

TvärGranska
**– Interaktiv webbmiljö för språkgranskning med
inriktning mot andraspråksinlärare**

CrossCheck
**– An Interactive Web Based Interface for Grammar
Checking with a focus on Second Language Learners**

Examensarbete i Publiceringsteknik 20 poäng

Av: Ylva Stenervall, PUB, T, KTH
ylvas@kth.se
Handledare: Martin Hassel, IPLab, Nada, KTH
& Ola Knutsson, IPLab, Nada, KTH
Examinator: Nils Enlund, GT, Nada, KTH
Uppdragsgivare: IPLab, Nada, KTH
2003-10-22

Sammanfattning

Ett interaktivt gränssnitt, TvärGranska, har utvecklats för att hjälpa till i fortsatta undersökningar i hur man ska utveckla Granska för att passa andraspråksinlärare bättre. Granska är en svensk språkgranskningsmotor som utvecklats på Nada, KTH. Syftet med det nya gränssnittet är att språkgranskning med Granska ska bli mer interaktivt och att Granska ska vara lättillgängligt för användaren genom att vara oberoende av operativsystem och webbläsare.

I början av utvecklingen delades en enkät ut, enkäten syftade till att ta reda på mer om de tilltänkta användarnas datorvana, hur de arbetar med ordbehandlare och språkgranskning i ordbehandlare samt deras inställning till en språkgranskare på webben. Applikationen är skriven i det serverexekverade språket JSP med användandet av JavaScript för att åstadkomma mer interaktion. Gränssnittet har utvärderats i en användarstudie med studenter/ andraspråksinlärare från Folkuniversitetet i Stockholm. Målet med användarstudien var att undersöka hur användarna interagerar med gränssnittet eftersom det ger viktig information till utvecklingen av gränssnittsdesignen.

Att utforma ett gränssnitt så att det passar alla användare i målgruppen är svårt, användarna kan ha olika mycket erfarenhet av datorgränssnitt, ha olika smak och olika sätt att arbeta på. Användarna i denna användarstudie speglade också detta, men alla i studien upplevde också att TvärGranska var bra. Det de framför allt uppskattade var att det var gratis och alltid tillgängligt för dem via webben.

CrossCheck – An Interactive Web Based Interface for Grammar Checking with a focus on Second Language Learners

Abstract

CrossCheck (TvärGranska) is an interactive web based interface for grammar checking that has been constructed in order to help with further studies on how to develop Granska to be more suited for second language learners. Granska is a Swedish grammar checking program that is being developed at NADA, KTH. The purpose of the new interface is to make working with Granska more interactive and Granska more accessible by being independent of operating system and web browser.

Before the development of the interface started a questionnaire was handed out in order to get to know something about the habits of the future users regarding computer use and how they work with word processors and grammar checking, and also their view on a grammar checker on the Internet. The application was then written in the server executed language JSP with the use of JavaScript to get more interaction. The interface was evaluated in a user study with students/second language learners from Folkuniversitetet in Stockholm. The goal with the user study was to observe how the users interacted with the interface as that gives important information on how to design the interface.

Designing an interface so that it suits all users in the focusgroup is hard, as users have different experience of computers. They also have different taste as well as different ways of working with a computer. The users in this user study were a reflection of this but all of them still found CrossCheck good. The thing they appreciated the most was that it was free to use and that it was always accessible to them by the Internet.

Förord

Detta examensarbete är utfört vid Institutionen för numerisk analys och datalogi (Nada) på Kungliga Tekniska Högskolan i Stockholm, i ämnet publiceringsteknik. Examensarbetet har gjorts på avdelningen Interaktions- och presentationslaboratoriet (IPLab) inom projektet CrossCheck – svensk grammatikkontroll för andraspråksskribenter¹ och, det tangerande projektet, Språkliga datorstöd och andraspråksinlärning². Forskningsprojektet CrossCheck syftar till att ta fram ett grammatikgranskningsprogram speciellt avpassat för användare med svenska som andraspråk, medan Språkliga datorstöd och andraspråksinlärning fokuserar på människa-datorinteraktion och pedagogik.

Tack till mina handledare på IPLab, Martin Hassel och Ola Knutsson, för deras synpunkter och råd.

Jag vill även rikta ett tack till Tessy Cerratto Pargman för hennes visade intresse för projektet och för chansen att visa upp TvärGranska på studiecirkeln som hon ledde på IPLab. Jag vill även tacka Tessy Ceratto Pargman för hjälpen med att hitta användare till användarstudien genom hennes kontakter med Folkuniversitet. Tack till Karl Lindemalm på Folkuniversitet för att han ställde upp och lånade ut sina studerande till användarstudien, Jenny Rahbek för samarbetet under användarstudien och självklart även ett stort tack till användarna som ställde upp. Sist men inte minst vill jag tacka min sambo Johan Driessen för hans hjälp och stöd.

¹ <http://www.nada.kth.se/theory/projects/xcheck/>

² <http://www.nada.kth.se/~knutsson/call.html>

Innehållsförteckning

1 Inledning	1
1.1 Problemformulering	1
1.2 Syfte och mål	2
2 Bakgrund	3
2.1 Språkgranskningsprogram på marknaden	3
2.2 Granska	3
2.3 Datorstödd andraspråksinläring	3
3 Tillvägagångssätt	5
4 Analys	6
4.1 Val av tekniker för utveckling av webbgränssnittet	6
4.1.1 Programmeringsspråk	6
4.1.2 Serverexekverade språk	6
4.1.3 Klientsideskriptspråk	7
4.1.4 Appletprogram	7
4.1.5 Rich Text Format	7
4.2 Integrering av TvärGranska i skrivprocessen	7
4.3 Grad av interaktion med TvärGranska	8
4.4 Presentation av information	9
4.5 Kommunikation på dataformatet XML	9
4.6 Återkoppling med Granska	10
5 Utvecklingen av TvärGranska fram till användarstudien	11
5.1 Integrering av TvärGranska i skrivprocessen	11
5.2 Grad av interaktion med TvärGranska	12
5.3 Presentation av information	14
5.4 Kommunikation på dataformatet XML	16
5.5 Återkoppling med Granska	17
6 Resultat	18
6.1 Enkät	18
6.1.1 Utförande	18
6.1.2 Diskussion	18
6.2 Användarstudie	19
6.2.1 Utförande	20
6.2.2 Diskussion	24
6.3 Ändringar efter användarstudien	26
7 Diskussion och slutsats	27
8 Framtida arbete	29
Referenser	30
Elektroniska referenser	30
Appendix	33
A Förkortningsordlista	33
B Enkät	35
B1 Enkät om språkgranskning på webben	35
B2 Svaren på enkäten om språkgranskning på webben	39
C XML	45
C1 XML-exempel	45
C2 XML Schema	47
D Användarstudie	50
D1 Medgivandeformulär	50
D2 Brev till användaren/deltagaren	51
D3 Instruktioner till användaren/deltagaren	52
D4 Försöksledarprotokoll	53
D5 Intervjufrågor	54
D6 Bakgrundsdataenkät	56
E Användarhandledning	57
F Teknisk dokumentation	59

1 Inledning

På Nada, KTH pågår projektet CrossCheck – svensk grammatikkontroll för andraspråks-skribenter och på Interaktions- och presentationslaboratoriet (IPLab), Nada, KTH projektet Språkliga datorstöd och andraspråksinläring. De har båda inriktning mot andraspråksinläring och det fanns ett behov av ett nytt gränssnitt till den på Nada utvecklade svenska språkgranskningsmotorn Granska. Det nya gränssnittet, TvärGranska, ska hjälpa till i fortsatta undersökningar i hur man ska utveckla Granska för att passa andraspråksinlärare bättre.

Man önskade sig ett mer interaktivt webbgränssnitt eftersom de gränssnitt som man hade på webben inte gjorde det möjligt för användaren att ändra i texten i webbläsaren. Föregångaren till TvärGranska som heter WebbGranska fungerar så att man får mata in en kortare text och sedan presenteras den granskade texten med den felmarkerade meningarna och ersättningsförslag osv. löpande. Användarna matas med all information på en gång och det finns inte någon möjlighet till interaktion såsom att klicka på ord eller liknade. WebbGranska är skriven i PHP och finns på <http://skrutten.nada.kth.se/scrut/svesve/>.

TvärGranska, det gränssnitt som utvecklats i detta examensarbete, finns på webbadressen <http://skrutten.nada.kth.se:8080/granska/> för den som vill testa.

I rapporten används de datatermer som rekommenderas av Svenska datatermgruppen (<http://www.nada.kth.se/dataterm>). Förklaringar till använda förkortningar finns längst bak i förkortningsordlistan, appendix A.

1.1 Problemformulering

Nadas språkteknikgrupp har utvecklat ett program, Granska, för automatisk språkgranskning för svenska. Detta examensarbete syftar till att inom ramarna för projekten CrossCheck och Språkliga datorstöd och andraspråksinläring ta fram en interaktiv granskningsmiljö för svenska med Granska i botten. Miljön är tänkt att vara en klient-serverbaserad miljö med flera tänkbara klienter och en serverversion av Granska. Huvuduppgiften i examensarbetet blir att utveckla en interaktiv webbmiljö, TvärGranska, som arbetar mot Granskaservern samt att utföra användarstudier med denna. Användarstudierna är till för att utvärdera miljön och gränssnittet som ett led i utvecklingsprocessen för att förbättra gränssnittet.

De tre viktigaste identifierade frågorna är:

- Hur man på bästa sätt integrerar TvärGranska i skrivprocessen, där den främsta frågan i det hänseendet är hur användaren infogar sin text i TvärGranska och, efter redigering, extraherar den på ett smidigt sätt.
- Hur man får TvärGranska så interaktivt som möjligt, dvs. vilka webbtekniker man ska använda för att åstadkomma ändamålsenlig interaktion.
- Hur presentationen av informationen (feltyper, ersättningsförslag, diagnoser, granskningsregler, hjälp) ska se ut för att det ska vara så enkelt som möjligt för användaren att förstå och använda webbgränssnittet.

Vidare frågor att hantera är att kommunicera med Granska på dataformatet XML, eXtensible Markup Language (<http://www.w3.org/TR/REC-xml>) och att om användaren ändrar ett stav- eller grammatikfel med följd att det uppstår ett nytt fel kommer detta inte att upptäckas om användaren inte kör texten genom Granska ytterligare en gång. Man bör alltså antingen upplysa användaren om detta eller automatiskt granska om texten.

TvärGranskas kompatibilitet med Explorer, Netscape och Mozilla på Windows, Mac, Unix och Linux kommer att tas hänsyn till.

Den avgränsning som gjorts är att det som skulle undersökas, utvecklas och göras användarstudier på är klienten TvärGranska och inte Granskaservern. Endast en användarstudie har gjorts.

1.2 Syfte och mål

Syftet med exjobbet är att språkgranskning med Granska ska bli mer interaktivt och att Granska ska vara lättillgängligt för användaren genom att vara oberoende av operativsystem och webbläsare.

Målet är att utveckla en interaktiv webbmiljö som kommunicerar med Granskaservern så att framför allt personer som har svenska som andraspråk kan granska och korrigera sin text med hjälp av Granska.

TvärGranska ska presentera informationen på ett för målgruppen ändamålsenligt sätt. Användaren ska inte bli överlastad med information utan bara visas det som är relevant och kunna få fram mer information om så önskas.

2 Bakgrund

Granska är en av få publika icke kommersiella språkgranskare som finns i världen idag. Även om det finns andra språkgranskare som inte är kommersiella för en del olika språk så är det en fråga om forskningsprototyper. Vad gäller kommersiella språkgranskare så står Microsoft med underleverantörer för den största utvecklingen inom språkgranskning idag. I detta kapitel ska vi titta lite på språkgranskningsprogram, Granska och datorstödd andraspråksinlärning.

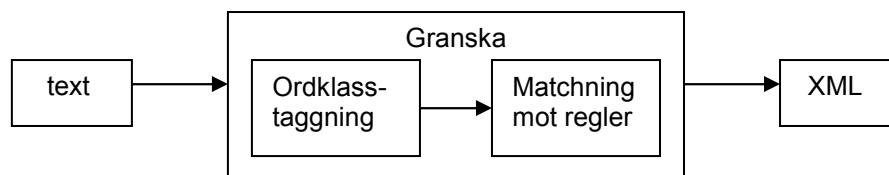
2.1 Språkgranskningsprogram på marknaden

Med Microsoft Office 2000 kom den första svenska grammatikgranskaren till en kommersiell ordbehandlare, innan den kom hade bara stavfel kunnat detekteras. Grammatikgranskaren hette Grammatifix och den har utvecklats av finska Lingsoft. Grammatifix kan bland annat hitta vanliga kongruensfel och enklare fel i verbkedjor (<http://www.lingsoft.fi/grammatifix/>).

2.2 Granska

Sedan början av 1990-talet har man på IPLab arbetat med olika projekt för utveckling av verktyg för språkgranskning, 1995 inleddes ett nytt sådant projekt nämligen Granska (Larsson, 1998). Granska kunde under denna tid utföra viss ord- och frasanalys vilket möjliggjorde att enklare grammatiska problem kunde upptäckas (Domeij et al, 1998). Under samma tid utvecklades på en annan avdelning under Nada, TCS (Teoretisk datalogi), ett program för svensk stavningskontroll, Stava. 1998 slog sig utvecklingarna av Granska och Stava ihop och det nya program som man utvecklade fortsatte att kallas Granska. Idag har Granska ett regelspråk som möjliggör konstruktion av både allmänna språkliga analysregler och specifika regler som detekterar, diagnostiserar och korrigerar grammatiska fel (Knutsson, 2001).

När Granska ska språkgranska en text börjar programmet med att ordklasstagga alla ord, t.ex. böcker är ett substantiv i obestämd form och i plural. När Granska har ordklasstaggat alla ord matchas de mot de olika granskningsregler som finns i Granska. Det finns t.ex. regler för att hitta felaktiga sårskrivningar och kongruensfel. När Granska kontrollerat texten mot reglerna för att hitta fel skapas det en XML-fil med texten och alla misstänkta fel. Se figur 2.1 för en förenklad bild över förloppet.



Figur 2.1 Förenklad figur över hur Granska arbetar.

Mer om Granska kan läsas i Granskarapporterna, <http://www.nada.kth.se/theory/projects/xcheck/> och <http://www.nada.kth.se/theory/projects/granska/> och om ordklasstagging Jurafsky & Martin, (2000).

2.3 Datorstödd andraspråksinlärning

Datorstödd andraspråksinlärning har de senaste åren blivit ett stort forskningsområde. I Chapelle (2001) kommer författaren efter omfattande tester av språkinlärningsprogram för andraspråksinlärare fram till att användning av ett språk- och grammatikgranskningsprogram

är det bästa lärosättet. Hon säger också att att bara markera ett ord som felaktigt inte är bra för inlärandet, men att markera ett ord som användaren sedan kan få mer information om är mycket bra. Det ger nämligen användaren en chans att reflektera över det markerade felet innan han/hon läser förklaringen och ett eventuellt rättningsförslag.

McGee & Ericsson (2002) utreder i sin artikel hur språkgranskningen i Microsoft Word påverkar oss genom att vara ”osynlig”, det vill säga ligga i bakgrunden och markera ord eller satser allteftersom vi skriver. De kommer fram till att det kan vara hämmande i skrivprocessen att grammatikgranskningen pågår under tiden som användaren skriver, t.ex. kan användarna distraheras av att ord markeras och därigenom glömma bort hur de hade tänkt att formulera en mening.

3 Tillvägagångssätt

Tillvägagångssättet har varit:

- identifikation av problemen när man ska utveckla ett interaktivt webbgränssnitt för språkgranskning
- inläsning på de identifierade problemen (se problemformuleringen) i synnerhet samt Granska och datorstödd andraspråksinlärning i allmänhet
- analys av tänkbara lösningar för de identifierade problemen
- utveckling av webbgränssnittet
- demonstration av en prototyp av gränssnittet på studiecirkel på IPLab inför personer med intresse för datorstödd andraspråksinlärning
- justering av gränssnittet efter synpunkterna som kom fram vid demonstrationen
- användarstudie för utvärdering av gränssnittet med personer som har svenska som andraspråk
- anpassning av gränssnittet efter vad som kom fram i användarstudien

4 Analys

Innan analysen av de olika identifierade problemen är det lämpligt att titta på vilka tekniker som är användbara i utvecklandet av interaktiva webbgränssnitt.

4.1 Val av tekniker för utveckling av webbgränssnittet

Man kan göra en del olika val när man ska välja programmeringsspråk och webbt tekniker, här är en kort överblick över tekniker som var aktuella att använda i det här fallet.

4.1.1 Programmeringsspråk

När man ska välja programmeringsspråk kommer gärna frågan om vilket språk som är snabbast upp. Det jag kommit fram till i denna fråga är att för det första är det svårt att jämföra olika programmeringsspråks snabbhet, det beror på många olika saker; hur snabbt ett program är vid testtillfället, vilken dator man kör det på, vilket operativsystem man kör på, hur optimalt man skrivit koden osv. För att få mätbara skillnader mellan olika språk kan man behöva köra programmet i en loop på 100 gånger dvs. det skiljer bara mikrosekunder i exekveringstid (D. Bagley). Olika språk kan också vara olika snabba på olika uppgifter (läsa från fil och dylikt), vilket gör det svårt att bestämma ett språk som är snabbare än de andra, det beror ju uppgiften. Språken uppdateras också regelbundet vilket medför att man måste testa de senaste versionerna av respektive språk för att en jämförelse ska kunna bli någorlunda rättvis.

D. Bagley har en omfattande sida där han försöker jämföra olika språks snabbhet (<http://www.bagley.org/~doug/shootout/>), men vill ändå poängtera att det finns viktigare aspekter i val av programmeringsspråk. Han menar att om man bara var ute efter snabbhet i ett programmeringsspråk skulle alla programmera i assembler. Det vill säga det är effektivare att välja ett språk som man känner sig bekväm med och bemästrar än att lära sig ett nytt för några millisekunder snabbare exekvering. Det är inte effektiv tid.

Det kan också nämnas att det är svårt att finna opartiska källor på detta område, den som skriver att det ena programmeringsspråket är snabbare än det andra gör det ofta i eget intresse.

För att summera så ska man först och främst välja ett programmeringsspråk som kan göra det man vill göra och som man känner sig bekväm att utveckla i. Det kan också vara bra att inte välja ett allt för obskyrt språk, om man utvecklar i ett av de större språken finns det alltid hjälp att få och de utvecklas också i större takt. (OBS! Att ett språk utvecklas i snabb takt är inte alltid en fördel.)

4.1.2 Serverexekverade språk

Serverexekverade språk körs på en server. För att kunna skriva serverexekverade språk behöver man en kommandotolk (server) som kan tolka språket. Inom speciella taggar i HTML-koden skriver man det serverexekverade språket, det tolkas av servern och resultatet visas i webbläsaren inbäddat med HTML-koden. Serverexekverade språk använder man bland annat när man har en databas innehållande information som man på olika sätt vill visa på webbsidan. De mest populära serverexekverade språken idag är JSP (Java Server Pages), ASP (Active Server Pages), ASP.NET och PHP (PHP: Hypertext Preprocessor) (<http://www.developer.com>). ASP och ASP.NET är utvecklade av Microsoft, JSP av Sun och PHP av The PHP Group. (Att kalla ovanstående tekniker för serverexekverade språk är egentligen lite slarvigt, de är inga renodlade programmeringsspråk utan en lös samling objekt som kan kommunicera med servern och klientens webbläsare.)

