VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

THE ÉTOILÉ PROJECT FEASIBILITY STUDY

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE   MICHAL ČÁP
AUTOR

BRNO 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
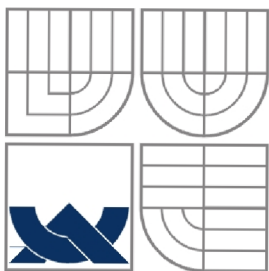DEPARTMENT OF COMPUTER SYSTEMS

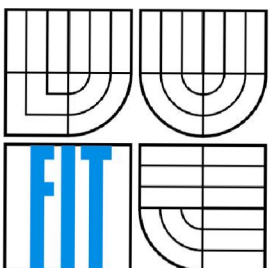# THE ÉTOILÉ PROJECT FEASIBILITY STUDY

BAKALÁŘSKÁ PRÁCE
BACHELOR´S THESIS

AUTOR PRÁCE   MICHAL ČÁP
AUTOR

VEDOUCÍ PRÁCE   ING. MILOŠ EYSSELT, CSC.
SUPERVISOR

BRNO 2007

Brno University of Technology - Faculty of Information Technology

Department of Computer Systems   Academic year 2006/2007

BSc. Project Specification

For: **Čáp Michal**
Branch of study: Information Technology
Title: **Feasibility Study of Benefits of Étoilé User Environment**
Category: Uživatelská rozhraní

Instructions for project work:

1. Research principles of an Étoilé user environment.
2. Describe user interface including fictional screenshots, and describe its parts behaviour.
3. Make a usability improvement study: Fictional task analysis performing tasks in different areas where computers are used. Investigation of users' approval on the new user interface concepts.
4. Make an underlying architecture feasibility study: Description of underlying architecture, analysis of advantages/disadvantages of an Étoilé's underlying architecture, ease of development, hardware requirements.
5. Consult with Dr. Nick Wright, Senior Lecturer, Logistics, Coventry University, UK.
6. Discuss the results and suggest possibilities for further development of the project.

Basic references:

- Wikipedia: Étoilé, available on http://en.wikipedia.org/wiki/%C3%89toil%C3%A9

Detailed formal specifications can be found at http://www.fit.vutbr.cz/info/szz/

The BSc. Thesis must define its purpose, describe a current state of the art, introduce the theoretical and technical background relevant to the problems solved, and specify what parts have been used from earlier projects or have been taken over from other sources.

Each student will hand-in printed as well as electronic versions of the technical report, an electronic version of the complete program documentation, program source files, and a functional hardware prototype sample if desired. The information in electronic form will be stored on a standard non-rewritable medium (CD-R, DVD-R, etc.) in formats common at the FIT. In order to allow regular handling, the medium will be securely attached to the printed report.

Supervisor: **Eysselt Miloš, Ing., CSc.**, DCSY FIT BUT
Beginning of work: November 1, 2006
Date of delivery: May 15, 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačových systémů a sítí
612 66 Brno. Božetěchova 2

Zdeněk Kotásek
Associate Professor and Head of Department

LICENČNÍ SMLOUVA
POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami

1. **Pan**

   Jméno a příjmení: **Michal Čáp**
   Id studenta: 84085
   Bytem: Hollarova 2158/3, 400 11 Ústí nad Labem
   Narozen: 02. 04. 1985, Ústí nad Labem
   (dále jen "autor")

   a

2. **Vysoké učení technické v Brně**

   Fakulta informačních technologii
   se sídlem Božetěchova 2/1, 612 66 Brno, IČO 00216305
   jejímž jménem jedná na základě písemného pověření děkanem fakulty:

   ....

   (dále jen "nabyvatel")

Článek 1
Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
    bakalářská práce

Název VŠKP: Feasibility Study of Benefits of Étoilé User Environment
Vedoucí/školitel VŠKP: Eysselt Miloš, Ing., CSc.
Ústav: Ústav počítačových systémů
Datum obhajoby VŠKP: ....

VŠKP odevzdal autor nabyvateli v:
    tištěné formě   počet exemplářů: 1
    elektronické formě   počet exemplářů: 2 (1 ve skladu dokumentů, 1 na CD)

II

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

Článek 2
Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užit, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizování výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti:
    - ☒ ihned po uzavření této smlouvy
    - ☐ 1 rok po uzavření této smlouvy
    - ☐ 3 roky po uzavření této smlouvy
    - ☐ 5 let po uzavření této smlouvy
    - ☐ 10 let po uzavření této smlouvy
    - (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

Článek 3
Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: ....

....
Nabyvatel   Autor

III

ABSTRACT AND KEYWORDS

ABSTRACT

The purpose of this thesis is to describe the goals of the Étoilé project and to analyse its feasibility. The Étoilé project is an open source community attempt to replace obsolete user interface concepts widely used in current systems such as Microsoft Windows or Mac OS X. This work consists of the following parts:

- description of the concepts proposed by the Étoilé community
- design of the fictional environment that would develop from Étoilé project in a future
- task analyses showing a work as if it took place on that environment
- survey researching user approval on these concepts
- brief description of the underlying architecture used in the project
- estimation of the effort needed until Étoilé gets usable

KEYWORDS

Étoilé, desktop environment, user-centric, document-oriented, user interface GNUStep, Objective-C, user environment, usability, object oriented environment, graphics interface, GUI, user interface concepts, Unix GUI, service oriented approach

ABSTRAKT

Tato práce popisuje cíle projektu Étoilé a analyzuje jejich proveditelnost. Étoilé je open-source projekt, který se snaží odstranit zastaralé koncepty ovládání počítače, často stále používané v současných operačních systémech jako je Microsoft Windows nebo Mac OS X. Práce obsahuje tyto části:

- popis konceptů uživatelského rozhraní navržených v rámci projektu Étoilé
- ztvárnění fiktivního prostředí, které ukazuje možnou budoucí podobu projektu Étoilé
- rozbor úkonů prováděných při práci v tomto fiktivním prostředí – psaní dopisu a kolaborativní úpravy dokumentu
- průzkum, jak uživatelé přijímají tyto nové koncepty
- stručný popis systémové architektury projektu Étoilé
- odhad náročnosti implementace projektu

KLÍČOVÁ SLOVA

Étoilé, desktopové prostředí, uživatelské prostředí, GNUStep, Objective-C, použitelnost, objektové prostředí, grafické rozhraní, GUI, koncepty ovládání počítače, Unixové grafické prostředí

TABLE OF CONTENTS

VI

1 PROJECT BACKGROUND

1.1 Introduction

Computers have become an integral part of our lives. Employees in the developed world are expected to work with them everyday to do their job. There are many professions that rely on their usage. Where once computers were mainly the domain of technically-oriented jobs, they have become increasingly important for companies in more traditional areas, such as law and medicine. A survey has revealed that the vast majority – 98% – of American lawyers take advantage of desktop computing to cope with the quantity of office documents. (KC Daily Record Staff, 2005)

That computers are now being used more and more by people who are not technology-oriented and do not understand the technology that makes them work represents a significant cultural shift. As they are specialist in some other area it is perfectly understandable that they do not have any interest in learning what drives these seemingly arcane devices.

However, this vast change in the types of users operating computers seems not to be reflected in modern user interfaces. They still remain grounded in the obsolete user interface concepts introduced more than 25 years ago. (Hiltzik 1999) They suffer from concepts that were designed with respect to the limitations of the computer architecture rather than to the user's style of work. These concepts force user to learn new ways of doing old things, which puts an unnecessary barrier between non-technical oriented people and computers.

Considering that virtually all current desktop environments, such as Microsoft Windows or Mac OS X still use these old concepts, the situation is even more disturbing and potentially detrimental. Attempts to come up with vastly different conceptual model for the user environment are very rare due to the high number of users familiar with the established paradigm.

Therefore, it might not be a coincidence that one of the serious attempts to introduce new innovative user interface concepts has come from the open source world. It is this project, called Étoilé, which has brought up ideas that are worth exploring. It has started work on an environment that seeks to be easy to use for all the users who have found current environments complex and too far removed from reality. Étoilé proposes documents instead of files, projects instead of folders and services instead of applications. Documents do not need to be saved. They do not even need to be named. Everything looks as if it was lying on the office desktop. Documents in a folder can be distinguished by their contents and everything can be manipulated and moved freely. (Ross, Mathé et al 2006)

This project has ambitious goals but is still at the beginning. The community around it is discussing how the ideas mentioned above should work in practice, but their vision is rather abstract. Decisions are made on the fly as the implementation progresses.

