BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
FAKULTA ELEKTROTECHNIKY
A KOMUNIKAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF FOREIGN LANGUAGES
ÚSTAV JAZYKŮ

INTERACTION DESIGN WITH REGARDS TO THE END USER
INTERACTION DESIGN S OHLEDEM NA KONCOVÉHO UŽIVATELE

BACHELOR'S THESIS
BAKALÁŘSKÁ PRÁCE

AUTHOR
AUTOR PRÁCE
David Macura

SUPERVISOR
VEDOUCÍ PRÁCE
Mgr. Pavel Sedláček

BRNO 2018

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Angličtina v elektrotechnice a informatice**
Ústav jazyků

Student: David Macura ***ID:*** 147393
Ročník: 3 ***Akademický rok:*** 2017/18

NÁZEV TÉMATU:

Interaction design s ohledem na koncového uživatele

POKYNY PRO VYPRACOVÁNÍ:

1. Úvod
2. Interaction design
3. Uživatelské rozhraní
4. Moderní požadavky na interakci a rozhraní
5. Závěr

DOPORUČENÁ LITERATURA:

Dix A., Finlay J., (2004) Human-Compoter Interaction, 3rd Ed., Pearson Education Limited, Essex, 834pp., ISBN-13: 978-0-13-046109-4

Galitz W.O., (2002) The Essential Guide to User Interface Design 2nd Ed., Wiley Computer Publishing, 760pp., ISBN: 0-471-084646

Termín zadání: 9.2.2018 ***Termín odevzdání:*** 25.5.2018

Vedoucí práce: Mgr. Pavel Sedláček
Konzultant:

doc. PhDr. Milena Krhutová, Ph.D.
předseda oborové rady

Anotace

Tato práce je zaměřena na uživatelská rozhraní, postupy spojené s jejich návrhem a vývojem a interakci mezi počítači a uživateli. Jejím cílem je kodifikovat a prozkoumat nejvýznamnější z těchto postupů a jejich vliv na výsledné produkty, dále popsat teoretickou stránku procesu návrhu a přístup designerů a uživatelů k designu a interakci. V části zaměřené na uživatelská rozhraní má práce za cíl prozkoumat a porovnat vybrané přístupy k tvorbě rozhraní a zmínit nejdůležitější vlivy na požadavky týkající se vývoje rozhraní pro aplikace.

Klíčová slova

Design, Software, Vývoj, Aplikace, Digitální gramotnost, Informační technologie, Interakce, Uživatelské Rozhraní

Abstract

This thesis focuses on user interfaces, methods connected with their design and the interaction between humans and computers. The goal is to codify and explore the most important of these methods and their influence on final products. Another goal is to describe the theoretical side of design and the approach of designers and users to design and interaction. In the part focused on the user interfaces the thesis sets the goal to research and compare select approaches to interface creation and to list the most important influences on demands concerning the application interface design.

Keywords

Design, Software, Development, Application, Digital literacy, Information technology, Interaction, User Interface

Bibliografická citace

MACURA, D. *Interaction design s ohledem na koncového uživatele.* Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2018. 36 s. Vedoucí bakalářské práce Mgr. Pavel Sedláček.

Prohlášení

Prohlašuji, že svou bakalářskou/diplomovou práci na téma Interaction design s ohledem na koncového uživatele jsem vypracoval samostatně pod vedením vedoucího bakalářské/diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské/diplomové práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské/diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.“

25.5.201 **.....**

Podpis

Acknowledgement

I would like to thank my supervisor for his guidance and patience through the creation of this thesis.

I would also like to thank my family for enabling me the opportunity to study and for support than allowed me to finally reach the end of a third year of a bachelor's program.

Table of contents

1. Introduction

This thesis deals with a very broad set of topics often collectively called the Human computer interaction (HCI). HCI describes and explores the way humans use computers and, to a degree, machines in general. Another part HCI relates to is the way computers present themselves to their users and provide them with feedback and new information – HCI covers the cooperation between man and machine. The topics related to HCI include elements of sociology, psychology, economics, ergonomics etc. For the purpose of this thesis, the focus will be on parts of HCI related to information technology directly and parts of the social and psychological aspect.

The three main chapters of this thesis focus on the thought processes, principles and reasons behind the design and creation of user interfaces. Chapter 2 primarily deals with terminology and theory behind designing interfaces. As the terms used to describe the interaction design process are not universal, this part of the thesis attempts to list and describe the most important ones.

Chapter 3 deals with interfaces and the technology behind them. It also deals with the way humans perceive user interfaces (UI), the main types of UI, their elements and traits of the main types of UI in use today. This part of the thesis includes comparison between typical properties of UI used in certain fields.

Chapter 4 shortly touches on the social aspect of UI design. How the needs of users and designers reflect on the design process and several methods of receiving information from them are both detailed in this chapter.

2. Explanation of interaction design

Interaction between computers and users is a method, or a collection of methods, utilized by the users to produce desirable results. In the context of this thesis, interaction is the way that users use computers as tools and the way these tools act in communication.

The term *interaction design* was first proposed by Bill Moggridge, (designer and co-founder of the company IDEO). It was proposed as a methodology for creating interfaces with the goal of end users in focus. Interaction design includes the creation of both hardware and software. [1, p. ix]

There are many ways the interaction between the user and a system can be conducted, from a barely interactive system to a system that is interacted with constantly. Some of the examples from this spectrum include [2, p. 124]:

- A system where the user provides input once while receiving little to no feedback, leaving the machine to perform the task (e.g. setting a timer on a coffee maker, then leaving)
- A system with periodic inputs that can be planned, unplanned, automatic or manual (e.g. using an antivirus software on several folders one by one)
- A system with constant feedback that allows for arbitrary input(e.g. monitoring and control software for a drilling machine)
- A highly interactive system with constant input and feedback provided between the user and the machine (e.g. virtual reality)

Systems providing no feedback are highly limited in the functionality they can provide and the tasks they can perform. For the purposes of this thesis, systems closer to the “more interactive” side will be considered.

The next subchapter deals with the basic framework of interaction, as well as the loop of utilizing systems as tools, the Execution – Evaluation cycle.

2.1 Basic framework of interactions

The basic interaction framework breaks the communication between a user and a system into four main components, as stated by the "Human Computer Interaction" book:

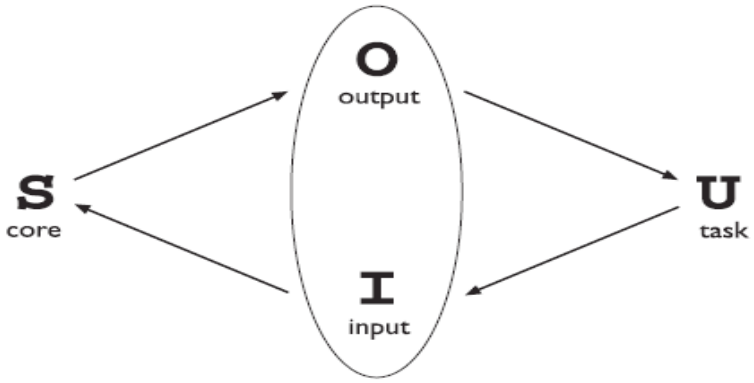

Figure 2.1: Application framework

The four components are **U**ser, **S**ystem, **I**nput and **O**utput. Figure 2.1 illustrates the basic relation between them: A user can influence a system only through input and a system can only present results through its output. A user presents a task to the system, which is then processed in the core of the system. [2, p. 127-130]

The purpose of an interaction is for the users to accomplish a pre-determined goal through their communication with the system. To this end, interaction design often approaches problems from the end, with the goal being the first, most important step in the design process. This approach is called the "*goal-oriented design*", as codified by the designer and programmer Alan Cooper in his book About Face [3, p. 25].

Execution – Evaluation cycle

The Execution–Evaluation cycle describes the basic sequence of interactions between a user and a computer: Coined by the usability consultant Don Norman in his book Design of everyday things. [4, p. 38-43]

Norman defines the cycle as a seven-part mental process from the side of the user.