4.1.3 Klientsideskriptspråk

Klientsideskriptspråken kom till när man upptäckte att man vill göra mer med sina webbsidor än att bara presentera information, man ville interagera med användarna. Med HTML presenterar man information medan man med klientsideskript utför händelser. Man kan med dessa skript t.ex. sätta och spara variabler och kommunicera med användaren. Det är webb-läsaren som tolkar både HTML:en och skripten. De klientsideskriptspråk som är aktuella idag är JavaScript (<http://devedge.netscape.com/library/manuals/2000/javascript/1.5/reference/>), JScript och VBScript (<http://msdn.microsoft.com/scripting/>), där JavaScript från början är utvecklat av Netscape och JScript och VBScript är utvecklade av Microsoft.

4.1.4 Appletprogram

De flesta textredigeringsprogram som finns på webben idag är appletprogram (<http://java.sun.com/applets/>). Ett appletprogram är ett Java-program som körs i en webb-läsare med JVM (Java Virtual Machine) installerat, programmet körs på klientdatorn, så att ingen kraft behöver tas från servern. Som standard har dock appletprogrammet inte rättigheter att skriva på klientdatorn eller att anropa någon annan dator än den den kom från. Detta kan man lösa genom att signera appletprogrammet, vilket innebär att det ber om tillåtelse att få komma åt klientdatorn och om det är förenligt med klientdatorns säkerhetsföreskrifter så får den det. Som standard är åtkomst inte tillåten, utan man får fråga användaren när han/hon försöker starta appletprogrammet. En fördel med appletprogram är att eftersom de inte körs på en server behöver man inte vara uppkopplad under tiden man använder dem utan bara när man laddar hem dem (om det inte igår i dess funktionalitet att anropa externa servrar). Webb-adresser till exempel på denna typ av textredigeringsprogram finns under referenserna.

4.1.5 Rich Text Format

Rich Text Format, RTF (http://www.biblioscape.com/rtf15_spec.htm) är intressant att titta på med avseende på hur TvärGranska ska integreras i användarens skrivprocess. RTF möjliggör nämligen en bibehållning av formateringen (stycken, typsnitt osv.) i en kommunikation med Microsoft Word. Det är ett beskrivningsspråk för textdokument. Det är en standard framtagen av Microsoft. En RTF-fil är egentligen en ASCII-fil (American Standard Code for Information Interchange) men med beskrivande attribut som typsnitt och marginal. En ASCII-fil är en fil där varje byte motsvarar ett ASCII-tecken, dvs. varje bokstav är representerad med ett tal, t.ex. ASCII-koden för M är 77.

4.2 Integrering av TvärGranska i skrivprocessen

För eventuell lösning på integreringen av TvärGranska i skrivprocessen tittade jag på textredigeringsprogram på webben för att se om den lösningen skulle vara önskvärd för TvärGranska. En annan möjlighet som undersöktes var att utveckla en funktion i Microsoft Word som tar användaren och texten till TvärGranska och när man granskat klart, användaren och texten tillbaka till Microsoft Word. För att texten ska behålla stilmallen, med rubriker och stycken behövs det en beskrivning av hur dokumentet är uppbyggt, textformatet RTF (Rich Text Format) kan användas till detta. Jag undersökte därför textformatet RTF för att se vad man har för möjligheter med det.

Integreringen av TvärGranska i skrivprocessen kan ses som kritiskt i avseende på hur pass mycket användarna kommer att använda sig av TvärGranska. Tycker användaren att det är värt att lämna sin skrivmiljö för att språkgranska sin text i TvärGranska? Hur svårt/besvärligt kan det vara innan användaren inte tycker att det är värt det? Det var bland annat dessa frågor som bidrog till att en enkät konstruerades (se appendix B1). Enkäten syftade till att ta reda på mer om de tilltänkta användarnas datorvana, hur de arbetar med ordbehandlare och språk-

granskning i ordbehandlare och deras inställning till en språkgranskare på webben. Frågorna utformades med hänsyn till att de som skulle besvara enkäten hade svenska som andraspråk. Det var framför allt ordval som beaktades för att göra enkäten både tydlig och lättläst. Bland de viktigare frågorna på enkäten var:

- Kan du tänka dig att göra stavnings- och grammatikkontrollen i ett annat program än det du skrivit din text i?
- Känner du till hur man kopierar, klipper och klistrar på en dator?
- Skulle du kunna tänka dig att klistra in och klippa ut din text i/ur programmet som kontrollerar din text för stavfel och grammatikfel?
- Kan du tänka dig att sitta och skriva på en dator uppkopplad till Internet för att kunna köra programmet som kontrollerar stavfel och grammatikfel?

Enkäten delades ut till en klass som läser Svenska som främmande språk, nivå 3 på Stockholms universitet. Det är den sista kursen man läser innan man blir behörig för andra högskolestudier på svenska.

4.3 Grad av interaktion med TvärGranska

När det gäller grad av interaktion med TvärGranska verkade ett appletprogram vara den rimligaste lösningen och detta undersöktes vidare bland annat genom att titta på textredigeringsprogram på webben. En annan tänkbar lösning var att presentera texten som HTML (Hyper Text Markup Language) och använda JavaScript för att skapa den interaktion som man vill ha, t.ex. att kunna klicka på felmarkerade ord och göra ändringar i texten.

Fördelen med ett appletprogram är att det nästan inte finns några begränsningar vad man kan göra interaktivt sett. Nackdelarna är att användaren måste ha en JVM (körmiljö) i sin webb-läsare och att appletprogrammet behöver ligga på samma dator som Granska för att kunna anropa Granska eftersom appletprogrammet bara har rättigheter att anropa den dator från vilken den kom. Detta kan dock lösas genom att man på datorn som appletprogrammet ligger på gör en servertjänst (t.ex. en servlet) som kan anropa en annan dator. Eftersom det blir ytterligare ett steg med denna lösning får man räkna med att det kanske tar längre tid för servern att svara då. Ett annat problem är att användarna behöver ha rätt version av JVM:en för att programmet ska fungera. Appletprogrammet har heller inte några rättigheter på klientdatorn, dvs. den kan t.ex. inte använda systemvariablerna eller spara en fil med den granskade texten på klientdatorn utan att vara signerad och fått lov från användaren. Detta kanske användaren inte vill ge om han/hon inte litar helt på den som tillhandahåller applikationen med tanke på virus och andra luckor i säkerheten.

Det andra förslaget som går ut på att presentera texten som HTML och använda JavaScript är en enklare lösning men som har fördelen att den är enkel för användaren. Om man t.ex. tittar på textredigeringsprogram på webben så har de menyer och en mängd knappar medan den här lösningen blir renare i utformningen och jag tror därför att den blir mer intuitiv för användaren. Textredigeringsprogram på webben är så gott som alltid utformade som appletprogram och därigenom finns det en risk med att göra ett appletprogram i det här fallet eftersom användaren kan tycka den ska kunna fungera som layoutprogram och det är inte ändamålet med TvärGranska. Se även nedan (kap. 4.6) att problemet med återkoppling med Granska får en enklare lösning med användandet av HTML och JavaScript än med ett appletprogram. Med ett appletprogram kan man åstadkomma stor interaktion, men i det här fallet räcker JavaScript för att åstadkomma interaktionen. Den interaktion som är viktig är möjligheten att

klicka på de ord som felmarkerats och att kunna ändra i texten utifrån de ersättningsförslag och den information man får om felet.

Jag kom fram till att jag skulle utveckla det andra förslaget eftersom det är enklare och kan göras tillräckligt interaktivt. Det är också det säkraste kortet i strävan att TvärGranska ska vara oberoende av operativsystem och webbläsare och därigenom lätt tillgängligt för användarna.

4.4 Presentation av information

Presentationen av informationen löstes genom att utveckla en prototyp av TvärGranska och göra användarstudier på den i olika stadier av utvecklingen. Hur användarstudierna bäst genomförs för att utvärdera webbgränssnittet undersöktes.

Det jag ville undersöka var om användaren på egen hand kunde tillgodogöra sig hur gränssnittet fungerade och därför lades studien upp så att en användare studerades åt gången. På så sätt kan man observera och protokollföra hur användaren interagerar med och reagerar på olika delar i gränssnitt. De som studerades tillhörde också den tilltänkta målgruppen, andraspråksinlärare, och de använde sig av en egenproducerad text. Detta för att deras intresse för att utforska gränssnittet skulle vara större.

Till skillnad från t.ex. Microsoft Word kan Granska visa hur den har ordklassat ett ord t.ex. *nn.neu.sin.ind.nom* (nomen (substantiv), *neutrum*, *singular*, *indefinit*, *nominativ*) och följaktligen också hur den tolkat en för Granska felaktig mening. Granska ställer en diagnos t.ex. *kassa biträde* är en *misstänkt särskrivning*, hänvisar till en regel, *sär2nn_nnA@sär* och föreslår ett ersättningsförslag, i det här fallet *kassabiträde*. Det har i tidigare versioner av Granska varit tänkt att användaren ska kunna få mer hjälp t.ex. en utförligare förklaring till ett visst fel och referens till var man kan läsa mer om detta fel, t.ex. i Svenska skrivregler (1992) (Larsson, 1998). Hur och om detta ska implementeras i TvärGranska undersöktes.

Det är också tänkt att användaren ska kunna välja vilka fel han/hon vill leta efter i sin text, t.ex. bara kontrollera särskrivningar eller kongruensfel osv. Denna funktion kan vara användbar om användaren har många fel i sin text för då kan han/hon fokusera på en sorts fel i taget. Det kan också hjälpa användaren att se vilken typ av fel han/hon har problem med och behöver träna mer på.

Hur denna information på bästa sätt presenteras för användaren testades genom att göra användarstudier i olika stadier av utvecklingen av TvärGranska. I början av utvecklingen kom handledarna med synpunkter och när utvecklingen kommit lite längre visades TvärGranska upp på en studiecirkel på IPLab, för att efter ändringarna som gjordes efter den, göra en större användarstudie med användare som har svenska som andraspråk. Med på studiecirkeln var personer med erfarenhet och kunskaper om andraspråksinlärare och språkgranskningsprogram och de hade många bra synpunkter (se nedan).

4.5 Kommunikation på dataformatet XML

När det gäller kommunikation på dataformatet XML tittades det på serverexekverade språk och en färdig teknik för att hantera XML. XML är ett beskrivningsspråk lämpligt för data som man vill kunna extrahera olika delar ur och göra olika saker med (se appendix C1 för ett XML-exempel). Man kan med XML själv bestämma vad de olika beskrivande taggarna ska kallas i motsats till t.ex. HTML där de är förutbestämda. Det man sedan får göra är att skriva en beskrivning över vilken struktur och vilka taggnamn man satt, t.ex. i form av en DTD

(<http://www.w3.org/TR/REC-xml#dt-doctype>) eller ett XML Schema (<http://w3c.org/XML/Schema>) (se appendix C2 för hur ett XML Schema kan se ut).

Efter diskussionen ovan om val av programmeringsspråk och vilket man därför ska välja kom jag fram till att jag skulle utveckla i JSP (Java Server Pages) eftersom jag redan kunde och kände mig bekväm med det serverexekverade språket. Dessutom är Java bra på att hantera XML. Därför blev det också naturligt att välja någon av de tekniker som finns i Java för att läsa och tolka XML; Java-XML Data Binding, DOM eller SAX (<http://java.sun.com>).

4.6 Återkoppling med Granska

Återkopplingen med Granska kan lösas genom att varje gång användaren har redigerat i en mening skickas den meningen till Granska igen för kontroll, detta innebär dock att användaren måste ha tillgång till en uppkoppling mot Internet under hela tiden han/hon jobbar med TvärGranska.

Om återkopplingen med Granska är önskvärd att lösa genom att man kontrollerar meningen som användaren redigerade nyss mot Granska en gång till, så är lösningen med ett applet-program mer komplicerad än lösningen med HTML och JavaScript.

5 Utvecklingen av TvärGranska fram till användarstudien

Här redogörs för arbetets gång, hur de olika identifierade problemen löstes och hur gränssnittet växte fram.

5.1 Integrering av TvärGranska i skrivprocessen

Hur man infogar sin text i TvärGranska löstes med att användaren får klistra in texten i ett fält i webbläsaren. Detta för att det är den enklaste lösningen, den är både oberoende av operativsystem och webbläsare. Enkäten visade också att användarna kunde kopiera och klistra i olika program på datorn och att hälften av dem använder sig även av kortkommandona (t.ex. Ctrl+C) för att göra det, se appendix B2.

Ett annat alternativ som undersöktes var att utveckla en tilläggsfunktion i Microsoft Word i form av en knapp som tog användaren och texten till TvärGranska och en knapp i TvärGranska som tog användaren och texten tillbaka till Microsoft Word. Den fördel som detta kunde ha givit i form av att kunna skriva i Microsoft Word, och efter granskning, komma tillbaka till Microsoft Word och formatet/stilen med stycken och rubriker skulle vara kvar visade sig vara svårt. RTF-formatet är inte helt likvärdigt med Microsoft Words sätt att formatera texten. I slutändan skulle alltså dokumentet antagligen inte se ut precis som det gjorde innan granskningen med TvärGranska vilket ju skulle vara poängen med denna lösning. Användaren skulle också med stor sannolikhet bli tvungen att spara sitt dokument som RTF, något som kan ses som knepigt för en datorovan användare. Enkäten visade att endast en tredjedel av de svarande visste hur man sparade i olika format på en dator (se appendix B2). En annan risk med tilläggsfunktionen i form av en knapp i Microsoft Word är att man skulle skicka väldigt långa texter till TvärGranska, vilket gör att det blir mer svåröverskådliga för användaren vid granskningen. Att göra en lösning till Microsoft Word, ett program som inte är gratis, kan också ses som orättvist av dem som inte har tillgång till Microsoft Word. En annan aspekt är också att TvärGranska blir mer som en produkt än som en lättillgänglig tjänst med denna lösning.

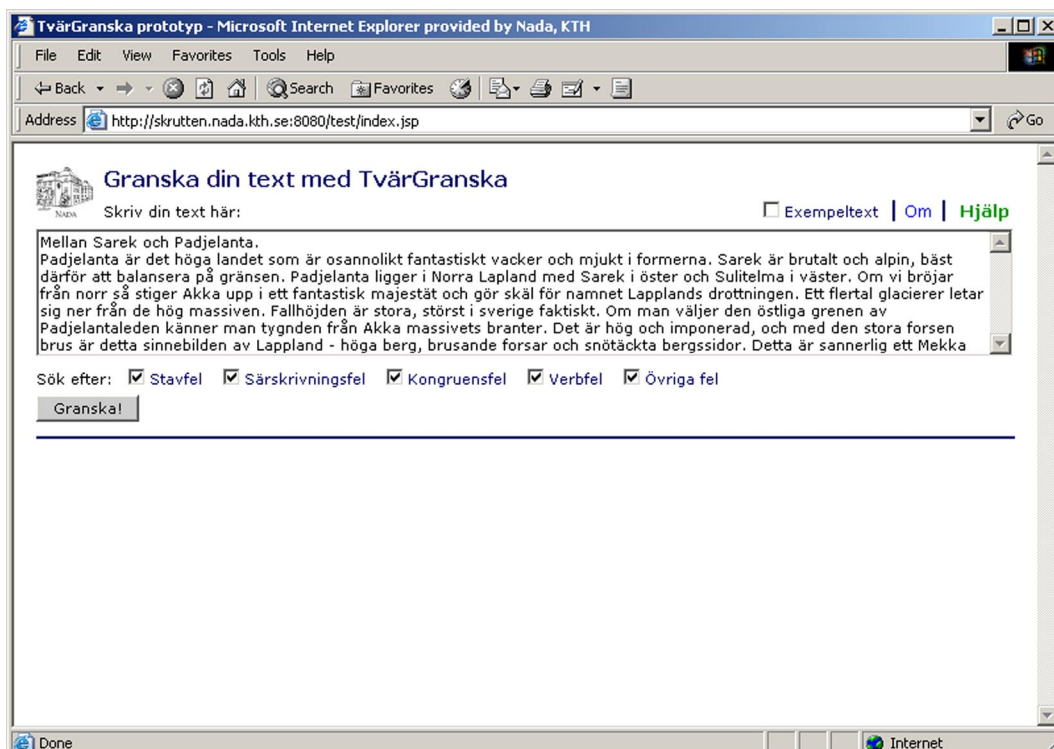
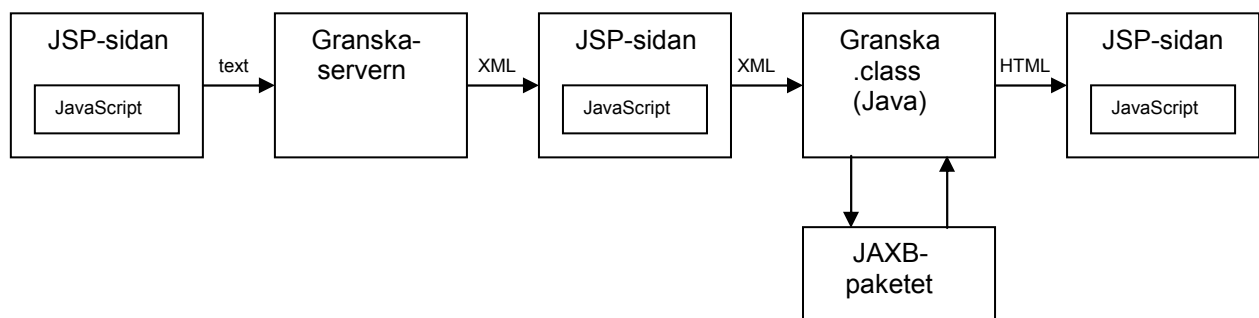


Bild 5.1 Huvudfönstret/Granskningsfönstret innan *Granska!*-knappen trycks på.

5.2 Grad av interaktion med TvärGranska

När det kommer till interaktionen med TvärGranska valdes lösningen med ett appletprogram bort, till stor del på grund av att TvärGranska ska vara lättillgängligt. Om man väljer lösningen med ett appletprogram behöver användaren ha en webbläsare med JVM. Det har visserligen de flesta läsare idag, men det finns olika versioner av JVM:en och det är inte säkert att appletprogrammet fungerar med alla. Ett exempel på det är KTHs studiedokumentation, PING, som bara fungerar med JVM version 1.3.1 och inte med den nyaste versionen av JVM 1.4. som är standard i nya webbläsare. Det vill säga att även om man utvecklar för den nyaste versionen av JVM kan det sluta fungera när det kommer en nyare webbläsare med en nyare JVM. Ett appletprogram har också den nackdelen att det tar tid att ladda den.