However, ideas proposed by the project's community could significantly affect a desktop user's experience and boost work productivity. It could help seniors and all people who are not following the technology cutting edge to take advantage of information technologies.

This project discusses the feasibility of Étoilé project. It shows how it would affect user experience once finished. It discusses feasibility from the viewpoint of the implementation and the eventual acceptance of the prospective users. It is trying to evaluate its goals and ambitions. Finally, it intends to offer good background on the project for everybody who is interested in it or just thinking to get involved.

1.2 Problem definition

Étoilé project was launched in June 2004 by Quentin Mathé and Nicholas Roard because they were not satisfied with the situation on the field of desktop environments. They had seen all the available systems as obsolete and lacking any innovation in terms of usability and user interface concepts. Thus, they set up the ideas and concepts to be followed and designed general system architecture of the Étoilé system. (Mathieu 2006)

This helped to create the community of people interested in that rather abstract and distant vision. Almost 3 years later, the environment is still not in a usable stage. Some basic frameworks are already coded, but all the planned innovative concepts remain unimplemented.

This may bring up the question whether the project has realistic goals in terms of its successful implementation. There are several aspects that may raise doubts.

As the software crisis in seventies proved, software projects are generally difficult to manage. They tend to run over-time and over-budget. Most of them failed before completion. Étoilé aims to implement complete desktop environment including some functionality which is normally handled by operating system kernel. It makes the project scope comparable to the biggest open source projects such as GNOME or KDE. It is surely legitimate to ask - will it ever be in a usable stage? Will it succeed?

To answer that question, Étoilé may be compared to similar projects in the past. The problems that led to their eventual failure need to be identified and related to the Étoilé project. It would help developers to get reasonable idea about the complexity of such project and to evaluate meaningfulness of possible personal involvement. As it is an open source project, it is fatally dependent on the community of contributing developers who keep the project active and development in progress.

Another aspect is related to a user acceptance of the concepts proposed by the project. Étoilé developers see them as the ideas that, once implemented, will change the way we work with computers. But are users ready for such a radical change? Would they accept it? Would these concepts be really beneficial in the real everyday tasks that people perform on their computers?

The project is now in an active state – frameworks are being implemented and conceptual questions discussed on the mailing list and dedicated internet chat. However, it lacks the detailed vision of the user environment that would serve as a ground for the discussion and overall presentation of the project's goals. At the moment, only partial and abstract thoughts exist on each user interface component available, which makes it extremely difficult for a novice user to figure out what are the concrete ideas behind the concepts and why is it so beneficial to employ it as a part of new-generation desktop environment.

1.3 Project objectives

The main goal of this thesis is to work out a summarising document that would help Étoilé community to solidify its shared vision and attract new contributors. It is intended to be a comprehensible explanation of the Étoilé project goal and basic but realistic feasibility study of its implementation.

The thesis is supposed to serve as an introduction into Étoilé project for everybody who is looking for a general information about the project. It discusses the conceptual project goals and demonstrates it by showing behaviour and appearance of the system that would develop from Étoilé project in the future. It also investigates prospective users' opinion on

such environment and aims to compare it with the similar attempts in past to analyse vitality of its ambitions in relation to the size of the contributing community.

To achieve these objectives, a custom fictional user environment will be used to demonstrate the concepts proposed by the community. Its design is derived from the Étoilé project's conceptual vision. Therefore, the environment shown in this report is likely to be similar to the one that will develop from the Étoilé project in the future.

1.4   Terminology

1.4.1   Application oriented approach

Application oriented approach is a general term for the user interface concepts used in current desktop environments such as MS Windows, Mac OS X or Linux desktop environments KDE and Gnome. In these systems, applications are the main entities handled by user. Documents belong to applications and are usually managed by them. To edit a document, at first it needs to be decided which application to choose. Document is then loaded into the application and saved in the application's native format. Different kinds of data must be edited in different applications and therefore in different contexts.

1.4.2   Document oriented approach

Document oriented approach is concept proposed by Étoilé project stressing documents as main entities in the environment. Tools operating on documents are called services and the concept of application remains only for tasks that are document unrelated.

Document oriented environment tends to group all activities into one workflow around a document. Document is always put into the centre of attention. All tools necessary to edit it are loaded around it on the fly as needed.

This approach is argued to support real-world workflow better than application oriented environments, which may distract user by changing work context along with every application.

1.4.3   Project inspired workflow

Project inspired workflow is another concept proposed by the Étoilé project. It is inspired by document management systems used by large companies to deal with thousands of documents needed to run a business. It usually includes versioning, which ensures that every change in the document is recorded and may be later reverted, access control and facilites to link documents to a particular project. Étoilé plans to go further and support spatiality, persistent session management, system-wide undo and the collaborative live editing of documents.

1.5   Similar attempts in the past

Étoilé is surely not the very first attempt to change the way we use computers. There have been a few notable projects with similarly ambitious goals in the computing history. Many of them aborted before completion. Other significant group of projects did not meet expectations or were commercially unsuccessful.

NEXTSTEP by NeXT Computer and Apple Newton can be mentioned as the examples of partially successful projects.

NEXTSTEP is probably the most successful system in terms of a technological maturity. Sometimes it is still seen as the best designed operating system ever, although it lacked any large scale commercial acceptance. In 1989, it introduced technologies that were sometimes more than ten years ahead to the rest of the computer industry. It introduced dynamic object architecture and the first really sophisticated graphical user interface.

Étoilé itself is based on the technology developed as underlying architecture for NEXTSTEP. It uses Objective C as implementation language and frameworks derived form the original NEXTSTEP objective frameworks. It is backwards compatible with NEXTSTEP and can run applications written for it.

Apple Newton was revolutionary product designed and marketed by Apple Computer since 1993 with the aim to redefine personal computing. Its operating system Newton OS already included lot of ideas that are now included in Étoilé project vision but failed to succeed commercially at the time, mainly due to the overpricing and initial problems with hand-writing recognition. Étoilé developers often admit that it had been significant source of inspiration when the project goals were formed. (Mathieu 2006)

In 1992, Microsoft also announced its own vision of system that should have employed the similar concepts which is Étoilé promoting now. This was code named Cairo, but never released and just few partial technologies are time to time included in the official release of Windows. Most promising part – WinFS – is still not released though.

2 ÉTOILÉ PROJECT VISION AND ITS CURRENT PROGRESS

2.1 Main goal

The leading members of the Étoilé community have published the project's goals on their website:

> Étoilé intends to be an innovative GNUstep based desktop environment built from the ground up on highly modular and light components with project and document orientation in mind, in order to allow users to create their own workflow by reshaping or recombining provided Services (aka Applications), Components, etc... Flexibility and modularity on both user interface and code level should allow it to be scaled from PDA to computer environment.

> The Étoilé project's goals are:

> - Light, focused applications, cooperating together to provide a rich user experience, using GNUstep services and its own Services model.

> - Fast, simple data sharing between tasks and documents without involving lot of context switches (between applications, windows, selection...).

> - Facility provided by default to composite, layout unrelated elements in first-class objects like documents, folders, etc...

> - Workflow based on project inspired management (versioning, indexing, sharing...).

> - Easy communication/collaboration among users, where users are first-class objects in the environment.

> - Assistant technology similar to the Newton's approach.

> It works in cooperation with GNUstep project and eventually Backbone project (http://www.nongnu.org/backbone/). It provides frameworks, UI guidelines and documentation to permit GNUstep developers to easily develop Étoilé compliant applications (called Services).

> (Ross, Mathé et al 2006)

The points raised here would be practical as the general description of course adopted by the project, but probably useless for somebody who has not sufficient background in operating systems concepts. Therefore, each concept mentioned here will be described in further details on the following pages.

2.2   Main conceptual ideas

2.2.1   Light, focused applications cooperating together

The application concept

In today's systems, we are mostly using the concept of application, which is usually understood as a monolithic bundle of functionality related to the application's main task. For example, a word processor, apart from text editing functions, includes tools for editing bitmaps, spell-checker, word counter, database interconnection tool and many other features. Cooperation with the host environment is limited only to saving and loading files and to the use of clipboard.