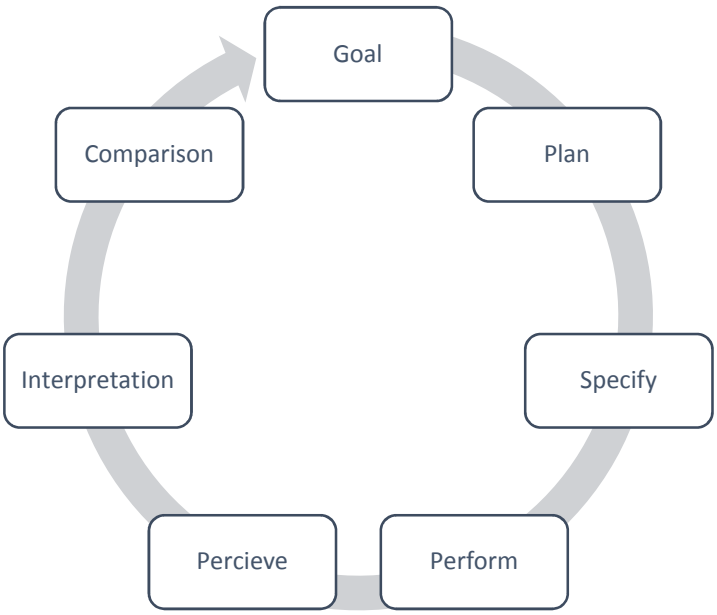

Figure 2.2: Visual representation of the Execution-Evaluation cycle

The full description of the parts is as follows:

1. Setting a goal
2. Forming the intention
3. Deciding on a sequence of actions
4. Executing the actions
5. Evaluating system feedback
6. Interpreting system feedback
7. Comparing the system state with respect to the goals and intentions

At the beginning of every process is the intention behind the process. The desired end state (step 1) and the way the state should be reached (step 2). The intended result, and the way the result is to be reached, become the intention and this intention then has to be translated to a machine, in a way the machine is designed to accept as a sequence of tasks, so the actions can be executed (4). The last steps (5, 6 and 7) are an evaluation of the new end state and the decision of further actions. If the end system state matches the desired result, the task is complete. Otherwise a new goal must be set, and the process is repeated.

Norman's model demonstrates one possible type of problems that interfaces may face in usage. The interaction can only be a success if the system's allowed actions match the actions intended by the user.

The effectiveness of an interaction can be measured by the amount of effort (and, consequently, how many times the Norman's cycle has to be performed by the users) a user has to expend to reach a desired goal. The lower the amount of effort, the more effective an interaction is.

Example: A user wants to crop the borders of an image and the image editor presents them with three options labelled "cut", "trim" and "take" with no visual indication. The ambiguity of the interface may potentially cause the user to try all three options, needlessly losing time that could be saved by visual indication of the action.

This cycle and the interaction framework are the most basic form of representing the way users and machines interact. Interaction design is based on these foundations. The basic parts connected to design will be outlined in the next subchapter.

2.2 Interaction design and design process

Interaction itself can be separated into different layers, according to the function and effect it has on a user. The layers, as codified by Gillian Crampton Smith (an interaction design academic) in an interview featured in the book "Designing interactions" [1, p. xvii-xix], are used to represent the various ways an interaction can affect a user. Smith herself described the layers as "languages" and "dimensions", likening the interaction design as "equivalent to the early stages of the cinema and therefore lacking fully developed language". The four dimensions defined by Smith are [1, p. xvii-xix]:

- **1-D:** Including words and their usage, the consistency of their usage and the "tone of voice" used for the messages
- **2-D:** The graphical dimension of design, performing a similar role to those of pictures, paintings and diagrams. The typography of a screen can imply depth in an otherwise completely flat image. The images can convey far more complexity with far smaller space than large amounts of text. A special case belonging to this layer are icons, simple images that stand as a representation of a larger, more complex object.
- **3-D:** The physical dimension. The means of control and interaction with a system for a user. This dimension includes various control elements and hardware used for feedback (monitors, headphones etc.)
- **4-D:** Time: A dimension that includes elements and objects and, most importantly, the user's own movement through time. This dimension includes sound and motion pictures.

Additionally, a fifth dimension was defined by Kevin Silver in his 2007's article "What Puts the Design in Interaction Design" [5]

- **5-D:** Behaviour: Including action, or operation and presentation or reaction.

Silver summarizes that all but the last layer enable interaction and time and behaviour define it. The first three layers could be used to characterize the input and output, and behaviour along with time describe how a system reacts to user input and provides feedback.

2.2.1 Interaction Design

The preceding texts served to establish the relationship between a user and a system in the context of the person using the system as a tool intended for certain tasks. In the article "What is Interaction Design?" Teo Siang (visual designer and a member of the Interaction design foundation) describes the term as "having a huge overlap with User Experience" [6]. Interaction design is the blanket term for the design of interaction

between a user and a system, with the systems connected with the term most often being software applications, websites and similar.

User Experience is the total sum of satisfaction and dissatisfaction that a user feels when operating a system. Many factors contribute into it, including visual design, ease of use, enjoyment, time spent using the system and others. One of the goals of design is to increase user satisfaction by providing a positive User Experience.

Interaction design is a process of making and planning for interactions. The book "Human Computer Interaction" describes design as "achieving goals within constrains" [2, p. 193]. In the context of this thesis, this definition could be paraphrased as "achieving user-defined goals within constrains presented by a system". Design is a process which evaluates, utilizes and builds upon using limited resources to create or modify an interaction or, more specifically, a user interface. In the case of designing interactions, some of the resources considered could be:

- Time
- Space (on screen or otherwise)
- Electricity
- processing power
- Materials
- Money

Constraints of UI design stem from the technological state of available hardware, the level of understanding of information systems within a group of users (Digital literacy), possible health and safety issues, effects of the environment (mainly for hardware) etc. One of the most important facets of design is the allocation of resources (with regards to constraints and goals) and finding compromises between desired results and available means.

Example 1: When developing a voice-controlled interface for a factory, omni-directional microphone was initially chosen as a main recording option. During initial testing, the microphone was deemed too sensitive, as it recorded all background noise, resulting in an unintelligible input. In the next cycle of testing, directional microphone with additional covers was introduced to eliminate the background noise.

Example 2: When changing platforms of an application (such as porting a mobile app to the PC), the designers have to consider how to utilize the screen space available on the new platform. A quick-and easy solution could be to have a forced display size, so the application appears the same way as on the mobile platform.

Another solution could be to force the interface to increase with the size of the new display, resulting in highly oversized, but functional UI. A more complex, but user-

friendly solution is to recreate the UI from the beginning with the increased display size in mind, even possibly adding new functions to fill in the space.

As illustrated, excess of resources can pose a problem in a similar way to the lack of them. The breaking down of interaction design into dimensions can help with describing the design process and the parts it influences. Identifying tools and obstacles can give developers the needed direction when developing an interactive software or hardware. The next chapter will focus more on the design process itself and the steps that can be followed in a goal-oriented design.

2.3 Design process

Goal oriented design is based on the fact that users will be satisfied when they accomplish their goals with the least amount of effort. The challenge designers face when creating interfaces is to use their limited resources in an effort to make the interface clear and readable. Additional time and effort can then be expended to reach as many users as possible, by localizing the interface in different languages, investing in graphical or sound improvements etc.

For proper utilization, designers have to understand the elements they are working with: the capabilities and limitations of computers and the motivations and psychological aspects of humans. The fact that humans can and do make mistakes and the fact that computer technology still continues to evolve should both be considered during the design process.

A design process can be both linear and iterative, depending on the circumstances. In its simple form, the design process can be considered to consist of three major parts: research, development (consisting of analysis and design) and implementation.