Valet föll således på att göra en JSP-sida (se bild 5.1) med användandet av JavaScript för att åstadkomma mer interaktion. När man har skrivit eller klistrat in sin text i formuläret på sidan och tryckt på *Granska!*-knappen kommer JSP-sidan att skicka texten till Granskaservern, se figur 5.2. Granskaservern returnerar en XML-fil (se appendix C1) som sedan skickas till Javaklassen Granska som med hjälp av den nya tekniken Java-XML Data Binding bygger ett objektträd av taggarna i XML-filen. Med hjälp av objekten tar man sen ut innehållet ur taggarna och presenterar det på ett ändamålsenligt sätt på JSP-sidan, se bild 5.4 och nedan kapitel 5.3.



Figur 5.2 Schematisk bild över förloppet efter att användaren tryckt på *Granska!*-knappen.

När man klickar på ett felmarkerat ord kommer, med hjälp av JavaScript, ett nytt litet fönster att öppnas med information om felet, se bild 5.3. Detta fönster är också en JSP-sida som kontaktar Granskaklassen för att hämta informationen om felet för att sedan presentera den. I fönstret finns också ett litet formulär med den felaktiga meningen i så att man kan rätta den. Om man gör det och trycker på *Uppdatera texten*-knappen kommer den nya meningen att skickas till Granskaservern för granskning och presenteras sedan igen i granskningsfönstret med samma plats i texten som tidigare, och informationsfönstret stängs.

För stavfel och särskrivningsfel finns det en finess i informationsfönstret som gör att användaren kan klicka på det ersättningsförslag som han/hon tror är riktigt och då kommer det automatiskt att ändras i meningen som ligger i formuläret. Detta görs också med JavaScript.

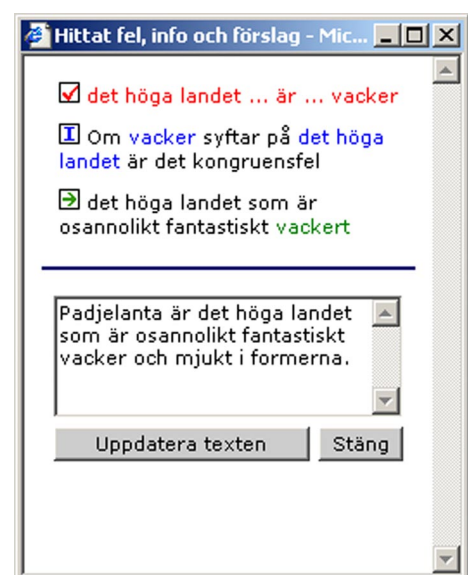


Bild 5.3 Informationsfönstret

Tyvär har det inte gått att göra för alla typer av fel då det som JavaScriptet gör är att byta ut det som står som *markerat fel* (☒) med *ersättningsförslaget* (☒) som användaren klickar på. Det vill säga det letar upp det ställe i texten där det som är fel står och byter ut det mot ersättningsförslaget, men i fallet som syns på bild 5.3 kommer inte ”det höga landet...är...vacker” att kunna matchas med något i meningen, eftersom det i meningen inte finns några ”...”.

I granskningsfönstret har man även möjligheten att redigera en mening som inte innehåller några felmarkerade ord, det gör man genom att klicka på bilden av pennan i slutet av meningen, se bild 5.4. Då plockas meningen med hjälp av JavaScript upp i formuläret och samtidigt dyker en *Uppdatera texten*-knapp upp. När man ändrat klart i meningen trycker man på den och som ovan skickas meningen till granskning innan den åter presenteras i texten.

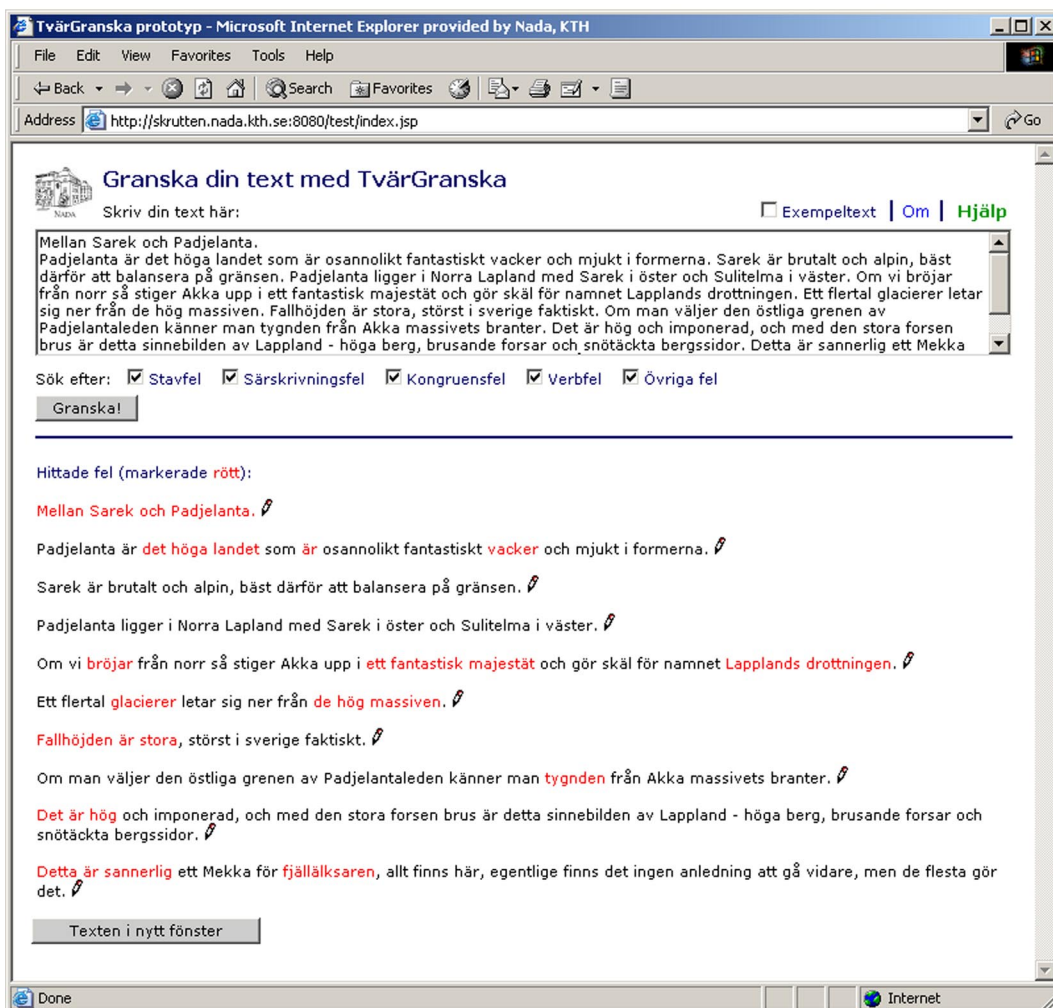


Bild 5.4 Huvudfönstret/Granskningsfönstret efter att *Granska!*-knappen tryckts på.

När man har granskat klart sin text trycker man på *Texten i nytt fönster*-knappen (se bild 5.4) och får då upp en ny JSP-sida med bara texten i, se bild 5.5. JSP-sidan kontaktar Granskaklassen som returnerar texten, men utan röda markeringar, pennor och mellanrum mellan meningarna den här gången. Härifrån kan användaren nu kopiera texten och klistra in den i ett textdokument till exempel.

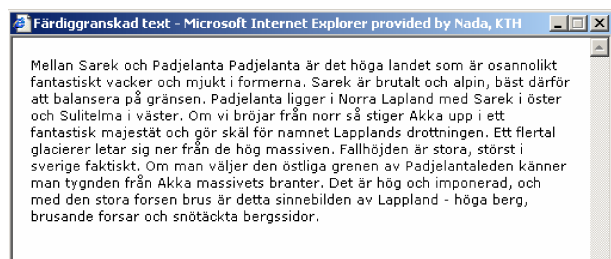


Bild 5.5 Fönstret med den färdiggranskade texten

5.3 Presentation av information

Texten som användaren valt att granska presenteras med varje mening på en ny rad, detta för att det ska bli tydligt och enkelt att fokusera på meningarna en i taget, se bild 5.4. Till markeringen av de ord som TvärGranska tycker är felaktiga valdes färgen röd. Det övervägdes att välja en annan färg eftersom rött tydligt signalerar att något är felaktigt och Granska inte är hundra procentig i felmarkeringen, men på demonstrationen av TvärGranska på studiecirkeln kom vi fram till att ingen annan färg uppfattas lika tydligt. Det är inte heller att förringa att användarna sannolikt har blivit vana vid röda markeringar från t.ex. Microsoft Word som använder den färgen för att markera felstavningar. Bilden av pennan efter en mening som ikon för att det är möjligt att redigera meningen är allmänt vedertagen på webben och kändes därför som ett bra val.

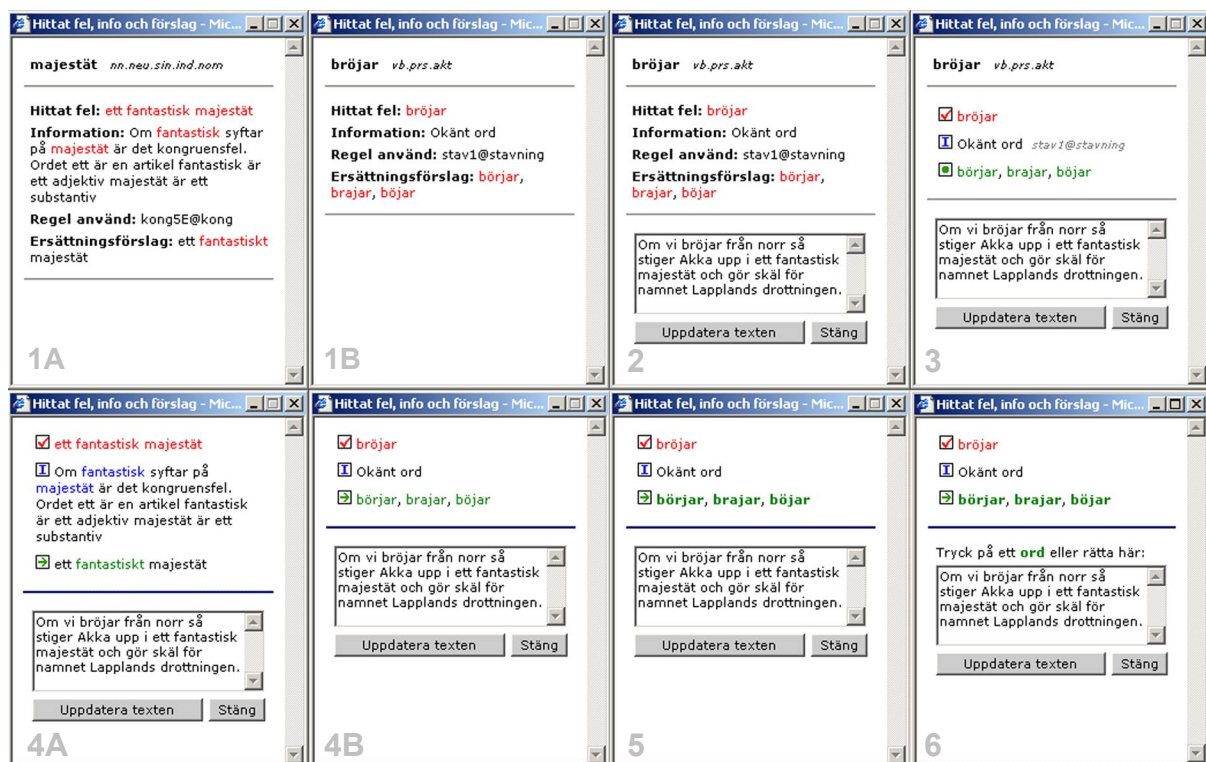


Bild 5.6 Olika steg i utvecklingen av informationsfönstret. A och B är olika exempel i samma utvecklingsstadium.

I informationsfönstret, som kommer upp när man klickar på ett rödmarkerat ord, valdes *hittat fel*, *information om felet* och *ersättningsförslag* att symboliseras av ikoner (se bild 5.3). När en användare väl lärt sig vad en ikon står för är ikonen enklare än text (Preece et al, 1994). *Hittat fel* symboliseras av ruta med en röd bock i , ikonen för *information om felet* är en ruta med ett blått versalt I i **I**, och för *ersättningsförslag* valdes en ruta med en grön pil pekandes åt höger i **➤**, eftersom Paulsson & Widengren (2002) kom fram till att det var en väl fungerande symbol för ersättningsförslag. När man håller muspekaren över ikonen får man ett så kallat ”tooltip”, det vill säga en förklarande text som berättar vad ikonen symboliserar.

Hur designen växte fram under utvecklingens gång kan ses i bild 5.6. ”Bröjar”-bilderna följer en tidsaxel i utveckling och de två bilderna till vänster är ytterligare exempel för att visa hur det ser ut när informationstexten är längre. De första bilderna är från ett tidigt stadium i utvecklingen och ganska snart insågs behovet av att använda sig av ikoner i stället för text. I de första bilderna är nämligen den beskrivande texten för dominant vilket gör att det

felmarkerade ordet, informationen och ersättningsförslagen blir mer svårlästa. Att denna design ändå blev kvar ett tag har att göra med skälet till att utvecklaren och designern till ett gränssnitt inte bör vara samma person, dvs. utvecklaren är så insatt i det funktionella och blir så nöjd när något fungerar att designen då hamnar i skymundan trots att den är minst lika viktig.

Den sista bilden på översta raden visar hur gränssnittet såg ut när det visades på studiecirkeln, i enlighet med synpunkterna som kom upp där togs sedan ordklasstaggen (t.ex. *vb.prs.akt*) och regeln (t.ex. *stav1@stavning*) bort med motiveringen att de inte var relevanta för användaren. Det blev en enklare design där de andra delarna blev tydligare för användaren. Det diskuterades också ifall ordklasstaggen skulle vara användaren till nytta om man skrev ut vad förkortningen står för dvs. skriva *verb, presens, aktiv* form istället för *vb.prs.akt* men konsensus i frågan nåddes aldrig. Detta är dessutom sådan information som inte finns i XML-filen och skulle alltså få läggas till för hand i gränssnittet för de 148 olika ordklasstaggar som finns i Granska. Annan information som inte finns i XML-filen idag som diskuterats är möjligheten att kunna få mer information om var man kan läsa mer om ett fel (t.ex. i Svenska skrivregler) även detta skulle då få läggas till för hand. Om man ska lägga till denna information borde det alltså göras i XML:en så att man kan använda den även om man utvecklar ett nytt gränssnitt, i detta examensarbete var detta inte prioriterat arbete. På studiecirkeln rekommenderades också att den gröna punkten för *ersättningsförslag* skulle bytas mot pilen i enlighet med vad Paulsson & Widengren (2002) kom fram till i sin användarstudie.

I informationsfönstret finns även en textruta med meningen som felet ingår i, så att man kan korrigera den utifrån informationen man får. Det som är att föredra här är att man kan klicka på de ersättningsförslag som ges och det ändras automatiskt i textrutan, tyvärr var detta endast möjligt att göra för de fel som tillhör kategorierna stavfel och särskrivningsfel, se ovan. (Om felet tillhör någon av de andra kategorierna måste man ändra själv i textrutan.) Detta medför att man behöver en urskiljning mellan vilka ersättningsförslag man kan klicka på och vilka man inte kan klicka på. Olika färger provades och även understrykning, men det upplevde jag som för plottigt och inte tillräckligt tydligt, så valet föll på att göra de ersättningsförslag som man kan klicka på i samma gröna färg som de ersättningsförslag som man inte kan klicka på men med den skillnaden att de står med fet stil (se bild 5.6). En liten risk med det är dock att när det för användaren presenteras två olika förslag på fel samtidigt, t.ex. särskrivningsfel och kongruensfel, kan användaren luras att tro att särskrivningsfelet är mer rätt eftersom dess ersättningsförslag kommer att stå med fet stil och ersättningsförslaget till kongruensfelet kommer att stå med tunnare stil. Givetvis är det önskvärt att alla ersättningsförslag i framtiden ska gå att klicka på, men det kräver att XML:en görs om så att det inte finns fall där det inte går att matcha felet med en del av meningen som i bild 5.3. Eventuellt skulle man kunna skriva ett mycket avancerat JavaScript för att matcha texterna, men då det blir komplicerat fanns det inte tid för det under utförandet av detta examensarbete.

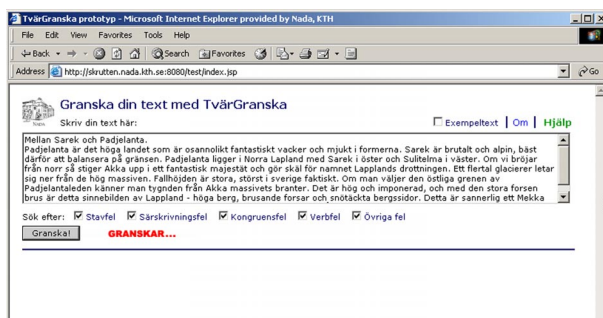


Bild 5.7 JSP-sidan med GRANSKAR...-bild.

Efter att användaren har tryckt på *Granska!*-knappen kommer det att dröja ett tag innan TvärGranska returnerar den av Granska granskade texten, medan detta arbete pågår visas en rörlig *gif*-bild med texten GRANSKAR ... där punkterna läggs till och tas bort, se bild 5.7. På så sätt får användaren återkoppling och vet att något händer.

Om man trycker på det gröna ordet Hjälpen i det vänstra hörnet i granskningsfönstret kommer man till hjälpen, se bild 5.8. Hjälpen utformades med tanke på att det skulle vara lätt att hitta information om en detalj man inte har förstått men att man även ska kunna läsa igenom hela för att få insikt i hur man kan arbeta med gränssnittet. Genom att lägga ikoner och knappar från gränssnittet i texten blir det enkelt för användaren att hitta det som han/hon söker efter.

