MALMÖ HÖGSKOLA
TEKNIK OCH SAMHÄLLE

Examensarbete
15 högskolepoäng, grundnivå

Grafiskt användargränssnitt kontra kommandoradsgränssnitt

En jämförande studie av gränssnitts effektivitet för expertanvändare

Graphical User Interface vs. Command Line Interface
A comparision study of the efficiency of interfaces for expert users

Pernilla Lindh

Emy Spjuth

Examen: Kandidatexamen 180 hp
Huvudområde: Datavetenskap
Datum för slutseminarium: 2010-05-30

Examinator: Göran Hagert
Handledare: Marie Gustafsson Friberger

Resumé

Denna studie är gjord med syftet att undersöka hur ett grafiskt användargränssnitt (GUI) håller måtten baserat på effektivitet jämfört med ett kommandoradsgränssnitt (CLI). Är det så att det generellt går snabbare att jobba i ett kommandoradsgränssnitt eller beror det på vilka uppgifter som skall utföras?

Vi har baserat denna studie på en generell uppfattning hos CLI-expertanvändare att ett CLI är betydligt effektivare och snabbare att jobba i än ett GUI. Vi undersöker därför i denna studie expertanvändares tillvägagångssätt i deras arbete framför datorn. Detta i form av ett utförandetest mätt på hastighet och svar från en enkätundersökning som komplement till användartestets resultat.

Det resultat vi får fram visar att val av gränssnitt i huvudsak är beroende på användningsområde och uppgifternas storlek, vilket enkätsvaren bekräftar. Ju mer uppgifterna ökar i storlek och omfång desto effektivare blir CLI i jämförelse med GUI. Användartestet visar att GUI i genomsnitt inte håller samma nivå som CLI när det gäller effektivitet. Vi analyserar dessutom vilka faktorer som påverkar gränssnittens effektivitet.

Nyckelord: Expertanvändare, gränssnitt, CLI, GUI, effektivitet, operativsystem.

Abstract

This paper has been made in order to examine how well a graphical user interface (GUI) measures compared to a command line interface (CLI). Is it true that a CLI generally is faster to work in or does it depend on the tasks to be performed?

We have based this study on a general perception among CLI-expert users that a CLI is much more efficient and faster to work in than a GUI. Therefore, in this study we investigate expert users' performance when they work with computers. We do this with thru performance test measured on the speed and from a survey response to complement the result of the usability test.

The result we obtain show that the choice of interface is mainly dependent on the tasks and data size, which is confirmed by the survey replies. The more the data increases in size and scale, the more efficient CLI gets in comparison with the GUI. The user tests shows that the GUI, on average, don't succeed to keep the same level of efficiency as CLI. We will also analyze the factors that affect the interfaces efficiency.

Keywords: Expert users, interfaces, CLI, GUI, efficiency, operation system.

Förord

”Tekniken har varit en del av vår miljö i generationer. Den ger kraft åt oss och den frustrerar oss, den förenklar och den försvårar våra liv, den skiljer oss och för oss närmare varandra. ” /Jesse James Garett

Vi vill tacka alla de från Tretton37 i Lund, Beepsend och Vergic i Malmö som tog sig tiden att medverka i våra användartester och därmed gav oss det underlag som vi behövde för att fullfölja denna studie. Ett stort tack även till vår handledare Marie Gustafsson Friberger som guidat och lotsat oss genom vår arbetsgång med sina kunskaper, till Johan Gustavsson för sin entusiasm och all inspiration han gett oss och även ett stort tack till alla ni som hjälp oss korrigera och förbättra texten längst vägen. Slutligen även ett tack till deltagarna och arrangörerna på tbd-Hackathon i Malmö i februari 2012 som utan att veta om det inspirerade oss till denna uppsats och utförandet av den.

Vi hoppas att denna studie ska vara till nytta och nöje för alla läsare oavsett bakgrund!

Malmö, Maj 2012

Pernilla Lindh och Emy Spjuth

Terminologi

Bash – GNU's skal för UNIX, en skal vanligt förekommande i UNIX-baserade operativsystem, som t.ex GNU/Linux och Mac OSX [i].

CLI – Står för "Command Line Interface", på svenska kommandoradsgränssnitt. I ett kommandoradsgränssnitt, dvs. i en kommandotolk/ett skal, skrivs textbaserade kommandon in för att kommunicera med datorn och dess program. Motsvarigheten till kommandoradsgränssnitt är grafiska användargränssnitt [ii].

DOS – Står för "Disk Operating System", en familj operativsystem för PC-datorer. Mellan 1981-1995 var DOS de mest dominerande operativsystemen på marknaden för IBM PC-datorer. DOS för IBM PC kallades PC-DOS och för restrerande återförsäljare kallades det MS-DOS[iii].

GNU – Står för "Gnu's Not Unix" och är ett unixliknande operativsystem som började utvecklas 1983. GNU-systemet kombinerades 1992 med Linux kärna och blev GNU/Linux, som vi i dagliga tal idag kallar Linux [iv].

GNU/Linux – I dagligt tal Linux (se Linux för definition).

GUI – Står för "Graphical User Interface", på svenska grafiskt användargränssnitt. I ett grafiskt användargränssnitt, exempelvis Windows, utförs kommandon, fil- och programhantering etc. med ett ikon-, meny- och knappbaserat pek- och klickgränssnitt som navigeras genom med datormus. [vi]

Kontrollsatser – Låter dig exekvera olika typer av satser och eventuellt repetera exekveringen av dessa satser eller exekvera en sekvens av dessa satser. Exempel på kontrollsatser är selektionssatser (ex. if-satser) och iterationssatser (loopar, ex. *for* och *while*). [viii]

Kommandoprompt – Motsvarar en symbol i skalet som visar att systemet är redo att ta emot ett kommando. [ix].

Linux – Ett operativsystem, skapat 1992, som till största delen består helt av fri programvara. Linux är byggt med en kombination av GNU-systemet, Linux kärna och annan mjukvara [x].

Linuxdistributioner – Operativsystem som distribueras som självständiga enheter och baseras på Linuxkärnan och tillhörande programvara. Ett fåtal exempel på Linuxdistributioner är Arch Linux, Ubuntu och Fedora [xi][xviii].

Linux kärna (kernel) – Stommen av Linux operativsystem som började utvecklas 1991 av Linus Torvalds [v][xviii].

Shell-script – En fil med körbara kommandon [xiii].

Skal – Synonymt med kommandotolk. Ett program som bearbetar de inskrivna kommandona och returnerar utdatat av dom. [xiv].

Skalprogrammering – När du i ett skal använder dig av kontrollsatser för att beskriva för på vilket sätt datorn skall lösa en uppgift [xv].

Terminal – En form av program som kör ett skal. En programvara som medföljer UNIX/Linux-operativsystem. Exempel på terminalprogram är Mac OSX's Terminal.app som när det startas öppnar ett terminalfönster som låter dig köra ett UNIX-skal i en grafisk skrivbordsmiljö i vilket man kan utföra sina kommandon [xvi].

UNIX – En grupp operativsystem. Det första Unix-systemet skrevs av Ken Thompson 1969. Unix används i huvudsak som operativsystem till arbetsstationer och servrar. Det används också på de flesta internetleverantörernas servrar vilket har utgjort en stor betydelse för internets utveckling. Bl.a. är operativsystemen GNU/Linux och Mac OSX baserat på Unix [xvii].

Innehåll

1. Introduktion

I dagens teknikutvecklade samhälle är det svårt att föreställa sig att de flesta människor för bara 30 år sedan aldrig använt en dator. Idag finns datorer överallt i vårt samhälle och inte minst i våra hem. 90% av alla människor i Sverige mellan 16-74 år har idag en dator hemma [1]. Persondatorn har på tre decennier gått från att vara en ovanlig ägodel som bara människor med ett specifikt intresse och med ett genuint behov hade, till att bli en generell ägodel för människor med olika typ av skäl och bakgrund. Med detta har även antalet datoranvändare ökat, samt människors allmänna kunskaper om datorer och teknik. Men i vilken utsträckning lär sig användaren om den teknik hon använder? Krug skriver:

> One of the things that becomes obvious as soon as you do any usability testing — whether you're testing Web sites, software, or household appliances — is the extent to which people use things all the time without understanding how they work, or with completely wrong-headed ideas about how they work. [2, sid 26]

En av de saker som blir uppenbara så fort man gör någon form av användartester, oavsett om du testar webbplatser, programvara eller hushållsapparater, är i vilken utsträckning människor använder saker hela tiden utan att förstå hur de fungerar, eller med helt befängda uppfattningar om hur de fungerar. Är det så att användare idag inte är intresserade av hur den bakomliggande tekniken fungerar, att vi lär oss göra det vi vill och inte mer? Gäller detta alla, eller är det skillnad på användare och användare?

Ser vi på dagens utvecklingen fokuserar den på att utveckla gränssnitten i operativsystemet så att det blir lätta att använda. Gränssnittet utformas med bilder, ikoner och andra metaforer för att gestalta objekt i verkligheten. Fördelen med detta är att människor som är nybörjare lättare kan börja lära sig att använda sin dator. Men när användaren utvecklas och lär sig hur systemet skall användas ändras prioriteringarna [3]. När det gäller vardagliga uppgifter och operationer behöver användaren inte längre guidas genom sitt system, utan istället prioriteras bl.a. att utförandena ska gå snabbt att uträtta, dvs. vikten av att gränssnittet låter oss jobba mer effektivt (baserat på snabbhet) väger mer [4][5].

Många expertanvändare väljer i vissa sammanhang bort GUI't helt på grund av detta och föredrar istället att arbeta i ett CLI. Nyckelorden i denna studie har vi härmed nämnt. Användare, operativsystem, gränssnitt och effektivitet. Det vi med denna studie undersöker är alltså hur ett GUI håller måtten baserat på effektivitet jämfört med motsvariga CLI. Stämmer det att ett CLI går snabbare att jobba i eller beror det på vad för uppgifter som skall utföras? I denna studie presenterar vi ett resultat för vid vilka situationer gränssnitten anpassar sig och lämpar sig bäst. Inför vår studie har vi studerat litteratur kring ämnet och utgått ifrån Whiteside m.fl. studie "User Performance with Command, Menu, and Iconic Interfaces" från 1985 [6].

I Whiteside's m.fl. studie testas CLI och GUI på användare och jämför dessa med varandra på olika system baserat på, som Whiteside m.fl. säger, *ease-of-use*, dvs. lätthet att använda. De klassificerar användare i tre kategorier; nybörjare, vardagsanvändare och expertanvändare. Vi utgår också ifrån en studie från 2007 av Chen och Zhang som testade gränssnitten i ett elektronisk tandjournalsystem på nybörjare och expertanvändare baserat på tid det tar att utföra uppgifter [7]. Vi ifrågasätter dock Chen och Zhangs metod gällande deras mätning av

effektivitet. Vi anser här att en mätning av effektivitet av gränssnitt ej kan göras baserat på nybörjaranvändare p.g.a. deras brist på erfarenhet och kunskap. Detta förklaras tydligare i avsnitt 2.4 som omfattar tidigare studier.

1.1 Syfte

En hypotes som denna studie bygger på är att CLI-expertanvändare uppfattar ett CLI i de flesta sammanhang som betydligt effektivare och snabbare att jobba i än ett GUI. Syftet med studien är att undersöka expertanvändares tillvägagångssätt i deras arbete framför datorn i ett användartest. Med detta användartest testas gränssnittens kapacitet och kraftfullhet. Detta görs genom ett experiment i form av ett utförandetest där tiden för utförandet används för att mäta prestationen. Vi jämför därefter de båda gränssnittens resultat för att ta fram ett resultat över vilket gränssnitt som är tidsmässigt effektivast.

För definition av expertanvändare, se kapitel 3.

1.2 Frågeställningar

Frågor som diskuteras i denna studier är:

- *Vilket gränssnitt av GUI och CLI är tidsmässigt effektivast?*
- *Vilka uppgifter är bäst lämpade för CLI respektive GUI?*
- *Vilka faktorer i gränssnitten GUI och CLI's utformning påverkar tidsmässigt resultatet av effektiviteten*?

1.3 Avgränsning

Vi har inte valt att göra vår studie baserat på hur lätt olika operativsystems gränssnitt är att använda. Vi har inte heller valt att fokusera på nybörjaranvändarna – inte ens att beräkna dem i denna studie överhuvudtaget. Vi har valt att studera redan erfarna användare som använder sitt system på expertnivå, och vill därmed testa gränssnittens effektivitet baserat på tid det tar att utföra givna uppgifter. Bakom detta val har vi haft Nielsen's [8] belägg i åtanke, att:

It's more difficult to conduct usability studies with experienced users than with novices, and the improvements are usually smaller. Still, improving expert performance is often worth the effort.

Det är svårare att utföra användartester på erfarna användare än nybörjare och resultaten ser man större skillnader, men att testa experters resultat är ofta värt besväret.

För denna studie är det i huvudsak användarnas personliga vardagssysslor på deras persondatorer som vi valt att avgränsa oss till att studera. Vi har inte valt att testa några uppgifter på programvara utanför standardprogram i operativsystemen. Detta för att försäkra

oss om att användarna vet hur systemet fungerar och inte behöver begränsas på grund av att de avsaknar erfarenhet om ett visst program. Det här innefattar vanlig mapp- och filhantering i operativsystem.

1.4 Disposition

Uppsatsen är uppdelat i sex kapitel. Det första kapitlet är en introduktion till studiens innehåll. Andra kapitlet är ett bakgrundskapitel där begrepp som är relevanta för uppsatsens senare delar utreds. Efterföljande avsnitt är metodkapitlet som belyser hur vi som uppsatsförfattare gått till väga för att få svar på våra frågeställningar. I metodkapitlet redovisas de metoder som använts samt hur de valda metoderna genomförts. Därefter kommer ett resultat och analys som visar resultat och som analyserar resultatet. Nästa kapitel innehåller en diskussionsdel med resultatgranskning och förslag till vidarestudier. Som avslutande del följer ett slutsatskapitel som sammanfattar denna uppsats.

2. Bakgrund

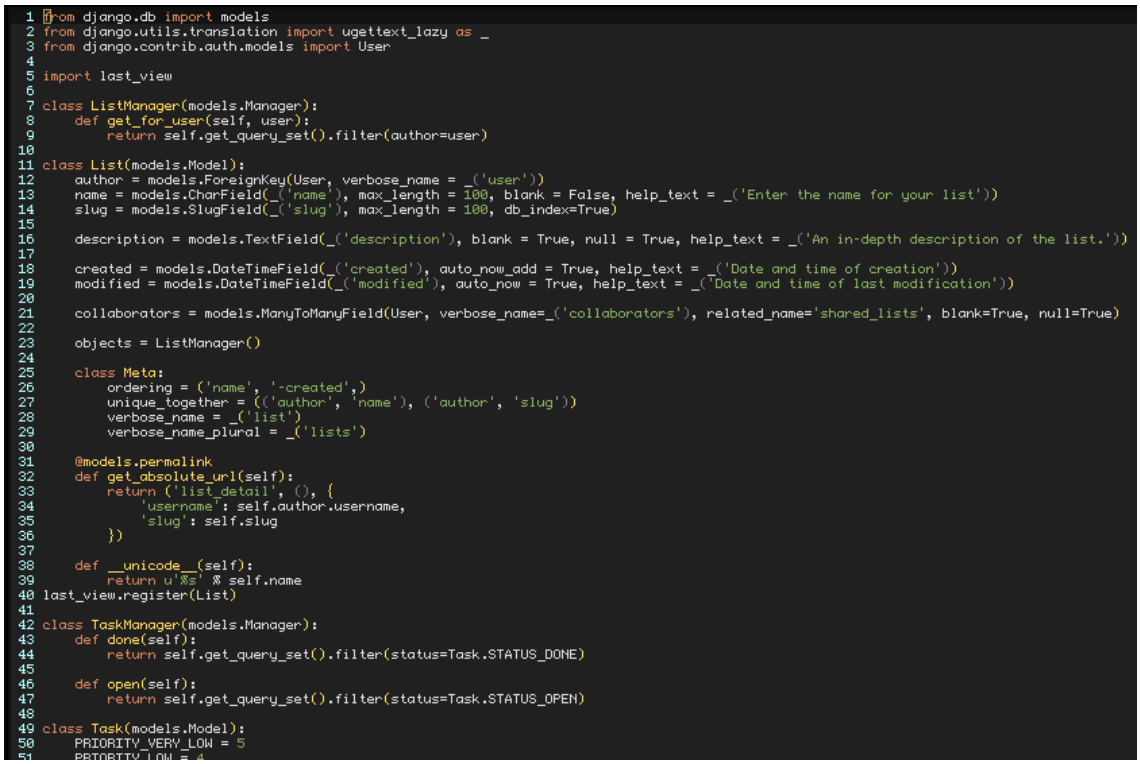
I detta kapitel förklarar vi CLI och GUI samt dess bakgrund för att bättre förstå oss på de bakomliggande faktorerna till de refererade studiernas resultat och vårt.