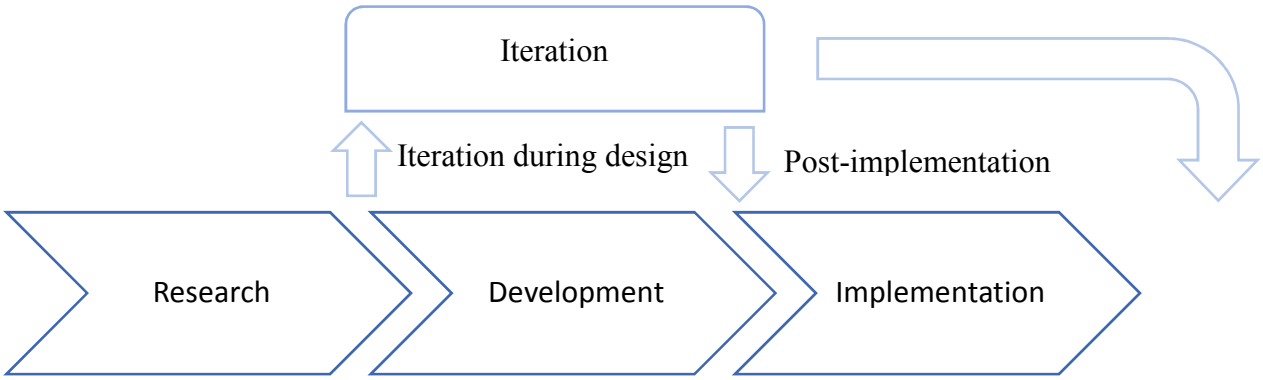

Figure 2.3: Visual representation of the design process

The book "Human Computer Interaction" breaks down the design process as being composed of five major parts [2, p. 195-196], which can be described as follows:

1. **Research:** The first step is the establishment of exact requirements. This is often preceded by finding out the present circumstances, for example: "What is the most common complaint discussed at the workplace?" This analysis can either be done directly, by interviewing a focus group or arbitrarily chosen individuals, or by observing their behaviour in either a controlled environment, or an uncontrolled one.

2. **Development – Analysis:** The results from observations and interviews need to be sorted and evaluated to reveal key issues and to give direction to the design

process. This stage could also be labelled as the "Planning" stage. The gathered data, together with possible pre-established goal, can influence the rest of the process in a dramatic way.

3. **Design:** The namesake stage and the main development stage where the resulting product or a service is given shape. An important part of design itself is archiving – recording the steps taken during design and the results of possible previous iterations to possibly be reviewed at a later date.

4. **Prototyping:** The design process can be very complex in practice, and the results are often not satisfactory on the first attempt. Various miscalculations of available data, wrong assumptions towards users or even the human error of the interviewees can lead to the necessity of repeating the design process. With certain products, feedback is almost impossible to get without a form of a prototype. The prototyping continues until the result is deemed satisfactory.

5. **Implementation:** Once the result is considered suitable enough, the designing process concludes, and the result is finalized, created and deployed among the intended users. This includes the writing of a code, assembling controls and all the parts that can be given to users. Depending on the type of product or a service, the design process can be repeated with the finished product, to add requested features or to remove errors only revealed by the stress testing of the full deployment.

Design accounts for many factors: space, budget, materials, and even psychological motivations of people who will utilize the final product. A designer often cannot consider all of these, as much of the information is unavailable, or too laborious to acquire. The lack of information itself can be considered as part of the constraints that the designers have to deal with. As long as a service or a product remains profitable for its providers, it can often be desirable to hold back the implementation part of the process until all major known issues are fixed, as it is the most resource and funds heavy part of the process.

Example: A Russian company deployed a new tablet-based inventory system for their logistics branch. Within an hour, the branch supervisor brought the tablets back with large blue error messages. The designers of the system failed to identify the need to accept input in Cyrillic and the application had to be reworked, costing time.

This subchapter touched upon the progress of a design process and its parts. When designing real interfaces, the process can be much less defined, and the step of research can be skipped if the end goal of users is already known. The following subchapter will deal with usability, with the general rules of design and will summarize one type of these rules – *the principles*

2.4 Design rules and principles

The way designers plan and create interfaces has many different rules that depend on the specific product being designed, but also a set of general rules, compiled from previous experiences with design, feedback on finished interfaces and applications, focus test groups and other factors. Following the general rules helps designers to achieve high levels of *usability* of the designed interface.

Usability, as defined by the Interaction design foundation [7], is another subset of User Experience and a general term for the ease-of-use of interfaces.

Example: An example of highly usable website would be Wikipedia.org. The site has a clear visual style, is easy to navigate and every major term is cross-hyperlinked between articles.

Figure 2.4: themagicofbaltimore.blogspot.com title page

In comparison, figure 2.4 shows a less usable website as it is first presented to visitors. The large banner, poor choice of font and cluttered site options all detract from the ability to navigate and from the User Experience in general. The site's search button, already hidden in the upper left corner, can easily be scrolled away from and entirely missed, which is another poor design choice.

While a design process is largely dependent on the desired goal, there is a degree of standard that designers can use to maximize the usability of resultant products. The design rules assist the designers in this effort by outlining features that the resulting product should exhibit.

This does not necessarily mean that the rules are rigid dogmas to be followed in every circumstance. Their interpretation can be both general and specialised, with more general rules more likely to contradict other rules to some degree.

The book "Human computer interaction" divides the rules into **principles, standards** and **guidelines** [2, p. 258-273].

Principles are not technology-specific and depend on the understanding of the human element of the interaction. They are derived from the experiences, records and reported reactions of users in typical model situations.

Guidelines are more oriented on technology and are less specialised, covering a broader range of topics. However, guidelines carry an increased risk of contradiction, therefore necessitating that the designer knows the underlying theory.

Standards are the most definitive of the three types of rules, and require less knowledge of underlying theory to be properly utilized, yet due to the higher "authority", they have to be much more accurate as a result.

The summary of design principles

The sum of all rules is too vast to be fully listed in a single chapter. This subchapter will briefly summarize the principles of interaction design, as principles focuses the most on interaction design theory.

Principles – the general principles are the most abstract part of interaction design of the three rules. In "Human Computer Interaction: Design Rule", Sarah Issack (a publisher for Supinfor.com) describes that the reason of defining principles (and rules in general) lies in the need to describe why certain algorithms were successful while others failed. [8]

Principles can offer a degree of repeatability into the otherwise arbitrary choice of design elements. They can be divided into three main classes, which are further divided into subclasses, as defined by the "Human Computer Interaction" book:

- **Learnability**

This class is involved with the ability of users to attain basic proficiency in using an interactive system, and the speed at which they are able to become fully proficient with it. One part of learnability a part concerning the ability of users to identify the options of an interactive system – *predictability*. This class of principles operates with an established assumption that humans have superior ability to recognize new patterns over recalling previously encountered ones. Predictability is differentiated from the inherent attribute of computers to produce expected results and concerns the design elements of interfaces and controls.

The system's predictability decides the ability of users to recognize at a glance which actions they can perform with the interactive system and which they cannot. In general, having a more predictable system is more desirable, as it enables users to become familiar with it faster, whereas an unpredictable system can cause delays, misunderstandings due to unexpected outputs and errors caused by unfamiliarity towards the system responses.

Another subclass is the *synthesizability.* Where predictability is connected to the ability of users to gain insight into the system without previously observed modification, synthesizability considers the ability of users to observe the results of past changes and their effects on the current system state. Internal changes of state should eb presented immediately, without further prompt. For example, when a new application is installed on the hard drive, it should be visible in the file system as soon as the install process is done.

The immediate feedback allows users to have precise overview of the system status and all changes happening to it at the cost of greatly increasing the number of variables the users must focus on at any given time. The issue with not highlighting changes to the system as they are made is that users have to look for the change. When users are unfamiliar with the interactive system they utilize, they may not be prompted to expect or observe the changes to the system, reducing their insight into the system state. An application that would change a save location of its files mid-project and only notifies the users after its next start-up could cause delays as the users attempt to find the moved data.

The other subclasses considered under learnability are *familiarity, generalizability and consistency.* All three previously mentioned subclasses deal with similarity between systems. When approaching a new system, users rarely have no previous experience whatsoever. Familiarity correlates between users' previous knowledge of other systems and the knowledge required for successful interaction. For example, when word processors first emerged, the input was purposefully fashioned after typewriters, to make the transition from using one easier. Regardless of intention, the previously encountered systems shape the users' expectations of new ones.

The subclass of generalizability supports features and traits of systems that were never encountered before by users but are similar in function or form to those of previously encountered systems. Consequently, a system designed with generalization in mind will have design elements that resemble those encountered elsewhere, e.g. a colour mixing tool representing choice of colours in word processors. Generalizability and consistency are very similar, with the difference that consistency is referred to when considering the system and its parts by themselves.

The learnability class of design rules supports a design philosophy that allows users to recognize the functions of an interactive system faster, easier and more consistently. The largest advantage the principles contained in its subclasses offer is speed and ease-of-use for both new and returning users.

- **Flexibility**

Where Learnability focuses on aiding the users in understanding and utilizing the functions of a system, as well as exchange of information, flexibility concerns the information exchange between the user and the system. The subclasses of flexibility begin with *multi-threading*. A thread is considered to be a part of an interaction. This subclass is centered on the ability of a system to perform multiple actions at the same time, as well as start new actions while previous ones are still being executed. A system should allow the users to utilize many actions at the same time.