Étoilé community defines traditional application as:

> Stand-alone, self-sufficient, usually closed process with graphical interface, usually for creating, editing and viewing documents. Application contains code for the application specific document handling, therefore all documents usually belong to the application. Relationship between application and document is one to many at a time.
>
> (Mathé et al 2005)

Thus, application is seen as an awkward tool for interaction with documents. Workflow is pre-designed and incapable of reshaping. Cooperation with other applications is very limited. Most of embedded functionality is often unused and makes user interface complex and complicated to use.

The service concept

In Étoilé, the concept of application is mostly removed in favour of the service concept. Services are light, focused applications operating on objects. The ones that can act as a part of a compound document are called Service Components.

Étoilé community describes these as:

> With Étoilé, in some way Applications are a thing of the past... To be precise, Applications are now special components which are loaded on the fly by Étoilé in whatever Workspace windows. Workspace windows which represent a piece of data like a folder or a document are very special windows called context windows which can be shared with the applications. (Mathé 2005)

Explanation of terminology

Although terminology has not settled yet, some attempts to clarify the core terms among the project developers is evident:

> Étoilé service: Stand-alone graphical interface wrapper (in form of a process) around framework provided mechanisms for document and object handling by the user. Relationship between application and document is many to many at a time. (See Service entry)

> Étoilé service provider: Stand-alone process providing advertised functionalities called service units or simply services

Étoilé Component (or Service Component): Set of code objects packaged together and coordinated by wrapper object implementing ComponentKit Master protocol.

(Mathé et al 2005)

Note that the general idea of service has been divided into two terms: service and service provider. This has been adopted to encourage developers to keep a user interface and underlying logic separately to comply with the model-view-controller methodology. Service and service provider represent controller/view and model respectively.

Although classic applications are for compatibility reasons still supported, Étoilé services are intended to replace them. Functionality of the present-day applications is planned to be broken apart to several components interacting together within the Étoilé component framework.

Typical workflow in the classical, application-centred environment involves numerous context switches as different data types need to be edited in the respective applications. It evidently requires to switch between different windows with divergent controls and then put data together again using clipboard or various export/import functions.

In the Étoilé environment, different kinds of data are edited directly in the document where they are actually used. The functionality needed to edit them is provided by wide choice of services loaded and unloaded on request. This approach is expected to avoid unnecessary context switches and workflow interruptions.

Étoilé service concept is comparable to the approach adopted by unix command line tools and plans to provide a similar flexibility within a graphical user interface.

A word processor example

Imaginary word processor running in Étoilé environment would be divided into several services. There might be an independent service responsible for vector drawing, one for tables, one for spell-checking and a very simple one for word counting. All these services would be system-wide and able to cooperate with other services from different vendors as a part of a compound document.

2.2.2   Fast, simple data sharing

In Étoilé, all data exist in the form of a first class object. Services can operate only on objects. Objects can be put together to form compound documents.

The process of creating a new document typically involves an activity of putting components responsible for different activities in a shared canvas. Thus, simple text document would consists of some components carrying formatted text, a component displaying picture and one rendering fancy borders. All these components cooperate seamlessly on the document canvas.

There is no need for the context switch once user decides to edit for example an image embedded in a text document. This image would be easily manipulated directly on it. Service operating on the image component would just display its own toolbars and menu around the image.

This approach seems to be similar to the Microsoft OLE technology. Clarification of its mutual relationship is therefore described at the Étoilé project website:

Objects in contexts

User, as a productive element, creates objects: pictures, photographs, pieces of text, articles, calendars, contact cards, music, to-do lists, source files, ...

Currently each object is separately handled by corresponding application, therefore each object is handled in different context. However, if we think of our tasks or projects as of a single thought-context, then why the computer should force us to think in more contexts? It should not! Therefore, we, the users, should be able to follow our thought stream in a single project oriented context and the computer should aid us in this creative process.

Let us take OLE-like mechanism known from MS Windows. It is an attempt to keep user in single thought context, however it helps only for desktop-publishing tasks. One has a textual document where he can embed charts, tables or music from other applications. Or one has a spreadsheet where he can embed graphics from a diagram drawing application or an equation from an equation editor. Where is the problem? First, you have many different types of containers for grouping objects of ones thought process. Before you start creating your project, you have to pick the right container: word document, spread-sheet,... But, how should i know, what type of container to pick? What if I realise that I wanted diferrent container later? Second: each container has different format, even it contains object for a project/thought process. One has to use specific application that created the container to be able to see objects in the container, even the objects were created in a different application. OLE was not a bad idea, however, it was not very polished.

(Ross Urbánek 2005)

In Étoilé, more systematic approach is used as everything in the system is treated as object. Objects can be used as input data for various services installed in Étoilé system. List of services capable of work on the particular kind of component is always offered in the menu, enabling user to decide which one to pick. Majority of document-oriented tasks may be then achieved without any context switch.

For example, when user needs to get word count of any part of the document, it is done by selecting a corresponding component and then choosing menu item called Services. It is the place where all services capable to operate on that object are listed. Once user chooses the word count item from the menu, small window pops up and displays the result immediately.

2.2.3   Facility for composite layout unrelated elements

In the current systems, there is a conceptual difference between the process of editing document and displaying them passively in the form of icon. When being edited, it displays its contents. When iconified, it shows its filetype. That is the reason why all MS Word documents look identically and may be distinguished only by a filename.

In Étoilé, this distinction has been removed. Documents tend to look similarly both when iconfied or when being edited in a window. Iconified version of document is still rendered by service set as default for the particular type of object.

Therefore, arranging contents in a folder is essentially the same action as the creation of compound document.

By default, the contents of folder are displayed spatially in the form of thumbnails representing each document in it. This serves as analogy to the real world environment where desktop may be covered with many piles of semi-finished documents intuitively organized in a space.

In traditional environments, files can be grouped together into a folder. In Étoilé, folder can contain whatever object. Thus, it is not unusual for folder to hold a running application or object that represents a real person. Once closed, objects are put into hibernation. Next time, the applications are opened in the identical state as when they were lastly closed.

2.2.4   Workflow based on the project inspired management

Étoilé intends to adopt the improved workflow concepts based on the way how people tend to organize their work in the real world. Spatial organization, focus on one task at a time or distinguishing documents by their contents instead of their filename are just some of them.

Furthermore, it adds concepts borrowed from corporate information systems such as versioning, indexing, sharing and access control.

Project

Étoilé introduces a new user interface element called project. It has been designed as an instrument to organize work more intuitively with a remarkable relation to the real world principles.

Project is described as:

> A desktop containing one or several documents. It can be saved and open as file. It is very similar to workspace in most window manager. Project can also be opened later from file manager. The concept is similar to session in most window manager.
>
> (Chen 2006)

Another explanation which can be found at the project website is:

> Similar to a folder in that it serves as a container for documents. Projects typically fill the whole screen when maximized, and contain metadata about the relationships between their contained documents. Projects can be annotated (See Annotations in Workspace). When minimized (zoomed out), a Project's thumbnail is a live representation of its contents, much like a virtual desktop.
>
> (Mathé et al 2005)

Conceptually, the project element is intended to carry documents and applications related to a particular task. Therefore there will probably exist many Étoilé projects related to the real-life projects of user. Once opened, it displays its contents in the same state as it was left last time. Projects allow to switch quickly between the work contexts and customise environment to suit the nature of task.

Versioning

Étoilé has a built-in document management system recording all modifications that user made on a document. This system allows user to track them and eventually revert document

to any of the previous versions. Whenever technically possible, this is supposed to act as a system-wide undo.

Indexing

Live searching is an important concept in Étoilé. Document content is saved in an object oriented database along with the associated metadata. It also maintains an index that allows fast fulltext searching within all objects administered by the system. Search text field is always available on the menubar. Typing a query will immediately highlight those documents and folders that contain searched phrase. This technology was named FogVision in Étoilé.

Sharing

Étoilé provides built-in facilities for document sharing. This topic is discussed in the next chapter.

2.2.5   Easy communication and collaboration

People as the first class objects

Étoilé provides support for an easy communication among users. Mainly, it is modelling people as first class objects capable of full interaction with the environment. People are represented as icons that can be placed anywhere an object is allowed.