2.1 CLI

I datorns operativsystem hittar du kommandoradstolken, eller skalet. Det är ett program som med ett CLI tolkar och kör de kommandon som skrivs in av en människa eller ett program för att interagera med sin dators filer och program [9]. Ett CLI är helt och hållet alfanumeriska (se Figur 1). Ett CLI används oftast av programmerare, tekniska systemadministratörer eller tekniskt avancerade persondator-användare eftersom det generellt är kraftfullare att använda [3]. Även synskadade kan föredra att använda ett skal eftersom kommandona och svaren kan visas med punkttext [10].

Ett CLI består av en specifik syntax och olika skal har olika inbyggda kommandon. Två exempel på inbyggda kommandon i ett Bash-skal är ls (list) som listar filer och information om filer, och mkdir (make directory) som skapar en ny mapp. [11] Det är även möjligt att skapa egna skräddarsydda variabler (s.k. *Alias*) som motsvarar de inbyggda kommandona, eller skapa sin egna innebörd av ett redan inbyggt kommando och överstyra detta till det inbyggda kommandot[12]. Operativsystemen MS-DOS och UNIX definierar sina egna regler som alla kommandon måste följa. När det gäller inbyggda system är det varje återförsäljare som definierar sin egna uppsättning av regler. Dessa regler bestämmer hur användarna navigerar sig genom systemet. [13].

I början av kommandoraden står prompten. I ett CLI är det möjligt att skräddarsy utseendet i kommandoprompten utifrån ens egna behov. Detta innebär att du kan räkna ut komplexa kommandon och spara ner dem till en fil. Sedan kan att du öppna upp kommandot för att använda det igen, utan att du behöver räkna ut det igen på nytt. Detta kallas för *Shell Scripting* [14].



```
from django.db import models
from django.utils.translation import ugettext_lazy as _
from django.contrib.auth.models import User

import last_view

class ListManager(models.Manager):
    def get_for_user(self, user):
        return self.get_query_set().filter(author=user)

class List(models.Model):
    author = models.ForeignKey(User, verbose_name = _('user'))
    name = models.CharField(_('name'), max_length = 100, blank = False, help_text = _('Enter the name for your list'))
    slug = models.SlugField(_('slug'), max_length = 100, db_index=True)

    description = models.TextField(_('description'), blank = True, null = True, help_text = _('An in-depth description of the list.'))

    created = models.DateTimeField(_('created'), auto_now_add = True, help_text = _('Date and time of creation'))
    modified = models.DateTimeField(_('modified'), auto_now = True, help_text = _('Date and time of last modification'))

    collaborators = models.ManyToManyField(User, verbose_name=_('collaborators'), related_name='shared_lists', blank=True, null=True)

    objects = ListManager()

    class Meta:
        ordering = ('name', '-created',)
        unique_together = (('author', 'name'), ('author', 'slug'))
        verbose_name = _('list')
        verbose_name_plural = _('lists')

    @models.permalink
    def get_absolute_url(self):
        return ('list_detail', (), {
            'username': self.author.username,
            'slug': self.slug
        })

    def __unicode__(self):
        return u'%s' % self.name
last_view.register(List)

class TaskManager(models.Manager):
    def done(self):
        return self.get_query_set().filter(status=Task.STATUS_DONE)

    def open(self):
        return self.get_query_set().filter(status=Task.STATUS_OPEN)

class Task(models.Model):
    PRIORITY_VERY_LOW = 5
    PRIORITY_LOW = 4
```

Figur 1: Linux kommandoradsgränssnitt (CLI) Bash i en GNOME-terminal.

Kommandona som skrivs till CLI-skalet är vanligtvis av följande syntax:

doSomething how toFiles

doSomething how < inputFile > outputFile

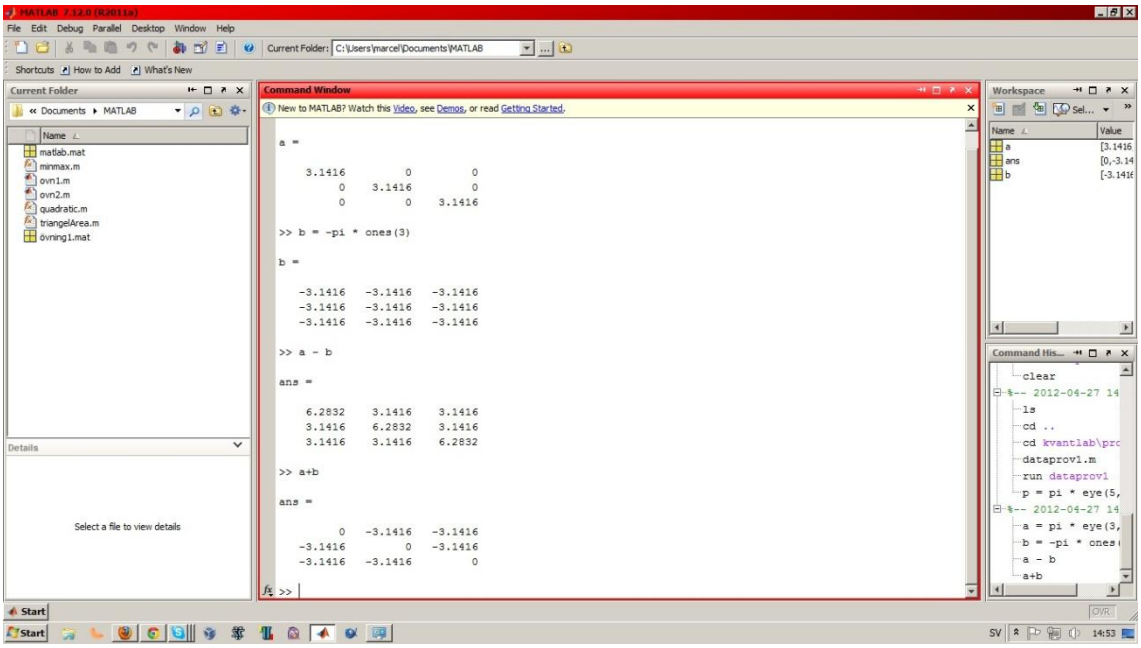
“doSomething” motsvarar här ett verb. ”how” förklarar på vilket sätt kommandot ska utföras och ”toFiles” motsvarar ett objekt, vanligtvis en eller flera filer, som kommandot skall köras utifrån. Tecknet ”>” symboliserar en omdirigering och talar om för kommandotolkskalet att skicka utdatat från tidigare kommandon (som står till vänster om ”>”) till annan destination (filnamnet som står till höger om ”>”).

En annan metod för omdirigering är att skriva ”|”. Detta berättar för kommandoradstolken att den ska behandla utdatat av detta kommandot som indata för nästa kommando, dvs. att du kan skicka text fram och tillbaks mellan olika kommandon. Denna typ av omdirigering kallas för en *pipe* eller *pipeline* [15].

Olika exempel på kommandoradstolkar är bl.a. de tidigare modellerna MS-DOS och IBM-DOS’s COMMAND.COM, UNIX-skalet bash och Windows cmd.exe-program. I de flesta operativsystem är det dessutom fullt möjligt att ersätta standardskalet med ett annat specialiserat skal [16]. Windows Vista och Windows 7 kommer med ramverket PowerShell som innehåller en kommandoradstolk och ett skriptspråk som är specifikt avsett för systemadministratörer. Det är en kombination av funktionerna från traditionella UNIX-skal med Microsoft’s egna objektorienterade .NET-ramverk, och har egna inbyggda

kommandon. För ett mer UNIX-liknande CLI för Windows finns bl.a. open source-biblioteket Cygwin för att få Windows att upplevas som en Linux-miljö [17].

I Mac's senare operativsystem OS X och i Linuxdistributioner som exempelvis Ubuntu medföljer en s.k. Terminal, ett program som möjliggör för användaren att använda olika skal baserade på Bash [18],[19]. Vissa program tillhandahåller dessutom en kombination av både GUI och CLI. I vissa fall görs ett grafiskt omslag runt ett CLI-program, andra gånger används ett CLI för att styra ett GUI-program. Ett exempel är programmet och programspråket MATLAB (se Figur 2) som används för tekniska, matematiska beräkningar. I MATLAB är CLI integrerat med GUI't och vi kan skriva kommandon för att öppna katalogfönster, för att skriva och läsa in saker till programmets arbetsyta och göra andra grafiska åtgärder från kommandoraden [20].



Figur 2: Bilden visar kommandofönstret i MATLABs GUI.

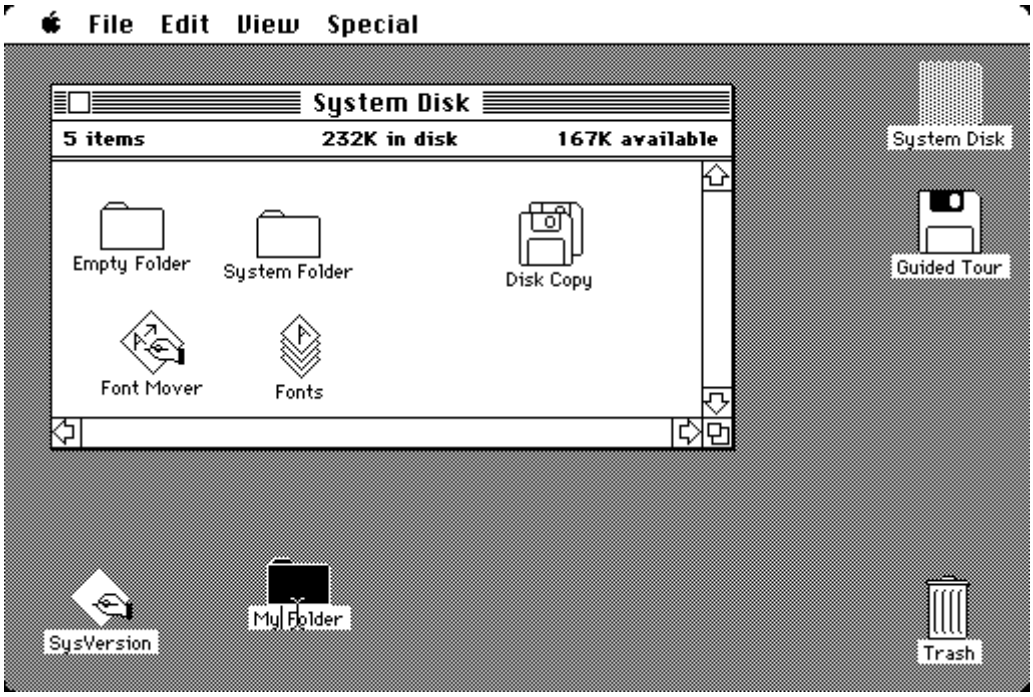
Idag innehåller de vanligaste grafiska operativsystemen ett kommandoskal. Mac OS X och GNU/Linux operativsystem har Unix-skalet Bash integrerat, och Windows Vista och 7 har Microsoft's PowerShell och cmd.exe-programmet som gör att vi kan interagera med datorn via ett CLI [21]. Bash bygger på en kombination av nyare verktyg och klassiska Unix-verktyg, PowerShell har sin egen uppsättning av kommandoradsprogram [22].

2.2 Grafiskt användargränssnitt (GUI)

Definitionen av ett GUI är att det består av bilder som utgör metaforer till objekt i verkligheten i form av knappar, ikoner, menyer och fönster som kan hanteras och manövreras med hjälp av en datormus och, i viss mån, ett tangentbord. Ett GUI utnyttjar

alltså igenkännandeprincipen och WYSIWYG-konceptet (*What you see is what you get)*, d.v.s. det du ser på bildskärmen är vad du får [23][24]. Metaforer ersätter den underliggande koden och komplexiteten i systemet och kommunicerar istället genom grafiska bilder. Dessa bilder underlättar inlärning och användning för användaren, möjliggör för interaktion och ger förståelse för informationen i gränssnittet [25]. Detta styrker ett GUI's anseende att vara lätt att lära och lätt att använda. T.ex är det för en nybörjaranvändare enklare att med så kallat *drag-and-drop* flytta en fil genom att dra den med muspekaren från en plats till en annan istället för att, som i ett CLI, behöva komma ihåg och skriva kommandon för att flytta filen. GUI är nu också standardgränssnitt i generell människa-datorinteraktion och har haft inflytande på många datoranvändares arbete från dess att det först släpptes till idag.

1984 släppte Apple systemet "Macintosh" som blev den första kommersiellt framgångsrika produkten på marknaden att använda sig av ett GUI (se Figur 3). Detta blev startskottet för en helt ny revolutionerande era för grafiska användargränssnitt [26].



Figur 3: GUI't från Apples "Macintosh" skrivbord från 1984 – Den första kommersiellt framgångsrika persondatorn med ett GUI [Med tillåtelse att publiceras från http://toastytech.com/guis/mac11deskicons.gif.]

Figur 4: GUI't i Mac's operativsystem OS X 10.7.3 Lion.

Microsoft's motsvarighet till det första kommersiellt framgångsrika operativsystem med ett GUI blev Windows 3 som blev lanserat 1990. Från Apple's Macintosh och Windows 3 har vi idag Mac OS X och Windows 7. GUI't har på dessa 30 år blivit det gränssnitt som vi idag tar för givet när vi arbetar med våra datorer, mobiltelefoner och läsplattor (se Figur 4).

2.3 GUI mot CLI

För nybörjaranvändare är det i högsta grad viktigt att ett gränssnitt är lätt att använda. Gränssnittsutvecklingen idag lägger stor fokus på detta. Krisler och Alterman [4] tydliggör dessutom att designen och utformningen av gränssnitt är relevant att göra utifrån en nybörjares synvinkel. Detta för att hjälpa en nybörjare med övergången till att kunna behärska gränssnittet.

> At the novice level, the interface should encourage exploration and promote the discovery of the functionality required to perform their tasks, within the context of their activity. Once the user establishes a comfort level with the interface, she should continue to learn, transitioning beyond the visual elements of the interface, so as to develop the deep structural knowledge characteristic of expertise that will enable her to work more efficiently.