The other subclasses are *task migratability, substituvity,* and *customizability.* The first two subclasses both focus on substitution of aspects of a system, migratability of control, and substituvity of equivalent values. Both also consider the choice of users in the matter. A user using an automated system has their control temporarily taken away to ensure that the task concludes correctly. This is an example of a purposefully inflexible system. On the other hand, substituvity lies in the user being allowed a choice in the form and state of values and information; e.g. the substituvity of input: specifically entering a value of 4 and inputting a formula 2*2 both produce the same value as input.

The customizability subclass concerns the ability of users to make changes to a part of a system presented to them – the user interface. A flexible system would allow the users to change and modify its form to better suit their needs, even removing functions that they consider redundant. Flexible customizability should be performed by the user.

- **Robustness**

The class of Robustness in interaction covers features that can support and assist users in attaining their goals. *Observability,* the first subclass of robustness, allows users to evaluate the current state of the system, similarly to synthesizability. An observable system has its components easily reachable and recognized by the users.

Other subclasses, *Recoverability* and *Responsivness,* cover the ability of a system to fix mistakes and the rate of communicating with users, respectively. A robust system needs to track mistakes, through the means of keeping all previous steps saved in memory, for example. The system also, ideally, responds to user input with the least delay possible and presents output as soon as it is processed. The last subclass is the *Task focus.* A robust system will allow for all actions necessary for it to perform its function completely and will avoid unnecessary additions that could detract from it.

Out of the rule types, principles are the most applicable in practice. The classes of principles can be summed up as follows:

- Learnability: How easy is it for users to learn and remember the system
- Flexibility: How many ways can users and the system Exchange information
- Robustness: How easy is it to make an error and how easy is it to fix it

3. User interface

In the previous chapter, the framework of an interaction was established to describe the basic relation between a user and a system: the user is using a system to accomplish a previously established goal. To this end a user and a system must have a shared platform for the user to convey their goals and for the system to provide feedback about the state of itself (the interface). In “The Essential Guide to User Interface Design”, Wilbert O. Galitz describes an interface as a part of a computer and its' software that the people can see, touch and otherwise interact with. [9, p. 4]

Described by Galitz, the basic structure of an UI is similar to the basic framework of an interaction, as described in chapter 2. The main difference comes from the fact that while the interaction is an abstract term for describing the relation between a user and an arbitrary system, the interface is a tangible part of a device.

One way to describe the objective of interaction design would be: the objective to create an interface that allows for maximum efficiency in translating a user's intentions and goals to a system while having the highest possible success rate of executing said intentions as actions. What this means is that a well-designed interface, following the previously outlined principles, should allow for faster and more reliable work, while a badly designed one would hamper the desired actions.

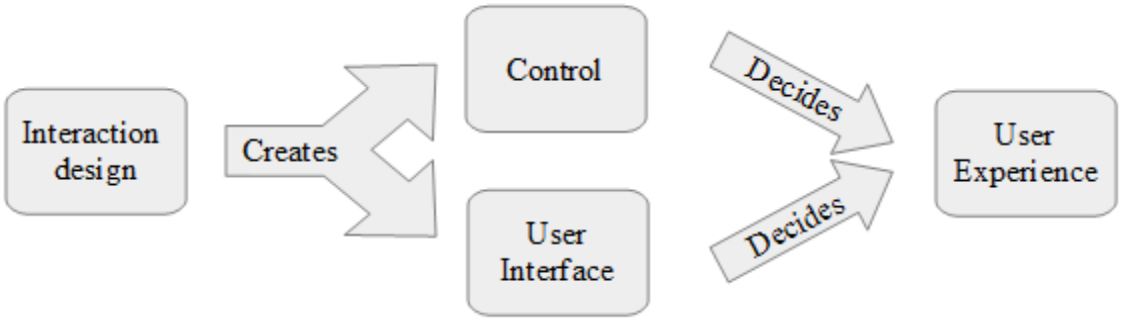

Figure 3.1: A visualisation of the design process and effects

The result of design is a user interface and certain means of controlling it. Depending on how the designers managed to follow the principles outlined in chapter 2.4, the controls and the interface influence the User Experience. With the myriad of possible hardware configurations, the approaches to interface design can vary greatly. For the purposes of this thesis interface of personal computers equipped with a keyboard and mouse will be considered unless otherwise stated.

3.1 Main styles of interfaces

The interfaces used throughout the information technology industry are constantly changing and evolving. However, despite the countless variations, two main types of interfaces can be considered prevalent. In order to describe them, it will be helpful to introduce the term *Manipulation.*

Manipulation describes the intended changes users make on an interface. It can be divided into two main categories – direct manipulation, a term coined by the computer expert Ben Shneiderman, and the derived indirect manipulation.

- **Direct manipulation**
  The main trait of direct manipulation is that the objects and interface are either continuously being represented and/or visible, or a physical interaction is taking place (e.g. pushing buttons, pulling switches etc.). [10, p. 1]
  Direct manipulation interfaces often present the users with a digital workspace or environment where their actions have a readily visible effect. A desktop is a typical direct interface, with UI elements that can be directly moved and highlighted.

- **Indirect manipulation**
  In contrast, indirect interfaces feature some kind of proxy between users and the manipulated object. An example of such interface would be various types of simulations where the users only input parameters and the simulation itself cannot be otherwise influenced.

Both manipulation types are present in interfaces to some degree and both serve a purpose. Parts of an indirect interface can be used in applications with too much data or too many functions to all be processed directly. Direct interfaces are more involved but should be more intuitively designed as a result, otherwise the users might have trouble using them.

The two main styles of interfaces in use today are:

Text based interface (TBI)

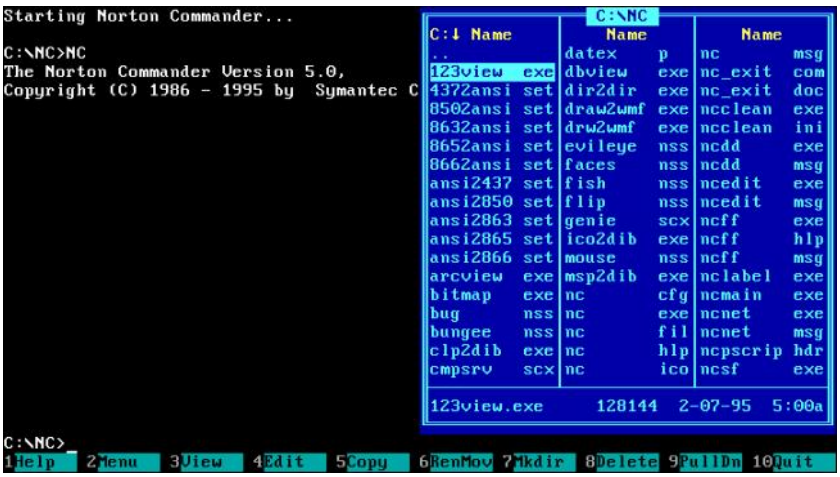

Figure 3.2: MS DOS running the Norton Commander software

This type of interface is the older one, allowing for output data to only be displayed in a pre-set grid made up of lines that contain text characters of a selected alphabet. Even though the name implies communication exclusively in textual form, TBI were eventually used to display elements of graphics using special alphabet symbols, essentially allowing developers to "draw" (example on figure 3.1).

Despite being technologically eclipsed, TBI still finds use as the "safe" option – the low demands on processing power and the lack of need for additional framework such as display drivers means that TBI can generally be expected to run in cases where the rest of the computer would fail. TBI is also generally cheaper to design and maintain.

Graphical user interface (GUI)

The chief difference between a GUI and a text-based interface is the scale on which the two operate. A GUI, unlike text interface can utilize the so-called pixels (an abbreviation of "picture element") to display images. The size of one pixel is not pre-determined and can therefore be exploited to draw images with far greater complexity than ones drawn with set symbols. Additionally, each pixel can be coloured independently of each other, allowing for gradual shifts of colour to be perceived. The size of pixels is only limited by the display hardware capabilities.

Another important difference between the TBI and GUI is the utilization of space. The TBI, by nature, utilizes only 2D, displaying an image without depth, while GUI can portray depth and space, effectively providing a 3D work environment.