Dragging a document onto that icon will initiate transfer to the person it represents. Selection of an actual transfer technology is postponed to the very end.

The addresses in a object representing a person need to be pre-set by user once created. It provides support for a variety of communication technologies such as email, Jabber, MSN, ICQ and others. One of them may be selected as a preferred one. Since then, it will be used as a default technology for all transfers.

The same applies when deciding upon choosing an appropriate export filter. Considering that the person does not use environment able to handle compound documents, object needs to be transformed to the file format suitable for recipient's system.

Live collaboration

Document might be made accessible via network using distributed object technology. It enables other remote users running the Étoilé system to work simultaneously on one shared document. Changes are immediately propagated to all users currently editing it.

2.2.6   Assistant technology similar to the Newton's approach

The last part of Étoilé vision is to provide an assistant tool. This technology is inspired by the idea which was build in the Apple Newton operating system.

This tool allowed user to type a statement in a natural language which was then converted into the set of steps performing the desired action.

Sean Luke describes the Newton's assistant technology in the article where he compares Nokia N800 with Apple Newton:

> The Newton has an application called "Assist", which basically consists of a text field and a "Do" button. ... In the Assist text field, I could write "Lunch Thursday with Glen in Fairfax". The system would realize that lunch is typically at 12:00 PM, then access one of the Names app's widgets to select which Glen I meant, and automatically figure out from Names where Fairfax might be. Then Assist would access the Dates application to register a calendar event with the relevant information.
>
> (Luke 2006)

Similar tool is planned to be a part of the Étoilé environment. This technology provides an assistance for simple everyday tasks. User types down a statement describing desired action in a natural language and system then automatically guides user through the set of steps that just need to be revised and confirmed.

2.3   Implementation background

Implementation of the concepts described above would not be possible without robust object oriented architecture underlying the whole system. Étoilé, unlike most other environments, adopts radical approach to the system design. It intends to be implemented as a fully dynamic, object-oriented system providing large extent of flexibility based on the built-in message passing system. This runtime platform holds information about the objects published in the system. All applications then exist in a common object environment which lets them share data and code freely as if it was one process.

Therefore, dynamic programming languages such as Objective-C, Smalltalk, Io or Ruby are the most suitable ones to implement the Étoilé system and its components.

2.3.1   GNUstep

After a critical consideration, GNUstep platform was chosen as a basis for Étoilé user environment. It provides set of object oriented frameworks based on the Objective-C language, which are ready for the development of Étoilé applications and services.

GNUstep community describe its product as:

> GNUstep is a cross-platform, object-oriented framework for desktop application development. Based on the OpenStep specification originally created by NeXT (now Apple), GNUstep enables developers to rapidly build sophisticated software by employing a large library of reusable software components.
>
> (GNUstep community 2007)

2.3.2   The current status of the Étoilé project

Étoilé project is under development for almost three years now, but it still have not produced any product that could demonstrate concepts stated in the project vision. The work has been done mainly on the clarification of the project vision, on the design of the underlying architecture and on the development of supporting libraries.

Étoilé version 0.2 has been released recently. It is a very basic self-sufficient environment with a few compatible applications such as an RSS reader, dictionary and basic text editor. However, this has been mainly done by bundling GNUstep system thogether with a few applications developed outside the Étoilé project. Étoilé project itself contributed a horizontal

menu, simple dock and a few frameworks such as LuceneKit for indexing and fulltext searching or IconKit for advanced icon manipulations. Some other libraries are in development but most of them just prototyped.

There are 16 registered contributors at the moment, of which about 7 - 8 seem to be committing on the regular basis.

3 USER INTERFACE DESCRIPTION

3.1 Introduction

Étoilé 0.2 does not include any of the innovative concepts described in the previous chapters. Therefore, to gather any relevant data about the user approval on them, it is necessary to work out the fictional environment that would describe what would be the behaviour and appearance of the future Étoilé system.

3.2 Task analysis

To describe the user interface elements and concepts used in Étoilé, series of fictional screenshots have been created. It shows the Étoilé system as if it was used to accomplish two typical desktop tasks. The attached CD includes these task analyses:

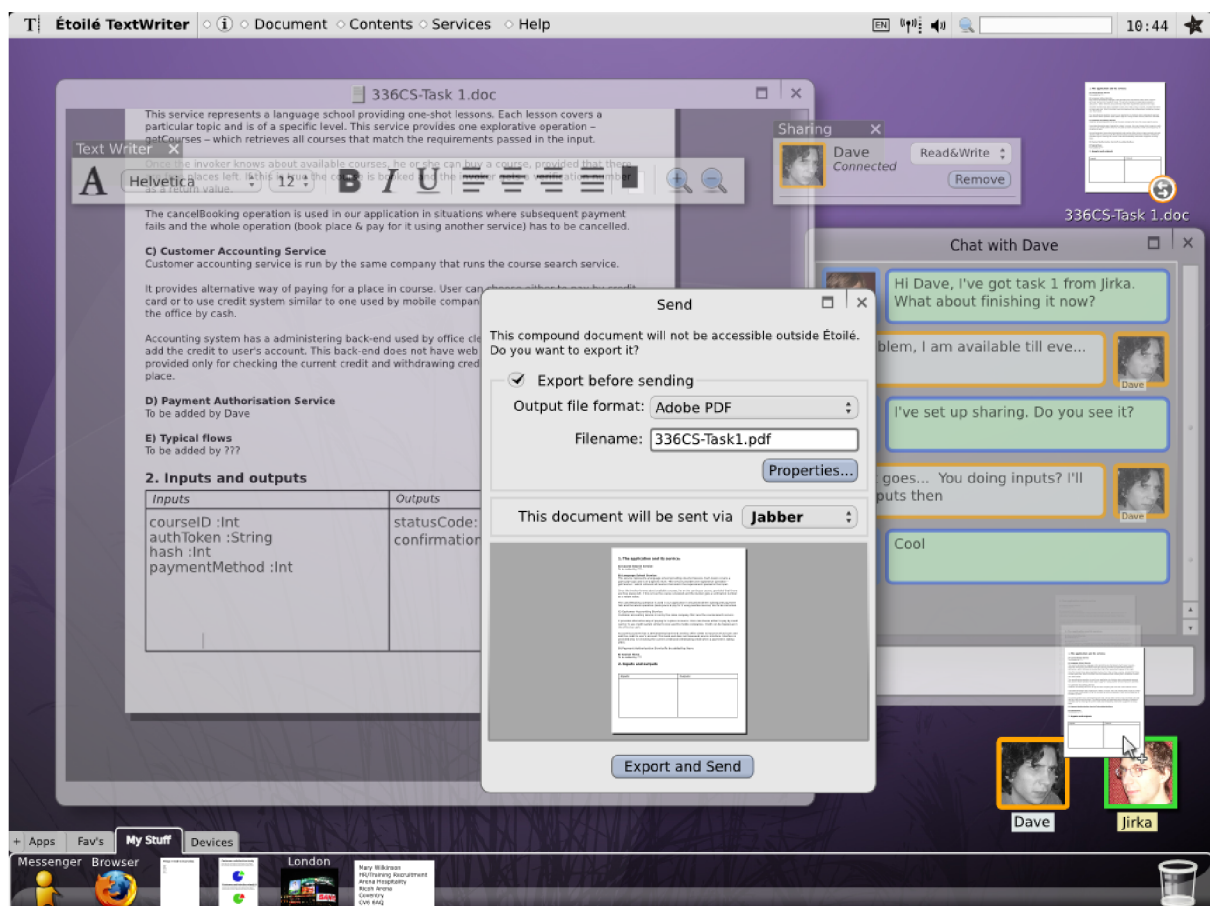- Task A – Creation of a covering letter
- Task B – Group project

Image 1: An example of fictional screenshot used in the task analysis B

3.3   User's approval research

The user interface principles discussed by Étoilé developers are supposed to be beneficial for general computer user audience. They were invented as a result of a theoretical work and may be intuitively evaluated as a better solution to the desktop usability problem. However, there is no feedback from people who are not involved in the Étoilé development which would prove that ordinary users are actually interested in such a radical change to desktop environment paradigms. A simple questionnaire asking about the user approval on the Étoilé concepts may help to resolve this doubt.

