelar är har bara relevans om det jämförs med andra språk eller att det finns begränsningar som syns senare i rapporten. Att det är ”managed” har låg, eller kanske ingen relevans för att förstå rapporten i övrigt. ASP.NET behöver läsaren kanske inte läsa 7 sidor om för att förstå att det är ett krav som inte uppfylls av vissa system. Det räcker med en översikt av vad ASP.NET är och vilka delar av systemet som är avgörande för ert projekt.

Kapitlet misstänks vara tänkt som teoridel men det finns då saker att tänka på. Så som vi tolkar teori:
En teori besvarar ”Varför”. Varför är ASP.NET att föredra framför andra system? Vilka begränsningar har C# och i så fall – vilka konsekvenser får det för arbetet?

- Exempel 1. Människan vet vad gravitation är, att det drar i materia. Newton, Einstein och Nordström beskriver varför – med olika teorier.

- Exempel 2. Vi vet nu vad ASP.NET är. Beskriv varför just ASP.NET. Varför inte något alternativt system t.ex. PHP eller JSP?

- Exempel 3. ”Jag har en teori om att C# är bättre lämpat än Java. Nu bygger jag upp en argumentation för at stödja min teori.”

Teori i det här sammanhanget är inte bara kunskap hämtad ur böcker. I så fall måste ”teorin” ha en stark förankring till det arbete som utförts. Det måste berätta något som är nödvändigt att kunna för att förstå arbetet, eller förklara varför ni valt en viss lösning.

Under avsnittet ”Project requirements” beskriver ni att den maximala kostnaden för en extern lösning är 200 USD (amerikanska dollar). Denna summa bör få stöd av kraftfullare argumentation. Varför just 200 dollar. 1400 (med dagens valutakurser) kronor låter som en mycket liten summa. Ställt i förhållande till dagens, förmodade, mycket höga administrationskostnad är det ingenting. Vidare, att utveckla egen mjukvara kostar många gånger mer. Bara att ha er undersöka bästa lösning kostar förmodligen mycket mer än 200 dollar. I samma stycke talar ni också om de tänkbara lösningar som finns för projektets problemställning och drar slutsatsen att antalet lösningar är tre. Här känns det dock som att ni missat möjligheten till att låta ett externt företag utveckla en helt ny produkt åt Ninetech. Detta är mycket vanligt, framförallt inom konsultbranschen.

Förväxla inte GNU GPL med öppen källkod. De är relaterade men inte samma sak. Öppen källkod innebär inte nödvändigtvis att produkten inte går att sälja vidare, och om det är er uppfattning bör tydligare argument anges.

Mycket av argumentationen i kapitlet hämtar stöd i kravspecifikationen, vilket kan vara okej. I så fall bör dock kraven ha någon typ av stöd i teori eller praktik. Varför är det exempelvis så viktigt att använda Active Directory? Varför ASP.NET? Varför SQL Server? Varför C#?

Mindre anmärkning finns i första stycket, avsnitt 2.3. En mening antyder att routrar hellre använder IP-adresser än värdnamn men det är kanske en mer korrekt beskrivning att säga att routrar inte kan routa på värdnamn och således kräver IP-adresser.

Sista stycket under 2.3.2 förslår att lövnoderna som är syskon inte får ha samma namn vilket kan tolkas som att övriga syskon (innernodssyskon) kan ha samma namn. Detta bör tydliggöras.

Andra stycket i avsnitt 2.3.5: Tydliggör att a.src.com adresserar sin lokala namnserver m.h.a. IP och inte använder värdnamnet dns.src.com, vilket mening föreslår.

Sista stycket i avsnitt 2.3.8 påpekar att det finns fler typer av DNS-records än de som redogjorts för men att dessa ligger utanför uppsatsens omfång. Detta föranleder frågan – behöver några records tas upp alls?