D.v.s. på nybörjarnivå bör gränssnittet uppmuntra utforskning och främja upptäckten av funktioner som krävs för att utföra uppgifter, inom ramen för verksamheten. När användaren upprättar blir bekväm med gränssnittet ska hon fortsätta att lära sig, övergå bortom de synliga

delarna av gränssnittet, för att utveckla den djupa strukturella kunskapen som är karakteristisk för expertis och som gör det möjligt för henne att arbeta mer effektivt.

Frågar vi expertanvändare är ett systems kraftfullhet, effektivitet, pålitlighet, robusthet och precision de främsta kriterierna för utformningen av ett användbart system [5]. I en studie gjord på systemadministratörers förtroende för olika gränssnitt i deras programvara nämner Takayama och Kandogan att utvecklingen för människa-dator-interaktiv programvara lägger fokus på användarvänlighet och att ett gränssnitt skall vara lätt att använda (”ease-of-use”). För expertanvändare, i detta fall systemadministratörer, har användarvänlighet och ”ease-of-use” en helt annan betydelse. Takayama och Kandogan nämner att återförsäljare av system erbjuder grafiska användargränssnitt åt administratörerna för deras arbete, med antagandet att grafiska användargränssnitt är lättare att använda. De avancerade användarna hade dock en annan åsikt om det hela [5].

I vår studie handlar det inte om att ta fram ett resultat för vilket av gränssnitten som generellt är bäst. Vi vet att de båda fungerar som mest optimalt tillsammans och kompletterar varandra utifrån vilka arbetsuppgifter som skall utföras. Exempelvis lämpar sig fotoredigering och surfning på webben bäst i det visuella GUI’t. Ska du däremot ändra MAC-adress på ett felaktigt ethernetkort krävs det i princip att du gör detta genom ett CLI.

2.4 Tidigare studier

I vår studie gör vi ett vidareexperiment på studien från 1985 av Whiteside m.fl [6] och Chen och Zhangs studie från 2007 [7]. Vi testar GUI och CLI genom att mäta vardera gränssnitts effektivitet baserat på faktisk tid det tar att utföra specifika uppgifter och operationer och därefter jämför gränssnittens resultat för att se vilket gränssnitt som är tidsmässigt effektivast.

Användargränssnitt har genom tiderna sett olika ut. 1985 gjorde Whiteside mfl. en studie där GUI och CLI testades på olika system med syfte att påvisa vilket system som var lättast att använda. När denna studie gjordes hade GUI’t precis kommit ut på den kommersiella marknaden. Tidigare hade människor enbart använt sig av ett CLI och resultatet visade inte helt överraskande därför att samtliga typer av användare (nybörjare, standardanvändare och expertanvändare) lättare orienterade sig i ett CLI.

I Chen och Zhangs studie [7] ser vi dock ett annat resultat. Här visar Chen och Zhang att ett GUI är effektivast för nybörjaranvändare och textbaserat användargränssnitt (TUI) är effektivast för experter. Ytterligare resultat beräknar Chen och Zhang därefter utifrån en medelhastighet av experternas och nybörjarnas tid kombinerat för de båda gränssnitten, och därmed får GUI den bästa medelhastigheten och summeras som det mest effektiva gränssnittet. Att ha i åtanke är att Chen och Zhang istället för CLI talar om TUI, d.v.s. textbaserat användargränssnitt. De ger ingen definition av TUI, men eftersom de refererar till studier som baseras på CLI kan vi bara anta att deras definition av TUI är närbesläktat med CLI.

Nybörjarna har i Chen och Zhangs test fått använda sig av manualer för att ta sig igenom uppgifterna. Utifrån att GUI vars fokus ligger mer på lätthet att använda jämfört med ett

TUI som kräver vissa förkunskaper av syntaxen, dvs. hur kommandona skall skrivas, är det förståeligt att nybörjarna gjorde ett avsevärt sämre resultat i utförande för TUI't.

Metoden att räkna ut hastigheten på utförda uppgifter anser vi är rättfärdigad. Att därefter presentera resultatet utifrån vardera användarkategori är också berättigat. Alltså, att säga att för nybörjare är det effektivast att använda sig av ett GUI är riktigt. Vi ifrågasätter dock deras metod om att beräkna just de faktiska gränssnittens effektivitet utifrån en kombination av resultaten från båda användarkategoriernas prestationer i de båda gränssnitten. Vi menar att bara för att en nybörjare presterar bättre i ett GUI än ett TUI betyder det inte att TUI't i sig är ineffektivt. Att mäta effektivitet med nybörjare i testet mäter därmed inte effektiviteten på det faktiskta gränssnittets uppbyggnad, utan enbart vilket gränssnitt som lämpar sig bäst beroende på användare. Som Krisler och Alterman [4] säger:

> Mastery, the state of knowing how to work efficiently with an application requires an understanding of the underlying conceptual model of the system.

D.v.s. det krävs att du förstår systemets underliggande struktur för att kunna arbeta effektivt i ett system, och detta är expertnivå. Vi gör härmed skillnad på användarnas resultat och det faktiska gränssnittens resultat. Därför kommer vi att borträkna användarnas betänketid och tid för läsning av uppgifter ifrån presentationen av utföranderesultaten och enbart presentera restrerande tid.

3. Metod

I detta kapitel presenterar vi vårt val av metod, riktlinjer från tidigare studier samt reflektioner om vårt metodologiska tillvägagångssätt. Fokus för denna studie ligger främst på att undersöka vilket gränssnitt som är mest effektivt beroende på uppgift. Vi undersöker expertanvändare och deras arbete i operativsystem på persondator.

Deltagarna i användartestet valdes utifrån ett krav om hög kunskap inom valt gränssnitt och klassificerades alla som ”expertanvändare”. Expertanvändare definieras som motsatsen till nybörjaranvändare. En användare som har hög förståelse, hög kunskap och långvarig erfarenhet för hur mjukvara generellt fungerar definierar vi i denna studie som expertanvändare.

Mer detaljerat definierar vi expertanvändare av CLI till att man som användare skall vara insatt i shell scripting och kunna utföra kommandon och lösa uppgifter med skalprogrammering (d.v.s. i form av kontrollsatser med ex. selektioner och iterationer). Expertanvändare av GUI definieras inte mer än att du som användare behärskar GUI’t till sin yttersta spets, men i användartesterna har vi försäkrat oss om GUI-användarnas expertis utifrån personer som arbetar som .NET-utvecklare i grafiskt gränssnitt i Windows.

3.1 Metodval

För att besvara våra frågeställningar (se avsnitt 1.2) har vi använt oss av en kombination av en enkät med ett användartest. Svaren från enkäterna kompletterar resultatet av användartesterna och kan styrka samt förklara testerna för att få ett så bra resultat som möjligt. Valet av att utföra ett användartest förefaller sig självklart för att testa gränssnittens prestationer. Med ett användartest kan vi få fram mätbar data i form av tid som möjliggör för oss att mäta effektivitet. Val för framställandet av enkät är gjort utifrån Ruane’s bok om vägledning i forskning [27]. Ruane skriver att enkäter är ett effektivt redskap för att samla in data eftersom en enkät i princip ”sköter sig själv”. Detta innebär att man kan utföra en stor undersökning utan att behöva lägga mycket tid på det. Vår önskan att snabbt ta reda på ett flertal svarspersoners datoranvändning och deras uppfattning om gränssnitten med utförliga motivationer gjordes därför enklast med en enkätundersökning. Fördelen med att snabb kunna sammanställa resultaten utifrån svarspersonerna är ytterligare en anledning i val av utförandet av enkäter.

Som underlag till vår studie har vi använt oss av Whiteside’s mfl. studie [6] som mäter hur lätta system är att använda baserat på hur många uppgifter de klarar att utföra på en specifik tid. Vår studie baseras också på Chen och Zhangs studie [7] som, precis som vi i denna studie, mäter gränssnitts effektivitet baserat på tid. Vårt ifrågasättande av Chen och Zhangs mätning av effektivitet baserat på nybörjare leder däremot till ett särskilt val av tillvägagångssätt i utförandet av användartestet. I testet mäter vi gränssnittets effektivitet utifrån de operationer som krävs för att utföra en uppgift. Vi menar att en nybörjares prestation i ett CLI inte innebär att ett CLI i sig är ineffektivt. Därför mäter vi enbart utifrån expertanvändare och borträknar det vi anser vara irrelevant för att utvärdera gränssnittens

faktiska prestation. Vi kommer därför att räkna bort användarnas betänketid och tid för läsning av uppgifter ifrån presentationen av utföranderesultaten och enbart presentera resterande tid. Denna resterande tid är uppdelad i faktiska operationer och övriga operationer.

3.2 Enkät

En enkät har sammanställts och skickats ut till utvalda personer. De företag som valdes ut var Tretton37 och Beepsend med syfte att få information till utförandet av användartestet. Vi har även valt att ställa frågor i enkäten med avsikt att använda dessa som komplement till användartesternas resultat. Det detaljerade syftet med denna enkät är att:

- Ta reda på hur testet ska utformas på bästa lämpliga vis för att kunna mäta effektiviteten.
- Ta reda på vilken typ av testdeltagare som anses vara tillräckligt avancerade expertanvändare för att kunna göra testerna.
- Få en generell uppfattning om vilka operativsystem som föredras. Detta för att ta reda på hur gränssnitten som användarna jobbar i ser ut, men också för att förbereda våra användartester i form av att undersöka vilka skärminspelningsprogram som är kompatibla med operativsystemen.
- Få svar som kan komplettera och bekräfta resultatet som vi får ut av användartesterna.

Enkäterna är medvetet utformade med öppna frågor.

Baserat på de frågor som vi ville ha svar på använde vi oss av Ruane's [27] rekommendationer för när öppna frågor bör användas. Ruane säger att öppna frågor är lämpligast att använda då frågorna som ställs är komplicerade och inte har några självklara svar eller för att ta reda på en svarspersons unika åsikt om ett ämne. Vi ville dessutom med enkäten ge svarspersonerna en frihet att svara på ett sätt som de själva anser lämpligast för att få ut så mycket information som möjligt. Ytterligare är enkätens omfång baserat på Holme och Solvangs synpunkt på att om enkäten blir för stor kan bortfall för svarspersoner komma att öka avsevärt och det finns en risk att svaren också blir mindre seriösa [28]. Vi valde därför att hålla enkätens innehåll på endast en sida men med frågor som ändå motsvarade vårt behov av information.

Delar av det resultat som enkäten gett har vi utgått från vid utformingen av användartestet. Ett betydande exempel som format användartestet är svarspersonernas åsikt om att gränssnitt lämpar sig bäst beroende på vilken uppgift som skall utföras.

Enkäten har skickats till utvalda personer på företagen Beepsend i Malmö och Tretton37 i Lund. För fullständig enkät , se bilaga 1.

3.3 Användartest

Användartester har gjorts för att ta fram resultat för vilket gränssnitt som är effektivast att använda vid fil- och mapphantering i operativsystem. Vi har inte valt att återge alla typer av arbetsområden för dessa gränssnitt utan valt att begränsa oss till ett personanvändande via operativsystem på persondator. Därmed räknar vi inte med någon form av användning utifrån företagssystem. Dock räknar vi med generella vardagssysslor, navigering och utveckling för eget intresse hos personanvändare vid deras personliga datorer.

För att kunna mäta effektiviteten har vi använt oss av Albert och Tullis metodik [29]. De har tagit fram fem viktiga punkter man skall ha i åtanke när man mäter effektivitet:

- Identifiera uppgiften som skall mätas.
- Definiera start och slut av uppgift.
- Räkna utförandena.
- Utförandena måste vara meningsfulla.
- Titta enbart på de uppgifter som har lyckats.

Albert och Tullis anser att det vanligaste sättet att anaylsera och presentera effektivitet i mått av tid är beräkna ett medelvärde för varje uppgift och deltagare. På detta sätt ser vi vilka uppgifter som kräver mer arbete och då tar längre tid att utföra. Därmed ser vi vilket gränssnitt som är mest effektivit per uppgift.

3.3.1 Pilottest

Ett pilottest utformades under tidigt stadie av denna studie. Detta test utformades utifrån Nielsen's anvisningar som handlar om att genomföra användartester på expertanvändare [8]. Nielsen nämner att ett pilottest skall göras på ett fåtal personer. Vi utförde därmed pilottestet på tre personer, och de åsikter och tankar som testdeltagarna återgav togs i beaktning. Testets utformning förnyades och pilottest-personerna fick se igenom det nya testet för att säkerställa att kvaliteten var god. Denna version av testet är det slutgiltiga testet vi använt för de skarpa användartesterna.

3.3.2 Skarpt användartest

Det skarpa användartestets utformning har inspirerats av Nielsen's tre komponenter för att studera användare genom användartester [30]:

1. Få tag i representativa deltagare.
2. Be användarna att utföra representativa uppgifter med systemet.

3. Observera vad användarna gör, när de lyckas och när de får problem i gränssnittet. Var tyst och låt deltagaren prata.

Användartestet är uppdelat i fem uppgifter. Dessa uppgifter innefattar att bygga hierarkier med hjälp av mappar, skapa en textfil samt göra sökning i en chattlogg (för fullständigt användartest, se bilaga 4). Uppgifterna är skapade för att efterlikna realistiska vardagssysslor. Valet av vardagssysslor och dess innehåll är medvetet för att inte en användares brist på eventuella kunskaper skall färga resultatet för gränssnittets effektivitet. Vi har därför valt att enbart testa fil- och mapphantering och har valt bort hantering av program utöver standardprogrammen i operativsystemet.

Tabell A: Tabellen beskriver användartestets uppgifter.

Uppgift	Uppgiftsbeskrivning
1	Skapa 5 kopior på en textfil och lägg i motsvarande 5 mappar. Zippa dessa mappar, packa upp på given plats och radera.
2	Skapa mappar från A-Z.
3	Skapa mappar med siffrorna 1-15, där mapp 2 ligger i mapp 1, mapp 3 i mapp 2 etc.
4	Skapa 7 mappar med givna specifika namn.
5	Leta reda på specifika textstycken i en chattlogg, räkna hur många gånger ett specifikt givnet ord angetts och räkna antal ord i ett stycke.

Varje deltagare har testats enkilt i enrum för att förhindra yttre störningsmoment och för att lättare bibehålla deras fokus på uppgifterna. Lindenberg och Neerincx framlägger i sin studie om användartester [31] att testets uppgifter bör presenteras en och en. Första uppgiften presenteras och förklaras, testdeltagaren utför uppgiften och först då uppgiften är slutförd presenteras och förklaras nästkommande uppgift. Denna metod har använts i denna studies användartest. Vid utförande av uppgift har deltagarna enbart fått se den uppgift de för stunden jobbar med. Deltagarna har fått direktiv om att tydligt säga när de börjat läsa uppgiften och när uppgiften är klar.

Testdeltagarna har utfört uppgifterna på egen medhavd dator. Likaså har de själva valt operativsystem efter egna preferenser. Detta för att bibehålla deltagarnas naturliga miljö och verktyg som de är vana att använda för att minimera risken för onödiga Övriga Operationer och andra diverse störningsmoment för användaren som kan påverka resultatet.

Samtliga användartester har filmats med hjälp av ett skärminspelningsprogram och klockats med tidtagarur enligt Lindenberg och Neerincx anvisningar för loggning [31]. De

operativsystem som använts är OS X , Windows 7 och Linuxdistrubutionen Fedora och Arch Linux.

3.3.3 Urval av testdeltagare

Testdeltagarna valdes utifrån vår definition av expertanvändare (för definition, se kapitel 3.). Antal testdeltagare valdes utifrån Nielsen's anvisningar i hans undersökning från 2000 [30] som visar att bäst resultat uppnås i små användartester med fem testdeltagare i vardera. I denna studie testades totalt 10 personer varav fem testades på ett CLI och fem testades på ett GUI. De personer som vi har utfört användartesterna på kommer samtliga från företaget Tretton37 i Lund samt Beepsend och Vergic i Malmö. Testdeltagarnas ålder varierade mellan 23-38 år.