3.3.1   Method

As there is no product that would demonstrate the concepts used in Étoilé, another way of demonstration needs to be applied. Textual description does not fit very well for its limited expressiveness. Therefore, fictional screenshots with comments have been chosen. It is organized into series, where the system is pictured during the accomplishment of some typical desktop task.

These tasks were intended to show the most of basic concepts used in Étoilé. The first one describes process of creating a letter, the second was showing a collaborative work on a group project.

Both task analyses have been published on the Internet along with an automated questionnaire. This website has been asking visitors to read through the task analyses and complete the provided questionnaire.

It has been gathering information about the participant's background and experiences with other environments. Visitors could provide answers regarding their computer background:

(Possible answers are in the brackets along with their coding)

- What is your age?

- What is your job?

- Gender? *(male = "m", female = "f")*

- How would you describe your computer skills? *(never used it = 0, beginner = 1, intermediate = 2, upper intermediate = 3, amateur expert = 4, professional expert = 5)*

- Do you have any programming skills? *(yes = 1, no = 0)*

- Do you have programming skills in any dynamic language such as Objective-C, Python, Ruby, Lisp, Io or Smalltalk? *(yes = 1, no = 0)*

- How often do you use computer at home? *(almost every day = 5, a few times a week = 4, once or twice a week = 3, a few times a month = 2, less often = 1, never = 0)*

- How often do you use computer at work? *(almost every day = 5, a few times a week = 4, once or twice a week = 3, a few times a month = 2, less often = 1, never = 0)*

User could also select environments he or she has experience with:

- MS Windows *(yes = 1, no = 0)*

- Mac OS X *(yes = 1, no = 0)*

- Mac OS 9 *(yes = 1, no = 0)*

- KDE *(yes = 1, no = 0)*

- Gnome *(yes = 1, no = 0)*
- Unix (other) *(yes = 1, no = 0)*
- GNUstep *(yes = 1, no = 0)*
- Unix (command line) *(yes = 1, no = 0)*
- NEXTstep *(yes = 1, no = 0)*
- Irix *(yes = 1, no = 0)*

Then, visitors could express their level of agreement with the statements about the Étoilé environment on the five-point Likert's scale from the "strongly agree" to "strongly disagree". Respondents were asked to express their opinion on these statements:

- If this systems really existed I would switch to it.
- If this systems was ready to use I would be interested in it.
- Étoilé uses more intuitive user interface than the current environments.
- Étoilé is easier to learn for absolute beginners (people with no computer knowledge) than the current environments.
- Concepts used in Étoilé will be difficult to learn.
- In reality, absence of classic file formats will cause serious problems
- Étoilé would be more suitable for home usage than the current systems
- Étoilé would be more suitable for work usage than the current systems
- I am thinking about getting involved into Étoilé project

  In the survey results list, the data are coded as follows:
  strongly agree = 5, agree = 4, neutral = 3, disagree = 2, strongly disagree = 1

Besides that, there were three more qualitative questions, where participants were allowed to express themselves more fully in the text fields:

- Is there anything you particularly like in Étoilé?
- Is there anything you dislike in Étoilé?
- Any other comments or ideas? e.g. How to further improve user experience.

The whole survey has been published at this address:
http://www.stud.fit.vutbr.cz/~xcapmi00/etoile/.

The link has been distributed among computer users who have been asked to read instructions and complete the questionnaire. The czech leading Linux-related server root.cz agreed to display the link on its frontpage, which proved to be the major source of survey visitors.

3.3.2 Results

Altogether, 100 visitors completed the survey. The majority of them expressed positive attitude to the Étoilé project. Only 10% of all participants answered that they would not switch to Étoilé if it was ready to use.

Table 1: Results of user approval survey

	I would switch to Étoilé	I would be interested in Étoilé	Étoilé is more intuitive	Étoilé is easier for beginners	Étoilé would be difficult to learn	File formats will cause problems	Étoilé is better for home use	I am thinking to get involved
1	2%	0%	0%	1%	9%	2%	4%	6%
2	8%	2%	4%	4%	63%	24%	15%	27%
3	33%	5%	33%	34%	24%	26%	47%	39%
4	42%	44%	51%	46%	1%	34%	22%	18%
5	15%	48%	13%	15%	1%	13%	12%	7%

Scale: 1 = strongly disagree, 2 = disagree, 3=neutral, 4=agree, 5=strongly agree

Due to a technical fault in the questionnaire script, the answers for statement "Étoilé would be more suitable for work usage than the current systems" have not been gathered.

Typical respondent was man in the age group 20 – 30 with an IT background. Summarised data about the participants are as follows:

Table 2: Gender

Males	97%
Females	3%

Table 3: Computer skills

Beginners	1%
Intermediate	4%
Upper intermediate	16%
Amateur experts	35%
Professional experts	44%

Table 4: Programming skills

General	89%
Dynamic languages	61%

Table 5: Computer usage at home

Empty	0%
Never	0%
Few times a month	0%
Once or twice a week	0%
Few times a week	3%
Almost every day	97%

Table 6: Computer usage at work

Empty	13%

Never	8%
Few times a month	0%
Once or twice a week	4%
Few times a week	7%
Almost every day	68%

Table 7: Familiarity with other environments

Windows	95%
Mac OS X	35%
Mac OS 9	18%
KDE	79%
Gnome	82%
Unix (other)	69%
GNUstep	17%
Unix (command line)	26%
NEXTstep	10%
Irix	9%

3.3.3 Discussion

Étoilé concepts illustrated in the both task analyses were generally accepted. About 57% of all respondents answered that they would switch to Étoilé if it was usable. The vast majority of 92% said that they would be at least interested in it. This is far beyond the expectations although the surveyed sample contained mainly computer experts and programmers familiar with Linux.

It has important consequences considering feasibility of the project as it largely depends on motivated contributors who are, as the survey revealed, interested in a radical change to desktop environments. They mainly appreciated concept of component model and its similarity to the Unix command line philosophy.

Main doubts concerned the ambitions of the project. One of the respondents suggested that it would be more feasible to incorporate these ideas into Gnome 3.

Although both task analyses have been created to support the understanding of Étoilé innovative concepts, many respondents had apparent difficulties in understand what is the Étoilé project about. Many of them tended to judge the system by its appearance rather than by the concepts explained in the task analyses. It proved that the explanation of such an abstract idea is a very complicated issue.

It has also shown that there is a considerable number of the potential developers willing to contribute. There seem to be just two main barriers for new developers: mainly, it is the unclear explanation of project goals on its website. Participants were often confused about the purpose of Étoilé project. Secondly, some respondents refuse programming in Objective-C language.

Non-expert users are usually not interested in switching to the new environment, but appreciate it as an alternative option.

4 UNDERLYING ARCHITECTURE

4.1 Introduction

Étoilé is targeting not only richer user experience, it is also planning to adopt a full-featured, object oriented system design that is meant to support it. The vision of target system is to have a special Étoilé system layer on top of the actual low-level operating system. This system layer, called CoreObject, allows processes to publish their objects for system-wide or even world-wide usage. CoreObject in connection with dynamic, object oriented language may serve as a basis for the Étoilé object architecure.

4.2 Dynamic object environment

Object oriented programming seems to be the most popular software development methodology. It is being used for application programming in the virtually all operating systems now. However, Étoilé claims to be more object-oriented than all major operating systems today. It aims to break classic process black box and enable objects to talk together without constraints. Main system entity is not a process any more, it is an object.

Objects as executable entities can be dynamically loaded or unload on request. Objects can be accessed via network to behave like a distributed system. Objects are capable to hibernate, to save itself on a disk.

The metadata about objects used in a program are not lost once it is compiled. Thus, the structure and state of every object may be inspected during a runtime.

As application is just a bundle or interconnected objects, it is possible to replace any part of application without restarting it. Object are even able to modify their own runtime structure.

All these features are available in the environments based on dynamic languages such as Squeak, which takes advantage of Smalltalk language or OpenStep based on the Objective-C language.

4.3 GNUstep

GNUstep has been considered as the most mature opensource platform offering dynamic object environment and therefore chosen to be a basis for the Étoilé system.

GNUstep is open source clone of the NEXTSTEP frameworks providing complete library for Objective-C developers with support for GUI programming. Almost identical library with some modern add-ons is used as a primary development platform in Mac OS X.