The GUI is a common form of interface due to it offering greater functionality and being better suited for use with keyboard and mouse (TBI has limited options for displaying cursor feedback). The GUI is more suited for direct manipulation, as it has the tools to display objects and model virtual environments to directly manipulate. TBI can simulate a virtual environment as well, it only lacks the option to display it.

The next chapter considers the elements of UI and the way users perceive it, which forms the basis for interface design.

3.2 Elements of UI

An important part of designing an interface is to ask how the user will see and understand it. Interfaces have to cater to a whole range of potential users with varying backgrounds and levels of familiarity. Demands on performance, usability, and the aesthetics of their applications also have to be considered. It became important for designers to recognize the way users recognize the user elements, what draws their attention and what evokes the possibilities of certain actions.

A book by Jeff Johnson, "Designing With Mind In Mind", describes how humans are pre-disposed to recognize structure, explained in the so-called Gestalt (unifying the body and mind) principles [11, p.11-24]. The principles in the book describe how the human visual system attempts to fill in gaps in structure, giving expected meaning to incomplete or visually similar shapes, as represented on figure 3.2.

Figure 3.3: Highlighted folders are perceived as grouped together, described as the gestalt principle of common fate and similarity

Through the 7 principles, Thompson describes that their combined usage can be utilized by designers to shape the perception of an interface. He adds that users don't often scrutinize details and read every word of an interface or a website, instead quickly scanning it for relevant information. It can therefore be deduced, that for interfaces, the most important elements to convey meaning or purpose are the most visually striking and visible ones – A small icon might give more information about a function than a sentence of text at a glance. Most important visual factors for non-text are colour contrast and basic shapes.

Due to the way users process information, organized data is more easily understood, noticed and remembered compared to arbitrarily structured or densely packed data. Thompson's book demonstrates how even identical information becomes easier to understand when segmented:

Credit card number: 2145862566
Credit card number: 214-586-2566

An even faster understanding and perception of function can be gained from visual perception. Interfaces are designed with similar parts organized in windows to allow users to quickly identify the function that the interface offers – the UI elements.

The UI elements are the functional parts of the interface, each serving one or more purposes in relaying information to users. As users utilize the elements of other UI, they expect certain functions from similarly behaving components of other interfaces. Over time, these elements became expected and accepted by both users and designers. Usability.gov describes the elements as fulfilling four major functions:

1. Input – elements that allow data to be fed into a system
2. Navigation – elements helpful in looking for certain parts of data or functions
3. Information – elements providing insight into state of the system and its changes
4. Containers – elements housing other elements and functions

Function	Example UI elements	Example element function
Input	Buttons	Represent an action to be performed after clicking
	Radio buttons	Allow choice from several pre-set options
	Text fields	Allow users to enter text
	Toggles	Allow for switching between several pre-set states
	Dropdown lists	A button which displays a choice of items
Navigation	Search fields	Enable the user to enter a keyword (query) and return the most relevant result
	Breadcrumbs	Identify the current location in a system
	Tags	Categorize content for easier searching
	Icons	A simplified image serving as an intuitive symbol, most usually visually linked with what it represents
Information	Progress Bars	Bars that fill as users pass through a system/process indicating their current progress.
	Tool tips	Shows a hint for an element that has been hovered over with a mouse cursor
	Message boxes	A small window with a message that requires users to select an action before continuing
	Modal windows (pop-ups)	A window that requires interaction before returning to the system
Container	Accordion	A vertical stack of items that show/hide their description when clicked. Default settings may show only one or more descriptions at a time

Table 1: Examples of UI elements and their functions, according to Usability.org [12]

Figure 3.4:An example of breadcrumbs UI element (source)

Figure 3.5: An example of Accordion UI element (source)

The UI elements do not necessarily follow only one function. Modal windows, for example, may allow for textual input before allowing the users to return to the system and windows as an UI element can be considered all of the above, as they incorporate functions of other UI elements.

3.2.1 UI comparison

As described in chapter 2, UI design is an iterative process. The previous chapter was an overview of UI elements that remain generally the same throughout iteration process, yet the way they are utilized can greatly differ.

Figure 3.6: An illustration of the Windows LIVE start menu

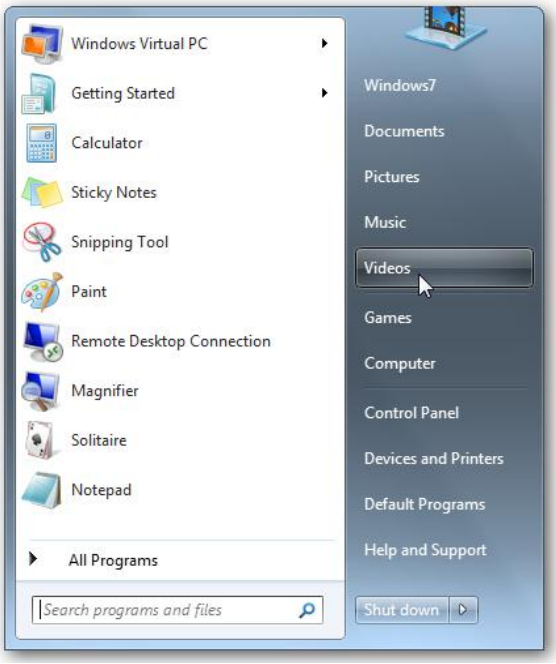

Figure 3.7: An illustration of the Windows 7 start menu

In 2012, the Microsoft corporation released another iteration of their operating system (OS) line Windows, the Windows 8. The OS was designed to be used with both desktops and mobile devices such as laptops. The issues with design choices became apparent as the users experienced the new start menu, clearly meant for use on mobile applications.

The intended mobile-centered design alienated the users using desktop computers [13]. The change was not only too intrusive visually, it also changed the mechanical side of the start menu. Previously, the start menu functioned as a deep dropdown list, allowing the user to find files directly from the desktop. The new system seemed wasteful and visually unappealing to some users, who sought ways to bring the old interface style back. Despite users eventually adjusting to the new style with varying degree of success, the change was deemed unpopular enough to be reverted in the next iteration, Windows 10.

Major changes do not necessarily have to always be negatively received and with enough iterations, large changes often happen, either to present the application in a new light or to change functionality.

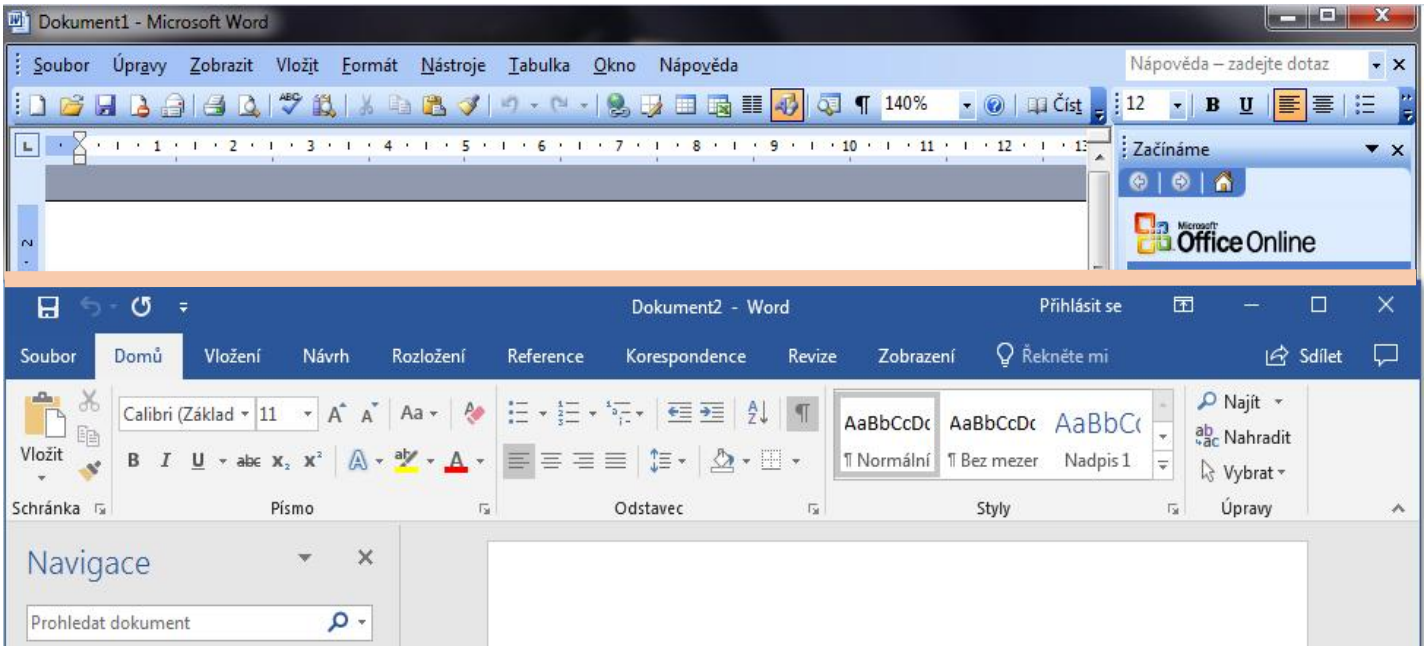

Figure 3.8: Side by side comparison of Word 2003 interface and Word 2013

Figure 3.7 shows the change of interface in iterations of the MS Word text processors. The change from toolbars to ribbons was much more positively received than the Windows 7 example because it added functions or at least didn't remove a majority of them. It also remained visually similar enough to the old interface to not feel new and confusing and as users got familiar with the new layout, the functions moved from drop down menus were more readily accessible.