3.3.4 Analys av användartester

Analysen av användartesterna delar vi in i fyra olika moment:

- Beräkna de Faktiska Operationer för CLI/GUI
- Beräkna Övriga Operationer för CLI/GUI
- Beräkna de Faktiska Operationerna + de Övriga Operationerna för CLI/GUI
- Jämföra tiden för varje uppgift för vardera gränssnitt

Den beräknade tiden för en fullständig uppgift är baserat på testdeltagarens aktiva arbetande för att lösa uppgiften. Inräknat är tiden för faktiska operationer och övriga operationer i uppgift. Restrerande tid, dvs. vid läsning, väntan och funderingar, har därmed tagits bort ur beräkningarna.

De Faktiska Operationerna är de steg som krävs av gränssnittet att utföra ett kommando eller en operation. Faktiska operationer gäller inmatningar, inslagningar och kommandon som måste utföras för att lösa uppgiften, dvs. de faktiska operationer som uppgiftens utförande kräver. Detta mäter gränssnittets effektivitet (utifrån en mänsklig användare), dvs. enbart de kommandon som uppgiften kräver ska utföras, oberoende av felinslag, betänketid etc.

Övriga Operationer innebär skrivfel (felstavningar, oavsiktlig nedtryckning av knapp på tangentbord), slarvfel (felaktigt kommando p.g.a. oavsiktlig nedtryckning av knapp på tangentbord eller datormus), användning av hjälpmanual och felaktig lösning, exempelvis en gissning av hur uppgiften borde lösas som visar sig efter prov inte fungera – och därmed även eventuell borttagning av felaktigt skapade mappar, filer etc. Korrigeringen av felaktigt lösta uppgifter räknas som övriga operationer tills dess att korrigeringen är gjort och användaren är på rätt spår igen. Det vi inte beräknar som övriga operationer är om användande program slutar att fungera eller om testdeltagaren inte förstår uppgiften och frågar testledarna om hjälp, dvs. företeelser som inte är aktivt utförda av användaren själv. Övriga operationerna mäter avgöranden som beror på den mänskliga faktorn, att oberoende

hur effektivt gränssnittet är i sig är en användare benägen till att göra fel som påverkar effektiviteten.

Kombination av de båda utgör den fullständiga uppgiften och är vad som utgör människa-dator-interaktion vid ett verkligt scenario.

De inspelade testerna redigerades i ett filmediteringsprogram för att lättare och mer exakt kunna mäta tiden. De faktiska operationerna och de övriga operationerna klipptes ut och sparades i separata filer. Resterande material klipptes bort och kasserades. Resultatet är därefter sammanställt i tabeller och diagram.

3.4 Metodkritik

I detta kapitel granskar vi vår egen studie och ger förslag till hur studien hade kunnat förbättras.

3.4.1 Granskning av egen enkät

Syftet med enkäten var att ta fram data som kunde användas för att skapa ett användartest. Enkäten gjordes alltså innan en tydlig plan för användartestets uppbyggnad gjorts, och därmed ställdes frågor i enkätundersökningen som i ett senare skede inte blivit relevanta för studiens innehåll. I en granskning av detta sammanhang är det dock lätt att vara efterklok. När enkäten utformades kände vi inte till vad resultatet av användartestet skulle visa. Hade enkäten gjorts om idag hade man däremot kunnat ställa frågor som rörde vad för uppgifter användarna vanligtvis utför och vilka program de använder för att därmed kunna göra ett bättre användartest. Om vi hade vetat vad användarna vanligtvis utför för sysslor vid sina datorer och vilka program de använder hade vi kunnat utforma uppgifter efter dessa sysslor och program och få ett bredare resultat från användartestet.

3.4.2 Granskning av eget användartest

Utförandet av detta användartest kan i efterhand kritiseras. Med facit i hand kan vi se att det hade varit relevant att ha en jämnare spridning av operativsystem, vilket hade möjliggjort för oss att också kunna jämföra användarnas prestationer mellan de olika operativsystemens gränssnitt. Vi ser i efterhand också att uppgifterna är ojämnt uppdelade baserat på typ av uppgift. Vi har tre uppgifter av typen mapphantering, en uppgift om filhantering och en uppgift om sökning och räkning av ord. Med fler uppgifter av samma typ hade vi i större utsträckning kunnat säkerställa tillförlitligheten av resultatet. Uppgifter hade även kunnat skapas med olika storlek och omfång för att mäta skillnaderna på tid mellan gränssnitten för när uppgifternas storlek ökar.

4. Resultat och analys

I detta kapitel analyserar vi enkätsvaren och resultatet av de utvärderade användartesterna transkriberat till tabeller och diagram för att besvara frågeställningarna som presenterats i avsnitt 1.2.

4.1 Resultat av användartest

Resultatet är analyserat och presenteras efter de tre variabler som vi kallar för Faktiska operationer, Övriga operationer och Kombination (fullständig uppgift). Tabell B och C visar medelhastigheterna av alla användares resultat för varje uppgift. Tabell B visar utförandet för GUI och Tabell C visar utförandet för CLI av de faktiska operationerna, övriga operationerna och kombinationen av de båda.

Tabell D visar genomsnittstiden av samtliga uppgifter för faktiska operationer, övriga operationer och kombinationen av de båda i GUI och CLI – angivet i minuter. Tabellen visar även tidsskillnaderna mellan de båda gränssnitten. Tabell E visar samma resultat angivet i procent.

Resultatet är analyserat och presenteras efter de tre variabler som vi kallar för Faktiska operationer, Övriga operationer och Kombination (fullständig uppgift).

Tabell B

Medelvärde GUI	Faktiska operationer	Övriga operationer	Kombination (fullständig uppgift)
Uppgift 1	02:10	00:22	02:41
Uppgift 2	01:46	00:41	02:27
Uppgift 3	01:53	02:08	02:18
Uppgift 4	00:49	00:11	01:00
Uppgift 5	01:12	00:32	01:44
Totalt:	08:12	02:00	10:03

Tabell C

Medelvärde CLI	Faktiska operationer	Övriga operationer	Kombination (fullständig uppgift)
Uppgift 1	02:04	00:51	02:51
Uppgift 2	00:35	00:48	01:27
Uppgift 3	00:56	00:55	01:05
Uppgift 4	00:32	00:04	00:36
Uppgift 5	01:34	00:58	02:32
Totalt:	05:32	03:18	08:26

Tabell D

Total tid i minuter	Faktiska operationer	Övriga operationer	Kombination (fullständiga uppgifter)
GUI	08:12	02:00	10:03
CLI	05:32	03:18	08:26
Skillnad	2:40	1:18	1:37

Tabell E

Total tid i procent	Faktiska operationer	Övriga operationer	Kombination (fullständiga uppgifter)
GUI	79%	21%	100%
CLI	63%	37%	100%
Skillnad	19%	25%	9%

4.2 Analys av användartest

Vi ser i Tabell C att CLI har 1:18 minuter mer övriga operationer än GUI, men trots detta får CLI ändå ett bättre kombinerat resultat.
Vi ser också i de uppgiftsspecifika tabellerna A och B att gränssnitten framförallt presterar olika bra i tid beroende på uppgift. För uppgift 5, Tabell I, presterar GUI bättre medan CLI presterar bättre på uppgift 1, 2, 3 och 4, Tabell D, E, F och G.

Spridningen av användarnas resultat per uppgift i CLI respektive GUI visas i ett s.k. lådagram. Tidsresultaten i dessa diagram anges i minuter. Det vi kan utläsa ur detta diagram är kvartil 1 (Q1), minsta värde, median, max-värde och kvartil 3 (Q3).

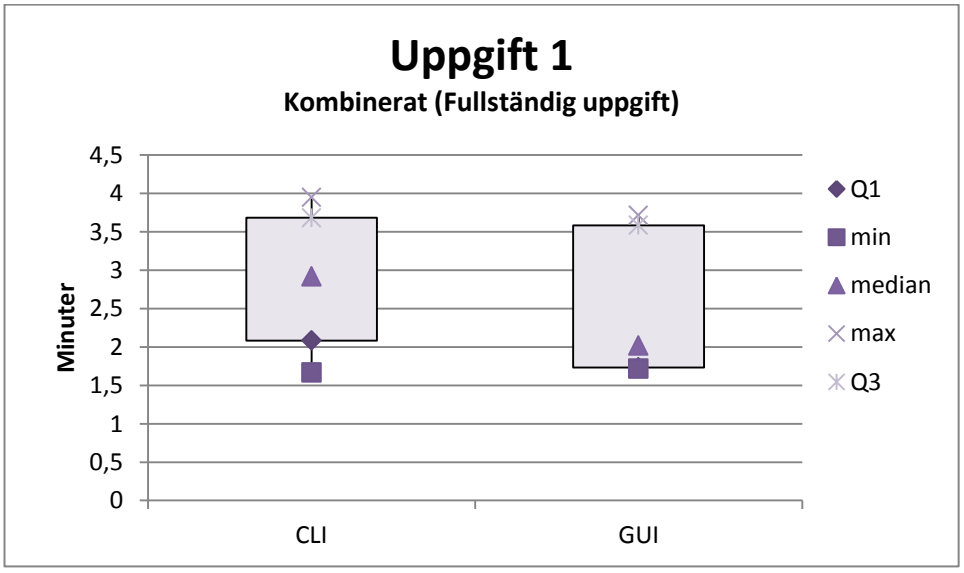
Nedan följer detaljanalys per uppgift. Uppgift 2, 3 och 4 innefattar mapphantering och kategoriseras därför tillsammans under samma rubrik.

4.2.1 Uppgift 1

Uppgift 1 handlar om att skapa fem stycken textfiler i fem motsvarande mappar, zippa dessa, packa upp dem och därefter radera dem igen. GUI-användarna har i denna uppgift inte haft några större problem. All tid för övriga operationer berodde på slarvfel. CLI-användarna fick ingen fördel från möjligheten med skalprogrammering framför GUI-användarna i denna uppgift, eftersom textfil- och mapphanteringen var såpass småskalig så att användarna ansåg att det inte var lönt att använda sig av kontrollsatser eftersom de inte hade vunnit någon tid på det. Istället hanterade de filerna manuellt precis som GUI-användarna gjorde. Följden av detta visar att GUI-användarna och CLI-användarna arbetar principiellt lika effektivt vid småskalig fil- och mapphantering. Vid faktiska operationer ser vi att GUI visar en tendens till att prestera bättre, om än med minimal marginal (se Tabell F, för spridning av tidsresultat för uppgift 1, se Figur 5.)

Tabell F: Tabell för Uppgift 1 över medelhastigheter angivna i minuter.

Uppgift 1	Faktiska operationer	Övriga operationer	Kombination (fullständig uppgift)
GUI	02:10	00:22	02:41
CLI	02:04	00:51	02:51
Skillnad	00:06	00:29	00:10



Figur 5: Spridning av tidsresultat för CLI och GUI för Uppgift 1.

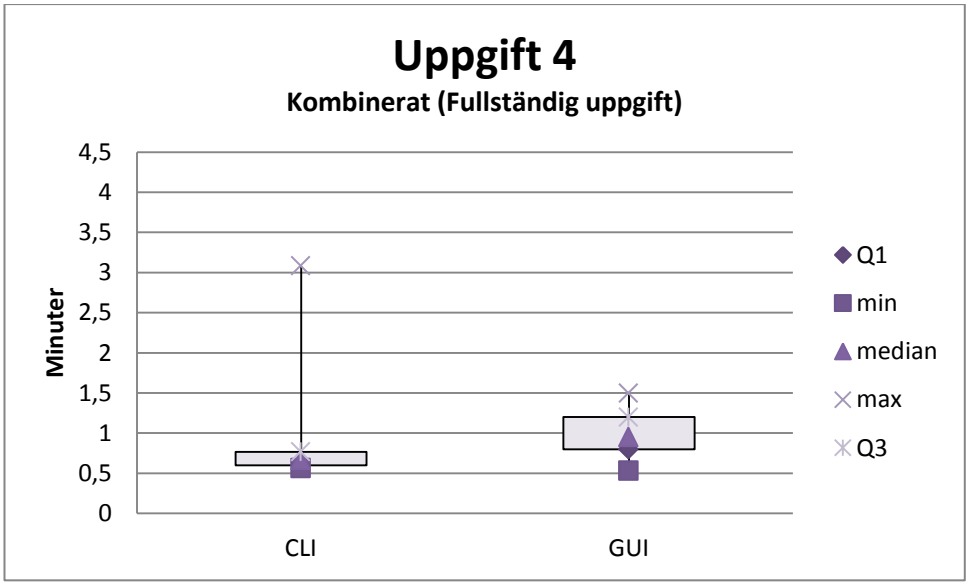
4.2.2 Uppgift 2, 3 och 4

Samtliga av uppgifterna 2, 3 och 4 (se Tabell I, H och G) är olika varianter på uppbyggnad av hierarkier genom skapandet av mappar. I uppgift 4 skapas minst antal mappar och i uppgift 2 skapas flest mappar. Därför presenterar vi resultatet från uppgift 4 först, uppgift 3 därefter och sist uppgift 2.

I uppgift 4 (se Tabell I) handlar det om att skapa sju stycken mappar. Tidsskillnaderna mellan GUI och CLI för de faktiska operationerna är här inte särskilt markanta, även om utförandena i CLI går snabbare.

Tabell G: Tabell för Uppgift 4 över medelhastigheter angivna i minuter.

Uppgift 4	Faktiska operationer	Övriga operationer	Kombination (fullständig uppgift)
GUI	00:49	00:11	01:00
CLI	00:32	00:04	00:36
Skillnad	00:17	00:07	00:24

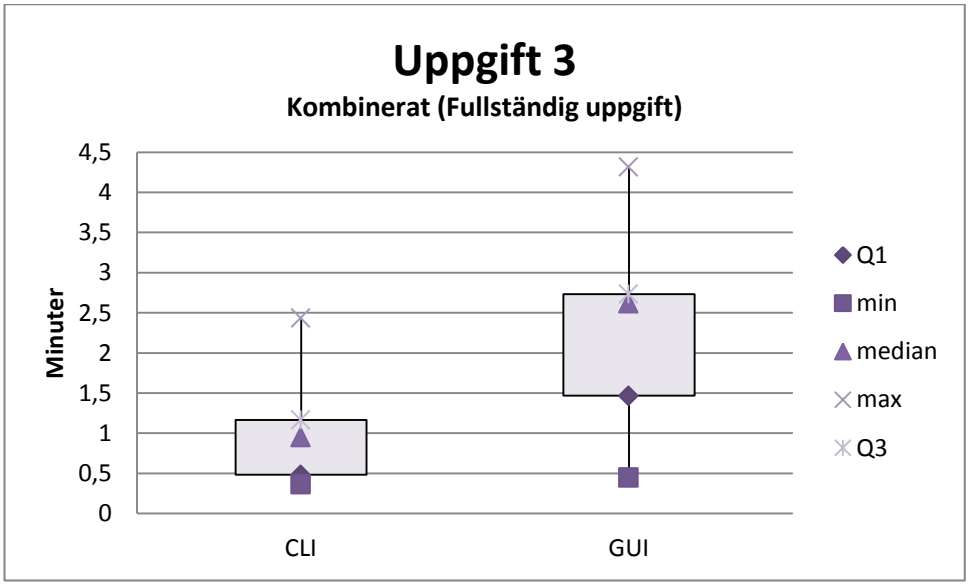


Figur 8: Spridning av tidsresultat för CLI och GUI för Uppgift 4.