Avsnittet som behandlar ITIL börjar med en tydlig förklaring av varför det är berättigat att ta upp ITIL och kortfattat varför det är ett måste att systemet följer standarden. Denna information saknas i övriga avsnitt. Mycket bra att den är med här!

Första stycket i avsnitt 2.4.1 saknar källa. Ni skriver att ITIL är ett resultat av många års erfarenheter från experter och ”myndighetspersoner”. Det här måste styrkas. Vetenskapen litar inte till auktoriteter bara för att de är auktoriteter. Likaså behöver 2.4.11 en källa. Vem är det som påstår att exempelvis Microsoft använder ITIL?

Sista punkten i punktlistan i 2.5.8 implicerar att ett ”Domain user account” endast ger access till nätverksresurser och att rättigheterna för den lokal datorn därmed inte kan påverkas av ”Domain user accounts”.

Under avsnitt 2.6.1 talar ni om en ”fysisk databas”. Utveckla gärna kortfattat vad ni menar med det. I kontexten är det inte helt klart vad som menas även om en begåvad gissning leder till att svaret förmodligen är ”själva informationsinnehållet och dess struktur”. Samma avsnitt specificerar dessutom DDL och DML som två programspråk vilket är ett påstående som måste styrkas. Vi håller inte med helt och därför behövs förmodligen en källa eller egen argumentation. DDL och DML kan klassas som ”computer languages” men saknar vissa konstruktioner (exempelvis programflöde och algoritmbeskrivning) för att vara ett fullfjädrat programspråk. (Jmfr. med HTML, XHTML och TeX som visserligen är språk men som traditionellt inte benämns som programspråk.) Senare kallar ni det för ett icke-procedurielt språk vilket i sig inte är fel. Med det ordval ni använder kan man kanske till och med kalla det för ett deklarativt språk!

I avsnitt 2.6.3 ger ni exempel på en ”create”-sats i SQL som påstås skapa en tabell enligt figur. Läsaren kan ledas att tro att även tabellens innehåll skapas av uttrycket, vilket inte är fallet.

Sista stycket i avsnitt 2.7.2 ger ni en direkt motsägelse till avsnitt 2.8.3. 2.7.2 menar att .NET är plattformsberoende medan 2.8.3 tydliggör att så inte är fallet då man kan skapa program för olika plattformar, bland annat Windows XP och Pocket PC. Vidare kan det hävdas att .NET är lika plattformsoberoende som exempelvis Java. Skillnaden mellan de båda är att .NET inte finns implementerat till lika många plattformar.

Avsnitt 2.7.3 är lite förvirrande så till vida att det påstår att ASP.NET har två sätt att lösa säkerhet på – och båda sätten stöds av ASP.NET. Problemet är att det inledningsvis låter som

att det rent generellt finns två sätt varav ett inkluderar Windows. Detta får läsaren att fundera på om säkerheten i ett godtyckligt system är beroende av Windows. Svaret läsaren ger sig själv är naturligtvis nej, vilket leder till slutsatsen att ASP.NET har två lösningar. Stycket direkt efter påpekar sedan att båda dessa lösningar stöds vilket alltså blir lätt förvirrande. Försök gärna omformulera styckena en aning. Sista stycket i detta avsnitt (2.7.3) är målande för stora delar av kritiken som hittills framförts. Ni skriver: ”Accordingly, ASP.NET provides ample functionality to implement security in a relatively simple and secure way”, och använder därmed ordet relativt. Frågan är då plötsligt uppenbar - relativt vad? Har ni undersökt andra system utan att redovisa det? Varför valdes just ASP.NET?