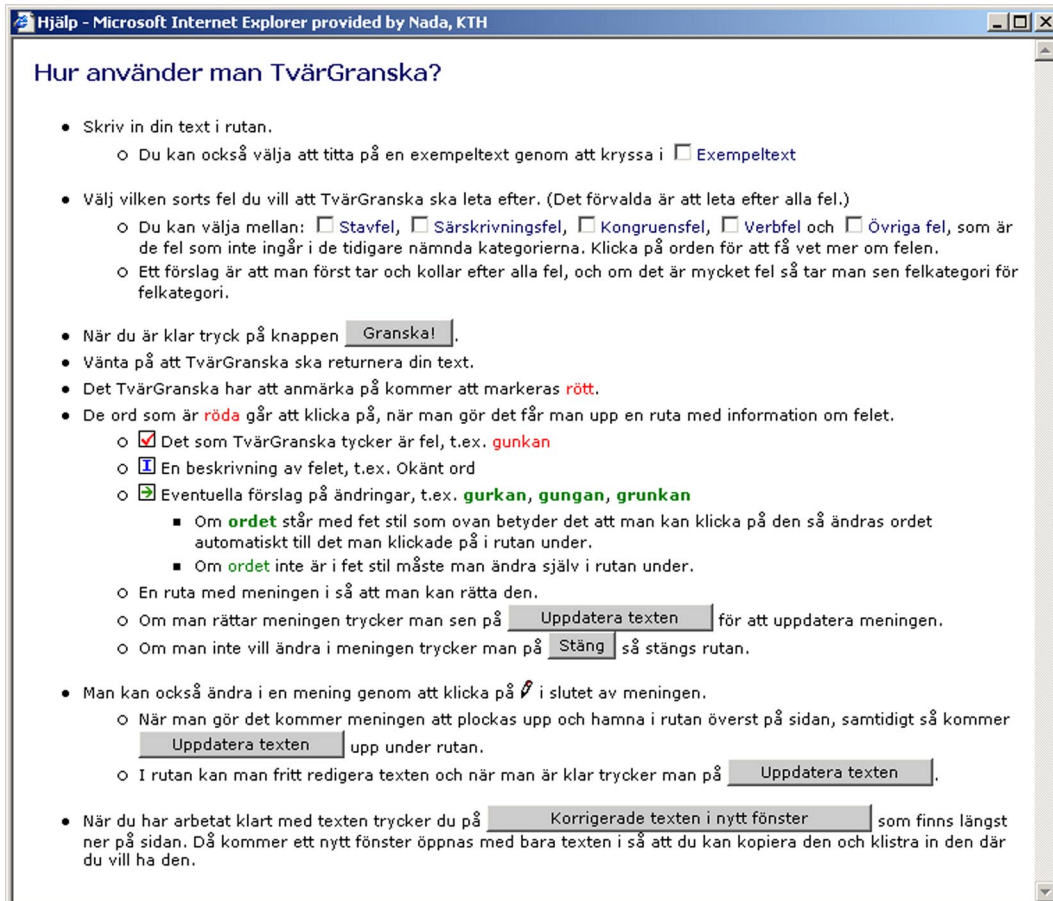


Bild 5.8 Hjälpen i Tvärgranska

5.4 Kommunikation på dataformatet XML

Valet av vilken av de färdiga tekniker som finns i Java för att hantera XML som skulle användas föll på Java-XML Data Binding. Det är en ny teknik som går ut på att man gör ett objektträd av alla taggar i XML-filen. Det Java-XML Data Binding-API jag valt att använda är det som kommer från Sun, Java Architecture for XML Binding (JAXB). JAXB går att ladda ner från Sun (<http://java.sun.com/xml/jaxb>). Detta API är väldigt nytt vilket betyder att jag har arbetat med JAXB v1.0 Beta, den slutgiltiga JAXB v1.0 kom samtidigt som sammanställandet av denna rapport.

För att kunna generera Java-objekten behöver man en beskrivning av XML-strukturen, i detta fall ett XML Schema. Då ingen tidigare beskrivning av den aktuella XML:en fanns fick jag utifrån en exempelfil (se appendix C1) skriva ett XML Schema (se appendix C2). För att man ska kunna generera objektträdet krävs det att XML Schema:t är validerat, det görs lämpligen med Suns Multi-Schema XML Validator som finns att ladda ner från Suns hemsida (<http://java.sun.com>). När man har ett validerat XML Schema genererar man med hjälp av JAXB ett träd med Java-objekt, ett Java-objekt per XML-tagga. Med hjälp av dessa Java-

objekt kan man sedan läsa innehållet i XML-filen och presentera det på t.ex. webben. (Möjligheten finns även att med dessa objekt skapa XML-filer, men det är ju inte aktuellt i det här projektet.)

5.5 Återkoppling med Granska

I och med att lösningen med ett appletprogram valdes bort löstes återkopplingen med Granska så att efter man redigerat i en mening skickas den igen till Granska för att se om några fel återstår eller om det blivit några nya fel i och med ändringarna. Lösningen att kontrollera en mening en gång till efter att användaren ändrat i den visade sig vara nödvändig eftersom användaren får ändra fritt i meningen och följaktligen kan göra slarvfel såsom att trycka på fel tangent eller dylikt.

6 Resultat

Det här är resultaten av mina undersökningar; enkäten och användarstudien. Presenteras görs också de ändringar som gjordes i gränssnittet efter användarstudien.

6.1 Enkät

Målet med enkäten var att undersöka hur mycket datorvana och vilka önskemål de tilltänkta användarna hade vad gäller integreringen av TvärGranska i skrivprocessen och att utifrån deras svar sedan konstruera TvärGranska efter vad man kom fram till var den bästa lösningen i denna fråga.

6.1.1 Utförande

Enkäten delades ut till en klass som läser Svenska som främmande språk, nivå 3 på Stockholms universitet. Efter att ha läst den kursen är man behörig för andra högskolestudier på svenska. Klassen bestod av 18 personer med olika språkbakgrund och med olika lång tid i Sverige.

6.1.2 Diskussion

När man analyserar svaren man fått på enkäten får man ha i åtanke att de som svarat inte behärskar svenska språket till fullo och även om alternativet ”förstår inte frågan” fanns på alla frågor kan det ha blivit fel ändå.

En av de mest grundläggande frågorna för program som Granska, nämligen *Kan du tänka dig att göra stavnings- och grammatikkontrollen i ett annat program än det du skrivit din text i?* fick det tillfredsställande svaret att majoriteten (67 %) kunde det, se diagram 6.1.

Nästa intressanta fråga var hur bra kunskaper i att kopiera och klistra på en dator de tilltänkta användarna hade. De fanns tre olika frågor som behandlade det ämnet i enkäten; *Känner du till hur man kopierar, klipper och klistrar på en dator?*, *Använder du dig av kortkommandona för att kopiera, klippa och klistra? (t.ex. Ctrl+C)* och *Skulle du kunna tänka dig att klistra in och klippa ut din text i/ur programmet som kontrollerar din text för stavfel och grammatikfel?*. Alla dessa frågor gav också tillfredsställande svar. Alla utom en (94 %) svarade att de visste hur man kopierade och klistrade på en dator (se diagram 6.2), hälften uppgav också att de använde sig av kortkommandona (se diagram 6.3) och ingen svarade att de inte kunde tänka sig att kopiera och klistra in i granskningsprogrammet (se diagram 6.4).



Diagram 6.1 Villighet att byta skrivmiljö



Diagram 6.2 Kan kopiera och klistra på dator



Diagram 6.3 Använder sig av kortkommandon för att kopiera och klistra på dator



Diagram 6.4 Kan tänka sig att kopiera och klistra in i granskningsprogrammet

Slutsatsen blir att man kan se det som allmänt känt hur man kopierar och klistrar text på en dator.

Enkäten innehöll även en fråga om kunskapen att spara i olika format, *Vet du hur man sparar i olika format i en ordbehandlare? (t.ex. .doc, .html, .rtf eller .txt)*, detta visade sig enligt enkätsvaren var ganska svårt, majoriteten (67 %) svarade nej på denna fråga (se diagram 6.5). Detta var en bidragande orsak till hur integrationen med TvärGranska i skrivprocessen valdes att göras.

En annan viktig fråga när man pratar om ett webbgränssnitt är användarnas vilja att vara uppkopplad under tiden de arbetar med granskningsprogrammet och därför ställdes frågan; *Kan du tänka dig att sitta och skriva på en dator uppkopplad till Internet för att kunna köra programmet som kontrollerar stavfel och grammatikfel?.* Svaret på den blev att 39 % sa ja, både hemma och i skolan, 33 % sa ja, i skolan och 22 % sa ja, hemma (se diagram 6.6). Denna fråga handlar till stor del om vilken sorts uppkoppling man har tillgång till och problemet kommer att minska i takt med att de gamla långsamma modemerna tas ur bruk och fler får tillgång till fast uppkoppling.

Den sista frågan på enkäten handlade om på vilket sätt användarna helst skulle se språkgranskningsprogrammet integrerat i skrivprocessen och inte så förvånande sade det flesta här (61 %) att de helst skulle se att man kunde komma till det från Microsoft Word (se diagram 6.7). Det näst vanligaste förslaget var att klippa och klistra (22 %) och efter det att skriva i ett textredigeringsprogram på webben (11 %).

Alla svar från enkäten redovisas i appendix B2.

6.2 Användarstudie

Målet med användarstudien var att undersöka om användaren kan agera interaktivt med gränssnittet utan förevisning eller mer förklaring om hur det fungerar än vad som finns att läsa i hjälpen. Mitt mål med gränssnittsutförningen är således att gränssnittet ska vara så pass intuitivt att man kan ge en användare webbadressen och han/hon ska själv kunna tillgodogöra sig tillräcklig information för att kunna använda gränssnittet. Att undersöka hur användarna interagerar med gränssnittet ger viktigt information till utvecklingen av gränssnittsdesignen.

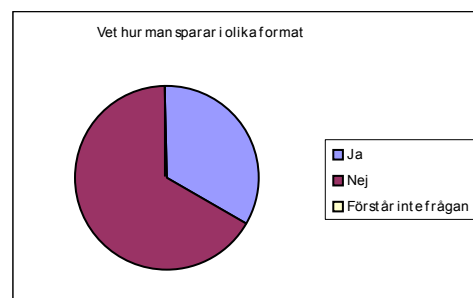


Diagram 6.5 Vet hur man sparar i olika format



Diagram 6.6 Kan tänka sig vara uppkopplad för att använda granskningsprogrammet

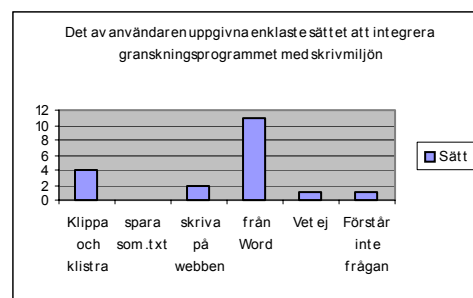


Diagram 6.7 Det av användaren uppgivna enklaste sättet att integrera granskningsprogrammet med skrivmiljön

De viktigaste detaljerna i gränssnittet att undersöka var:

- Att det gick bra att kopiera och klistra in texten i gränssnittet.
- Att det var tydligt vilken knapp man skulle trycka på för granskning.
- Att färgen röd var bra som markering av felaktiga ord och att det var tydligt att man kunde klicka på det rödmarkerade ordet för information om felet.
- Att ikonerna i informationsfönstret var tydliga.
- Att funktionaliteten hur man korrigerar meningar var bra.
- Att knapparnas namn speglade vad de gjorde.
- Att hjälpen var bra utformad.

Och om användarna fann gränssnittet funktionellt och något de kunde tänka sig att använda sig av i framtiden.

6.2.1 Utförande

Användarstudien genomfördes med fyra studenter från Folkuniversitetet som läser Svenska för akademiker på nivå B2 enligt Europarådets nivåskala, vilket motsvarar följande:

Jag kan följa huvuddragen i komplexa texter om konkreta såväl som abstrakta ämnen. Inom mitt specialområde förstår jag även fackdiskussioner. Jag kan kommunicera med så pass hög grad av ledighet och spontanitet att ett samtal med en infödd samtalspartner förlöper i stort sätt problemfritt. Jag kan uttala mig klart och detaljerat om ett stort antal varierande ämnen, jag kan förklara min ståndpunkt i en problemsituation och kan diskutera olika lösningars för- och nackdelar.

Användarstudien gick till på följande sätt:

- Användaren fick börja med att fylla i ett medgivandeformulär (se appendix D1) där han/hon säger ja till att delta i studien och att det insamlade materialet får användas i forskningssyfte.
- Användaren har i ett brev (se appendix D2) blivit ombedd att ta med sig en egenproducerad ogranskad text som han/hon nu öppnar i Word och enligt instruktionerna (se appendix D3) klistrar in i TvärGranska.
- Användaren granskar sin text med TvärGranska och försöksledaren observerar användarens interaktion med gränssnittet. Försöksledaren ska i möjligaste mån endast observera hur användaren interagerar med gränssnittet och inte svara på frågor annat än att hänvisa till hjälpen. Studien dokumenteras genom att försöksledaren antecknar hur användaren agerar i olika moment i granskningsprocessen i försöksledarprotokollet (se appendix D4). (Att videofilma användaren anses i detta fall som ett hinder för att användaren ska känna sig självsäker i sitt utforskande av gränssnittet.)
- Efter att användaren granskat sin text ställer försöksledaren frågor om hur användaren upplevde sitt arbete med granskningsgränssnittet (se appendix D5).
- Användarstudien avslutas med att användaren fyller i en kort enkät om modersmål, ålder, datorvana etc. (se appendix D6).

I utförandet av användarstudien hade jag hjälp av en annan exjobbare på IPLab, Jenny Rahbek från Språkteknologiprogrammet på Uppsala Universitet. Vi observerade två användare var, jag observerade först användare 1 medan Jenny Rahbek observerade användare 3, därefter observerade jag användare 2 medan Jenny Rahbek observerade användare 4. Efter användarstudien sammanställde vi materialet vi fått in tillsammans.

Användarna har olika bakgrund och olika mycket datorvana, se tabell 6.1 och de interagerade med gränssnittet på ganska olika sätt. Användare 1 var snabb och entusiastisk, användare 2 var mycket passiv, användare 3 var systematisk och användare 4 var snabb och otålig. Användare 1 och 2 hade korta egna texter med få fel i med sig, det gjorde att försöksledaren fick visa dem exempeltexten för att de skulle ha någon text att jobba med, tyvärr gjorde det att användare 2 inte känner behovet av att ändra några fel utan bara klickade och tittade.

Bakgrundsdata	Användare 1	Användare 2	Användare 3	Användare 4
Kön:	Kvinna	Man	Kvinna	Man
Ålder:	30-39 år	30-39 år	19-29 år	19-29 år
Utbildning:	teknisk högskoleutbildning	teknisk högskoleutbildning	ekonomisk högskoleutbildning	teknisk högskoleutbildning
Modersmål:	kinesiska	bosniska	kroatiska	arabiska
Andra språkkunskaper:	engelska	albanska	engelska, tyska	engelska
Tid i Sverige:	6 år	4,5 år	1 år	2 år
Svenskstudier:	SFI: 2,5 månader svenska B2 i 10 veckor (den kurs de läser nu)	SFI svenska för akademiker (den kurs de läser nu)	SFI: 8 månader självstudier med Linguaphone	SFI: 29 veckor
Datorvana:	Stor	Liten	Tillräckligt stor	Mycket stor
Använder ordbehandlingsprogrammen:	Microsoft Word	Microsoft Word e-postprogram	Microsoft Word e-postprogram	Microsoft Word e-postprogram
Använder stav- och grammatikkontroll:	Ja.	Ja.	Ja.	Ja.
Försöksledarkommentarer:	Användare 1 och försöksledaren har ibland svårt att förstå varandra p.g.a. språket.	Användare 2 har liten datorvana och blir väldigt passiv i utforskandet av gränssnittet, testar först efter att försöksledaren har förklarat en viss funktion.	Användare 3 gick igenom gränssnittet systematiskt och noggrant.	Användare 4 missade många funktioner eftersom han gick igenom gränssnittet snabbt och otåligt.

Tabell 6.1 Användarnas bakgrundsdata (resultatet av bakgrundsenkäten + försöksledarkommentarer)

Tabell 6.2 visar resultatet av observationen, nämnas kan att det är svårt att veta hur användaren hade gjort i vissa situationer om han/hon inte hade haft en försöksledare bredvid sig. Trots att försöksledaren försökte att hänvisa användaren till hjälpen, var det för de flesta av användarna inte naturligt att titta där när det dök upp något de inte förstod. En reaktion var också att användaren tyckte att det var bra att det fanns hjälp men förväntade sig ändå att försöksledaren skulle förklara den här gången.

Uppgift:	Användare 1	Användare 2	Användare 3	Användare 4	Kommentarer:
att skriva in rätt adress i Internet Explorer?	Ja	Ja*	Ja	Ja	* Fick hjälp att öppna IE.
att kopiera sin text i Microsoft Word?	Ja	Nej	Ja	Ja	
att klistra in sin text i TvärGranska?	Ja	Nej	Ja	Ja	
att hitta rätt knapp att trycka på för granskning?	Ja	Ja	Ja	Ja	
att hitta hjälpen om det är något han/hon undrar över?	Nej*	Nej*	Nej*	Nej*	* Fick påpeka att det fanns hjälp i gränssnittet att få.
att uppfatta att klicka på de röda orden?	Ja	Nej*	Ja	Ja	* Bara siffror markerades rött i användarens text, förstod när han såg exempeltexten.
Om nej, efter att ha tittat i hjälpen?		Nej*			* Läser inte hjälpen så noggrant.
att i lilla nya fönstret förstå ikonerna?	Ja	Ja	Ja	Ja	
att uppfatta att man kan hålla muspekaren över för att få en förklaring till ikonen?	Inget behov	Inget behov	Ja	Inget behov	
att uppfatta att ändra i ändringsrutan?	Ja*	Inget behov	Nej	Ja	* Klickade först på knappen <i>Uppdatera texten</i>
Om nej, efter att ha tittat i hjälpen?			Ja		
att uppfatta när man kan klicka på ersättningsförslagen och när man inte kan det?	Nej	Inget behov	Nej	Nej	
Om nej, efter att ha tittat i hjälpen?	*		Ja	*	* Förstår inte att de behöver läsa i hjälpen.
att uppfatta att man ska trycka på <i>Uppdatera texten</i> efter att man ändrat i rutan?	Ja	Inget behov	Ja	Ja	
att uppfatta att man kan stänga rutan genom att trycka på <i>Stäng</i> ?	Ja	Vet ej*	Ja	Ja	* Använder X för att stänga rutan.
att uppfatta att man kan ta bort regler, för att titta på ett sorts fel i taget?	Ja	Nej	Ja	Ja	
Om nej, efter att ha tittat i hjälpen?		Nej*			* Försöksledaren får förklara.
att uppfatta att man kan få en förklaring av felregeln genom att klicka på den?	Nej	Nej	Ja	Vet ej	
att uppfatta att man i huvudfönstret kan ändra i alla meningar genom att trycka på pennan?	Nej*	Nej*	Nej*	Tveksamt*	* Ingen har behovet, försöksledaren får uppmärksamma användaren på pennan.
Om nej, efter att ha tittat i hjälpen?	*	*	Ja	*	* Tittade inte i hjälpen.
att uppfatta att meningen efter man tryckt på pennan hamnar i rutan överst på huvudsidan?	Nej	Nej	Ja*	Nej	* Läst om pennan i hjälpen.
att uppfatta att man efter att ha ändrat ska trycka på <i>Uppdatera texten</i> som var gömd förut?	Nej	Nej	Ja	Ja	
att uppfatta vad knappen <i>Texten i nytt fönster</i> gör?	Inget behov	Inget behov	Ja*	Ja*	* Tryckte på knappen efter anmodan.
att efter granskning kopiera och klistra tillbaks texten i Word?	Inget behov	Inget behov	Ja*	Ja	* Skulle nog ha förstått om det inte blivit fel i textrutan.

Tabell 6.2 Försöksresultatet

Användarna är överlag positiva till gränssnittet och säger att de kommer/kan tänka sig att använda det igen. De tycker det är bra för det hittar fel som inte Microsoft Word hittar och det är gratis. Hjälpen anses dock som lite svår om man inte är så bra på svenska. Vad användarna svarade mer på i intervjun framgår av tabell 6.3.