Men om vi tittar på uppgift 3 (se Tabell H) som handlar om att skapa 15 stycken mappar ser vi att skillnaden på tiden för de faktiska operationerna nu har ökat. Detsamma gäller för den fullständiga uppgiften där fel och faktiska utföranden kombineras.

Tabell H: Tabell för Uppgift 3 över medelhastigheter angivna i minuter.

Uppgift 3	Faktiska operationer	Övriga operationer	Kombination (fullständig uppgift)
GUI	01:53	02:08	02:18
CLI	00:56	00:55	01:05
Skillnad	00:57	01:13	01:13

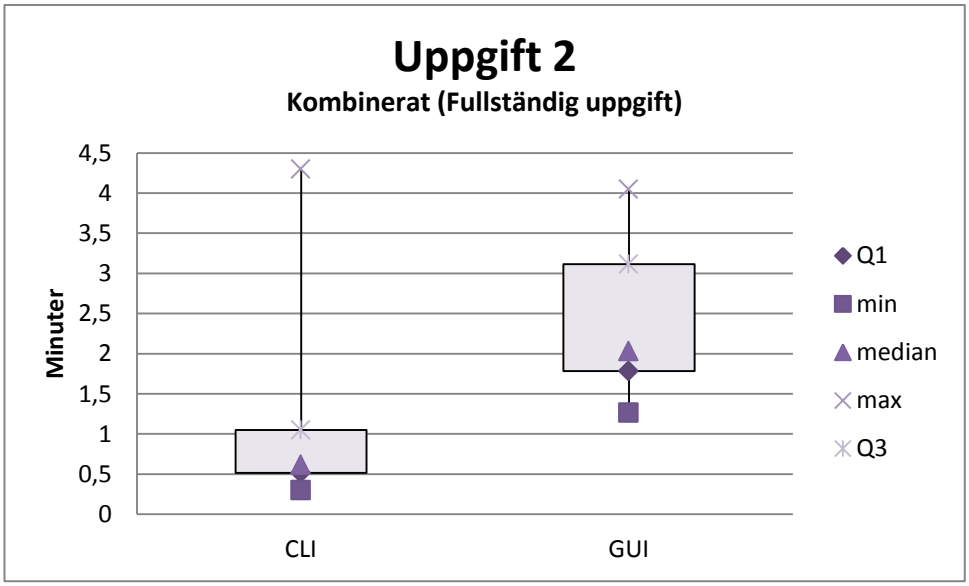


Figur 7: Spridning av tidsresultat för CLI och GUI för Uppgift 3.

Tittar vi slutligen på uppgift 2 (se Tabell I) som handlar om att skapa 26 stycken mappar, ser vi återigen en ökning i tidsskillnaderna på faktiska operationer och fullständiga uppgifter. För GUI ökar tiden succesivt ju fler mappar som måste skapas medan CLI inte verkar påverkas av antalet mappar.

Tabell I: Tabell för Uppgift 2 över medelhastigheter angivna i minuter.

Uppgift 2	Faktiska operationer	Övriga operationer	Kombination (fullständig uppgift)
GUI	01:46	00:41	02:27
CLI	00:35	00:48	01:27
Skillnad	01:11	00:07	01:00



Figur 6: Spridning av tidsresultat för CLI och GUI för Uppgift 2.

Följden av detta är att CLI till skillnad från GUI verkar lämpa sig bättre ju större uppgifterna blir och ju större mängd filer och mappar som skall skapas. GUI-användarna arbetar mer manuellt och skapar varje mapp individuellt var för sig till skillnad från i ett CLI där användarna har möjligheten att lösa uppgiften med selektioner och iterationer i kontrollsatser. Ska du exempelvis skapa mapparna 1-500 kan vi lätt föreställa oss problematiken för en GUI-användare. En CLI-användare däremot hade kunnat lösa denna uppgift i ett enda kommando. Flera svarspersoner instämmer även i detta i enkäten och nämner att de upplever att GUI fungerar bra i småskalig fil- och mapphantering men att CLI fungerar bättre för större och mer generell fil- och mapphantering.

Vi ser i uppgift 3 (se Tabell H) att CLI har en längre utförandetid jämfört med dess tid i uppgift 2 och 4 (se Tabell I och G). I denna uppgift skulle mapparna ligga i varandra, dvs. mapp 15 ligger i mapp 14, mapp 14 ligger i mapp 13 osv. Detta gjorde uppgiften aningen mer svårlöst utöver situationen att enbart skapa mapparna, vilket kan förklara att denna uppgift har längre utförandetid än uppgift 2 och 4. (För spridning av tidsresultat för uppgift 2, 3 och 4, se Figur 6, 7 och 8).

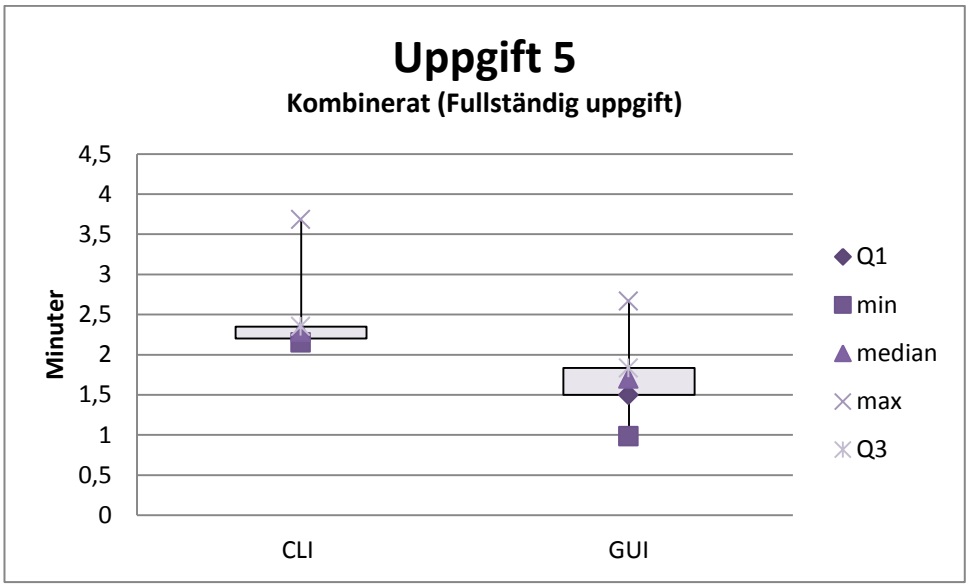
4.2.3 Uppgift 5

Uppgift 5 handlade om att leta reda på specifika meningar i en chattlogg, räkna antalet ord som skrivits i ett specifikt inlägg och räkna antalet gånger en specifik person skrivit ett inlägg. Frustration uppstod för GUI-användarna i denna uppgift då de skulle räkna specifika ord i en stor datamängd. 4 utav 5 GUI-användare framförde spontant under testet att de hade velat lösa denna uppgift med ett CLI istället. Dock ser vi att CLI-användarna hade lite svårare för att lösa uppgiften.

Några GUI-användare tog hjälp av räkningsverktyget i Microsoft Office Word för att lösa uppgiften. En användare använde sig av en ordräknare på internet. Alla CLI-användare använde sig av kommandot wc (word count). CLI-användarna har i denna uppgiften precis som i uppgift 1 inte vunnit några fördelar från skalprogrammeringen i sitt gränssnitt. Genomförandet för GUI- och CLI-användarna var desamma och båda parter sökte i chattloggen på samma premisser. CLI-användarna har dessutom en större tidsmarginal för övriga operationer än GUI-användarna. (För spridning av tidsresultat för uppgift 5, se Figur 9).

Tabell J: Tabell för Uppgift 5 över medelhastigheter angivna i minuter.

Uppgift 5	Faktiska operationer	Övriga operationer	Kombination (fullständig uppgift)
GUI	01:12	00:32	01:44
CLI	01:34	00:58	02:32
Skillnad	00:22	00:26	00:48

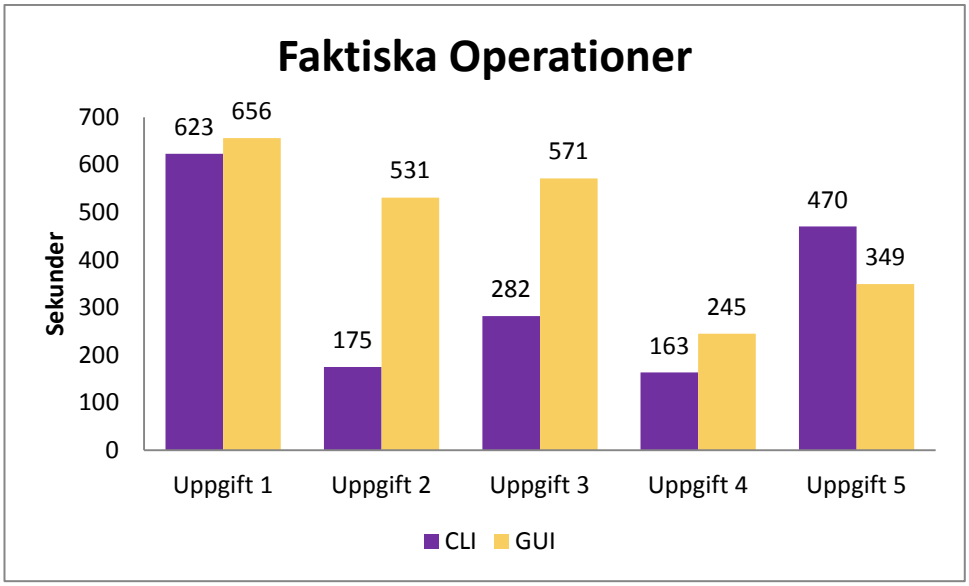


Figur 9: Spridning av tidsresultat för CLI och GUI för Uppgift 5.

4.3 Analys av gränssnitt

I kombinationen av faktiska operationer och övriga operationer är CLI tidsmässigt effektivast. Men gränssnittens prestationer skiljer sig om vi tittar på de faktiska operationerna och de övriga operationerna separat. CLI har bättre resultat för faktiska operationer men ändå avsevärt mer tid för övriga operationer jämfört med GUI.

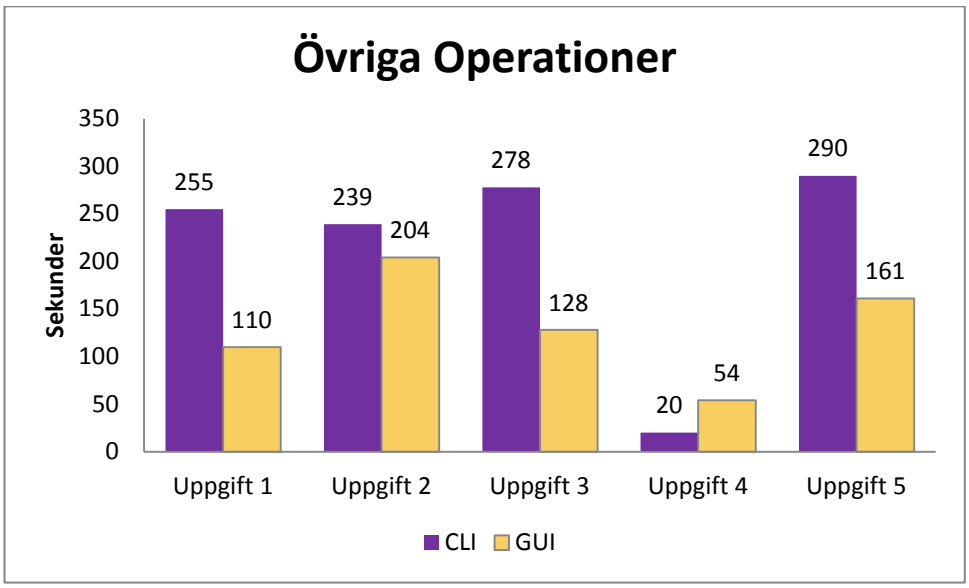
Resultatet i Figur 10 visar vilket gränssnitts faktiska operationer som är tidsmässigt effektivast. Vi ser att CLI presterar bäst i samtliga uppgifter utom i Uppgift 5.



Figur 10: Stapeldiagram över Faktiska Operationer

Beräknar vi gränssnittens utföranden ser vi att CLI kräver färre steg. För att exempelvis skapa en hierarki med mappar från a-z kan man med ett CLI skapa detta med ett enda steg i ett kommando med hjälp av kontrollsatser, medan man i ett GUI får skapa var och en mapp i 26 steg. Detta bekräftar att CLI är tidsmässigt effektivare än GUI på faktiska operationer.

Analyserar vi däremot gränssnittens prestationer utifrån övriga operationer har GUI en betydligt bättre tidresultat (se Figur 11).

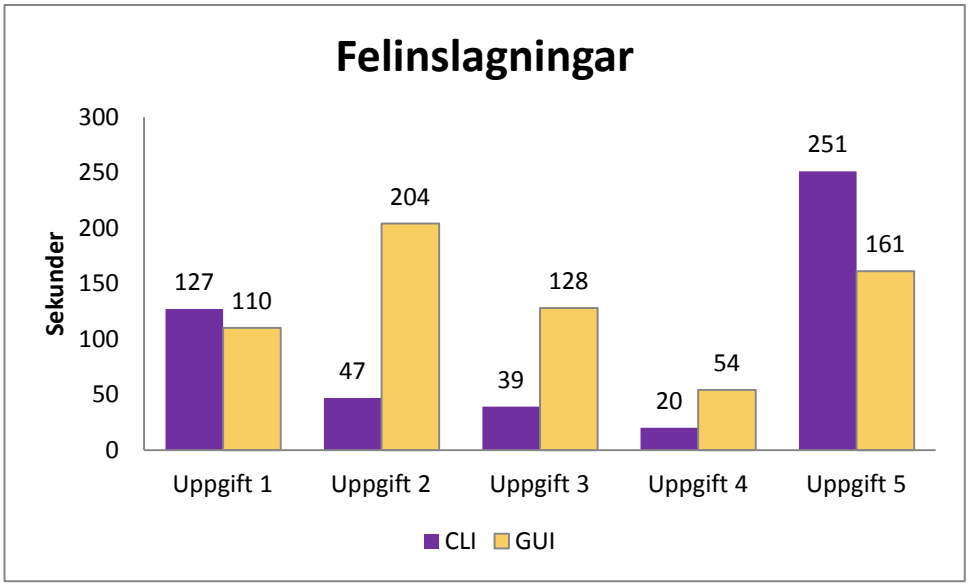


Figur 11: Stapeldiagram för Övriga Operationer

De variabler som spelar in i den beräknade tiden för övriga operationer beror övergripande på den mänskliga faktorn. Denna faktor grundas bland annat i mentala modeller som vi inte tänker analysera i detalj i denna studie (se Avgränsningar, avsnitt 1.4). Däremot kan vi analysera hur gränssnitten medbringar felinslagningar på grund av dess design och utformning.