Inledningen till avsnitt 2.8, som behandlar C#, proklamerar att programmeringsdetaljer inte ligger i rapportens omfång och därför inte kommer att tas upp. Däremot tar rapporten upp C# i ganska ingående detaljer. Här kan läsaren få känslan av att just programmeringsdetaljer hade varit intressant. Hur har ni löst implementationen, med vilka algoritmer? Detta kan kännas mer relevant än att läsa om C#s historia och vad innebörden av ”managed code” är. Vi skulle alltså vilja vända på resonemanget. Programmeringsdetaljer hade bidragit med mycket till rapporten. En läsare som är intresserad av C# väljer däremot med stor sannolikhet en annan källa. Beskrivningen är dock mycket bra och kan med fördel finnas kvar som bakgrundsinformation. Vad som eftersöks är varför valet av språk föll på C# och kanske en omformulering till varför vissa saker inte tas upp. Slutligen, i delavsnittet 2.8.3 så skriver ni att kommandopromptsprogram är bra för att skapa enklare program. Detta är sant men det finns många avancerade server-system som inte körs med grafiska gränssnitt.

Vad gäller kapitlets avslutande avsnitt, 2.9, så kan det förtydligas en aning vad som menas med klient-server-modellen. Den tolkning vi gör av uttrycket är i kontexten av ett nätverk med en eller flera klienter som vill ha vissa tjänster utförda och då frågar en server som tillhandahåller dessa tjänster. Detta känns inte fullt jämförbart med MVC. Vidare ger avsnittet uttryck för att klient-server inte är skalbart för webapplikationer. Här behövs mer argumentation än en källhänvisning för att övertyga. En webapplikation använder sig av webben, en infrastruktur som bygger på just klient-server. Om inte klient-server är tillräckligt skalbart för en webapplikation, kan det då vara skalbart för webben som helhet? Vi vet att klient-server klarar av att upprätthålla web-servrar och även en hel del underliggande infrastruktur så som DNS-system och DHCP. Borde det då inte klara av webapplikationer? I delavsnittet 2.9.3 skriver ni om ”lagerarkitektur”. Hur kan MVC kopplas till detta? Utveckla gärna.

Genomgående för kapitlet är att introduktionerna till avsnitten tycks syfta till ett specifikt system. Efter att ha läst hela rapporten står det klart att det system ni hänvisar till är det ni själva implementerat. Detta framgår dock inte av kapitel 2 eller något av dess avsnitt vilket gör att läsaren kan känna sig förvirrad. Har ni redan bestämt er för vad resultatet ska bli – en egenutvecklad produkt? I kapitel 1, avsnitt 2 så säger ni att rapporten ska läsas ”uppifrån och ner” vilket ytterligare förvirrar då ni inte förklarar förrän senare vad som avses med ”DNS-administrationssystemet”.

3.3 Kapitel 3

Kapitlets innehåll är en viktig del i en sådan här uppsats då det beskriver de befintliga system på marknaden. En sådan undersökning förhindrar att man "uppfinner hjulet på nytt".

Generellt sätt kan man säga att utredningen av de olika systemen har skett på ett bra sätt; En kortare beskrivning av deras olika egenskaper och krav har undersökts och sedan jämförts med kravspecifikationen.

I avsnitt 3.2.3 så skriver ni att skaparna av mjukvaran Men & Mice Suite anser att de marknadsledande. Vi tycker att sådan, extremt subjektiv, information inte hör hemma i en D-uppsats, såvida inte en pålitlig tredje parts-källa kan styrka påståendet. Alla företag, mer eller mindre, skryter om att de är marknadsledande.

Avsnitt 3.4 behandlar de olika kriterier som ni har valt att evaluera på de tre olika system som till slut valdes att undersöka vidare. Som läsare är man intresserad av att veta varför ni har valt att evaluera med utgångs punkt från dessa kriterier. Motivera varför dessa är viktiga för just erat projekt.

Kostnaden för ett eventuellt system får inte överstiga 200 USD, konstateras i avsnitt 3.4.1. Vi anser att det från början låter som ett orimligt krav då det ska sättas i relation med t.ex. nuvarande kostnad att ha en anställd som sköter arbetet manuellt, kostnader för utveckling av en egen produkt i form av löner, arbetsstationer, lokaler m.m. Men det intressanta med ett sådant krav är att Ninetech implicit anser att eran totala arbetsinsats inte är värd mer än 200 USD.