Apple as a new owner of NeXT Computer advocates its main language:

> Compared to other object oriented languages based on C, Objective-C is very dynamic. The compiler preserves a great deal of information about the objects themselves for use at runtime. Decisions that otherwise might be made at compile time can be postponed until the program is running. This gives Objective-C programs unusual flexibility and power.

(Apple 2006)

4.4 Performance

Dynamic languages are often thought to be poor performing compared to the static languages such as C or C++. Which is apparently true for the languages that are interpreted or JIT (just-in-time) compiled from bytecode. But Objective-C is rather special. It is built on classic C language and adds just one language construct and few keywords to support true object-oriented programming. It is compiled into native machine code identically as C programs are. Dynamic features rely on a very small (about 66KB) message passing runtime. When used without dynamic features it has identical performance to classic C code. Dynamic features adds a little overheads, but due to their lightweight implementation perform faster than similar features in C# or Java.

Thanks to the hybrid nature of Objective-C, it can be comfortably used as a system language. It is evident from example of Mac OS X or NEXTSTEP where it was used as the main system language.

4.5 Ease of development

The introspection feature of the Objective-C runtime system makes the development process more interactive:

> Objective-C's dynamism enables the construction of sophisticated development tools. An interface to the runtime system provides access to information about running applications, so it's possible to develop tools that monitor, intervene, and reveal the underlying structure and activity of Objective-C applications.

> (Apple 2006)

By employing a dynamic object architecture, development may take place even on a running application. If there is just a few objects that need to be modified, it is possible to link them dynamically into running application during runtime. There is no need to recompile whole application.

Traditionally, rapid application development techniques are intensively used in connection with Objective-C. First rapid application development kit - Interface Builder - was part of NEXTSTEP system. It takes a full advantage of the dynamic features available in the language and offers a truly interactive development style.

GNUstep developers are confident about an efficiency boost arising from the Objective-C advanced features:

> GNUstep provides an excellent, mature framework for writing good applications. There's good anecdotal evidence that OpenStep developers have written very complex commercial applications in weeks or months, rather than years (or often, never) in the case of other development environments. Using the visual interface modeling application, you can construct a decent user interface skeleton in a day or so with no coding.

> (GNUstep community 2007a)

5 PROJECT FEASIBILITY DISCUSSION

5.1 Introduction

The Étoilé project has the ambitious goals and large scope. This chapter discusses its feasibility from the two different viewpoints: Firstly, it rises the question about the viability of the user interface concepts introduced by the Étoilé environment: Will they work in a real-world setting? Will users accept them?

Secondly, it brings up questions about the actual implementation: Will the Étoilé project develop the usable product in a reasonable time? Will the Étoilé community grow up enough to deal with the enormous amount of the implementation work waiting to be done?

5.2 Possible drawbacks

Concepts on which is Étoilé based have not proved viability in real world setting. Some issues that seems to be rather problematic are discussed in the next paragraphs.

5.2.1 Cooperation with the mainstream systems

It is very probable that Étoilé will exist as a minor system with most people using application oriented environments such as Microsoft Windows. Information interchange with them may be seen as a serious drawback of the Étoilé concepts.

Traditionally, data are saved into files, but Étoilé uses a different way to store data. Each component of a document is stored separately and bundled together only if this document needs to be transferred over a network.

Documents are not edited as a single element, but rather as a bundle of components, each one by its respective service. This concept makes it very impractical to natively edit classic monolithic documents such as MS Word "doc" in Étoilé. To take advantage of its component-driven technology, this MS Word document needs to be converted into the Étoilé compound document.

The same problem arises if user wants to transfer Étoilé document across the network to the person who uses application-oriented system such as Microsoft Windows. Data needs to be first exported into a file with a pre-agreed format. Although all the conversions are mainly done automatically behind the scenes, it makes the system usability depend highly on the quality of import and export filters.

It must be said that this problem can be partially overcome by running classic application to edit the document in a compatibility mode or outside the Étoilé system. Obviously, most of the Étoilé component-related features will not be available then.

5.2.2 User's resistance to the new concepts

The majority of ordinary users, who use computers just as a secondary tool to do their jobs, would probably resist any effort to change the principles they are used to. The vast majority of computer users has been taught Microsoft Windows or Mac OS X as the way computers work.

These concepts has been generally accepted and people might have troubles to understand that desktop computer can exist without files and applications.

Similar doubt has been expressed in the survey:

> I´ve never had any problems with using Windows. I don´t know any reason why should I learn any different system. From my level of PC knowledge (almost 0) I can´t see any difference between these systems. I am used to work with Windows.

> (the survey result, 27 year old teacher)

On the other hand, the majority of respondents with IT background or other professions with high computer utilization, such as graphic designers, expressed interest in the Étoilé ideas. More than half of them said that they would switch to that environment if it was ready to use.

5.2.3   Very few suitable applications

Different software architecture stressing components and services instead of applications implies the need to rewrite everything from scratch. Étoilé project is not able to take advantage of large repository of open source applications as they are usually written in static languages, which do not fit into Étoilé dynamic architecture. It is dramatically broadening project scope as all the functionality normally provided by large open source projects such as OpenOffice.org need to be rewritten.

5.2.4   Responsibility boundaries are not obvious

Strict boundaries set for applications by present operating systems affect flexibility but seem to serve as handy principle to let user determine application's responsibilities. As user deals with just one monolithic application during a work on a particular task it is obvious who to blame when anything goes wrong. Whenever application stops responding, user can make up that there is a bug in the application and it is therefore responsibility of the application's vendor to fix it.

In the service oriented environment, responsibilities are strictly set as well, but on the underlying, object interaction level, which might not be as naturally understood by the end users. User might not be able to determine which component caused a system instability and may blame the system architecture for it.

5.3   Complexity of implementation

Étoilé developers set themselves an ambitious goal and it is very questionable if their effort will ever end in a usable product. To illustrate how much programming work may be expected, Étoilé project modules are compared to the established projects with similar goals. The number of lines of code (LOC) is used as an approximate measure for the amount of work that needs to be done. The data about open source projects have been taken from server http://www.ohloh.net, which is able to automatically analyse their source repositories and generate reports about it.

Expressiveness of Objective-C

Different programming languages need different amount of source code to implement an identical task. Objective-C is often argued to be more expressive language than C++.

One isolated case will be used to illustrate difference in expressiveness between Objective-C and C++.

Small example:
Use xine-lib media engine to create a media player
GNOME's Totem: 27931 lines of C
xine-ui (the main xine player): 96791 lines of C
Cocoaxine on Mac OS X: 1436 lines of Objective-C
apfelXine (Mac xine front-end): 2653 lines of Objective-C

(Pang 2004)

Although this is not a representative research as Linux versions of xine frontends include many more features, it shows that similarly focused application can be coded with significantly less effort.

Etoile project has been divided into three parts to make the comparison easier: CoreObject, desktop environment and desktop applications.

CoreObject framework

Distributed object layer allowing universal access with broad language support. It consist of object name server, serialization engine and facilities for data storage abstraction. It is intended to be easier-to-use replacement of CORBA with some features of VFS (virtual file system).

Table 8: Sources for CoreObject estimation

Xstream	Simple Java serialization backend	28,179 LOC
TAO	Complete C++ CORBA implementation	867,094 LOC
JacORB	Java implementation of CORBA	180,876 LOC
GnomeVFS	VFS implementation for Gnome	84,474 LOC

Considering the light-weight architecture of CoreObject and fact that it builds on GNUstep distributed object implementation may lead to the approximate estimation of **150,000 LOC** for initial, but fully usable version of CoreObject.

Desktop environment

It consist of several parts:

● File/Object manager

● Étoilé specific frameworks (ComponentKit, FogVision etc.)

● User interface elements (menu, dock etc.)

Table 9: Sources for the desktop environment part estimation

Nautilus	File manager for gnome	104,435 LOC
XFCE	Lightweight desktop environment based on GTK+	600,879 LOC
Beagle	Fast full-text search	114,893 LOC

Considering better expressiveness of Objective-C and good support provided by GNUstep, the effort needed to create the first version of usable Étoilé desktop environment may be estimated to **500,000 LOC**.

Desktop services and applications

As the classic applications do not fit into the Étoilé component architecture, it will be necessary to implement new services for all basic desktop tasks: text editing, browsing, sending emails, drawing etc.