I Figur 11 ser vi att de övriga operationerna i CLI är överrepresenterade. Vår definition av övriga operationer beräknar vi med att bland annat trycka på fel tangent och när man utför fel kommandon. Eftersom CLI är textbaserat och manövreras via ett tangentbord, gentemot GUI's klicka- och *drag-and-drop*-gränssnitt via datormus, är sannolikheten större att man gör felinslagningar baserat på antal komponenter som möjliggör felinslagningar. Ett tangentbord har ca 85-100 tangenter mot GUI's 1-3 knappar på datormusen vilket gör CLI mer benäget till felinslagningar. Därmed inte sagt att felinslagningarna i ett CLI tar upp mer tid än felinslagningarna i ett GUI. Att göra slarvfel i ett CLI genom att omedvetet trycka på fel bokstavstangent är det fel vi i huvudsak noterat och är tidsmässigt ett snabbt problem att lösa. I GUI har vi sett att användaren ex. missar sitt mål i drop-down-menyer och klickar sig in i fel mappar vilket tidsmässigt tar längre tid att åtgärda i jämförelse. GUI-användarna får ofta korrigera fel med att upprepa hela operationen på nytt, exempelvis gå in i drop-down-menyn igen efter att ha siktat utanför menyn med muspekaren.

Beräknar vi enbart felinslagningarna för sig ser vi att CLI generellt får ett bättre resultat än GUI per uppgift (se Figur 12).



Figur 12: Stapeldiagram för Felinslagningar

Skillnaderna i Figur 11 och Figur 12 visar att CLI-användarna lägger mycket av sin tid i Figur 11 på användning av manual. Att använda sig av hjälpmanualen är dock ofta en nödvändighet för en CLI-användare oavsett kunnighet p.g.a. antalet kommandon som användaren inte har möjlighet att lära in utantill. Detta är alltså en faktor som påverkar resultatets helhet för CLI.

I Figur 11 och Figur 12 beräknas även användarens skrivhastighet till CLI. Dessutom kan vi se ifrån användartestet att flera CLI-användare skriver in långa kommandon, kommer på att de är inkorrekta och raderar dem igen. Detta är i sig tidsödande oberoende skrivhastighet och bidrar till den längre tiden för övriga operationer och felinslagningar.

Användning av hjälpmanual är visserligen en nödvändighet för CLI, men då uppgifter upprepas och görs på daglig basis kan vi föreställa oss att de kommandon som krävs att utföras lär sig användaren så småningom utantill. Detta bidrar till att hjälpmanualen succesivt behöver användas i betydligt mindre utsträckning vid repetativa uppgifter, vilket gör Figur 12 mer relevant ju mer användaren upprepar en uppgift.

4.4 Resultat och analys av enkät

3 av 11 personer anser att gränssnitten lämpar sig bäst beroende på uppgifter och användningsområden. Övriga personer var konsekventa gällande preferens av gränssnitt. 4 av 11 föredrog att orientera sig i sitt operativsystem med ett GUI, 4 av 11 med ett CLI. Svarspersonerna nämnde att ett CLI fungerar bäst till versionshantering, enklare installationer av program, generell fil- och mapphantering och utveckling. De anser att GUI naturligtvis fungerar bäst till att surfa på web med och vid bild- och filmredigering, men även vid installation av program med mycket konfigurationer och vid småskalig filhantering.

5 av 11 svarspersoner tycker att dagens GUI i operativsystem kan vara hämmande för expertanvändare. En svarsperson tillägger att det är lättare att automatisera och anpassa ett CLI-baserat arbetsflöde än ett GUI-baserat. 4 av 11 anser inte att GUI är hämmande. 2 personer anser att det beror på vad som skall åstadkommas i ett GUI där man nämnde att GUI't utvecklas i en positiv riktning och blir bättre och bättre med tiden.

Vi frågade vilket gränssnitt av CLI och GUI som var mest bekvämt och effektivt att använda vid installation av programvara. 4 svarspersoner föredrar här att använda sig av ett GUI. En person motiverade detta val med att han aldrig använt sig av ett CLI för att installera programvara, men övriga föredrog GUI för att det var lättare. Restrerande svarade att det på något sätt beror på. Man skiljer på bekväm och effektiv och svarar att GUI kan uppfattas som mest bekvämt och ett CLI mest effektivt. Om ett CLI har installationsanvisningar eller om det är mindre installationer av program föredrar man CLI eftersom det anses snabbare och smidigare. Om man arbetar i ett operativsystem som har stöd för CLI-baserad programinstallation, ex. Linuxdistributioners pakethantering eller Mac OS X's Homebrew där man enbart behöver skriva in ett kommando (apt-get) för att hämta hem programmet, anses det vara överlägset. Vid större och mer avancerade installationer med mycket konfigurationer anses GUI ge en bättre överblick och är därmed att föredra.

De operativsystem som svarspersonerna använder är framförallt Windows och Mac OS X. Linux operativsystem används också men i mindre utsträckning. Vid användandet av Windows används ett CLI i mindre utsträckning än vid Mac OS X och Linux-operativsystem.

För mer detaljerad enkätsammanfattning, se bilaga 2 och 3.

4.5 Sammanfattning

Användartesterna har möjliggjort för oss att åskådligöra faktorer som annars hade varit svåra att få fram med enbart intervju- eller enkätdata från testpersonerna. Resultatet av användartestet hjälper oss att svara på flera essentiella frågor. Vad innebär detta resultatet för användarna? Och vad innebär det för hur man designar ett system? Användandet och tillhandahållandet av grafiska användargränssnitt (GUI) är idag överlägset. Vi har även konstaterat att stor fokus läggs på nybörjaranvändare och att utvecklingen av system därför går i riktningen att göra gränssnitten så lätta som möjligt att använda. Däremot ser vi i detta användartest att GUI i vissa sammanhang inte håller samma standard som CLI när det gäller effektivitet. GUI-användarna säger dessutom att de vid lösandet av vissa uppgifter hellre hade använt ett CLI. Effektiviteten betyder mycket för expertanvändarna.

I Tabell D ser vi att CLI har 25% mer övriga operationer i tid än GUI. Men trots detta är CLI ändå snabbare än ett GUI i uppgiftens helhet med 9%. Detta bevisar CLI's effektivitet för faktiska operationer. Figur 11 visar däremot att ett GUI är tidsmässigt effektivare för hantering av övriga operationer. Att GUI får mindre tid i övriga operationer skulle dessutom kunna vara ett bevis på konstaterandet att GUI är lättare att använda.

Baserat på användartestets uppgifter i detalj ser vi att GUI lämpar sig bättre vid småskalig fil- och mapphantering och CLI vid storskalig fil- och mapphantering. CLI presterar alltså inte bättre vid samma manuella och småskaliga arbete som man utför i ett GUI eftersom kontrollsatser tidsmässigt inte lönar sig. GUI och CLI har båda sina fördelar, men att kombinera de båda när man designar ett system kan lösa många problem för användarna, oavsett kunnighet och erfarenhet.

5. Diskussion

GUI är idag det dominerande gränssnittet för designen av system. Utvecklingen för användargränssnitt är främst inriktad på lätthet att använda och studier skrivna om ämnet bekräftar i många fall väsentligheten att utforma gränssnitt baserat på lätthet för nybörjaranvändare. Och precis som Nielsen poängterar [8] bör man ha i åtanke att ingen användare blir expertanvändare utan att ha varit nybörjare först. Det är i hög grad viktigt att utveckla gränssnitt utifrån användarvänlighet och lätthet. I våra empiriska användartester ser vi däremot att GUI i genomsnitt inte mäter sig med CLI då det gäller effektivitet, det som expertanvändare anser vara något av det viktigaste [5]. Både Whiteside [6] och Chen och Zhang [7] bekräftar i sina studier att expertanvändarna presterar bäst i ett CLI. Krisler och Alterman [4] nämner att det krävs en förståelse för systemets underliggande struktur för att kunna använda det effektivt. Takayama och Kandogan [5] visar också att expertanvändarna anser CLI vara det snabbaste och mest förtroendeingivna gränssnittet att arbeta i. Men är ett CLI alltid snabbast?

Oavsett om CLI ger ett bättre resultat i genomsnitt ser vi att resultatet för effektiviteten i högsta grad är beroende på användningsområdet och framförallt dess omfång. Resultatet av våra användartester visar att ju mer uppgifterna ökar i storlek och omfång desto effektivare blir CLI i jämförelse med GUI. Vi kan även konstatera utifrån enkäterna att expertanvändare anser att CLI lämpar sig bättre vid versionshantering, enklare installationer av program och utveckling. GUI i sin tur fungerar bäst vid installationer av program med konfigurationer, bild- och filmredigering och används för att surfa på internet med (se bilaga 2) . Detta är svaret på frågeställningen: ”*Vilka uppgifter är bäst lämpade för CLI respektive GUI ?*”.

Utifrån detaljanalysen för varje uppgift är CLI tidsmässigt effektivast i fyra uppgifter av fem. Samtliga innefattar mapphantering. En uppgift är GUI tidsmässigt effektivare i. Detta är svaret på frågeställningen ”*Vilket gränssnitt av GUI och CLI är tidsmässigt effektivast?*”.

Under användartestets gång kunde vi utläsa att flera faktorer i gränssnittens styrkor och svagheter påverkade effektiviteten. CLI kräver i många fall färre steg i utförandet av kommandon tack vare kontrollsatser än GUI och är därför tidsmässigt effektivare än GUI på faktiska operationer. GUI har dock betydligt mindre beräknad tid för övriga operationer än CLI, vilket gör GUI tidsmässigt effektivare vid utförandet och hantering av övriga uppgifter. En avgörande faktor för detta är att ett CLI har ett stort antal kommandon som användarna inte har möjlighet att lära sig utantill. Användarna behöver titta i manualen vilket bidrar till ett långsammare utförande av uppgift. Om vi däremot enbart tittar på felinslagningarna för vardera gränssnitt ser vi att resultatet för CLI blir bättre. Vi kan därmed dra slutsatsen att ju mer man lär sig kommandona för utförandet av en uppgift och ju mindre tid man behöver lägga på att använda manualen, desto mer effektivt blir CLI än GUI. Detta är svaret på frågeställningen ”*Vilka faktorer i gränssnitten GUI och CLI’s utformning påverkar tidsmässigt resultatet av effektiviteten*?”.

5.1 Vidarestudier

Vad vi kan konstatera är att val av gränssnitt för vardagliga uppgifter på persondator beror på uppgift och uppgiftens omfång, men givetvis inte uteslutande. Användarens mentala modell är en faktor som spelar in i varför användaren väljer att arbeta i det gränssnitt hon valt. En vidarestudie gällande detta hade därmed kunnat validera användarens anledning till val av gränssnitt. Av vilken anledning använder man respektive gränssnitt? Hur ser inlärningsprocessen ut? Vi uppmuntrar även till vidare forskning för att säkerställa resultatet från denna studie med en större grupp testdeltagare och med uppgifter som mäter gränssnittens effektivitet utifrån uppgifternas storlek, omfång och datamängd. Ytterligare forskning som hade varit intressant att studera är kombinationen av GUI och CLI, en vidarestudie för att se hur vardera gränssnitts fördelar kan kombineras – kommandoradsgränssnittets effektivitet och det grafiska gränssnittets lätthet.

6. Slutsats

Resultatet av denna studie pekar på att vår hypotes om att CLI är effektivast stämmer till viss del, men inte fullt ut. CLI är inte effektivare i alla sammanhang. CLI presterar i flera fall bättre än GUI tack vare möjligheten till skalprogrammering, men inte vid alla typer av upgifter. GUI presterar bättre i 2 uppgifter av 5, dock inte med avsevärt stora skillnader. Därmed ser vi i resultatet dessutom att skillnaderna mellan gränssnittens prestationer inte är så väldigt stora som hypotesen antog förutom vid hanterandet av stora mapp- och filhierarkier.

Där GUI får sämre resultat kan vi dock märka GUI's brister och var någonstans som grafiska system skulle behöva effektiviseras. De operativsystems GUI som vi testat har inga smarta lösningar för att bl.a. skapa, bytan namn på, flytta och kopiera mappar och filer i storskaliga hierarkier. Att integrera textbaserade lösningar till systemet i kombination med GUI hade kunnat lösa detta problem. Om vi tar till vara på dessa observationer skulle vi i framtiden kunna ta fram ett grafiskt operativsystem som är ännu mer effektivt i sin mapphantering med en grafisk motsvarighet till CLI's kontrollsatser.

Litteraturlista

Referenser för ordlistan

[i] *Bash Reference Manual*
(Elektronisk)
http://www.gnu.org/software/bash/manual/bashref.html#What-is-Bash_003f
(2012-04-29)

[ii] The free dictionary *CLI*
(Elektronisk)
http://encyclopedia.thefreedictionary.com/cli
(2012-04-29)

[iii] The free dictionary *DOS*
(Elektronisk)
http://encyclopedia.thefreedictionary.com/DOS
(2012-04-29)

[iv] The free dictionary *GNU*
(Elektronisk)
http://encyclopedia2.thefreedictionary.com/GNU
(2012-04-29)

[v] The free dictionary *Kernel*
(Elektronisk)
http://encyclopedia.thefreedictionary.com/kernel
(2012-04-29)

[vi]The free dictionary *GNU*
(Elektronisk)
http://encyclopedia2.thefreedictionary.com/GNU
(2012-04-29)

[viii] ToggleOn: *Grundläggande programmeringskoncept förklarade: Iteration och Rekursion*
(Elektronisk)
http://toggleon.wordpress.com/2010/08/29/grundlaggande-programmeringskoncept-forklarade-iteration-och-rekursion/#vad_ar_kontrollsatser_och_kontrollstrukturer
(2012-04-29)

[ix] The free dictionary *Command/Prompt*
(Elektronisk)
http://encyclopedia2.thefreedictionary.com/command+prompt
(2012-04-29)

[x] The Free Dictionary "*GNU/Linux",*
(Elektronisk)
http://encyclopedia2.thefreedictionary.com/GNU%2FLinux
(2012-04-29)

[xi] The Free Dictionary "*Linux Distribution"*
(Elektronisk)
http://encyclopedia2.thefreedictionary.com/Linux+distribution
(2012-04-29)

[xviii] – Distrowatch.com "*Top Ten Distributions"*
(Elektronisk)
http://distrowatch.com/dwres.php?resource=major
(2012-04-29)

[xviii] The Free Dictionary "*Linux Kernel*"
(Elektronisk)
http://encyclopedia.thefreedictionary.com/Linux+kernel
(2012-04-29)

[xiii] The Free Dictionary, *"Shell Script"*
*(*Elektronisk)
http://encyclopedia2.thefreedictionary.com/shell+scripting
(2012-04-29)

[xiv] The Free Dictionary, *"Shell*",
(Elektronisk)
http://encyclopedia2.thefreedictionary.com/Shell
(2012-04-29)

[xv] Johansson, Per, Uppsala Universitet, "*Skalprogrammering och reguljära uttryck*"
(Elektronisk)
http://stp.ling.uu.se/~perjo/gd03/gd_f9.html
(2012-04-29)

[xvi] The Free Dictionary, "*Terminal"*
(Elektronisk)
http://encyclopedia.thefreedictionary.com/Terminal+(application)
(2012-04-29)

[xvii] The Free Dictionary, "*Linux"*
*(*Elektronisk)
http://encyclopedia.thefreedictionary.com/unix
(2012-04-29)

Referenser för uppsatsen

[1] SCB (Statistiska Centralbyrån), *Privatpersoners användning av datorer och Internet 2009.* ISSN 1654-7624

[2] Krug, Steve, *"Don't Make Me Think! A common sense approach to web usability"*, Second edition, Berkeley, USA, 2006.

[3] The Linux Information Project (LINFO), "*Command Interpreter*", Elektronisk – Tillgänglig på URL:
http://www.linfo.org/gui.html
(2012-04-22)

[4] Krisler, Brian och Alterman, Richard, "*Training towards mastery: overcoming the active user paradox*", Proceeding – NordiCHI '08, "Proceedings of the 5th Nordic conference on Human computer interaction: Building Bridges", sida 239-248, New York, USA, 2008.