I avsnitt 3.6 diskuteras slutsatsen för kapitlet och en rad olika argument presenteras för och emot de olika systemen som har evaluerats. Detta genomförs på ett tillfredställande sätt men vi vill betona att vissa argument som förkastar många av de existerande lösningarna är irrelevanta. Tabell 3-1 illustrerar på ett överskådligt sätt de tekniska krav som anses viktiga. Men av vilka anledningar måste t.ex. ett existerande system vara utvecklat i .NET, använda

SQL Server osv.? Om det finns starka argument för att använda just .NET när ni utvärderar andras system så är det viktigt att presentera dem. I en studie som utvärderar existerande lösningar så är det i synnerhet de funktionella kraven som bör stå i fokus, t.ex. användaren ska kunna spara dokumentet till hårddisken. Det är däremot inte intressant med de tekniska kraven som t.ex. beskriver att applikationen ska utvecklas i Lisp, såvida det inte finns unik funktionalitet i Lisp som gör att det är absolut nödvändigt, men då bör den funktionalitet motiveras tydligt.

3.4 Kapitel 4

I första meningen i avsnitt 4.2 påstår att ni att tidigare kapitel har beskrivit ett "dns-administrationssystem" och att följande kapitel beskriver implementationen av ett sådant. De föregående kapitlen har inte beskrivit något "dns-administrationssystem" utan beskrivit fakta kring de tekniker som kravspecifikationen påbjuder (t.ex. ASP.NET, C#, SQL SERVER, Active Directory) samt hur existerande system fungerar.

Säkerhet beskrivs som en (mycket) viktig del i applikationen men samtidigt så poängteras att systemet endast ska vara tillgängligt inom det lokala nätet på företaget. Är det verkligen så viktigt med hög säkerhet på trafiken inom intranätet? Om ja, beror det i så fall på att det är dålig säkerhet på det interna nätet (dvs. kan man komma in på intranätet utifrån enkelt) eller litar man helt enkelt inte på sina anställda? Kanske finns det andra orsaker som gör att det är viktigt med säkerheten i just det här systemet, men ange i så fall dessa.

SSH beskrivs som "den enda säkra teknologin" för att exekvera shell-kommandon remote. Finns det inte andra alternativ, t.ex. att använda Telnet via en VPN-tunnel eller över SSL? Senare i samma avsnitt så konstateras det inte vara möjligt att överföra filer eller att utföra shell-access remote m.h.a. SSL. Författarna måste ha missförstått hur SSL fungerar om man antyder sådana felaktigheter, för SSL kan användas t.ex. med FTP för säker filöverföring och tillsammans med Telnet för säker remote shell-access.

Under avsnittet om säkerhet, 4.5.4, så beskrivs rättigheterna för en s.k. "Change Manager". En "Change Manager" är den enda som kan godkänna eller förkasta förslag på ändringar. Kan en "Change Manager" föreslå en ändring och själv godkänna den samma eller måste en annan "Change Manager" godkänna den ändringen? Finns det faror med det?

I avsnitt 4.5.5 beskrivs bl.a. hur en godkänd ändring genomförs genom att man startar om namnservern. Finns det inget annat sätt att ändra inställningar än att starta om servern? Ser ni några potentiella risker eller faror med att starta om servern?

Sista meningen i avsnitt 4.6 (Summary) förklarar att mer detaljer finns tidigare i kapitlet. Är inte det definitionen av en sammanfattning?

3.5 Övriga kommentarer

På de två sista kapitlen, kapitel 5 (resultat och utvärdering) och kapitel 6 (slutsats), har vi inga specifika anmärkningar utan de följer samma språkliga standard som övriga kapitel och beskriver resultaten på tillfredställande sätt. Vi anser dock att man bör överväga att flytta avsnitt 6.1, som behandlar problem, till kapitel 5. Olika problem och hur de löstes passar bättre i kontexten ”utvärdering”, vilket kapitel 5 behandlar.