3.3 UI structures

The previous chapter introduced UI elements as functional parts of the interface. The functions can be presented to users either verbally, visually and through structure. The interface (content and elements) and applications can be sorted and presented in a way that contributes to increased usability.

The Elements of User Experience by Jesse James Garret describes two possible approaches to hierarchies of information (or content) structure [14, p.89-92]:

- **Top down approach**
  This approach begins with the purpose of a finished system in mind, as well as the goals and needs of the users. The content and UI elements are divided into broad categories according to those goals and then narrowed down into logical subsections. The result is an empty container of categories ready to be filled.

- **Bottom-up approach**
  Where the top down approach prepares the conditions for information to be filled in, bottom-up approach starts with either pre-existing information, or with information that will soon be added. The existing information is grouped up into subsections, then into broad overlapping categories according to user goals.

In the book, neither of the approaches is stated to be better or worse than the other, it is stated, however, that information approached from the top may lose important low-level details, while bottom-up approach may result in a structure that is too suited to the existing content, and unaccommodating to possible future adjustments.

These approaches can be considered even out of context of websites in Garret's book, even though that makes them not equally viable for every type of UI. A UI that accommodates user input made with bottom-up approach would essentially dictate what the input would need to look like and as result in arbitrary restrictiveness.

UI Hierarchy

The UI design differs depending on the purpose of its system. An application may use one or several of these approaches to information/function structure at once. This chapter will list and comment on the usage and possible applications of some of approaches to UI layout.

"The Elements of User Experience" lists the following hierarchies [14, p.92-95]:

1. Tree

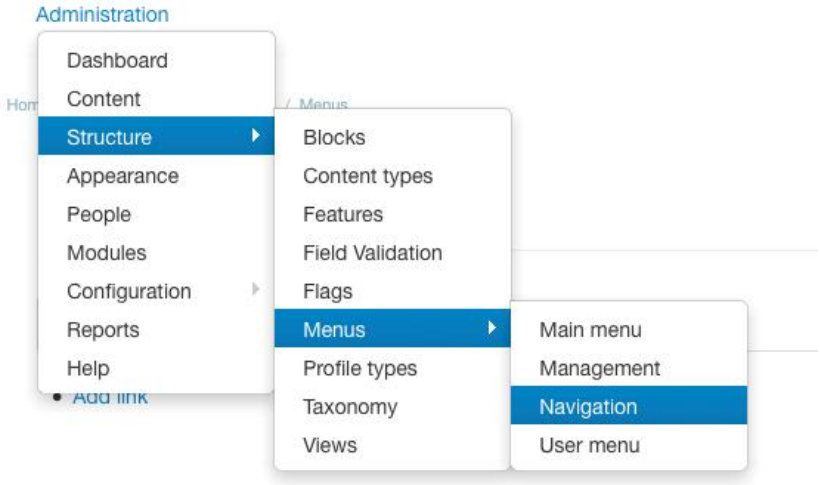

Figure 3.9: Nested dropdown list and element example of tree structure

A hierarchy that utilizes a parent-child relationship. There are options that have no children, but all options have parents, all the way to a single starting point. The tree hierarchy is well-understood by users because of its widespread use and logical progression of parent/child relationship.

This hierarchy is efficient when presenting related options. However, even trees designed with the familiarity concepts in mind are not necessarily easily navigable by users from the first moment. The entire sum of information/functionality is not present at a glance, only its parent term (in the case where all information is visible, the favourable information density is sacrificed instead). To properly utilize the tree structure, users often have to take time to explore the child options, which prolongs the learning process.

2. Sequential

Figure 3.10: The most basic sequence of actions

A sequence is a basic and ubiquitous structure. Due to its simplicity, it can be readily understood by nearly any user – the flow of language is a sequence, for example. Sequential structures are most often used on a small scale, and sequential structures offer limited interactivity.

3. Matrix structure

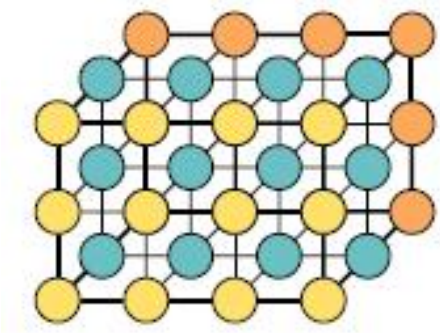

Figure 3.11: A possible matrix structure

Matrix structures allow users to access information along two or more axes. This structure doesn't necessarily result in a grid-like placement of elements and content, even though that is one possible result. An example of a matrix structure could be a sorted list, with conditions for sorting representing an imaginary axis of the matrix.

The purpose of this structure is to accommodate users with differing approaches and needs for accessing information. With the increasing number of axes, however, the matrix becomes increasingly hard to navigate and is therefore a bad choice for a main navigational tool.

These architectures describe a way of organizing the information architecture. However, if the resulting interface retains high level of usability, they can be freely (as presented on figure 3.9) exchanged and mixed.

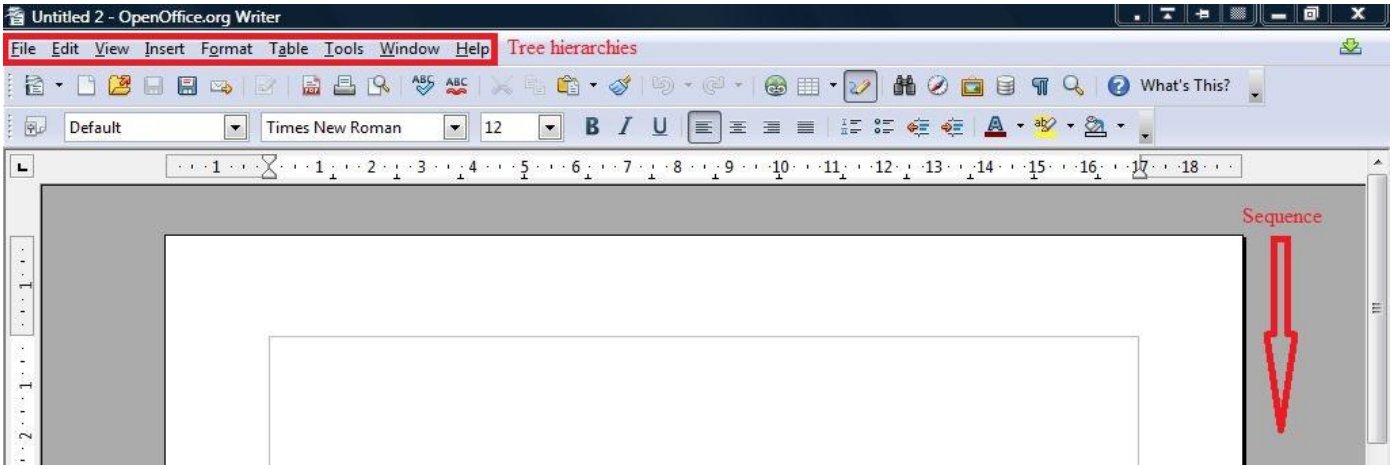

Figure 3.12: An example of mixing hierarchies in the OpenOffice Writer

3.4 2D and 3D interfaces

As text-based interfaces were the only option in the early days of UI, a large part of user interfaces throughout the history of interactive systems could be considered to be a *2D interface* – utilizing only two dimensions or a single plane. The interface typically consists of a type of display or similar output. The designers could use them to convey a sense of depth through creative usage of the interface elements. However, the elements still only existed in a single plane.