[5] Takayama, Leila och Kandogan, Eser, "*Trust as an underlying factor of system administrator interface choice*", Proceeding – CHI EA '06, "CHI '06 extended abstracts on Human factors in computing systems", sida 1391-1396, New York, USA, 2006.

[6] Whiteside, John; Jones, Sandra; Levy, S. Paula och Wixon, Dennis, "*User performance with command, menu, and iconic interfaces",* Proceeding – CHI '85, "Proceedings of the SIGCHI conference on Human factors in computing systems", sida 185-191, New York, USA, 1985.

[7] Chen, Jung-Wei och Zhang, Jiajie, *"Comparing Text-based and Graphic User Interfaces for Novice and Expert Users"*, Proceeding – American Medical Informatics Association, "AMIA Annual Symposium Proceeding", Houston TX, USA, 2007.

[8] Nielsen, Jakob, "*Testing Expert Users*". Elektronisk – Tillgänglig på URL:
http://www.useit.com/alertbox/experienced-users.html
(2012-04-29)

[9] SearchWinIT, TechTarget, "*Command Interpreter*". Elektronisk – Tillgänglig på URL:
http://searchwinit.techtarget.com/definition/command-interpreter
(2012-04-22)

[10] Gite, Vivek, "*Linux/BSD Command Line Interface – Ideal For Blind Users*". Elektronisk – Tillgänglig på URL:
http://www.cyberciti.biz/tips/linux-command-line-interface-ideal-for-blind-users.html
(2012-06-10)

[11] "*An A-Z Index of the Bash command line for Linux",* Elektronisk – Tillgänglig på URL:
http://ss64.com/bash/
(2012-04-05)

[12] Finnie, Scot, LINUX CLUES *"Command-Line Aliases*", Elektronisk – Tillgänglig på URL:
http://www.linuxclues.com/articles/18.htm
(2012-04-28)

[13] Cisco.com, ”Introducing the CLI”, Elektronisk – Tillgänglig på URL:
http://www.cisco.com/en/US/docs/security/ips/7.0/command/reference/crIntro.pdf
(2012-04-05)

[14] Burgess, Andrew, NetTuts+, “*How to Customize the Command Prompt*”, Elektronisk – Tillgänglig på URL:
http://net.tutsplus.com/tutorials/other/how-to-customize-the-command-prompt/
(2012-04-05)

[15] Wilk, Jacub, Debian Wiki, *”CommandLine”,* Elektronisk – Tillgänglig på URL:
http://wiki.debian.org/CommandLine
(2012-04-05)

[16] The Linux Information Project *(LINFO), “Change Shell”,* Elektronisk – Tillgänglig på URL:
http://www.linfo.org/change_shell.html (2012-04-05)

[17] Cygwin.com, *”This is the home of the Cygwin Project”,* Elektronisk – Tillgänglig på URL:
http://www.cygwin.com/
(2012-04-05)

[18] macblog.se ”*Terminal - Shell - bash - OSX – Tiger”,* Elektronisk) – Tillgänglig på URL:
http://www.macblog.se/terminal-shell-bash-osx-tiger-1
(2012-04-05)

[19] , Orr, Giles, ”*Bash Promt HOWTO*”, Elektronisk – Tillgänglig på URL:
http://www.faqs.org/docs/Linux-HOWTO/Bash-Prompt-HOWTO.html#AEN29
(2012-04-05)

[20] MathWorks, “*MATLAB, The Language of Technical Computing*”, Elektronisk – Tillgänglig på URL: http://www.mathworks.se/products/matlab/
(2012-04-05)

[21] Ilijason, Robert, MacWorld, *”Terminal med OS X-smak*”.

[22] Nasarek, Marcus, “*SHELL GAMES - Comparing Bash with the Window Vista Shell*”, Elektronisk – Tillgänglig på URL:
http://w3.linux-magazine.com/issue/78/Bash_vs._Vista_PowerShell.pdf (2012-04-05)

[23] The Free Dictionary, “*WYSIWYG”*, Elektronisk – Tillgänglig på URL:
http://www.thefreedictionary.com/wysiwyg
(2012-04-28)

[24] Umeå Universitet, ”*Grafiska Användargränssnittet*”, Elektronisk – Tillgänglig på URL:
http://www8.cs.umu.se/kurser/TDBB11/HT00/src/gui.html
(2012-04-28)

[25] Stockhaus, Anna, “*Metaforens innebörd och betydelse för användargränssnitt*”, Stockholm, Sverige, 2003.

[26] Reimer, Jeremy, Arstechnica, "*A History of the GUI*", Elektronisk – Tillgänglig på URL: http://arstechnica.com/old/content/2005/05/gui.ars/4
(2012-04-05)

[27] Ruane, M. Janet, "*A och O i samhällsvetenskaplig forskning"*, Lund, Sverige, 2006.

[28] Holme, Magne, Idar och Solvang, Krohn, Bernt, "*Foskningsmetodik: Om kvalitativa och kvantitativa metoder",* Lund, Sverige, 1991.

[29] Albert, Bill och Tullis, Tom, "*Measuring the user experience*", USA, 2008.

[30] Nielsen, Jacob,*"Usability 101: Introduction to Usability",* Elektronisk – Tillgänglig på URL: http://www.useit.com/alertbox/20030825.html
(2012-05-06)

[31] Lindenberg, J., Neerincx, M. A., "*A Generic Usability Test Environment for Web-based Services",* Soesterberg, Nederländerna, 1999.

[32] Nielsen, Jacob, "*Why you only need to test with 5 users",* Elektronisk – Tillgänglig på URL: http://www.useit.com/alertbox/20000319.html
(2012-05-05)

Bilaga 1 Enkät

Enkätundersökning

I vår studie definierar vi GUI, Graphical User Interface, för operativsystem som det visuella pek- och klick-gränssnittet, där du navigerar dig med hjälp av framförallt datormus bland exempelvis mappar (dvs. bilder som utgör metaforer till objekt i verkligheten).
Med TUI (eller CLI), textbaserat användargränssnitt, menar vi här en terminal där du skriver kommandon med exempelvis olika Unix-skal.

1. Föredrar du att orientera dig i ditt operativsystem med TUI framför GUI?

   Motivera:

2. Anser du att dagens GUI i OS kan vara hämmande för avancerade användare?

   Motivera ditt svar:

3. Vilket/vilka operativsystem använder du?
4. Vilket beslut låg bakom ditt val av operativsystem?
5. Vid installation av ny programvara –Av TUI och GUI, vilket anser du mest bekvämt och vilket ser du som mest effektivt att använda?
6. Hur navigerar du dig vanligtvis i din dator?
7. I hur stor utsträckning använder du dig av kortkommandon?
   Motivera så omfattande du kan:

Bilaga 2 Sammanställning enkät

FRÅGA 1 - Föredrar du att orientera dig i ditt operativsystem med TUI framför GUI?

4 av 11 föredrog GUI

D		GUI	”Jag föredrar GUI där alternativet TUI också finns dvs det erbjuds i Windows, Mac OS X och unix/linux”
F		GUI	”För mig känns det självklart att ha ett GUI-baserat OS”
I		GUI	”Jag föredrar GUI. Detta pga att jag förespråkar mer det visuella. Vad jag ser, är vad jag får.”
J		GUI	”Jag föredrar GUI framför CLI. Motivering är att det för min känns enklare och mer intuitivt.”

4 av 11 föredrog CLI

B	CLI	”Enklare och oftast snabbare.”
C	CLI	”Det känns som att allting går mycket snabbare när man slipper använda musen.”
E	CLI	”Terminal om man behöver mata in sökvägar (antar då att det finns path/file-completion) som man vet ungefär hur de ska se ut”
H	CLI	”Det är för mig är mer effektivt med CLI.”

3 av 11 föredrog att kombinera de båda.

a	BÅDA	”Beror på vilket operativsystem. *Nix baserade (OSX, Linux etc.) föredrar jag TUI över GUI. I Windows använder jag GUI, antagligen av vana då det var det jag började med.”
G	BÅDA	”Det beror på vad jag ska göra.”
K	BÅDA	”Mycket beroende på vilken platform jag är på.”

FRÅGA 2 Anser du att dagens GUI i OS kan vara hämmande för avancerade användare?

5 av 11 ansåg JA

B	Ja
E	Ja
G	Ja
J	Ja
K	Ja

B: Ja

E: . Helt klart

G: Ja, det är lättare att automatisera och anpassa ett TUI-baserat arbetsflöde, än ett via GUI.

J: Ja.

K: Absolut.

4 av 11 ansåg NEJ

A	Nej
D	Nej
F	Nej
I	Nej

A: Nej det tror jag inte.

D: Nej.

F: Nej

I: Det tror jag inte.

2 av 11 ansåg att det berodde på.

H	Beror på
C	Beror på

H: Beroende på vad man gör.

C: Jag tycker det blir bättre och bättre med GUI.

FRÅGA 3 – Vilket/vilka operativsystem använder du?

A: OSX och Windows
B: Windows (7 & Vista), OSX, Linux
C: Windows 7, Ubuntu och OSX
D: Windows 7, Windows XP, Mac OS X,
E: OS X, Win XP ochWin 7
F: Windows 7 och Mac OS X Lion.
G: Windows, Linux och Mac OS X
H: GNU/Linux och Windows (XP)
I: Snowlion
J: OSX och Windows 7.
K: Mac OS X och Win 7.

FRÅGA 4 Vilket beslut låg bakom ditt val av operativsystem?

A: Inte ett medvetet beslut.
B: Kommer med, jobbet kräver det, bäst anpassning för mina system
C: Tvungen till, har en liten server, Jobbet kräver det.
Det är lite rätt verktyg för rätt jobb på något sätt som ligger bakom beslutet.
D: Mest använt
E: Självvalt, krav från arbete
F: Vill köra det senaste.
G: Jobbet kräver det, hemma baseras på övriga familjemedlemmar
H: Nyfikenhet.
I: Jag vill ha något som fungerar och behöver fungera.
J: Krav från jobb, hansyn till prestanda,
K: Det som kom med datorn + arbetasgivare

FRÅGA 5 Vid installation av ny programvara –Av TUI och GUI, vilket anser du mest bekvämt och vilket ser du som mest effektivt att använda?

	TUI	GUI	BÅDA
A:			x
B:			x
C:			x
D:		x	
E:			x
F		x	
G:			x
H:			x
I:		x	
J:		x	
K:			x
Totalt:	0	4	7

FRÅGA 6 - Hur navigerar du dig vanligtvis i din dator?

Se rådata. Då frågan ställts på ett vis som gjorde att den kunde tolkas både hur man navigerade sig inuti sin dator och hur man navigerade sig utanför sin dator blir resultatet svårt att redovisa. Vi såg inte frågan som relevant till vår studie och strök denna fråga.

FRÅGA 7 hur stor utsträckning använder du dig av kortkommandon?

	Mycket	Lite
A	X	
B	X	
C	X	
D	X	
E	X	
F	X	
G	X	
H	X	
I		x
J		x
K	X	

Bilaga 3 Rådata enkät

RÅDATA FRÅGA 1

A: Beror på vilket operativsystem. *Nix baserade (OSX, Linux etc.) föredrar jag TUI över GUI. I Windows använder jag GUI, antagligen av vana då det var det jag började med.

B: TUI, enklare och oftast snabbare

C: TUI
Det känns som att allting går mycket snabbare när man slipper använda musen. Dessutom är man inte lika låst vid en terminal som man är i ett gui. För att utvecklaren/designern av GUI't begränsar vad man kan göra och i vilken ordning man kan utföra något. Vid en terminal har man istället alla småfunktioner som man själv kombinerar ihop/skräddarsyr till just den funktionen man ska utföra.

D: Jag föredrar GUI där alternativet TUI också finns dvs det erbjuds i Windows, Mac OS X och unix/linux. De flesta vardagliga uppgifter som att surfa, kopiera filer drag n drop etc kan jag göra i GUI medans jag kan köra program där GUI är ivägen i skalet. Tex nuget och curl.

E: Terminal om man behöver mata in sökvägar (antar då att det finns path/file-completion) som man vet ungefär hur de ska se ut. Vet man inte det så är en trädstruktur i ett GUI att föredra.

F: GUI , Movivering: För mig känns det självklart att ha ett GUI-baserat OS. Jag tycker nog att frågan är lite konstig år 2012 ;) Vi kommer snart se opertivsystem (Windows 8) där vi använder GUIt ännu mer med ett touch baserat gränssnitt (för vissa applikationer). Jag känner själv att det är klart jobbigt då man ibland måste använda konsollen för att köra text-kommandon då det inte finns något GUI tillgängligt. Dock så har vissa applikationer både ett GUI och möjlighet till vissa kommandon i konsollen, vilket kan underlätta i vissa avseenden, t ex så kan man i IIS:en i Windows snabbt starta om den med ett enkelt kommando istället för att öppna upp applikationen och navigera i menyerna.

G: Det beror på vad jag ska göra. Vissa saker är enklare via TUI, t.ex. väl definierade (ev. repetetiva) uppgifter som inte kräver större grafisk översikt. Operationer som kräver mer information än vad som ryms i en skärmbild åt gången är generellt lättare via GUI.

H: TUI då det för mig är mer effektivt och tillåter mig att vara produktivare i de saker jag gör framför datorn.

I: Jag föredrar GUI. Detta pga att jag förespråkar mer det visuella. Vad jag ser, är vad jag får.
Detta ger även en bättre översikt av vad som finns och dess möjligheter.

J: Nej, jag föredrar GUI framför TUI. Motivering är att det för min känns enklare och mer intuitivt.

K: Mycket beroende på vilken platform jag är på. På Windows föredrar jag GUI. På Mac OS föredrar jag TUI. Men det beror mycket på vad jag gör. Vi utveckling eller när jag håller på med textfiler så föredrar jag TUI. Vid hantering av bilder och/eller filmer så föredrar jag GUI.
Antar att jag är bi eller switch eller nått ^^

RÅDATA FRÅGA 2

FRÅGA 2 - Anser du att dagens GUI i OS kan vara hämmande för avancerade användare?

A: Nej det tror jag inte. Avancerade användare använder det verktyg som hjälper mest och är mest effektivt.

B: Ja, både Windows & OSX dummas, krävs alltid extra klickande för att få fram samtliga inställningar

C: Jag tycker det blir bättre och bättre med GUI, när wizards var "hett" så var det överallt och det är bra första gången man ska göra något men sen är det bara ivägen. Nuförtiden finns det nästan alltid en liten checkbox för visa guide eller göm direkt så man kan komma åt funktionerna direkt istället. Jag vill gärna tro att det blir bättre och bättre...

D: Nej. Eftersom det alltid erbjuds en möjlighet att starta ett skal i de mest använda OS är det inte hämmande.

E: . Helt klart, Jag vill ha betydligt mer information tillgängligt än det som presenteras i många gränssnitt. Antagligen för att man som van användare kan sortera bort information som är irrelevant för den aktuella uppgiften så man inte uppfattar någon information-overload. T.ex. om man kör en detaljerad fillistning så tittar man inte ens på de extra kolumnerna om man inte är intresserad av det så det blir inget problem.

F: Nej, Motivering: Det finns möjlighet i de OS som jag i alla fall använder att köra avancerade kommando i konsollen.

G: Ja, det är lättare att automatisera och anpassa ett TUI-baserat arbetsflöde, än ett via GUI. Det går också ofta fortare att skriva kommandon via tangentbordet än att leta sig fram till motsvarande operation i ett GUI.