Table 10: Sources for the services and applications estimation

KOffice	Office suit for KDE	956,290 LOC
GNOME	Desktop environment + wide range of applications such as f-spot, evolution, GIMP etc.	16,407,706 LOC
OpenOffice.org	Office suit comparable with MS Office	10,793,134 LOC
MozzilaCore	Rendering core of the Firefox browser	2,609,461 LOC

The effort to develop services that would provide functionality similar to the current desktop applications could be approximately estimated to **10,000,000 LOC.**

The repository of Étoilé project now includes 313,394 lines of code. First piece of code was contributed 2,5 year ago and over the entire history of the project, 10 contributors have submitted code. (Ohloh 2007)

Considering that Étoilé will still follow its growing tendency, it may release its first usable version in the time of one or two years.

However, although base environment will be probably ready to use, it will not provide applications supporting full-featured desktop work.

5.4   Relation to the other attempts in past

There have been many attempts to redefine personal computing in the past – to provide fully integrated, flexible desktop environment along with a sophisticated object technology underneath. Although it is generally accepted as a better approach to operating system design, all those attempts have effectively failed as we are still using application based systems for everyday tasks.

It may lead to the belief that Étoilé is predetermined to fail as well, that there is something wrong with these concepts in general. But it does not necessarily need to be the case. Let's have a critical look on what caused these failures.

NEXTstep was a sophisticated product aimed to the research and education sector. It fitted well to that market, but it was tied to the hardware called NeXTcube, which proved to be problematic and overpriced. This caused serious sales problems considering that students were primarily targeted. (Deutschman, Alan 2001)

Another extraordinary operating system, Newton OS, was developed by Apple computer as the part of a product that aimed to redefine personal computing. It was initially supposed to be a device that we know as a tablet PC nowadays, but it was later transformed into PDA (personal digital assistant) in the fear that it would interfere with Apple Macintosh sales. The lack of easy communication infrastructure such as wireless networks at the time and

problematic cooperation with PCs made it difficult for users to take advantage of such an advanced device. (David MacNeill 1998)

Other attempts trying to employ pure object architecture usually suffered from trade-offs needed to keep the compatibility with traditional systems. Industry leading application software has been written using classic paradigms and it is impossible to expect its vendors to carry the costs connected with rewriting applications to fit into a new object-oriented environment. Likewise, main software products such as Adobe Photoshop, OpenOffice.org etc. tend to have identical user interfaces at all platforms to reduce the porting costs. This pushes software industry to maintain a user interface uniformity. Mac OS X may be taken as a good example. Although it is based on the object architecture derived from NEXTstep, it had to keep classic user interface elements (e.g. horizontal menu instead of vertical one) in order to make the transition easy for the users coming from previous systems such as Mac OS 9.

Demand for a maintained continuity with preceding systems seems to be an impenetrable barrier for the efforts trying to introduce a radical change to user environment design.

However, Étoilé is growing up from different conditions.

Contrary to the systems mentioned above, Étoilé is not expected to keep such continuity. It does not need to be complete desktop environment at the time it is released. As an open source project, it does not rely on a commercial success and smooth transition for the end users. Therefore it has better chance to resist the unification pressure and truly contribute to the redefinition of user interface concepts.

6 CONCLUSION

6.1 Project summary

The aim of this work has been to provide a document that summarises the goals of the Étoilé project, explains its innovative concepts and discusses the feasibility of its successful implementation. The analysis presented in this document has drawn mainly on the various bits of information produced by the Étoilé community and the literature discussing the relevant technologies and products.

Details about the project vision have been gathered on the project website and in the Étoilé mailing list archives. The project's vision is summarised in chapter two, which may also serve as a quick introduction into the Étoilé project.

In order to make this vision less blurred, a new fictional environment has been designed. This environment intends to reflect future vision of Étoilé project and illustrate its main concepts. It has been used in chapter three to research the user approval on the proposed concepts. Opinions from more than 100 respondents have been gathered and analysed. The survey results are available at the same chapter.

Underlying architecture is briefly explained in chapter four. The next chapter is discussing the feasibility of the project. It tries to estimate a quantity of work that needs to be done until Étoilé gets usable by comparing it with similar, established projects.

6.2 Summary of results

User interface description:

This work uses two different methods to describe the Étoilé user interface. In chapter two, it describes user interface concepts proposed by the Étoilé community. In the following chapter (and on the attached CD), it presents the fictional environment that employs these concepts. This may be used as a self-explanatory tool for the description of the Étoilé user interface.

Usability improvement study:

Both task analyses are intended to show the improved usability brought by innovative concepts used in Étoilé. Each one shows typical desktop task as if it was performed in the future Étoilé environment. No formal methods have been used as there is no existing system to which they could be applied to.

Feasibility study:

Chapter four is dedicated to the feasibility of the Étoilé project. Different aspects are considered. Possible drawbacks of the Étoilé environment are analysed and project is compared to the similar open source projects in order to estimate amount of implementation work behind the Étoilé concepts.

6.3 Feasibility study findings

As the survey revealed, ordinary users are not interested in such a radical change to user interface concepts. However, developers and technically-oriented users would appreciate it. Étoilé is therefore likely to become an innovative, experimental platform targeted to research and development tasks. As the estimated complexity of implementation has shown, it is simply unrealistic to expect Étoilé to become full-featured desktop environment capable of

office work in a short term. However, it may turn into an important source of innovative user interface ideas for the established main stream environments. It may act as an incubator for a new user interface concepts, which could be later incorporated into the large systems such as Gnome or KDE.

First usable version of Étoilé may be anticipated in two years' time considering the current speed of development. Whether it ever becomes an environment capable of full-featured desktop work is effectively unpredictable as it highly depends on the overall acceptance of the platform by developers.

The survey also pointed out that there is a large number of potential developers wiling to contribute. The project appears to suffer from a poor marketing and the incomprehensible explanation of its goals. Some developers also expressed doubts about the primary project language – Objective-C. Thus, Étoilé project seems to be able to significantly enrich its community by improving the project's promotion, clarifying its vision and improving support for other object oriented languages.

6.4   Critical Appraisal

This thesis is a very basic attempt to evaluate the viability of ideas proposed by the Étoilé project. As the topic of the work has been rather general, it have not applied any sophisticated formal method to evaluate the usability of the Étoilé user interface. The sample used in the user approval survey is not representative as it does not reflect the typical social profile of a computer user. More balanced sample would mean to survey more beginners, which in turn requires to prepare very detailed and explanatory task descriptions.

Similarly, estimation of the complexity uses a very basic method and may prove to be considerably wrong. However, more accurate estimation relies on a more precise development plan than the one available now.

The task analyses do not cover all the concepts proposed by the Étoilé project. It would be necessary to create several more just to illustrate all the basic concepts. Although it was quite incomplete description of Étoilé environment, it gained lot of participants' attention.

6.5   Further work

As this thesis has had rather general scope, there exists a potential to go deeper and research on any of the points raised here. There is a possibility to choose one specific concept of Étoilé user interface and prove its usability using some strong formal method. Another suggestion may be to formally compare programming languages Objective-C and C++ in terms of performance and expressiveness. Other option would be to implement some part of the Étoilé's underlying architecture. I would suggest serialization engine in CoreObject to be concrete.

6.6   Personal evaluation

Although I did like the ideas raised by the project community, I expected this thesis to prove that the Étoilé project is rather infeasible. According to my research, idea of complete full-featured environment seems to be rather unrealistic in short or middle term, but it does not mean that the entire Étoilé project will fail. It will probably become a very interesting experimental platform for research and development in some reasonable time.

After a deep research not only on Étoilé itself, but also on technologies and systems related to it – such as dynamic programming, NEXTstep, Apple Newton, Smalltalk Squeak and

others – I started to work on fictional system design. Once I had finished the first task, I sent it to the Étoilé mailing list. After an attention it had gained I kept in touch with them. Mainly with artistic leader of Étoilé – Jesse Ross. The second task was therefore more influenced by the community's comments. As the screenshots I had created provided a more concrete draft of Étoilé environment, it set up a new vital discussion on the topic. The community even started to produce new mock-ups and internal task analyses based on my design.

When I released the questionnaire I experienced some difficulties to convince people to take part on it. Non computer-oriented people often could not understand the purpose of the research. Luckily, I managed to publish link to the questionnaire on the frontpage of the Czech leading Linux server called root.cz. The response was much larger than I would ever expect.