This type of UI remains common for applications where the lack of spatial depth is not an issue. In the early 2000s, a new concept of interfaces, dubbed *3D interfaces* began to appear. In the book 3D User Interfaces: Theory and Practice, Doug Bowman describes 3D interfaces as: "a human-computer interaction in which the user's tasks are performed directly in a 3D spatial context. "

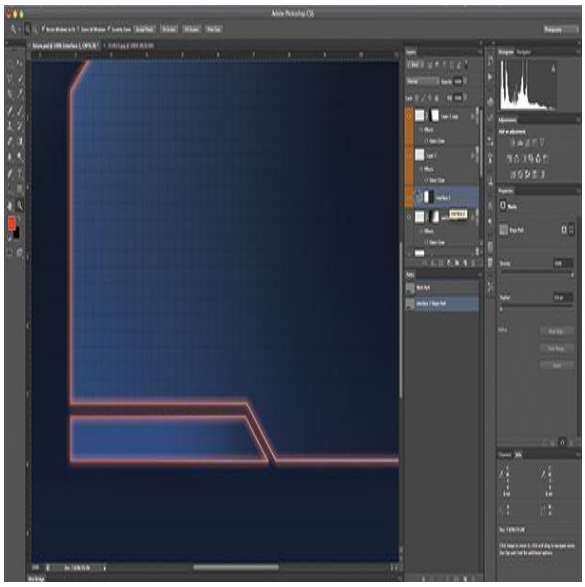

Figure 3.13: Figure: An interface of Adobe Photoshop Even though the application allows to model a perceived 3D object, the displayed interface is 2D

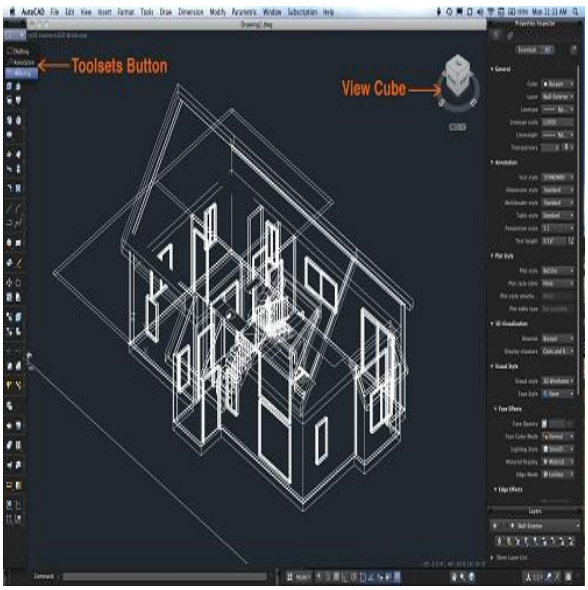

Figure 3.14: AutoCad 3D workspace. The interface allows users to directly interact

By Bowman's definition, an interface manipulating in a 3D space only with the usage of coordinates would still be considered 2D, since the manipulation is not performed directly. [18] 3D Interfaces allow the users to work in a simulated space using either the same control elements as their 2D counterparts or their own specialized variants, designed with control in a 3D environment in mind. The controls need to provide a spatial input for the system for them to be considered 3D controls. The UI elements themselves may be 2D, but as long as they allow direct manipulation with the virtual space, the interface can be considered 3D.

The 3D systems are often utilized in entertainment (video games) and in mobile applications, due to the small mobile display benefiting from the additional UI space in a 3D system. A special type of a 3D system became available to the public in the recent years, the so-called *virtual reality (VR).* VR uses both specialized display and control options to allow users to experience simulated 3D environment as if they were present within them. Namely, a VR headset to allow a wide-angle view of the 3D environments and spatial input devices. VR cannot utilize control elements a 2D system effectively as the nature of control in such a system is often reliant on a much wider range of motions.

Utilization and comparison of 2D and 3D interface environments

As the first and baseline UI, 2D interface is naturally vastly more widespread. However, even after the inception of its more advanced counterpart, 3D UI has not become the norm and instead remains a specialized type of interface. This chapter aims to describe the specific application of 2D, 3D and the VR interface technologies and compare their parameters in several select fields.

Interface application

1. 2D

The 2D UI is utilized in nearly ubiquitous fashion. The 2D interface elements are present wherever text, images in common formats (.jpg, .png, gif, etc.) are used. Most operating systems are inherently two-dimensional, as this environment is the least demanding to both designers and users. An interface in two dimensions simply demands less visual information to identify one's location within the system.

2. 3D

As mentioned before, 3D systems gained strong foothold as an entertainment interface, however, 3D interfaces have a broader range of utilization. 3D modelling has become a norm for many facets of manufacturing, architecture, route plotting in transportation, flight simulations both in space and in atmosphere and many others. The chief advantage of 3D UI is the increased visual complexity, as well as increased functionality.

Users can perceive 3D differently, even when viewed through the same display hardware as 2D. The perceived depth and the ability to rotate the displayed image freely helps with spatial perception.

3. VR

The chief difference of placing the user viewpoint into the 3D environment can have very noticeable effects on the familiarity of the situation. Aside from entertainment, efforts are already underway for utilizing VR for training in a multitude of situations. The option of creating scenarios on a chosen scale present the opportunity to safely familiarize users with situations that would be dangerous to health, should the users conduct such training in actual conditions.

Usage of this technology still has very clear limits. Even though force feedback technology is being developed, VR cannot fully replicate the sensations of touch, and

therefore muscle memory for certain tasks has to be taught separately. The question remains whether the inherent "disconnect" stemming from the fact that the users' perception of VR as a simulation could hamper the learning process. Despite the drawbacks, VR is being utilized as a tool for training of military personnel [15] *Cit Virtual Reality in the Military: Present and Future René ter Haar*, medical personnel, civilian pilots and others.

Figure 3.15: An airplane operator training application utilizing VR. The screen replicates the image sent into the HMD

4. AR

Figure 3.16: Promotional image of the Google Glass wearable device, showing what an AR interface could look like

Augmented reality is a concept blending elements from VR and 3D interfaces with physical reality. AR uses a display or a camera to apply an interface for a real world instead of a virtual environment. A theoretical AR device could feed relevant information about objects the user can see on demand or automatically through a wearable interface. The AR could also provide control without needing additional hardware, e.g.: the user hovering their hand over a business to bring up relevant information such as the owner, opening times etc.

Experimental AR projects such as Google Glass have been met with a lukewarm reception. The low battery life, limited functionality and the camera required for the Glass to function causing privacy concerns have made the device impractical to use in public, as stated in an article by the journalist Matt Swider [16]. As public use was the intended design, this AR project needs adjustments to be usable.

Performance

With increasing complexity comes the increased demand for processing power. It is exponentially more demanding to simulate space than it is to simulate a plane at the same level of visual fidelity.

It is difficult to estimate the minimal requirements in terms of processing power for 2D and 3D interfaces in terms of memory capacity or processing speed, yet hardware requirements can be estimated. A minimum for a 2D interface would be a type of display capable of displaying a line of text, connected with input controls (even a single button or a lever) and a chip performing a function.

For a 3D interface, the requirements are increased, both for processing power and hardware. To simulate a space would require a stronger processing chip, and possibly external memory to store the environment in during use. The display and input controls have to be more complex than a single button, needing to facilitate and visualize spatial movement, yet with creative design choices, for example delegating Y and Z axis controls to certain buttons, even keyboard and mouse can be used.

VR, a technology still ostensibly in a somewhat experimental stage, has the most specialized requirements. The technology requires a reasonably powerful PC [17] to even operate, as well as a HMD (head mounted display) and a set of spatial controls. The creation of a fully interactive environment has high performance demands. The issues with performance arise from the fact that the HMD needs to display a wide (110°) image at a high (2160 – 1200) [18] resolution for comfortable viewing by human eye. High levels of detail at this resolution result in immense demands on processing power.

A large difference between 2D, 3D and VR is the way users utilize them. 2D and 3D and their toolsets are perceived as viewed from the outside, similar to a book or real tools, while VR provides an environment in which the user seemingly operates, using the tools from the inside. Every technology has its use and neither has truly eclipsed the others in all fields.