H: Beroende på vad man gör. Programmering lämpar sig utan någon GUI-editor medan exempelvis bild/videoredigering förenklas med ett GUI. Däremot kan en kombination av båda vara lämplig då man kan skapa en "mall" för bilder i ett GUI och sedan göra batchexportering i terminalen med hjälp av den mall man skapat.

I: . Det tror jag inte. Det beror helt på vad din arbetsuppgifter är. Även som en avancerade användare kan "enklare" saker att göras. Jag tror det istället beror på uppgifterna som skall lösas.

J: Ja. Mycket information som är bra för nya användare kan lätt skapa ett visuellt brus för avancerade användare, och den grafiska ytan används således inte alltid optimalt.

K: Absolut. Jag tror att man vid TUI blir mer utforskande medans vid GUI så slutar utforskandet när man tittat genom menyerna. En fördel med GUI är att det generellt är lättare att lära sig kortkommandon då de oftast skrivs ut i menyerna.

RÅDATA FRÅGA 3

FRÅGA 3 – Vilket/vilka operativsystem använder du?

A: OSX och Windows

B: Windows (7 & Vista), OSX, Linux (enbart i terminal form)

C: Windows 7, Ubuntu och OSX

D: Windows 7, Windows XP, Mac OS X, har använt linux en hel del men för tillfälligt inte (kört mest som server både med och utan GUI)

E: OS X, Win XP, Win 7

F: Windows 7 samt Mac OS X Lion.

G: Windows (hemma/i arbetet), Linux(hemma), Mac OS X (i arbetet), iOS

H: GNU/Linux på desktop samt servrar. Även *BSD-server och Windows (XP) då

kompabilitet med vissa program krävs.

I: Snowlion

J: OSX och Windows 7.

K: Mac OS X och Win 7.

RÅDATA FRÅGA 4

FRÅGA 4 Vilket beslut låg bakom ditt val av operativsystem?

A: Var nog inget medvetet beslut. Fick Windows 3.1 för många år sedan och har hållt mig till det. OSX har jag börjat använda senaste året för att kunna använda verktyg som inte fungerar särskilt bra på Windows.

B: Windows: jobb os, utveckling, spel. OSX: ok unix, kommer med macboken, Linux: bäst möjlighet att anpassa för det man ska ha det till (ex: filserver behöver inte ha något gui då den inte har en skärm)

C: Windows har jag alltid använt och det är jag dessutom tvungen att använda i mitt arbete.
Ubuntu för jag har en liten server hemma utan tangentbord/mus/skärm inkopplat som bara står och gör sitt jobb.
OSX för jag fick en macbook via jobbet, tycker det är lite kul att testa något nytt.

Det är lite rätt verktyg för rätt jobb på något sätt som ligger bakom beslutet.

D: Använder mest Windows och det är mest använt, utvecklar för det i .NET.

E: OS X: självvalt, Win XP: krav från kund, Win 7: självvalt

F: Vill köra det senaste

G: I arbetet är det bestämt av arbetsgivare/uppdragsgivare. Hemma baserades besluten på vad som var praktiskt för användningsändamålen och övriga medlemmar av familjen.

H: . Nyfikenhet. Jag ville testa något nytt, få större kontroll över det system jag dagligen använder och helt enkelt lära mig mer om de alternativ som finns att tillgå.

I: Jag vill ha något som fungerar och behöver fundera så mycket kring uppsättningar och konfiguration. Snowlion bara fungerar! Ingen blå skärm.

J: Jag använder idag OSX privat och Windows i jobbet. Valet av OSX är i kombination av valet av fysisk dator, Mac, vilket jag valde med hänsyn till prestanda och awesomeness. Windows använder jag i jobbet då övrog infrastruktur kräver det (Visual Studio, Microsoft CRM).

K: Det som kom med datorn + arbetsgivare

RÅDATA FRÅGA 5

FRÅGA 5 Vid installation av ny programvara –Av TUI och GUI, vilket anser du mest bekvämt och vilket ser du som mest effektivt att använda?

A: I OSX är TUI både mer bekvämt och mest effektivt tycker jag. I Windows är GUI mest bekvämt men de få gånger jag använt TUI har det känts mer effektivt.

B: Beror på, föredrar commandline för ex versionshantering fast andra saker (ex web) funkar en gui ofta bättre

C:) Det beror lite på vad det är för slags programvara. Ifall det är ett program man inte behöver installera så är TUI mycket enklare t ex apt-get systemet i linux eller homebrew i osx. Den tar bara hand om allting och laddar ner det bara jag vet vilket program jag vill ha.
Är det å andra sidan massa konfiguration som måste göras t ex vid installation av någon serverprogramvara så föredrar jag GUI med lite mer stöd för konfigureringen.

D: . Lätt GUI, bara att klicka på en ikon.

E: Terminal är att föredra för enklare installationer eftersom man då ofta bara kör ett kommando och då slipper all GUI-spam i form av EULA etc. För avancerade installationer behöver man ett GUI för att få en överblick på alternativen.

F: GUI

G: I OS som har stöd för bra TUI-baserad installation (t.ex. pakethanteringen i Debian Linux; apt-get) känns det som ett överlägset alternativ. Det går snabbare att söka sig fram till paket och genomföra installationen med tangentbordet än att omväxlande skriva och klicka med musen

H: . Precis som svaret till fråga två så är det lite baserat på vad som skall utföras. I de flesta fall blir det TUI-program då de flesta Open Source GUI-program oftast bygger på ett befintligt TUI-API och därav väljer jag själv TUI över GUI för att kunna interagera med det på det sätt jag behöver.

I: Jag har aldrig installerat TUI, så har svårt att svara på denna frågan.

J: GUI.

K: Oftast är GUI mest bekvämt. MEN det beror mycket på dokumentation.
Om TUI har installationsanvisningar så går det oftast snabbare och är smidigare.

RÅDATA FRÅGA 6

FRÅGA 6 - Hur navigerar du dig vanligtvis i din dator?

A: . OSX med TUI (Terminalen), i Windows GUI (Utforskaren).

B: Med tangentbord så mycket som möjligt

C: Med hjälp av launchy på windows och quicksilver på osx, dvs bara via tangentbordet. Använder total commander i windows för att slippa utforskaren och kunna göra allting utan att använda musen.

D: GUI

E: GUI

F: Tangentbord och mus

G: En blandning av TUI och GUI, beroende på vad jag ska göra

H: Via terminalen för att finna vad jag söker och sedan öppna det med lämpligt program.

I: Med pekplatta/mus.

J: Med tangentbordet och kortkommandon.

K: Beror på platform och omständigheter. Se punkt 1.

RÅDATA FRÅGA 7

FRÅGA 7 hur stor utsträckning använder du dig av kortkommandon?

A: Använder kortkommandon i väldigt stor utsträckning både i operativsystemet och verktygen jag använder. Uppmärksammade efter en tid att du sparar mest tid om du kan behålla båda händerna på tangentbordet så mycket som möjligt och då gjorde jag ett aktivt val att lära mig så många kortkommandon. Dessutom läste jag mycket om muskelminne man får när man sitter vid datorn som gör det mycket snabbare att använda kortkommandon än att använda GUI.

B: Finns det så använder jag det, vid programmering så försöker jag navigera utan att använda mus för att inte behöva flytta händerna.
de flesta browsers har stöd för enklare saker (byta tab, ny tab, öppna senast stängd, reload osv),
tabba mellan input fält

C: Så mycket som möjligt! Finns väldigt bra stöd för kortkommandon i Visual Studio som jag använder och jag försöker hela tiden undvika att använda musen/gui för att göra saker och ting. Bara lägga till en ny fil är antingen 3-4 musklick eller alt+insert och sen enter.
I kommande version av Visual Studio kommer det med ett inbyggt stöd liknande launch/spotlight/quicksilver där man bara skriver i ett sökfält så letar den upp kommandon/inställningar/etc i visual studio och öppnar den fliken.

Med Launchy (http://www.launchy.net/) är det otroligt enkelt att öppna vad som helst i windows (program, dokument, filmer etc etc) bara genom att tryck alt+enter och skriva vad man vill ha oavsett vart det ligger på datorn.

Ser man på hårdvaran så har det ju utvecklats en hel del över tiden speciellt när det gäller tangentbord. Nu är det väl nästan standard att det kommer med dedikerade volymknappar, play, stop etc etc vilket jag använder hela tiden.

D: Väldigt mycket kortkommando. Nästan uteslutande för tex navigering.

E: Så ofta jag kan. Om jag måste gå in i en meny eller klicka på en knapp för att göra något så lägger jag tid på att se om det inte finns ett kortkommando för det. All tid som spenderas med musen när man väl kodar är waste och kan man få ihop ett bra workflow där man kan undvika att använda musen är det nice. Gillar personligen kombinationen med CTRL bättre än CMD eftersom CMD sitter lite dumt till.

F: Ganska stor, Motivering: I de verktyg jag använder försöker jag lära mig eller själv sätta kortkommando på vanligt förekommande operationer för att snabba upp arbetet.

G: För texteditering, framförallt inom programmering använder jag nästan uteslutande kortkommandon. För vanliga funktioner i Operativsystemet använder jag också kortkommandon.

H: Konstant. Då vi(m) bygger på kortkommandon och sekvenser av tangentbordstryckningar så utförs diverse kommandon baserat på dessa sekvenser. De flesta av de program jag använder dagligen har stöd för vim-bindings varpå jag alltid kan navigera och utföra diverse moment med liknande kortkommandon oavsett vad jag skall göra, om det så är att navigera på en webbsida, programmera, bläddrar genom bilder eller chattar. Det har däremot lett till att de tillfällen då dessas vim-bindings saknas i ett program så kan man känna sig aningens handikappad. Även den fönsterhanterar jag använder styrs helt av kortkommandon. Det finns inga menyer, inget sätt att öppna program genom ikoner man kan klicka på utan allt är kortkommando-styrt. Därav startas program genom en terminal som i sin tur öppnas via ett kortkommando varpå terminalen mer eller mindre alltid är utgångspunkten för ett program (körs som en child-process för det shell som öppnas via terminalen).

I: Väldigt lite. Men skulle gärna vilja göra det med.

J: Jag försöker medvetet att använda tangentbordet och kortkommandon i så stor utsträckning som möjligt. Både i webbläsaren, när jag navigerar i filsystemet och i de applikationer och system som jag använder mest. Jag skulle vilja påstå att jag använder tangentbordet 70% och musen 30%, ungefär.

K: Beror på platform och program. Jag använder då och då en texteditor
som heter Vim och där måste man använda såkallade "kortkommandon".
Annars är det bara de vanligaste kortkommandon.

Bilaga 4 Användartest

Uppgift 1: Ett skämt i en txt-fil!

(Börja från hemkatalogen)

1. **Skapa en mapp och döp den till ”Testing”.**

2. **I mappen ”Testing”, skapa ytterligare 5 mappar och döp dom till ”mapp1”, ”mapp2”, ”mapp3”, ”mapp4” och ”mapp5”.**

3. **I mappen ”Testing”, skapa en textfil som du döper till ”häst”. Öppna textfilen och skriv in denna text:**

- **Hej! Nu ska jag berätta en rolig historia. En cowboy och hans häst satt och spelade kort. Plötsligt började hästen äta på korten. Cowboyen frågade:**
  - Varför äter du på korten?
  - Det är ju klöver, sa hästen!

4. **Spara din text.**

5. **Gör 5 kopior på textdokumentet och döp dom till ”häst1”, ”häst2”, ”häst3”, ”häst4” och ”häst5” och lägg var och en av dessa filer i vars en av de tidigare skapta mapparna med respektive siffra.** (dvs. ”häst1.txt” ska ligga i mappen ”mapp1”, ”häst2.txt” ska ligga i mappen ”mapp2” osv.)

6. **Zippa de 5 mapparna.**

7. **Packa upp mapparna på skrivbordet.**

8. **Ta bort mapparna på skrivbordet och ”Testing”-mappen <u>permanent</u>.**

9. **KLAR!**

Uppgift 2: Bygg en hierarki med mappar 1!

(Börja från hemkatalogen)

1. **Skapa en mapp och döp den till ”Bibliotek”.**
2. **I mappen ”Bibliotek”, skapa en mapp och döp den till ”Alfabet”.**
3. **I mappen ”Alfabet”, skapa mappar från a-z** (dvs. första mappen heter ”a”, andra ”b” etc.).
4. **KLAR!**

Uppgift 3: Bygg en hierarki med mappar 2!

(Börja från hemkatalogen)

1. **Skapa en mapp och döp den till ”Bibliotek”.**
2. **I mappen ”Bibliotek”, skapa en mapp och döp den till ”Siffror”.**
3. **I mappen ”Siffror”, skapa mappar från 1-15 som läggs i varandra** (dvs. 1 ligger i ”Siffror”, 2 ligger i 1, 3 ligger i 2 osv.).
4. **KLAR!**

Uppgift 4: Bygg en hierarki med mappar 3!

(Börja från hemkatalogen)

1. **Skapa en mapp och döp den till ”Bibliotek”.**
2. **I mappen ”Bibliotek”, skapa en mapp och döp den till ”Färger”.**
3. **I mappen ”Färger”, skapa de 7 mapparna ”Yellow”, ”Orange”, ”Red”, ”Purple”, ”Blue”, ”Green” och ”White”.**
4. **KLAR!**

Uppgift 5: Stora datamängder

(För att göra denna uppgift, utgå ifrån bilagan ”chattlogg.txt” du fått. När du hittar svaren på frågorna, säg svaren högt.)

1. **Vad skrivs i chat-loggen kl 5:00:48 PM?**
2. **Vad skrivs i chat-loggen kl 5:31:44 PM?**
3. **Hur många ord skrivs i post’en kl 5:54:48 PM?** (bortsett från tiden och postarens namn.)
4. **Hur många gånger har ’glenn c elliott’ skrivit en post?**
5. **KLAR!**

Bilaga 5 Användarresultat - Användartest

Faktiska operationer

Användare	Operativsystem	Uppgift 1	Uppgift 2	Uppgift 3	Uppgift 4	Uppgift 5
Testperson 1	Window 7	03:03	02:36	02:32	01:01	01:43
Testperson 2	Windows 7	01:33	01:50	01:06	00:39	00:42
Testperson 3	Windows 7	01:42	01:41	01:23	00:48	00:51
Testperson 4	OS X Lion	03:04	01:37	04:03	01:08	00:51
Testperson 5	Windows 7	01:34	01:07	00:23	00:29	01:42

Övriga Operationer

Användare	Operativsystem	Uppgift 1	Uppgift 2	Uppgift 3	Uppgift 4	Uppgift 5
Testperson 1	Window 7	00:40	00:31	00:05	00:29	00:57
Testperson 2	Windows 7	00:28	00:12	00:22	00:09	01:00
Testperson 3	Windows 7	00:01	00:06	01:21	00:09	00:28
Testperson 4	OS X Lion	00:31	02:26	00:16	00:04	00:08
Testperson 5	Windows 7	00:10	00:09	00:04	00:03	00:08

Kombination

Användare	Operativsystem	Uppgift 1	Uppgift 2	Uppgift 3	Uppgift 4	Uppgift 5
Testperson 1	Window 7	03:43	03:07	02:37	01:30	02:40
Testperson 2	Windows 7	02:01	02:02	01:28	00:48	01:42
Testperson 3	Windows 7	01:43	01:47	02:44	00:57	01:30
Testperson 4	OS X Lion	03:35	04:03	04:19	01:12	00:59
Testperson 5	Windows 7	01:44	01:16	00:27	00:32	01:50

GUI – Tabell över varje testdeltagares resutat för faktiska operationer, Övriga Operationer och kombinationen av de båda och de operativsystem som använts.