Intervjufrågor	Användare 1	Användare 2	Användare 3	Användare 4
Hur tycker du granskandet av texten gick?	Bra, TvärGranska anmärkte på grammatik- och stavfel som Word inte hittade.	Mycket bra.	Ganska bra.	Inte så bra, t.ex. hittades inte vissa stav- eller grammatikfel.
Var det svårt att komma igång?	Nej, mycket lätt.	Jag är inte så van vid datorer så lite svårt.	Inte så svårt, men jag är inte så van vid datorer.	Nej.
Var det något som var svårt att förstå?	Ordet exempeltext förstod jag inte helt, förstod dock ordet exempel. Förstod först inte att texten i ändringsrutan inte ändrades automatiskt när man tryckte på <i>Uppdatera texten</i> i informationsfönstret.	Nej.	Inget spontant.	<i>Texten i nytt fönster</i> var inte självförklarande, man förstod inte förrän man hade provat. Förväntade mig att hela texten skulle finnas i ”Skriv din text här”-rutan hela tiden.

Tabell 6.3 Del 1 Svaren på intervjufrågorna

Intervjufrågor	Användare 1	Användare 2	Användare 3	Användare 4
Var det tydligt vilken knapp du skulle trycka på när du skulle granska texten första gången?	Ja.	Ja.	Ja.	Ja.
Förstod du att du skulle klicka på de ord som var markerade rött?	Ja.	Ja.	Ja, det står här. (Pekar på texten "Hittade fel (markerade rött):")	Ja.
Förstod du att du kunde avmarkera vissa regler för att leta efter bara en sorts fel?	Inte först.	Ja. (Efter att försöksledaren förklarar.)	Ja.	Ja.
Förstod du de olika reglerna? Märkte du att man kunde klicka på dem för att få en förklaring?	Lite oklart, förstod inte särskrivningsfel. Bra att man kan klicka.	Inte kongruensfel. Nej.	Ja, vi har pratat om alla fel på lektionerna.	Ja.
Förstod du efter att ha läst förklaringen/exemplen?	Ja.	Ja.	Ja.	Läste aldrig förklaringen.
Förstod du vad pennan användes till? Tryckte du på pennan?	Nej. Nej.	Nej. Nej.	Förstod efter att ha läst i hjälpen. Pennan är en möjlighet till att göra ändringar.	Nja. Ja.
Om du tryckte på pennan, märkte du att det kom fram en knapp till? Var det tydligt var den knappen var till för?	Nej. Nej.	Nej. Nej.	Nej, jag såg inte den knappen först. Själva knappen är tydlig när man sett den.	Nej. Vet ej.
När du tryckte på ett ord som var markerat rött kom det upp en ny liten ruta, vad tyckte du om den?	Jättebra. Skulle vilja ha en markering i ändringsrutan av det ord som är fel eftersom det är svårt att hitta det ordet i rutan annars.	Tydlig.	Bra med egen ruta och bra med förklaringar.	Inget konstigt.
Var ikonerna tydliga? Förstod du dem?	Ja.	Ja.	Ja, de var tydliga och begripliga.	Inte helt lättbegripliga.
Vad tyckte du om att det ibland gick att klicka på ersättningsförslagen, de gröna orden och ibland inte? Förstod du när det gick och när det inte gick?	Vore bra om det gick alla gånger. Nej.	Kom inte upp i och med att användaren tittade på exempeltexten.	Ganska bra. Förstod efter att ha läst hjälpen.	Bättre att klicka på orden. Nej.
Vad tyckte du om ändringsrutan?	Bra.	Förstod inte ändringsrutan. Jag är inte så van vid datorer, men det är nog bra.	Bra att se hela meningen.	Den var jobbig att använda vid lång mening.
Var knapparna under ändringsrutan bra? Förstod du vad de gjorde?	Ja, lätt.	Ja.	Ja.	Bra.
Vad tycker du om hjälpen?	Lite mycket text om man inte är så bra på svenska.	Bra.	Kortfattat och enkelt men inte bra för nybörjare, orden kan vara för svåra för SFI-nivå.	Jag tittar inte i hjälpen om jag inte har problem.
Tycker du att du blev hjälpt av att läsa hjälpen?	Bilderna av knapparna är bra i hjälpen, det gör det lätt att se.	Just det.	Ja!	(Användaren läste aldrig hjälpen.)
Var det något i hjälpen som var svårt att förstå?	Svårt med mycket text.	Jag förstod ej pennan. (Användaren misstolkar frågan lite.)	Nej, men för några månader sedan hade många ord varit för svåra.	(Användaren läste aldrig hjälpen.)
Kan du tänka dig/Kommer du att använda dig av det här gränssnittet i fortsättningen?	Ja, det hjälper mycket.	Ja, säkert. Bra att det är gratis, fick lov att köpa Word för kanske 800 kr för att få grammatikgranskning.	Ja.	Ja, varför inte.
Är det något du tycker borde förbättras?	Markering av ordet i ändringsrutan och automatiskt byte av ordet när man trycker på <i>Uppdatera texten</i> .	Jag är inte så van vid datorer, jag tycker det är helt okej.	Vill ha all text i "Skriv din text här"-rutan hela tiden.	Vill ändra texten på knappen <i>Texten i nytt fönster</i> till t.ex. <i>Korrigerade texten i nytt fönster</i> . Vill ha en förtydligande text i slag med "Klicka på orden för att korrigera" i informationsfönstret.
Är det något du saknar?	Tabell över hur många fel som hittades i procent.	Nej, jag är ju inte så van vid datorer.	Nej.	Vill ha olika färg på olika typer av fel.
Spontan helhetsomdöme:	Jättebra. Bra design. Schysst.	Mycket bra.		

Tabell 6.3 Del 2 Svaren på intervjufrågorna

6.2.2 Diskussion

Användarna är positiva till TvärGranska. De mest grundläggande funktionerna är tydliga för alla fyra, t.ex. *Granska!*-knappen, rödmarkerade ord och förklaringen man får när man klickar på ett rödmarkerat ord. De delar av gränssnittet som innebär vissa svårigheter är:

Hjälpen

Ingen av användarna tittade självmant i hjälpen. Det kan bero på att när försöksledaren sitter bredvid är det enklare för användaren att fråga henne. Detsamma gäller om försöksledaren ber användaren läsa i hjälpen, användare 1 och 2 tittar i hjälpen men tar inte till sig informationen, användare 4 är helt ointresserad av hjälpen, endast användare 3 använder hjälpen som det är tänkt.

Eventuell ändring: Ingen ändring kommer i åtanke, det är rimligt att tro att om användaren sitter hemma själv skulle han/hon använda sig av hjälpen i större utsträckning.

Användare 1 tycker att det är bra med bilderna av knapparna i hjälpen, hon tycker att det gör hjälpen tydligare. Användare 1 och 3 tycker dock att det är mycket text i hjälpen och de tror också att det kan vara svårt för dem som inte är så bra på svenska att förstå allt.

Eventuell ändring: Eftersom textmassan upplevs som stor kan man tänka sig att göra den lite luftigare, då kommer dock vissa användare, beroende på dator, att behöva använda rullningslistan för att se hela hjälpen och det kan leda till att textmassan upplevs som ännu större. En annan åtgärd skulle kunna vara att försöka förenkla språket mer genom ytterligare funderingar kring t.ex. ordval, eventuellt med hjälp av någon lämplig person med kunskaper inom området.

Informationsfönstret

Användare 1 tror att om man trycker på *Uppdatera texten* så har man godkänt TvärGranskas förslag och att det automatiskt ändras i meningen.

Eventuell ändring: Valet av knappens namn kan tänkas över.

Användare 3 förstod inte från början att man skulle ändra i rutan med meningen i.

Eventuell ändring: En förtydligande text i stil med ”Rätta meningen här:” ovanför rutan skulle kunna vara bra att lägga till.

Användare 1 tycker det är svårt att se vilket ord det är som ska ändras i meningen som ligger i ändringsrutan, hon föreslår att det ordet markeras.

Eventuell ändring: Ingen ändring kommer i åtanke eftersom det inte går att markera eller förtydliga ord i en vanlig HTML-textruta (med den nuvarande implementeringen).

Ikonerna

Alla förstod ikonerna, men användare 4 tyckte inte att de var självklara, skulle hellre se bokstäver på alla t.ex. F för förslag istället för en pil.

Eventuell ändring: Slutsatsen blir ändå att ikonerna är bra.

Ersättningsförslagen

Ingen användare förstod intuitivt skillnaden mellan klickbara och icke klickbara förslag. De upptäckte inte att ersättningsförslaget ibland stod i fet stil och ibland inte.

Eventuell ändring: En åtgärd kan vara att ändra presentationen av de klickbara ersättningsförslagen så att skillnaden blir större och tydligare mellan de klickbara och de icke klickbara förslagen. En idé kan vara att ha olika färg eller använda sig av understrykning, detta skulle

dock behöva undersökas i en ny användarstudie. Eventuellt skulle man även kanske kunna ha en speciell knapp som bara syns för de klickbara förslagen.

Huvudfönstret/Granskningsfönstret

Användare 3 och 4 vill att hela texten ska finnas i ”Skriv din text här”-rutan under hela granskningen.

Eventuell ändring: Om funktionaliteten med pennan ska finnas kvar kan inte hela texten ligga i rutan hela tiden, man kan dock tänka sig en lösning med en knapp som återställer hela texten till ”Skriv din text här”-rutan när användaren vill det. (Nu kan man lite omständligt trycka på *Texten i nytt fönster* kopiera texten från det fönstret och klistra in den i ”Skriv din text här”-rutan om man vill ha hela sin text där.) Ett annat alternativ kan vara att ha två textrutor, en som texten ligger i hela tiden och en som man kan redigera meningar i. Vad som är bäst ur användarperspektiv är dock svårt att veta, t.ex. kan fler vyer leda till ytterligare förvirring kring vad som kan göras var. Ytterligare ett alternativ är att man får upp ett mindre nytt fönster att redigera meningen i.

Olika typer av fel

Användare 1 och 2 förstår inte alla de olika typerna och märker eller förstår inte heller att det går att klicka på dem.

Eventuell ändring: En tänkbar åtgärd kan vara att försöka göra det tydligare att det går att klicka på dem, hur är dock oklart. (Knappar signalerar klickbarhet, men även händelse.)

Användare 1 tycker att det vore bra om det fanns en tabell som visade hur vanliga de olika felen är i texten.

Eventuell ändring: Informationen om hur många fel av varje kategori som hittats i texten finns i XML-filen så med den informationen kan man skapa en tabell.

Användare 4 vill ha olika färg på de olika typerna av fel.

Eventuell ändring: Detta diskuterades på studiecirkeln och vi kom där fram till att den röda färgen är bra på att signalera att här är något du borde titta på och att om man använder flera olika färger kan man förlora det.

Pennan

Ingen av användare reagerade på pennan alls. När de uppmanades att testa den förstod de inte heller vad den gjorde för nytta. Användare 3 och 4 såg inte att meningen hamnade i den översta rutan eftersom de var längre ner i sidan. Användare 1 märkte inte riktigt vad som hände, men upptäckte sen meningen i rutan men trodde att det kanske var så att man kontrollerade denna mening en gång till på detta sätt. Användare 2 förstod inte alls.

Eventuell ändring: Att ingen av användarna reagerade på pennan kan bero på att pennan visserligen är en vedertagen ikon på webben för redigering av text, men den förekommer främst i administrationsgränssnitt och användarna i denna studie har kanske inte kommit i kontakt med administrationsgränssnitt i någon större omfattning. Slutsatsen blir att pennan är vad man kan kalla överkurs, användarna i den här studien hade inte användning av pennan och noterade den alltså inte. Om en användare känner behovet av att ändra i en mening som det inte markerats några fel i kanske han/hon läser om det i hjälpen och finner pennan användbar. Användare 3 och 4 såg inte vad som hände eftersom de var för långt ner på sidan när de tryckte på pennan, detta kan lösas med hjälp av JavaScript genom att man fokuserar på ”Skriv din text här”-rutan när användaren trycker på pennan. En annan lösning kan, som tidigare nämnts, vara att man får upp meningen i ett nytt fönster där man kan redigera den.

Texten i nytt fönster

Ingen av användarna insåg vad det innebar att trycka på knappen innan de testade.

Eventuell ändring: Knappen kan kanske förtydligas genom att man ändrar texten på den, användare 4 föreslog t.ex. ”Korrigerade texten i nytt fönster”.

För grundmaterialet se tabell 6.1-6.3.

6.3 Ändringar efter användarstudien

Det är svårt att veta vilka ändringar som kommer att göra att gränssnittet blir mer förståeligt för användarna i målgruppen och vilka som kanske gör det mer krångligt, därför har jag inte gjort några övergripande ändringar efter användarstudien. Att det gjorts restriktivt med ändringar beror även på att det inte fanns tid till mer. I kapitel 8 nedan listas det framtida arbete som skulle kunna göra gränssnittet ännu bättre.

Det som ändrades efter användarstudien var framför allt att förklarande texter lades till och ändrades på ställen där användarstudien visade på det inte var tillräckligt tydligt hur tillvägagångssättet var tänkt. En text som lades till var den ovanför ändringsrutan i informationsfönstret, se bild 6.1, där står nu ”Tryck på ett ord eller rätta här:”. Även texten på knappen *Texten i nytt fönster* ändrades till *Korrigerade texten i nytt fönster* i enlighet med vad användare 4 tyckte skulle vara tydligare.

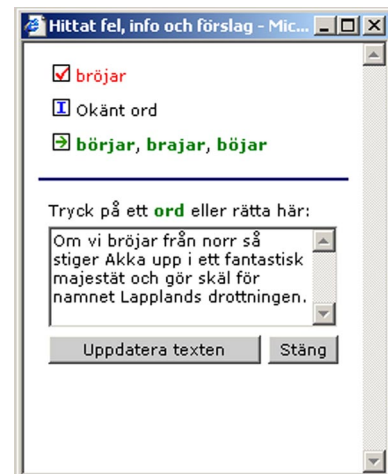


Bild 6.1 Informationsfönstret efter användarstudien

En annan sak som gjordes i avsikt att det ska bli tydligare för användaren hur pennan fungerar är att ett JavaScript lades till. Dess funktion är att om användaren trycker på pennan långt ner på sidan så kommer sidan att rulla upp så att ”Skriv din text här”-rutan med meningen i hamnar i fokus. För att vara säker på att detta upplägg är bättre behövs det göras ytterligare studier med användare.

Mer information om gränssnittets olika funktioner och om hur det är tänkt att man ska arbeta med gränssnittet finns att läsa i användarhandledningen, se appendix E.

Hur gränssnittet är uppbyggt, vilka filer som finns och vad de olika metoderna i Javaklassen Granska gör, kan läsas i den tekniska dokumentationen, se appendix F.

7 Diskussion och slutsats

TvärGranska får övervägande bra kritik i användarstudien, det finns ett behov att den här sortens språkhjälp och användarna uppskattar att den är gratis och lätt tillgänglig genom ett webbgränssnitt. Självklart behövs det dock utvecklas vidare, fler användarstudier kan göras på gränssnittet för att kunna förbättra det vidare i enlighet med målgruppens behov och Granska behöver utvecklas så att fler fel kan detekteras.

Att designa och att presentera information så att alla i målgruppen förstår är nästintill omöjligt, det som är självklart för en användare kan vara otydligt för nästa, det man får göra är att söka efter en design som de flesta förstår och tycker är bra. En bra väg att gå i det arbetet är att försöka designa i enlighet med vad andra inom samma område gjort tidigare, eftersom användarna då känner igen funktionaliteten. Ett exempel i språkgranskningsprogram är att rödmarkera ord som kan vara felaktiga, rött för att det används av andra språkgranskningsprogram, att t.ex. markera ord med lila färg skulle vara mindre intuitivt för användaren.

Att tidsuppskatta hur lång tid ett programmeringsprojekt tar är svårt, felet man oftast gör är att man är för optimistisk och räknar med att allt man skriver ska fungera som man tänkt det med en gång. Så är tyvärr oftast inte fallet utan väldigt ofta fungerar det inte alls som man trodde och man får göra justeringar i det man skrivit eller i värsta fall tänka om och skriva nytt. Man kan också som utvecklare hamna i situationer där man inte själv kan påverka framskridningen i projektet för att en server eller dylikt som man inte själv har ansvar för inte fungerar.

När man utvecklar för webben finns det många olika tekniker som man kan använda sig av, men trots det har webbläsarna sina begränsningar, det går inte att utnyttja högerklick för egna funktioner, ett appletprogram behöver ett JVM som kan tolka den version av Java som det är utvecklat i etc. Olika webbläsare på olika operativsystem tolkar HTML på olika sätt, vilket gör att sidorna ser olika ut. Alla dessa begränsningar måste man ta hänsyn till i valet och utvecklingen av webbgränssnitt.

En annan sak man ska vara försiktig med är att låta personen som står för de tekniska lösningarna i ett gränssnitt att även stå för den grafiska utformningen. Den som är fokuserad på de tekniska lösningarna kommer antagligen inte att lägga ner lika mycket tid på designen eftersom det ur hans/hennes synvinkel är det tekniska lösningarna som är det viktigaste. Ett annat problem kan vara att veta vad som uppfattas som svårt av användaren när man själv är väldigt insatt i hur allt fungerar eftersom man konstruerat det. Eftersom gränssnittsdesignens betydelse inte ska underskattas bör man alltså anlita en separat person för att göra den.

För att undersöka om man utvecklat ett gränssnitt som är förståeligt för användarna gör man en eller flera användarstudier. Användarstudien i den här rapporten var nog förberedd men ändå så uppstod problem under genomförandet. Meningen var nämligen att försöksledaren inte skulle hjälpa användaren i någon högre grad men detta visade sig svårt när användaren hade lite datorvana och hjälpen i gränssnittet dessutom var på ett språk skilt från användarens modersmål. Slutsatser man kan dra av detta är att man kan planera en användarstudie in i minsta detalj men måste ändå vara beredd på att allt kanske inte går helt enligt planerna och att det är svårare för användaren att ta till sig nya saker när förklaringarna inte är på modersmålet.

Att göra användarstudier ger mycket, man får se gränssnittet användas av den tilltänkta målgruppen och det kommer fram saker som man inte hade kunnat komma på själv. För den som konstruerat alla funktioner är det självklart hur de fungerar och hur de ska användas, i användarstudien kom det fram att det inte var lika självklart för användarna (t.ex. vad gäller ”pennan”). Det är bland annat detta som gör användarstudier till en självklar del i utvecklingen av gränssnitt.

8 Framtida arbete

När tiden för det här examensarbetet tog slut fanns fortfarande några tänkbara funktioner och önskemål kvar, dessa har jag samlat här under framtida arbete i hopp om att någon någon gång kanske ska implementera dem i gränssnittet.