Finally, I had to compare complexity of implementation. I registered Étoilé project with the ohloh.net server, that can automatically analyse source code repositories and display reports about a number of lines of code used in a project. I had to download the sources of the projects that have not been included there and analyse it myself using special tools on my computer.

During the work on my thesis I have developed a strong interest in Étoilé project and I am considering to stay involved.

REFERENCES

KC Daily Record Staff (09/21/2005), **American Bar Association releases survey of lawyers' technology usage**, Daily Record, The (Kansas City, MO)

Hiltzik A. M. (1999), **Dealers of Lightning: Xerox PARC and the Dawn of the Computer Age**, HarperCollins, New York

Ross J., Mathe Q. et al (2006), **EtoileWiki: About**, [Online] Available: http://www.etoile-project.org/etoile/mediawiki/index.php?title=EtoileWiki:About

Mathieu R. (2006), **News: The Project Étoilé**, [Online] Available: https://gna.org/forum/forum.php?forum_id=1046

Mathé Q., Ross J. (2005), **Etoile Project: Glossary**, [Online] Available: http://www.etoile-project.org/etoile/mediawiki/index.php?title=Glossary

Mathé Q. (2005), **Étoilé Project: Components**, [Online] Available: http://www.etoile-project.org/etoile/mediawiki/index.php?title=Components

Ross J., Urbánek S. (2005), **Étoilé Project: Document**, [Online] Available: http://www.etoile-project.org/etoile/mediawiki/index.php?title=Document

Chen Yen-Ju (2006), **Etoile Project: User Interface Glossary**, [Online] Available: http://www.etoile-project.org/etoile/mediawiki/index.php?title=User_Interface_Glossary

Luke S. (2006), **The Nokia N800, And Its GUI Shortcomings From a Newton Developer's Perspective**, [Online] Available: http://www.cs.gmu.edu/~sean/stuff/n800/

GNUstep community (2007), **GNUstep overview**, [Online] Available: http://www.gnustep.org/

Apple Computer, Inc. (2006), **The Objective-C Programming Language**, Apple Computer, Inc.

GNUstep community (2007a), **GNUstep - about GNUstep**, [Online] Available: http://www.gnustep.org/information/aboutGNUstep.htmlhttp://www.gnustep.org/information/aboutGNUstep.html

Pang A. (2004), **Beyond C, C++, Python and Perl**,

Ohloh (2007), **Ohloh - Étoilé analysis report**, [Online] Available: http://www.ohloh.net/projects/4841

Deutschman, Alan (2001), **The Second Coming of Steve Jobs**, Broadway

David MacNeill (1998), **Why did Apple kill the Newton?**, Pen Computing Magazine

Bibliography

Mandel Theo (1993), **The GUI-OOUI War, Windows versus OS/2,** Von Nostrand Reinhold, New York

Stefan Görling (2003), **A critical approach to Open Source software**, [Online] Available: http://opensource.mit.edu/papers/gorling.pdf

APPENDICES

Appendix A – Project management documentation

Log Book

Activity	Date	No of hours
Research on the Étoilé project	2.11.2006	6
Project Brief preparation	10.11.2006	5
Research on the project goals	6.12.2006	4
Research – the Etoile project	18.12.2006	3
Report writing requirements research	19.12.2006	4
Research – the Etoile project	29.12.2006	3
Research – the Etoile project	30.12.2006	4
Learning Objective-C	1.1.2007	3
Installing Étoilé	2.1.2007	4
Research on dynamic languages (ObjC vs. C#)	3.1.2007	5
Research on dynamic languages	3.1.2007	3
Research on OLPC project	4.2.2007	5
Research on OLPC project	6.2.2007	5
Research in HCI field	8.2.2007	2
Design of fictional target environment	26.2.2007	4
Design of fictional target environment	28.2.2007	1
Writing report – chapter 2	4.3.2007	9
Creating mock-ups and preparing for presentation	9.3.2007	4
Writing report – chapter 2	14.3.2007	7
Creating mock-ups (Task A)	23.3.2007	5
Creating mock-ups (Task A)	24.3.2007	8
Creating mock-ups (Task A)	25.3.2007	11
Creating mock-ups (Task A)	26.3.2007	10
Creating task analysis webpage (Task A)	27.3.2007	14
Creating task analysis webpage (Task A)	28.3.2007	7
Creating mock-ups (Task A)	29.3.2007	12
Creating mock-ups (Task B)	30.3.2007	13
Writing report – chapter 1	31.3.2007	6
Creating mock-ups (Task B)	1.4.2007	10
Report writing – chapter 1	2.4.2007	12
Report writing, Task A shown to Etoile community	3.4.2007	10

Creating questionnaire	4.4.2007	13
Finalizing questionnaire, published at root.cz	5.4.2007	12
Gathering and analysing results, discussion with Étoilé comunity	6.4.2007	12
Writing report – chapter 3	8.4.2007	10
Analysing the complexity of implementation	9.4.2007	11
Discussions with community about implementation, gathering data about similar projects	10.4.2007	10
Writing report – chapter 4, 5	11.4.2007	13
Writitng report – chapter 6	12.4.2007	12
Revisions	13.4.2007	14
Total		**306**

Project gant chart
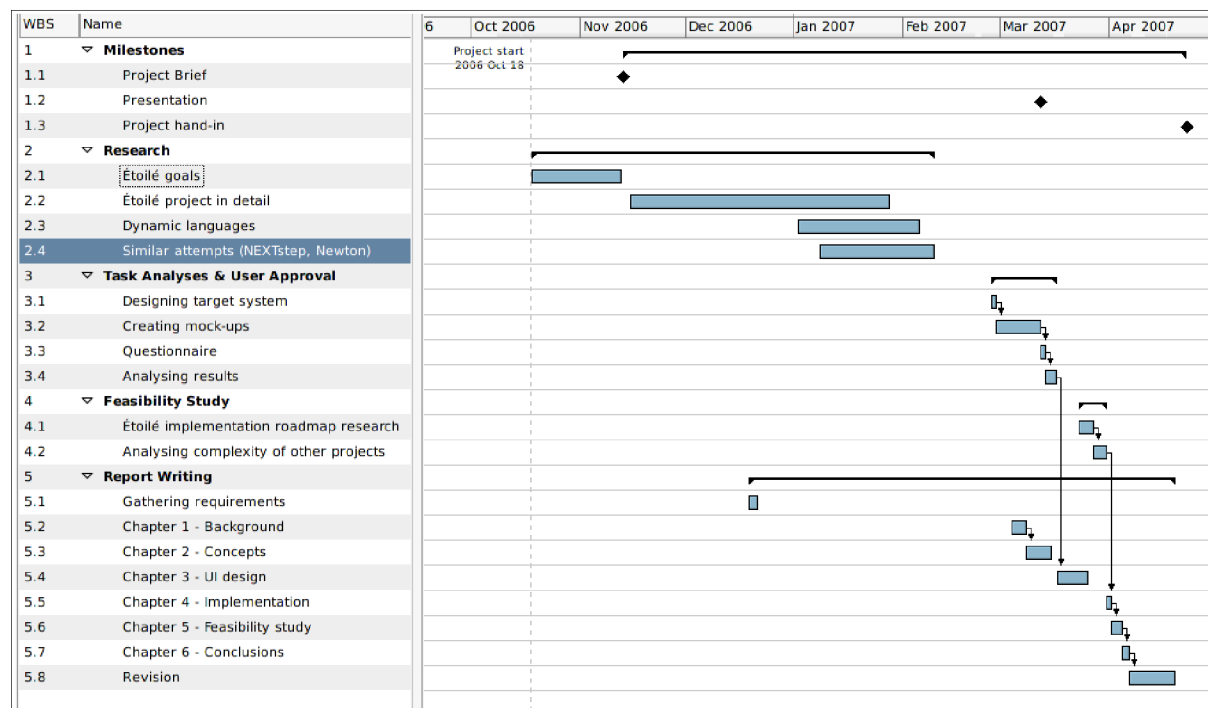
Contents of the attached CD

report.pdf	Electronic version of this report
survey_results.xls	Data gathered from the questionnaire
task_analyses/	Fictional task analyses used to illustrate Étoilé user interface concepts
UI_design_sources/	Inkscape source files used to create fictional user environment