4. Demands on UI

Technology and society both progress and change in time. Infrastructures and methods dedicated to a certain field may become obsolete as new advancements present a universally more efficient solution. Likewise, the needs and demands of users change with technological advancements and they vary depending on their goals. An important part of accepting new technologies and methods is also the difficulties and costs required in their implementation. This chapter aims to shortly summarize the demands on interactive system from the stakeholders – all people influenced by it.

"Human Computer Interaction" states that the stakeholders can be divided into four categories:

- Primary, who directly use and benefit from a system
- Secondary, who do not directly use it, but receive output or provide input
- Tertiary, who are directly affected by the results of the system, despite not using it or providing I/O
- Facilitating, who are involved in design, development and manufacture. [2, p. 458-466]

Example: For software in a cash register, the primary users are the cashiers, secondary customers and suppliers, tertiary the owners and other employees of the software-using company. Facilitating stakeholders are the developers of the software.

Demands on interface came from primary and secondary stakeholders directly, and tertiary stakeholders indirectly. HCI states that the priority of needs generally diminishes – the primary stakeholders usually have a higher priority than secondary ones. This is not always the case, though. E.g. a life support machine – the primary users are the medical staff, yet the greatest interest in the system's success lies with he tertiary users, the patients. Additionally, the Facilitating developers may be Primary stakeholders themselves, resulting in a blend of demands.

CUSTOM requirement analysis

To accomplish the goals and fulfil the needs of stakeholders, it is important to identify them first. One method for identifying them is CUSTOM. The methodology is focused on stakeholders and their circumstances and can be used both when designing a new system entirely, or when transitioning from an existing system to a new one. The book describes CUSTOM as a 6-step process, which can be simplified as:

1. Identify the stakeholders, their background and skills, characteristics of employment and their role in it.
2. Identify the work groups, meaning people who all work on a related task. Outside of business context this would mean identifying who the potential users are.

3. Identify the object-action pairs. In context of interfaces, what are the goals and what UI elements will be used to achieve them.
4. Identify stakeholder needs. The needs are considered from the viewpoint of the new system – what changes need the stakeholders do for the new system to function properly? (training, changes work group structure, etc.)
5. Consolidate the stakeholder demands. The list is checked to see if it conforms to identified stakeholder goals.

CUSTOM provides insight for designers to make informed decisions when drafting and developing interfaces and interactive systems. However, the stakeholder role in the design process isn't only reduced to being the point of evaluation and end users. The stakeholders can take active part in the design process, too.

Participatory design *Participatory design and democratizing innovation*

Participatory design could be described as blending of the stakeholder types, where primary and secondary stakeholders participate in facilitation. This type of design approach is typical for an organization, where the users and their environment can typically be safely identified. In Participatory design and "Democratizing innovation", Erling Björgvinsson, Pelle Ehn and Per-Anders Hillgren describe that this methodology is slowly gaining new ground with including the public in the design process [19].

Their article talks about the Malmö University Living Labs experiments and their efforts in including groups in the design process. The Living Labs often set out to consult and gather input on specific issues from only tangentially related focus groups in attempts to gather unexpected new insight into the design process that an experienced user might overlook.

Bug reports

A more widely-spread participatory design are the often-utilized bug reports. Bugs are errors in code, interface and function that are an obstacle to the desired operation of an application or a system. The users are encouraged to document and describe the appearance of bugs, which is then reviewed by the facilitating design team.

The reports may not be accurate, or one of the features may be mistaken for a bug, due to the disconnect between the designer and user perception. This in itself is a message to the designers about the visual readability and usability of their system and whether changes should be implemented in the next version.

Beta testing

The version-based iteration increases the development time on a large margin. In order to free up time of the design team, unfinished versions expected to have bugs, oversights and performance issues, are often released with the intent of users stress-testing them and bringing attention to as many issues as possible. The group can be closed – select

individuals, often with known background and trusted associates, or open – the version is publicly available, and anyone can make a contribution in testing it.

The demands on applications are driven by current technology, user needs and available resources. The companies and users often prefer tried and tested solutions instead of experimental technologies, even ones with high potential. The demands can be identified before the design process is commenced (such as the CUSTOM methodology), during the process (bug reports, beta testing) or after it has concluded, for the next iteration.

5. Conclusion

The thesis starts with the definition of important terms as stated in the introduction and with the introduction of the mental processes behind users interacting with a computer. Based on the research done by several publications, chapter 2 compiles and defines the terms and framework of interaction and interface design. Additionally, chapter 2 summarizes the basic structure utilized when approaching design and principles conductive to creating good and usable design.

Chapter 3 is focused on interfaces and their iterations. The building blocks of interfaces – UI elements are identified and summarized and possible negative effects of major changes of said elements are explored in chapter 3.2.1. The chapter further explores existing and experimental utilization of interfaces in 2D and 3D environment. The VR and AR technologies are still in experimental stages, and due to financial and privacy concerns, VR is becoming more widespread in the present. Older 2D and 3D technologies remain in use due to their lower cost and hardware demands, as well as familiarity to users.

The last chapter briefly notes the effects stakeholders have on the development process and some of the methodologies used to gather information about user demands – CUSTOM more suited to a closed work environment, universally usable bug reports and beta testing, active monitoring of users using an unfinished product.

The contribution of the creator of this thesis lies in compiling information, considering the applications of defined terms and selecting suitable examples from own experiences.

The closer description of defined terms and changes of meaning from cited articles was done in an effort to unify the terms for the purpose of this thesis, as stated in the introduction, and to clarify their meaning through the author's consideration.

List of used literature and references

1. Moggridge (October 2006), *Designing interactions,* p. ix; xvii-xix; ISBN: 9780262134743, United States, MIT Press ltd.
2. Dix et al. (2004), *Human computer interaction third edition p. 124; 127-130; 193; 195-196; 258-273; 458-466,* ISBN-10: 0-13-046109-1, England, Pearson Education Limited
3. Cooper (2007), Robert Reimann, and Dave Cronin, *About Face, The Essentials of Interaction Design,* p. 25, ISBN: 978-0-470-08411-3, USA, Wiley Publishing Inc.
4. Norman (2013), *Design of everyday things revised edition p. 38-43,* ISBN 978-0-465-00394-5, New York, Basic books
5. Silver (2007), *What puts the design in interaction design chapter Pondering form,* available online at www.uxmatters.com
6. Siang (2018), *What is interaction design, available online at www.interaction-design.org*
7. *What is usability?,* available online at *www.interaction-design.org*
8. Issack (2017), *Human computer Interaction: Design rule, available online at www.supinfo.com*
9. Galitz (2002), *The Essential Guide to User Interface Design 2nd edition p.:4,* ISBN: 0-471-084646, USA, Wiley Computer Publishing
10. Frohlich (June 1993), *The History and Future of Direct Manipulation p.: 1,* HPL-93-47*,* HP Laboratories Bristol
11. Johnson (2010), *Designing With The Mind In Mind* p.: *11-24,* ISBN: 978-0-12-375030-3, typeset in Chennal, India, Elsevier Inc.
12. *User Interface Elements*, Available online at www.usability.org
13. Ranger (November 2014), *Windows 8: Why Microsoft's giant gamble didn't pay off*, available online at www.zdnet.com
14. Garret (2011), *The Elements of User Experience 2nd edition p.: 89-92; 92-95,* ISBN 13: 978-0-321-68368-7, USA, Pearson Education
15. Haar (2005), *Virtual Reality in the Military: Present and Future, University of Twente,* Faculty of Electrical Engineering, Mathematics and Computer Science, available online at www.researchgate.net
16. Swider (February 2017), *Google Glass review,* available online at www.techradar.com
17. NVIDIA GeForce VR system requirements, www.geforce.com
18. Jon Martindale (April 2018), *Oculus Rift vs HTC Vive,* available online at www.digitaltrends.com
19. Björgvinsson, Ehn and Hillgren, *Participatory Design and "democraticizing innovation"*, ACM 978-1-4503-0131-2/10/0011, Malmö university

List of figures with sources

List of signs and abbreviations

2D	Two dimensional
3D	Three dimensional
AR	Augmented reality
GUI	Graphical user interface
HMD	Head mounted display
I/O	Input/output
PC	Personal computer
TBI	Text based interface
UI	User interface
UX	User experience
VR	Virtual reality