- En tabell som visar hur många fel ur varje felkategori (stavfel, särskrivningsfel, kongruensfel, verbfel och övriga fel) som användaren har i sin text.
- Med en ändring i XML:en skulle man kunna klicka på rätt ord varje gång (inte bara när det är stavfel och särskrivningsfel) för att få en ändring i informationsfönstret.
- Med en utbyggnad av informationen i XML:en skulle man också kunna få en hänvisning till Svenska skrivregler för ytterligare information om ett speciellt fel.
- Något som arbetas på nu och som förhoppningsvis kommer att ändras i XML:en är att förenkla termerna i informationen om ett fel så att de ska vara tydligare för användare med svenska som andraspråk. (Till exempel har termen ”Okänt ord” ogillats.)
- Något som vidare skulle kunna undersökas är också hur språket i hjälpen till gränssnittet skulle kunna förenklas för att nå högre förståelse hos användare med annat modersmål än svenska.
- Det vore också önskvärt att Granskaservern kunde ta emot och tolka tecken som < > & ” ’ och radbrytning och även kunna hålla reda på mellanrummen som var i texten med tecken som . , ; : ! ? () _ och /. Detta skulle nämligen leda till att formateringen av texten kan behållas på ett enkelt sätt och de mellanrum och stycken som användaren har i texten bevaras som de var.

Som avslutning måste ändå påpekas att användarstudien faktiskt visar att användarna tycker att TvärGranska är bra och min förhoppning nu är att någon utvecklar den vidare så att TvärGranska kan bli ännu bättre i framtiden.

Referenser

Chapelle C. 1999. *Computer Applications in Second Language Acquisition – Foundations for Teaching, Testing and Research*. Cambridge University Press. Cambridge. ISBN 0521626463

Domeij, R., Knutsson, O., Larsson, S., Rex, Å. & Severinsson-Eklundh, K. 1998. *Granskapprojektet 1996–1997*. IPLabrapport, IPLab, Nada, KTH. Stockholm.

Jurafsky D. & Martin J. 2000. *Speech and Language Processing – an Introduction to Natural Language Processing, Computational Linguistics and Speech Recognition*. Prentice-Hall, Inc. New Jersey. ISBN 0130950696

Knutsson, O. 2001. *Automatisk språkgranskning av svensk text*. Licentiatavhandling TRITA-NA-0105, Nada, KTH. Stockholm.

Larsson, S. 1998. *Interaktivitet och användbarhet vid datorstödd språkgranskning och redigering i en integrerad skrivmiljö*. Exjobbssrapport TRITA-NA-E9833, Nada, KTH. Stockholm.

McGee & Ericsson. 2002. The politics of the program: Microsoft Word as the invisible grammarian. *Computers and Composition* 19 (2002) s. 453-470

Paulsson, J. & Widengren, M. 2002. *Datoriserad språkgranskning för lärande? – En studie av verktyget Granska och andraspråksinlärning*. Projektarbete Kognitionsvetenskap, DSV/Nada, SU/KTH/KI. Stockholm.

Preece J., Rogers Y., Sharp H., Benyon D., Holland S. & Carey T. 1994. *Human-Computer Interaction*. Addison-Wesley, Harlow, England. ISBN 0201627698

Svenska skrivregler. 1992. Svenska språknämnden. Almqvist & Wiksell.

Elektroniska referenser

Bagley D. 2001. The Great Computer Language Shootout, <http://www.bagley.org/~doug/shootout/>, 2003-08-27

Developer.com
<http://www.developer.com>, 2003-08-27

Java Architecture for XML Binding (JAXB), [java.sun.com](http://java.sun.com/xml/jaxb/),
<http://java.sun.com/xml/jaxb/>, 2003-08-27

Lingsoft – Grammatifix
<http://www.lingsoft.fi/grammatifix/>, 2003-08-27

The MSDN Library – JScript, VBScript
<http://msdn.microsoft.com/scripting>, 2003-08-27

Netscape Devedge – JavaScript
<http://devedge.netscape.com/library/manuals/2000/javascript/1.5/reference>, 2003-08-27

Rich Text Format (RTF) Version 1.5 Specification
http://www.biblioscape.com/rtf15_spec.htm, 2003-08-27

The Source for Java Technology
<http://java.sun.com>, 2003-08-27

The Source for Java Technology, API Documentation
<http://java.sun.com/api/>, 2003-08-27

The Source for Java Technology, Applets
<http://java.sun.com/applets/>, 2003-08-27

Svenska datatermgruppen
<http://www.nada.kth.se/dataterm>, 2003-08-27

Webopedia: Online Dictionary for Computer and Internet Terms
<http://www.webopedia.com>, 2003-08-27

World Wide Web Consortium – DTD
<http://www.w3.org/TR/REC-xml#dt-doctype>, 2003-08-27

World Wide Web Consortium – Extensible Markup Language (XML) 1.0 (Second Edition)
<http://www.w3.org/TR/REC-xml>, 2003-08-27

World Wide Web Consortium – XML Schema
<http://www.w3.org/XML/Schema>, 2003-08-27

Andra länkar:

Forskningsprojekt: CrossCheck – svensk grammatikkontroll för andraspråksskribenter
<http://www.nada.kth.se/theory/projects/xcheck/>, 2003-08-27

Forskningsprojekt: Grammatikgranskning
<http://www.nada.kth.se/theory/projects/granska/>, 2003-08-27

Forskningsprojekt: Språkliga datorstöd och andraspråksinlärning
<http://www.nada.kth.se/~knutsson/call.html>, 2003-08-27

TvärGranska
<http://skrutten.nada.kth.se:8080/granska/>, 2003-08-27

WebbGranska
<http://skrutten.nada.kth.se/scrut/svesve>, 2003-08-27

Textredigeringsprogram på webben:

edit-on® Pro by RealObjects
<http://realobjects.de/eopro/index.htm>, 2003-08-27

jEdit – Open Source programmer's text editor
<http://www.jedit.org/>, 2003-08-27

Pote – The Online Text Editor

<http://www.look-designs.com/pote/>, 2003-08-27

Simredo 3.4 – Java Unicode Editor

<http://www4.vc-net.ne.jp/~klivo/sim/simeng.htm>, 2003-08-27

Appendix

A Förkortningsordlista

API, Application Program Interface – de funktioner, protokoll och verktyg som behövs för att kunna utveckla. Ett bra API gör det enklare för programmeraren att utveckla genom att alla delar i språket är beskrivna. Ett bra exempel finns på <http://java.sun.com/api/>

appletprogram – ett program som kan köras i webbläsaren och som exekveras på klientdatorn

ASCII, American Standard Code for Information Interchange – standard som representerar varje bokstav med ett nummer och där varje bokstav är en byte stor.

ASP, Active Server Pages – ett serverexekverat språk utvecklat av Microsoft

ASP.NET – det senaste serverexekverande tekniken utvecklat av Microsoft

C – ett programmeringsspråk

C++ – ett objektorienterat programmeringsspråk

DOM, Document Object Model – en specifikation för hur objekt i en webbsida ska visas

DTD, Document Type Definition – en DTD beskriver hur en XML- eller HTML-fil är uppbyggd, t.ex. vilka taggar som finns och var de finns

gif, graphics interchange format – ett bildformat mycket använt på webben

HTML, Hyper Text Markup Language – ett beskrivningsspråk till för att beskriva webbsidor

Java – ett objektorienterat programmeringsspråk

JavaScript – ett skriptspråk utvecklat av Netscape (inget egentligt samband med Java)

Java-XML Data Binding – en teknik i Java för att läsa och tolka XML

JAXB, Java Architecture for XML Binding - ett Java-XML Data Binding-API från Sun

JScript – ett skriptspråk utvecklat av Microsoft (närmast identiskt med JavaScript)

JSP, Java Server Pages – ett serverexekverat språk utvecklat av Sun

JVM, Java Virtual Machine – finns till nästan alla operativsystem och används av webbläsare för att kunna köra appletprogram. Eftersom Javaprogram kompileras till operativsystemoberoende bytekod behövs JVM:en för att köra programmen i ett specifikt operativsystem.

PHP, PHP: Hypertext Preprocessor – ett serverexekverat skriptspråk utvecklat av The PHP Group

RTF, Rich Text Format – ett dokumentbeskrivningsspråk

SAX, Simple API for XML – en teknik i Java för att läsa och tolka XML

VBScript – ett skriptspråk utvecklat av Microsoft

Visual Basic – ett programmeringsspråk utvecklat av Microsoft

XML, eXtensive Markup Language – ett sätt att märka upp information

XML Schema – en standard för att beskriva hur en XML- eller HTML-fil är uppbyggd, t.ex. vilka taggar som finns och var de finns

För längre förklaringar till ovanstående förkortningar rekommenderas
<http://www.webopedia.com>

B Enkät

B1 Enkät om språkgranskning på webben



Enkät om språkgranskning på webben

Interaktions- och presentationslaboratoriet

Jag heter Ylva Stenervall och gör exjobb på Nada, KTH inom forskningsprojektet CrossCheck – svensk grammatikkontroll för andraspråksskribenter (<http://www.nada.kth.se/theory/projects/xcheck/>). Min uppgift är att göra ett nytt interaktivt webbgränssnitt till Granska. Granska är till för att man ska kunna granska sin text och få hjälp med att korrigera stavfel och grammatikfel. Den här enkäten är till för att ta reda på vad de tänkta användarna har för vanor, kunskaper och åsikter om hur man enklast granskar sin text på webben. Jag skulle därför vara väldigt tacksam om ni tog er tid att besvara denna enkät som kommer att vara till hjälp för mig i utvecklingsarbetet av Granska.
Tack på förhand!

1. a) Kön:

- Kvinna
 Man

b) Ålder:

- 0-18 år
 19-29 år
 30-39 år
 40-49 år
 50-59 år
 60 år eller äldre

c) Modersmål:.....

d) Andra språk du kan:.....

e) Hur länge har du varit i Sverige?.....

f) Utbildning:

- Gymnasial utbildning, om Ja: Teknisk eller Humanistisk
 Högskoleutbildning eller motsvarande, om Ja: Teknisk eller Humanistisk
 Högre utbildning, nämligen.....

2. a) Vilka program (ordbehandlare) använder du dig av för att skriva text?

- Microsoft Word
 E-postprogram
 LaTeX
 Förstår inte frågan
 Annat, nämligen.....

b) Brukar du använda dig av stavnings- och grammatikkontrollen i ditt ordbehandlingsprogram?

- Ja
- Nej
- Bara stavningskontroll
- Förstår inte frågan

Kommentar:.....

c) När använder du stavnings- och grammatikkontrollen?

- När texten är färdig
- När ett stycke är färdigt
- När en mening är färdig
- Förstår inte frågan
- Annat, nämligen.....

d) Kan du tänka dig att göra stavnings- och grammatikkontrollen i ett annat program än det du skrivit din text i?

- Ja
- Nej
- Tveksamt
- Förstår inte frågan

Kommentar:.....

e) Vilka texter är intressanta för dig att få stavnings- och grammatikhjälp med?

- Uppsatser/Rapporter
- E-post
- Förstår inte frågan
- Andra, nämligen.....

3. a) Känner du till hur man kopierar, klipper och klistrar på en dator?

- Ja
- Nej
- Förstår inte frågan

Kommentar:.....

b) Använder du dig av kortkommandona för att kopiera, klippa och klistra? (t.ex. Ctrl+C)

- Ja
- Nej
- Förstår inte frågan

Kommentar:.....

c) Skulle du kunna tänka dig att klistra in och klippa ut din text i/ur programmet som kontrollerar din text för stavfel och grammatikfel?

- Ja
- Nej
- Förstår inte frågan

Kommentar:.....

4. Vet du hur man sparar i olika format i en ordbehandlare? (t.ex. *.doc*, *.html*, *.rtf* eller *.txt*)

- Ja
- Nej
- Förstår inte frågan

Kommentar:.....

5. a) Vilket/Vilka operativsystem använder du?

- Windows, om Ja: 2000/NT/XP 95/98/ME Vet ej
- MacOS, om Ja: MacOS X MacOS 9 eller äldre Vet ej
- Vet ej
- Förstår inte frågan
- Annat, nämligen.....

b) Vilken/Vilka webbläsare använder du?

- Explorer, om Ja: 5 eller nyare 4 eller äldre Vet ej
- Netscape, om Ja: 6 eller nyare 4 eller äldre Vet ej
- Vet ej
- Förstår inte frågan
- Annan, nämligen.....

c) Vet du hur man sparar en sida med text i en webbläsare?

- Ja
- Nej
- Förstår inte frågan

Kommentar:.....

6. a) Kan du tänka dig att sitta och skriva på en dator uppkopplad till Internet för att kunna köra programmet som kontrollerar stavfel och grammatikfel?

- Ja, i skolan.
- Ja, hemma.
- Nej
- Förstår inte frågan

Kommentar:.....

b) Vilket av nedanstående sätt för att få in din text i programmet som kontrollerar stavfel och grammatikfel tror du är enklast för dig?

- Klippa ut och klistra in texten
- Spara texten på formatet *.txt* och ange filnamnet i webbläsaren
- En ordbehandlare med programmet som kontrollerar stavfel och grammatikfel inbyggt och som man måste vara uppkopplad till Internet för att använda
- Skriva i Word och sedan genom att trycka på en knapp komma till programmet som kontrollerar stavfel och grammatikfel som man måste vara uppkopplad till Internet för att använda
- Vet ej
- Förstår inte frågan
- Annat, nämligen.....

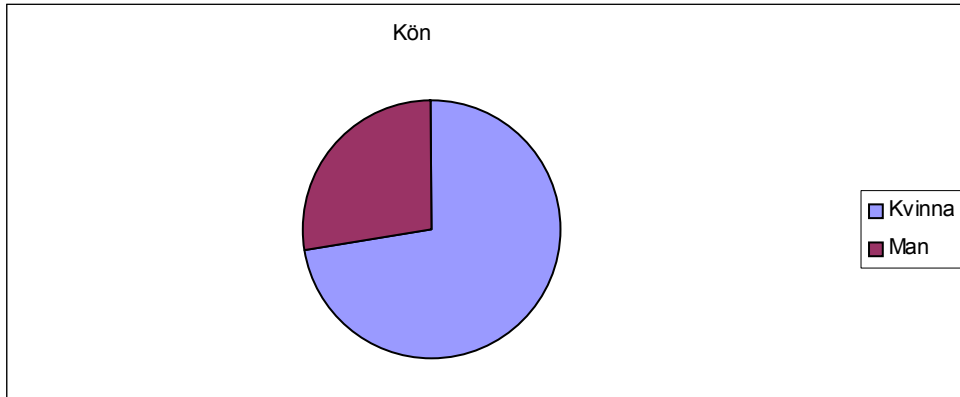
Tack för att du tog dig tid att svara på denna enkät!

Testa gärna de äldre versionerna av Granska på <http://skrutten.nada.kth.se>

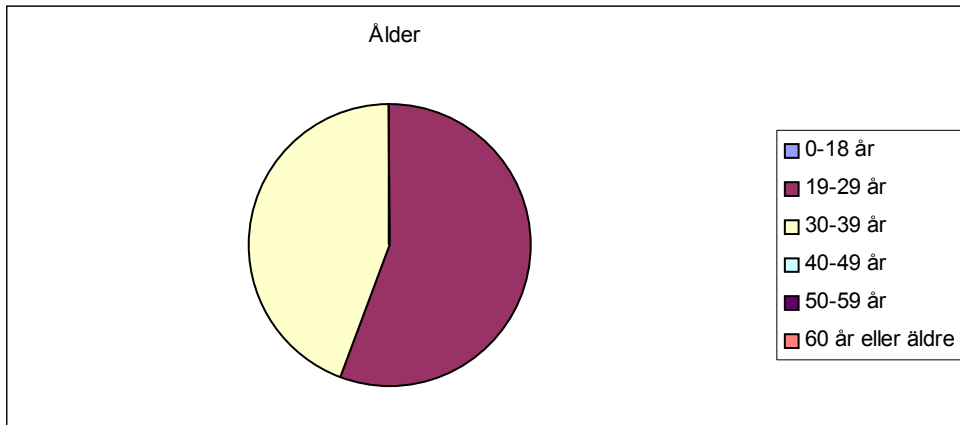
B2 Svaren på enkäten om språkgranskning på webben

Enkäten delades ut i november 2003 och 18 andraspråksinlärare svarade på den.

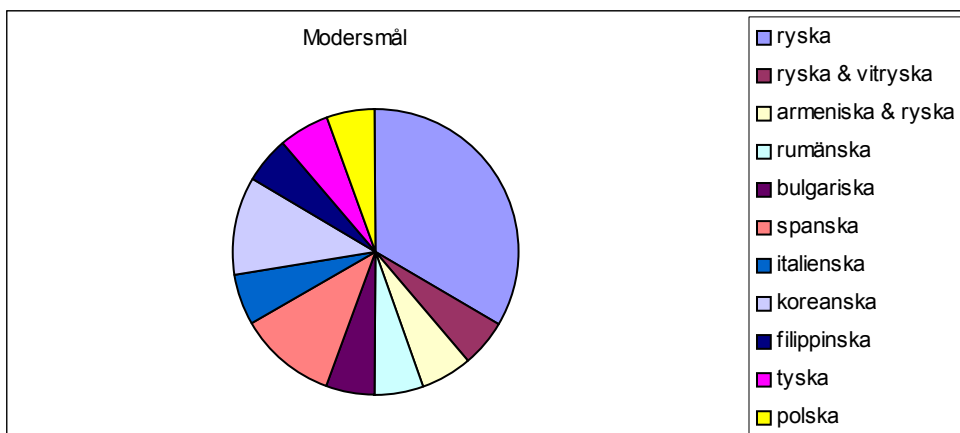
1. a) Kön:



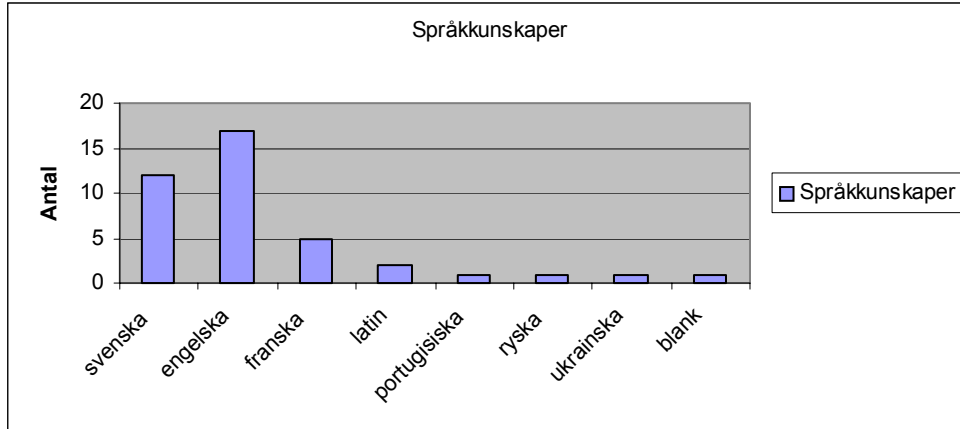
b) Ålder:



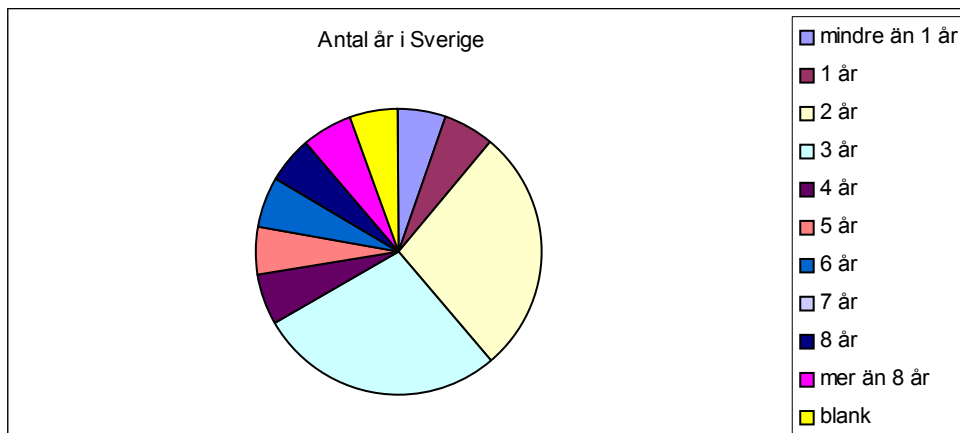
c) Modersmål:



d) Andra språk du kan:



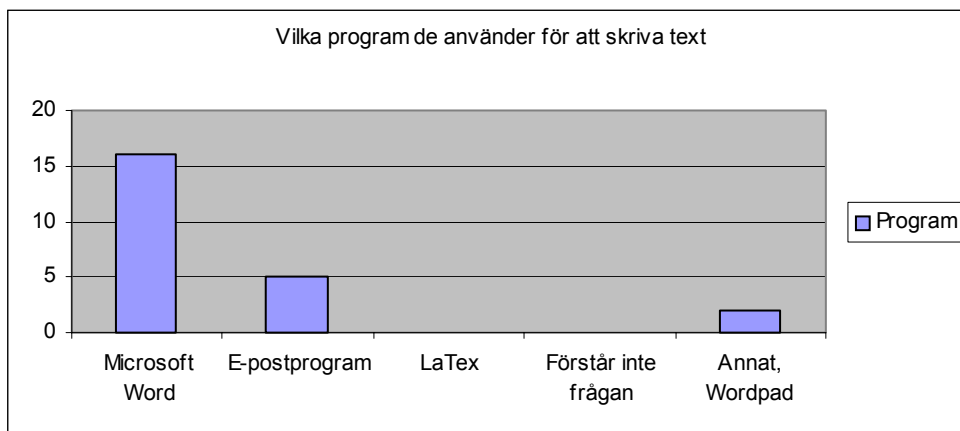
e) Hur länge har du varit i Sverige?



f) Utbildning:

Gymnasial:	5 st	varav Teknisk: 2 st	Humanistisk: 4 st
Högskole eller motsv.:	14 st	varav Teknisk: 2 st	Humanistisk: 13 st
Högre utb.:	1 st, nämligen Masters in Development Studies, Masters in Business Administration		

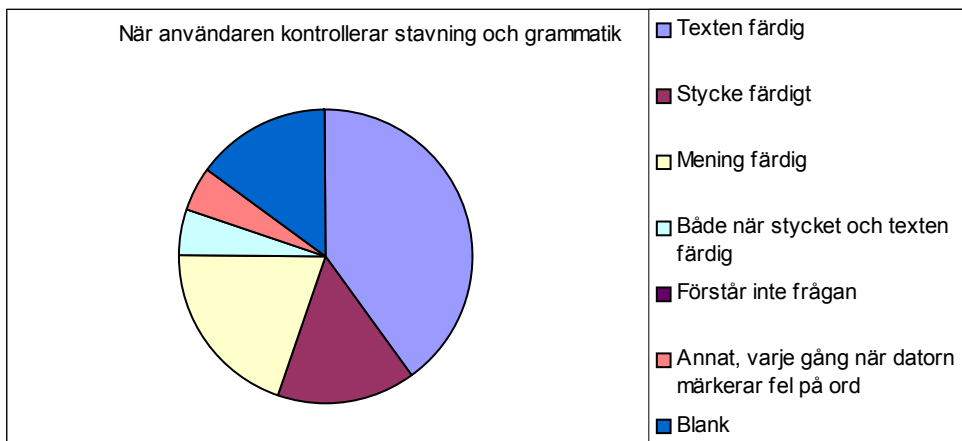
2. a) Vilka program (ordbehandlare) använder du dig av för att skriva text?



b) Brukar du använda dig av stavnings- och grammatikkontrollen i ditt ordbehandlingsprogram?



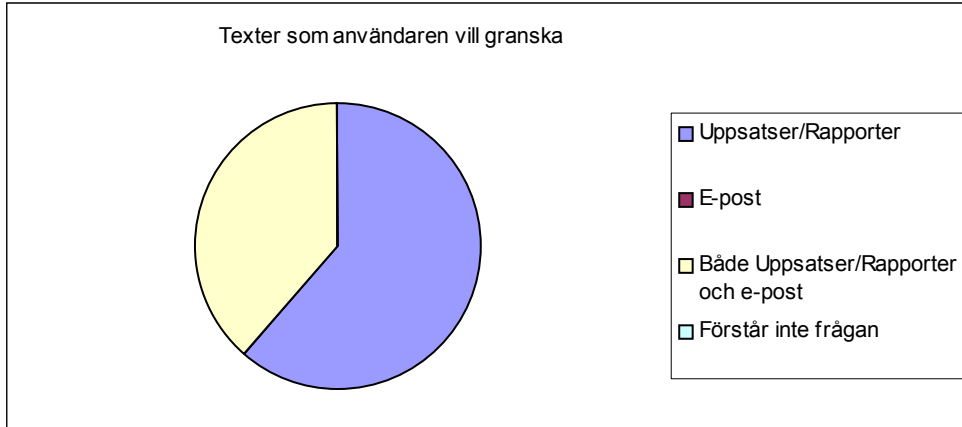
c) När använder du stavnings- och grammatikkontrollen?



d) Kan du tänka dig att göra stavnings- och grammatikkontrollen i ett annat program än det du skrivit din text i?



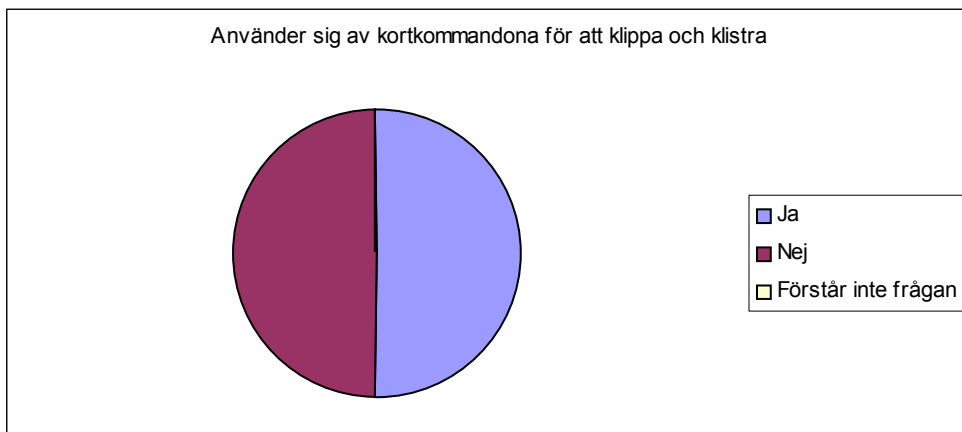
e) Vilka texter är intressanta för dig att få stavnings- och grammatikhjälp med?



3. a) Känner du till hur man kopierar, klipper och klistrar på en dator?



b) Använder du dig av kortkommandona för att kopiera, klippa och klistra? (t.ex. Ctrl+C)



c) Skulle du kunna tänka dig att klistra in och klippa ut din text i/ur programmet som kontrollerar din text för stavfel och grammatikfel?



4. Vet du hur man sparar i olika format i en ordbehandlare? (t.ex. *.doc*, *.html*, *.rtf* eller *.txt*)



5. a) Vilket/Vilka operativsystem använder du?

Windows: 18 st varav 2000/NT/XP: 9 st 95/98/ME: 9 st Vet ej: 1 st Blank: 1 st
MacOS: 0 st
Vet ej: 0 st
Annat: 1 st, nämligen Matlab, Femlab, Java

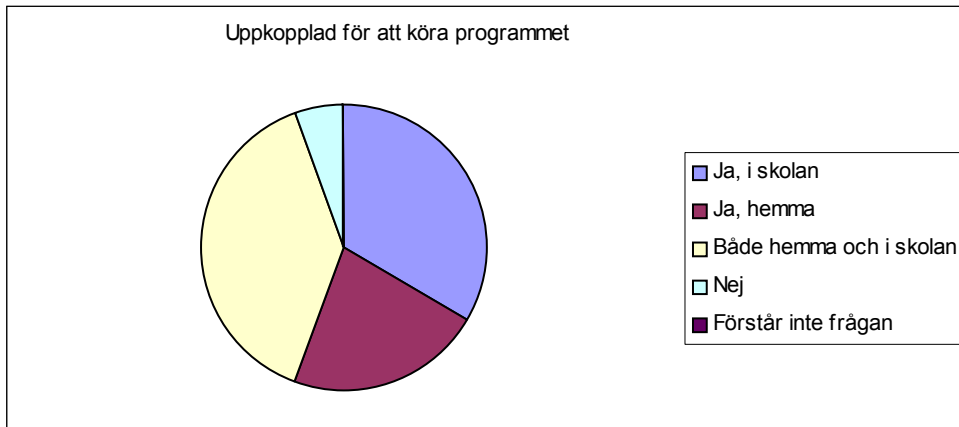
b) Vilken/Vilka webbläsare använder du?

Internet Explorer: 12 st varav 5 eller nyare: 4 st 4 eller äldre: 2 st Vet ej: 4 st Blank: 2 st
Netscape: 5 st varav 6 eller nyare: 2 st 4 eller äldre: 2 st Vet ej: 1 st
Vet ej: 3 st
Förstår inte frågan: 1 st
Blank: 1 st

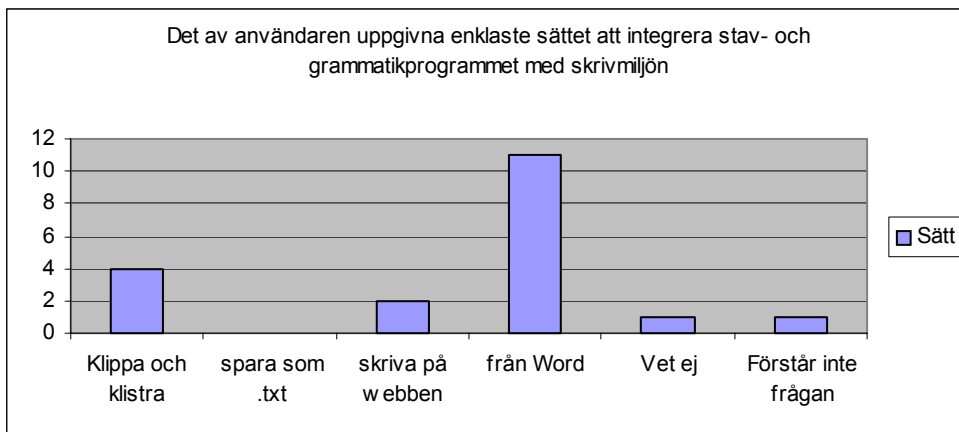
c) Vet du hur man sparar en sida med text i en webbläsare?



6. a) Kan du tänka dig att sitta och skriva på en dator uppkopplad till Internet för att kunna köra programmet som kontrollerar stavfel och grammatikfel?



b) Vilket av nedanstående sätt för att få in din text i programmet som kontrollerar stavfel och grammatikfel tror du är enklast för dig?



C XML

C1 XML-exempel

```

<?xml version="1.0" encoding="ISO-8859-1" standalone="yes" ?>
<Root>
  <s ref="0">
    <words>5</words>
    <text>Mellan Sarek och Padjelanta ./text>
    <paragraph />
    <contents>
      <w no="0" tag="133" lemma="mellan">Mellan</w>
      <w no="1" ref="7" tag="40" lemma="sarek">Sarek</w>
      <w no="2" ref="13" tag="61" lemma="och">och</w>
      <w no="3" ref="17" tag="40" lemma="padjelanta">Padjelanta</w>
      <w no="4" ref="27" tag="109" lemma=".">.</w>
    </contents>
  </s>
  <s ref="31"><words>15</words>
    <text>Padjelanta är det höga landet som är osannolikt fantastiskt vacker och mjukt i formerna ./text>
    <contents>
      <w no="0" ref="0" tag="40" lemma="padjelanta">Padjelanta</w>
      <w no="1" ref="11" tag="134" lemma="vara">är</w>
      <w no="2" ref="14" tag="71" lemma="den">det</w>
      <w no="3" ref="18" tag="51" lemma="hög">höga</w>
      <w no="4" ref="23" tag="0" lemma="land">landet</w>
      <w no="5" ref="30" tag="27" lemma="som">som</w>
      <w no="6" ref="34" tag="134" lemma="vara">är</w>
      <w no="7" ref="37" tag="147" lemma="osannolikt">osannolikt</w>
      <w no="8" ref="48" tag="147" lemma="fantastiskt">fantastiskt</w>
      <w no="9" ref="60" tag="80" lemma="vacker">vacker</w>
      <w no="10" ref="67" tag="61" lemma="och">och</w>
      <w no="11" ref="71" tag="147" lemma="mjukt">mjukt</w>
      <w no="12" ref="77" tag="133" lemma="i">i</w>
      <w no="13" ref="79" tag="12" lemma="form">formerna</w>
      <w no="14" ref="87" tag="109" lemma=".">.</w>
    </contents>
  </s>
  ...
  <s ref="11288"><words>14</words>
    <text>Att hitta lä är underbart och äntligen kan man slappna av en smula ./text>
    <paragraph />
    <contents>
      <w no="0" ref="0" tag="138" lemma="att">Att</w>
      <w no="1" ref="4" tag="79" lemma="hitta">hitta</w>
      <w no="2" ref="10" tag="11" lemma="lä">lä</w>
      <w no="3" ref="13" tag="134" lemma="vara">är</w>
      <w no="4" ref="16" tag="67" lemma="underbar">underbart</w>
      <w no="5" ref="26" tag="61" lemma="och">och</w>
      <w no="6" ref="30" tag="110" lemma="äntligen">äntligen</w>
      <w no="7" ref="39" tag="63" lemma="kunna">kan</w>
      <w no="8" ref="43" tag="69" lemma="man">man</w>
      <w no="9" ref="47" tag="79" lemma="slappna">slappna</w>
      <w no="10" ref="55" tag="133" lemma="av">av</w>
      <w no="11" ref="58" tag="56" lemma="en">en</w>
      <w no="12" ref="61" tag="31" lemma="smula">smula</w>
      <w no="13" ref="66" tag="109" lemma=".">.</w>
    </contents>
  </s>
  <scrutinizer>
    <s ref="0">
      <gramerrors>
        <gramerror>
          <marked />
          <rule>inget_verb@verb</rule>
          <info>Verb verkar saknas i satsen. Om detta är en rubrik bör den inte avslutas med punkt.</info>
          <marked_section>
            <mark begin="2" end="6" />
          </marked_section>
        </gramerror>
      </gramerrors>
    </s>
  </scrutinizer>
  <s ref="31">

```

```

<gramerrors>
<gramerror>
  <marked>
    <emph type="red">det höga landet ... är ... vacker</emph>
  </marked>
  <rule>pred1@pred</rule>
  <info>
    Om
    <emph type="italic">vacker</emph>
    syftar på
    <emph type="italic">det höga landet</emph>
    är det kongruensfel
  </info>
  <suggestions>
    <sugg>
      det höga landet som är osannolikt fantastiskt
      <emph type="green">vackert</emph>
    </sugg>
  </suggestions>
  <marked_section>
    <mark begin="4" end="6" />
    <mark begin="8" end="8" />
    <mark begin="11" end="11" />
  </marked_section>
</gramerror>
</gramerrors>
</s>
...
</scrutinizer>
<tags>
  <tag no="0" name="nn.neu.sin.def.nom" />
  ...
  <tag no="148" name="pm.sms" />
</tags>
<rules>
  <categories>
    ...
    <category>
      <name>stavning</name>
      <info>stavning</info>
      <linkurl>http://www.nada.kth.se/~viggo/stava/manual.html</linkurl>
      <linktext>Stava</linktext>
      <count>33</count>
    </category>
    ...
  </categories>
  <rules_used>
    ...
    <rule>
      <name>stav1@stavning</name>
      <category>stavning</category>
      <linkurl>http://www.nada.kth.se/~viggo/stava/manual.html</linkurl>
      <linktext>Stava</linktext>
    </rule>
    ...
  </rules_used>
</rules>

</Root>

```

C2 XML Schema

```

<?xml version="1.0" encoding="ISO-8859-1" standalone="yes" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

  <xs:element name="Root">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="s" minOccurs="1" maxOccurs="unbounded" />
        <xs:element ref="scrutinizer" minOccurs="1" maxOccurs="1" />
        <xs:element ref="tags" minOccurs="1" maxOccurs="1" />
        <xs:element ref="rules" minOccurs="1" maxOccurs="1" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="s">
    <xs:complexType>
      <xs:choice>
        <xs:sequence>
          <xs:element ref="words" minOccurs="1" maxOccurs="1" />
          <xs:element ref="text" minOccurs="1" maxOccurs="1" />
          <xs:element ref="paragraph" minOccurs="0" maxOccurs="1" />
          <xs:element ref="contents" minOccurs="1" maxOccurs="1" />
        </xs:sequence>
        <xs:element ref="gramerrors" minOccurs="1" maxOccurs="1" />
      </xs:choice>
      <xs:attribute name="ref" type="xs:NMTOKEN" use="required" />
    </xs:complexType>
  </xs:element>

  <xs:element name="words" type="xs:string" />
  <xs:element name="text" type="xs:string" />
  <xs:element name="paragraph" type="xs:string" />

  <xs:element name="contents">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="w" minOccurs="1" maxOccurs="unbounded" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="w">
    <xs:complexType mixed="true">
      <xs:attribute name="no" type="xs:NMTOKEN" use="required" />
      <xs:attribute name="ref" type="xs:NMTOKEN" use="optional" />
      <xs:attribute name="tag" type="xs:NMTOKEN" use="required" />
      <xs:attribute name="lemma" type="xs:string" use="required" />
    </xs:complexType>
  </xs:element>

  <xs:element name="scrutinizer">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="s" minOccurs="0" maxOccurs="unbounded" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="gramerrors">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="gramerror" minOccurs="1" maxOccurs="unbounded" />
      </xs:sequence>
    </xs:complexType>
  </xs:element>

```

```

<xs:element name="gramerror">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="marked" minOccurs="1" maxOccurs="1" />
      <xs:element ref="rule" minOccurs="1" maxOccurs="1" />
      <xs:element ref="info" minOccurs="1" maxOccurs="1" />
      <xs:element ref="suggestions" minOccurs="0" maxOccurs="1" />
      <xs:element ref="marked_section" minOccurs="1" maxOccurs="1" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="marked">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="emph" minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="emph">
  <xs:complexType mixed="true">
    <xs:attribute name="type" type="xs:string" use="required" />
  </xs:complexType>
</xs:element>

<xs:element name="rule">
  <xs:complexType mixed="true">
    <xs:sequence>
      <xs:element ref="name" minOccurs="0" maxOccurs="1" />
      <xs:element ref="category" minOccurs="0" maxOccurs="1" />
      <xs:element ref="linkurl" minOccurs="0" maxOccurs="1" />
      <xs:element ref="linktext" minOccurs="0" maxOccurs="1" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="info">
  <xs:complexType mixed="true">
    <xs:sequence>
      <xs:element ref="emph" minOccurs="0" maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="suggestions">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="sugg" minOccurs="1" maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="sugg">
  <xs:complexType mixed="true">
    <xs:sequence>
      <xs:element ref="emph" minOccurs="1" maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="marked_section">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="mark" minOccurs="1" maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="mark">
  <xs:complexType>
    <xs:attribute name="end" type="xs:NMTOKEN" use="required" />
    <xs:attribute name="begin" type="xs:NMTOKEN" use="required" />
  </xs:complexType>
</xs:element>

```

```

<xs:element name="tags">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="tag" minOccurs="1" maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="tag">
  <xs:complexType>
    <xs:attribute name="no" type="xs:NMTOKEN" use="required" />
    <xs:attribute name="name" type="xs:string" use="required" />
  </xs:complexType>
</xs:element>

<xs:element name="rules">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="categories" minOccurs="1" maxOccurs="1" />
      <xs:element ref="rules_used" minOccurs="1" maxOccurs="1" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="categories">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="category" minOccurs="1" maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="category">
  <xs:complexType mixed="true">
    <xs:sequence>
      <xs:element ref="name" minOccurs="0" maxOccurs="1" />
      <xs:element ref="info" minOccurs="0" maxOccurs="1" />
      <xs:element ref="linkurl" minOccurs="0" maxOccurs="1" />
      <xs:element ref="linktext" minOccurs="0" maxOccurs="1" />
      <xs:element ref="count" minOccurs="0" maxOccurs="1" />
    </xs:sequence>
  </xs:complexType>
</xs:element>

<xs:element name="name" type="xs:string" />
<xs:element name="linkurl" type="xs:string" />
<xs:element name="linktext" type="xs:string" />
<xs:element name="count" type="xs:string" />

<xs:element name="rules_used">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="rule" minOccurs="1" maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>
</xs:element>
</xs:schema>

```

D Användarstudie

D1 Medgivandeformulär

Consent to participate in the research project:

The Use of Language Tools in the Context of Writing Swedish as a Second Language

You are invited to participate in a research project that is aimed at assessing the role of language tools in the context of second language learning. We believe that your participation in this project will make an important contribution to our research. As a participant in the study, you will examine a essay in Swedish in the interactive interface *TvärGranska*. While you work with *TvärGranska* you will be observed and afterwards you will be asked some questions.

By participating in this study you will provide us with following:

- Your text (in electronic form) produced in the context of a course of Swedish as second language
- Participating in an interview
- Filling out a questionnaire

Only the research staff will have access to the texts, questionnaires, interviews and other non-public information collected in this project. Your name and other information that could identify you will not appear in any reports of this research without your permission. We will notify you of any reports that are produced in the study and send copies upon your request.

Your participation in this research is entirely voluntary. You may refuse to participate now or withdraw your participation in the research project at any time with no adverse consequences to you. If you have any questions, please ask the responsible for the project : Teresa Cerratto Pargman, tessy@nada.kth.se, phone : 8- 790 63 41 or 070 49 48 405.

If you agree to participate, please sign in the appropriate places below and provide the requested information. A copy of this signed form will be provided for you.

Further information can be found on the following web page:

<http://www.nada.kth.se/~knutsson/call-en.html>

I have read the information about the project and am willing to participate in the project. I hereby give the right to use the material I submit to the project for research, teaching, publication and presentation (conferences, workshops, etc.). I consent to the possible publication of my material (as a whole or in part) in various forms, including paper and electronic media.

Signature of Subject

Signature of Investigator

Printed name

Email

Date

Jenny Rahbek & Ylva Stenervall
Kungliga Tekniska Högskolan

D2 Brev till användaren/deltagaren

Jag heter Ylva Stenervall och gör exjobb på Nada, KTH inom forskningsprojektet CrossCheck – svensk grammatikkontroll för andraspråksskribenter (<http://www.nada.kth.se/theory/projects/xcheck/>) och Språkliga datorstöd och andraspråksinlärning (<http://www.nada.kth.se/~knutsson/call.html>).

Jag har gjort ett nytt interaktivt webbgränssnitt till Granska. Granska är till för att man ska kunna granska sin text och få hjälp med att korrigera stavfel och grammatikfel. Den här användarstudien är till för att ta reda på om gränssnittet är lättförståeligt och användbart för personer med svenska som andraspråk. Jag vore därför väldigt tacksam om du ville ställa upp.

Användarstudien kommer att gå till så att du har med dig en **egen ogranskad text på diskett**. Texten behöver inte vara lång, det räcker med ca 20 meningar.

Det börjar med att du får sitta vid en dator och granska din text med hjälp av webbgränssnittet. Sedan kommer jag att ställa några frågor om vad du tyckte och hur du upplevde webbgränssnittet.

Det hela kommer att ta ungefär en halvtimme och du kommer att få en biocheck som tack för hjälpen.

Tack på förhand!
/Ylva

D3 Instruktioner till användaren/deltagaren

1. Stoppa i disketten i diskettstationen och öppna filen med din text i Word
2. Öppna Explorer och surfa till <http://skrutten.nada.kth.se:8080/test/>
3. Kopiera din text och klistra in den i webbgränssnittet
4. Använd webbgränssnittet för att granska din text
5. Testa att söka efter stavfel
6. När du granskat klart klistra tillbaks din text i Word och spara

D4 Försöksledarprotokoll

Klarar användaren av:

att skriva in rätt adress i Internet Explorer?	Ja	Nej		
att kopiera sin text i Word?	Ja	Nej		
att klistra in sin text i TvärGranska?	Ja	Nej		
att hitta rätt knapp att trycka på för granskning?		Ja	Nej	
Om nej, efter att ha tittat i hjälpen?	Ja	Nej		
att hitta hjälpen om det är något han/hon undrar över?		Ja	Nej	
att inse att klicka på de röda orden?		Ja	Nej	
Om nej, efter att ha tittat i hjälpen?	Ja	Nej		
att i nya lilla fönstret förstå ikonerna?	Ja	Nej		
Om nej, efter att ha tittat i hjälpen?	Ja	Nej		
att inse att man kan hålla muspekaren över för att få en förklaring till ikonerna?			Ja	Nej
Om nej, efter att ha tittat i hjälpen?	Ja	Nej		
att inse att ändra i ändringsrutan?	Ja	Nej		
Om nej, efter att ha tittat i hjälpen?	Ja	Nej		
att inse när man kan klicka på ersättningsförslagen och när man inte kan det?			Ja	Nej
Om nej, efter att ha tittat i hjälpen?	Ja	Nej		
att inse att man ska trycka på ”Uppdatera texten” efter man ändrat i rutan?			Ja	Nej
Om nej, efter att ha tittat i hjälpen?	Ja	Nej		
att inse att man kan stänga rutan genom att trycka på ”Stäng”?			Ja	Nej
Om nej, efter att ha tittat i hjälpen?	Ja	Nej		
att inse att man kan välja bort regler, för att titta på ett sorts fel i taget?	Ja			Nej
Om nej, efter att ha tittat i hjälpen?	Ja	Nej		
Att inse att man kan få en förklaring av felregeln genom att klicka på den?			Ja	Nej
Om nej, efter att ha tittat i hjälpen?	Ja	Nej		
att inse att man i huvudfönstret kan ändra i alla meningar genom att trycka på pennan?	Ja			Nej
Om nej, efter att ha tittat i hjälpen?	Ja	Nej		
att inse att meningen efter man tryckt på pennan hamnar i rutan överst på huvudsidan?	Ja			Nej
Om nej, efter att ha tittat i hjälpen?	Ja	Nej		
att inse att man efter man ändrat ska trycka på ”Uppdatera texten” som var gömd förut?	Ja			Nej
Om nej, efter att ha tittat i hjälpen?	Ja	Nej		
att inse vad knappen ”Texten i nytt fönster” gör?		Ja	Nej	
Om nej, efter att ha tittat i hjälpen?	Ja	Nej		
att efter granskningen kopiera och klistra tillbaks texten i Word?			Ja	Nej

D5 Intervjufrågor

1. Hur tycker du granskandet av texten gick?

.....

2. Var det svårt att komma igång?

.....

3. a) Var det något som var svårt att förstå? (Visa skärmdump på huvudsidan och sidan man får upp när man tryckt på ett fel.(Gäller hela fråga 3.))

.....

b) Var det tydligt vilken knapp du skulle trycka på när du skulle granska texten första gången?

.....

c) Förstod du att det gick att klicka på de ord som var markerade rött?

.....

d) Förstod du att du kunde avmarkera vissa regler för att leta efter bara en sorts fel?

.....

e) Förstod du de olika reglerna? Märkte du att man kunde klicka på dem för att få en förklaring?

.....

f) Förstod du efter att ha läste förklaringen/exemplen?

.....

g) Förstod du vad pennan användes till? Vad tyckte du om pennan? Tryckte du på pennan?

.....

h) Om tryckte på pennan, märkte du att det kom fram en knapp till? Var det tydligt vad den knappen var till för?

.....

i) När du tryckte på ett ord som var markerat rött kom det upp en ny liten ruta, vad tyckte du om den?

.....
j) Var ikonerna tydliga? Förstod du dem?

.....
k) Vad tyckte du om att det ibland gick att klicka på ersättningsförslagen, de gröna orden och ibland inte? Förstod du när det gick och när det inte gick?

.....
l) Vad tyckte du om ändringsrutan?

.....
m) Var knapparna under ändringsrutan bra? Förstod du vad de gjorde?

.....
4. a) Vad tycker du om hjälpen? (Visa skärmdump på hjälpen.(Gäller hela fråga 4.))

.....
b) Tycker du att du blev hjälpt av att läsa hjälpen?

.....
c) Var det något i hjälpen som var svårt att förstå?

.....
5. a) Kan du tänka dig/Kommer du att använda dig av det här gränssnittet i fortsättningen?

.....
b) Om inte, varför då?

.....
6. Är det något du tycker borde förbättras? (Visa alla skärmdumpar.)

.....
7. Är det något du saknar? (Visa alla skärmdumpar.)

D6 Bakgrundsdataenkät

1. Kön:

Kvinna Man

2. Ålder:

0-18 år 19-29 år 30-39 år 40-49 år 50-59 år 60 år eller äldre

3. Utbildning:

Gymnasial utbildning, om Ja: Teknisk eller Humanistisk

Högskoleutbildning eller motsvarande, om Ja: Teknisk eller Humanistisk

Högre utbildning, nämligen.....

4. Modersmål:.....

5. Andra språk du kan:.....

6. Hur länge har du varit i Sverige?.....

7. Har du studerat svenska i Sverige? Om ja, hur länge och på vilken nivå?

.....

8. Hur stor datorvana skulle du säga att du har?

Mycket stor Stor Tillräckligt stor Mindre stor Liten Mycket liten

9. Vilka program (ordbehandlare) använder du dig av för att skriva text?

Microsoft Word

E-postprogram

LaTeX

WordPad

Annat, nämligen.....

10. Brukar du använda dig av stavnings- och grammatikkontrollen i ditt ordbehandlingsprogram?

Ja

Nej

Bara stavningskontroll

Kommentar:

Tack!

E Användarhandledning

Infoga din text i TvärGranska

- Skriv eller klistra in din text i textrutan.
eller

Om du inte har en egen text och vill se hur TvärGranska fungerar eller om du vill se exempel på de olika fel som TvärGranska kan hitta:

- Kryssa i rutan framför **Exempeltext**. Då kommer du att få se en text med mycket fel i när du trycker på knappen märkt *Granska!*.
- Om du vill se exempeltexten innan du tryckt på *Granska!*-knappen tryck på **Exempeltext** då öppnas ett nytt fönster med texten i.

Val av fel att leta efter

- När du startar är rutorna framför **Stavfel**, **Särskrivningsfel**, **Kongruensfel**, **Verbfel** och **Övriga fel** ikryssade, det betyder att TvärGranska letar efter alla sorters fel.
- Om du bara vill söka efter en av dessa tar du bort ikryssningen framför de fel du inte vill leta efter.
- Om du vill ha en förklaring av vad de olika alternativen innebär så kan du klicka på det (t.ex. kan man klicka på **Stavfel**), då kommer ett nytt litet fönster med en förklaring upp.
- Det rekommenderade är att man först söker efter alla fel, och om TvärGranska hittar många fel så väljer man att visa en felkategori i taget.

Redo för granskning




- När din text är redo för granskning och du har valt vilken/vilka felkategorier du vill söka efter klickar du på knappen märkt *Granska!*.
- Medan du väntar på att TvärGranska ska returnera din text med de fel som hittats visas en bild där det står **GRANSKAR...** med röd text.

Efter att TvärGranska granskat texten

- Det TvärGranska har att anmärka på kommer att markeras rött.
- De ord som är röda går att klicka på, när man gör det får man upp en ruta med information om felet.

Information om felet

Det som visas i informationsfönstret är följande:

-  Det som TvärGranska uppfattar som fel, t.ex. ordet *gunkan*.
-  En beskrivning av felet, t.ex. *Okänt ord*.
-  Eventuella förslag på ändringar, t.ex. *gurkan*, *gungan*, *grunkan*.
- En ruta med meningen som felet ingår i, så att man kan rätta den.
- Två knappar som tillhör rutan med meningen i, nämligen en märkt *Uppdatera texten* och en märkt *Stäng*.

- Efter man ändrat i meningen trycker man på *Uppdatera texten*. Meningen kommer då att granskas igen innan den placeras tillbaka i texten.
- Och om man vill stänga rutan utan att göra ändringar trycker man på *Stäng*.



Möjligheten att klicka på ersättningsförslag

Vid stavfel och sårskrivningsfel står ändringsförslagen med fet stil, det betyder att det går att klicka på dem.

- Om man klickar ändras det felaktiga ordet automatiskt till det ersättningsförslag du klickade på. Du behöver alltså inte ändra i rutan med den felaktiga meningen själv.

Redigering av mening

Det finns två sätt att redigera en mening:

- I ändringsrutan i informationsfönstret som beskrivits ovan.
- eller
- Genom att klicka på  i slutet av en mening.
 - Om man klickar på  kommer meningen att plockas upp och hamna i rutan överst på sidan, samtidigt så kommer en knapp märkt *Uppdatera texten* upp under rutan.
 - Efter att man i rutan ändrat meningen trycker man på *Uppdatera texten*, då kommer meningen att granskas igen innan den placeras tillbaka i texten.

Extrahera din text ur TvärGranska

- När du har arbetat klart med texten trycker du på knappen märkt *Texten i nytt fönster* som finns längst ner på sidan. Då kommer ett nytt fönster öppnas med bara texten i så att du kan kopiera den och klistra in den där du vill ha den.

Om

Om du vill veta mer om de projekt på KTH som TvärGranska ingår i:

- Tryck på **Om**.

Hjälp

Det finns också hjälp att få i programmet:

- Tryck på **Hjälp** uppe i det högra hörnet.

F Teknisk dokumentation

För att hantera de felmarkerade orden har en klass Word skapats (se nedan), den innehåller attribut som ordets taggnummer, taggnamn, om ordet ingår i något fel och vilken regel som använts i detta fall. Ett antal Word skapar en sentence[] som i sin tur hashas in i hashtabellen allSentence, med meningens nummer som nyckel. De meningar som inte innehåller några fel hashas in som String.

Hashtabellen allSentence behövs för att efter man har ändrat i en mening skickas den till Granskaservern igen för att granskas om och den behövs alltså för att hålla reda på meningarna. Hade Granskaservern varit snabbare hade man kunnat skicka om hela texten och då hade man inte behövt göra på detta sätt.

Klassen BubbleSort används till att sortera nycklarna från hashtabellen allSentence så att meningarna kan skrivas ut i rätt ordning.

Klassen GramerrorList finns för att hantera de fall när ett ord ingår i flera fel, GramerrorList hashas in i hashtabellen gramError med meningens nummer i kombination med ordets nummer som nyckel.

Hashtabellerna allSentence och gramError sätts till sessionsvariabler i index.jsp så att de kan användas i mark.jsp och out.jsp utan att anropa Granska-klassen igen, då detta tar mer tid.

På skrutton.nada.kth.se under /disk0/httpd/jwsdp/webapps/granska finns följande filer:

about.html	Om
exempeltext.html	Exempeltexten
exempeltext.xml	Filen med exempeltexten
help.html	Hjälpen
index.jsp	Huvudfönstret/Granskningsfönstret
kong.html	Kongruensfel
mark.jsp	Informationsfönstret
out.jsp	Färdiggranskad text
rest.html	Övriga fel
saer.html	Särskrivningsfel
stavning.html	Stavfel
style.css	Stilmallen
verb.html	Verbfel

och följande bilder:

error.gif	Ikon på mark.jsp
granskar.gif	Granskar...-bilden på index.jsp
info.gif	Ikon på mark.jsp
Nadalogo.gif	Bild på about.html
Nadalogo.jpg	Bild på index.jsp
pen.gif	Bild på index.jsp
sugg.gif	Ikon på mark.jsp

Under /disk0/httpd/jwsdp/webapps/granska/WEB-INF/classes finns filerna:

BubbleSort.class
BubbleSort.java
GramerrorList.class
GramerrorList.java
Granska.class
Granska.java
Word.class
Word.java

Huvudklassen Granska har metoderna:

public Granska(String fel, InputStream is)
Tolkar XML:en.

public String getGranska()
Hämtar tolkningen av XML-filen.

public Hashtable getAllSentence()
Hämtar hashtabellen som innehåller alla meningar, meningens nummer är nyckel.

public Hashtable getGramError()
Hämtar hashtabellen som innehåller alla hittade fel (GramerrorList), meningens nummer kombinerat med ordets nummer är nyckel.

public static String getMark(String sRef, String wNo, Hashtable gramError)
Hämtar informationen om ett felmarkerat ord.

public static String getUpdatedAllSentence(String sref, Object obj, Hashtable allSentence, String fel)
Används efter kontrollerandet av en enskild mening i en text. Hämtar den uppdaterade hashtabellen med meningarna.

public static String sentenceGranska(String fel, InputStream is, Hashtable allSentence, String sref, Hashtable gramError)
Tolkar XML:en i en enskild mening i texten.

BubbleSort har metoden:

public static Vector sortBubble(Vector vec)
Sorterar en vektor med nummer.

GramerrorList har metoderna:

public void setKey(String key)
Sätter nyckel.

public String getKey()
Hämtar nyckel.

Word har metoderna:

public Word(String wContent, int wTag)
Sätter ett ord och dess taggnummer.

public String toString()
Ger ordet som String.

public void mark()

Sätter ett ord till att vara markerat.

public boolean isMarked()

Kontrollerar om ett ord är markerat.

public void setRule(String regel)

Sätter regeln som använts till det felmarkerade ordet.

public ArrayList getRule()

Hämtar regeln eller reglerna som använts till ett felmarkerade ordet.

public int getTag()

Hämtar ett ords taggnummer.

public void setTagName(String tagname)

Sätter ett ords taggnamn.

public String getTagName()

Hämtar ett ords taggnamn.

(Namnen på variablerna hänvisar till namnen i XML-filen, se appendix C1 och C2)