HELSINKI UNIVERSITY OF TECHNOLOGY

Department of Electrical and Communications Engineering

Osmo Tolvanen

User assistance in a complex application for experts

Master's Thesis, which has been submitted in partial fulfilment of the requirements for the degree of Master of Science in Espoo, Finland, October 17th, 2005

Supervisor **Marko Nieminen, prof., Dr.Sc.(Tech.)**

Instructor **Pia Nakari, M.Sc.(Tech.)**

HELSINKI UNIVERSITY OF TECHNOLOGY
Department of Electrical and Communications Engineering

ABSTRACT OF MASTER'S THESIS

Author
Osmo Tolvanen

Date
October 17th, 2005
Pages
91

Title of thesis
User assistance in a complex application for experts

Professorship
User interfaces and usability
Professorship Code
T-121

Supervisor
Marko Nieminen, prof., Dr.Sc.(Tech.)
Instructor
Pia Nakari, M.Sc.(Tech.)

The objective of this study was to find out what kind of assistance and help the users of a complex application for experts need, and how this information could be delivered to them by the software. The user assistance is closely integrated to the user interface and can also be called embedded help. This study was initiated because it is known that many computer software applications boast features that the users cannot use and don't easily learn to use, and consequently the software isn't used efficiently. At the same time these software applications often lack the ability to efficiently give the user information about their features, and how and where to use them.

This question was addressed with a user centered study. Users' ways of working and getting help were analyzed with the case study of the software called Tekla Structures. Six different methods of data gathering were used for this, the most important being user interviews, and observations, and getting familiar with the other complex software. Other methods of collecting data were use diaries, web questionnaires and interviews with the Tekla's user support personnel. After analyzing the data and getting a picture of the users' world of work and tasks, solution proposals for user assistance were created and their functionality was validated by testing and interviewing with the users. The six users who participated in the study represented the average target group of this study: they used the complex application daily and were experts in their own profession.

28 solution proposals on how to assist the user were created. They are concentrated on the most important properties of the user assistance identified on the basis of the analyzing of the data: supporting the users' tendency to use and learn the software by trial and error by giving a better visibility to the undo function, users' actions and their results and by giving short instructions while using the command; giving the users information about their possibilities and supporting the users in finding and learning new functionalities in the software; giving the user a good, quick and easy control of the assistance provided; giving the information in stages so that first a brief version is given and more upon request; providing information that contains examples and directions on where different commands can be used.

The main outcome of this study is that the users' typical needs of assistance while using complex software applications were identified and solution proposals given to address these needs.

Keywords: usability, user assistance, embedded help, complex applications, complex software, experts

TEKNILLINEN KORKEAKOULU
Sähkö - ja tietoliikennetekniikan osasto

DIPLOMITYÖN TIIVISTELMÄ

Tekijä Osmo Tolvanen	Päiväys 17. lokakuuta 2005
	Sivumäärä 91
Työn nimi Käyttäjän avustaminen monimutkaisessa asiantuntijasovelluksessa	
Professuuri Käyttöliittymät ja käytettävyys	Koodi T-121
Työn valvoja Prof., TkT Marko Nieminen	
Työn ohjaaja DI Pia Nakari	

Tämän diplomityön tavoitteena oli tutkia millaista avustusta ammattilaisille suunnattujen monimutkaisten tietokonesovellusten käyttäjät tarvitsevat, ja miten sovellus voi tämän käytönaikaisen avustuksen ja tiedon parhaiten antaa. Diplomityötutkimus sai aiheensa tiedosta, että monet, erityisesti monimutkaiset sovellukset ovat tulvillaan ominaisuuksia joita käyttäjät eivät osaa käyttää, tai joiden oppiminen on hankalaa, ja ohjelman käyttö on siten hyvin tehotonta. Sovelluksissa useimmiten ei kuitenkaan ole juuri minkäänlaista tapaa millä käyttäjille annettaisiin tietoa toiminnoista, sekä missä ja miten niitä tulisi käyttää.
Lähestymistapa ongelman ratkaisussa oli käyttäjäkeskeinen tutkimus. Käyttäjien työskentelytapoja ja avunhankintatapoja selvitettiin tutkimuksen kohdesovelluksen Tekla Structuresin käyttäjien keskuudessa. Tietoa kerättiin kuudella eri metodilla, joista tärkeimmät olivat käyttäjähaastattelut, ja –havainnoinnit sekä muihin ohjelmistoihin tutustuminen. Muita käytettyjä menetelmiä tiedonkeruussa olivat käyttöpäiväkirjat, web-kyselylomake sekä Teklan käyttäjätuen työntekijöiden haastattelut. Kerätyn tiedon analysoinnin jälkeen saatiin muodostettua kuva käyttäjien työskentelytavoista ja tähän tietoon pohjautuen ideoitiin ratkaisuehdotuksia joiden toimivuuteen haettiin varmistusta testaamalla ja keskustelemalla niistä käyttäjien kanssa. Tutkimukseen osallistuneet kuusi käyttäjää ovat tyypillisiä monimutkaisen sovelluksen käyttäjiä: he käyttävät monimutkaista ohjelmaa päivittäin ja he ovat oman alansa asiantuntijoita.
Tutkimuksen tuloksena luotiin yhteensä 28 ratkaisuehdotusta. Ne keskittyvät tunnistettuihin käyttäjän avustamisen tärkeimpiin ominaisuuksiin: käyttäjän tukemiseen yritykseen ja erehdykseen perustuvassa ohjelmankäyttötavassa antamalla heille ohjeita käytön aikana, parantamalla perumistoiminnon näkyvyyttä sekä toimintojen tulosten näkyvyyttä; kertomalla käyttäjille erilaisista mahdollisuuksista käyttää ohjelmaa sekä auttaa heitä löytämään ja oppimaan uusia komentoja; antamalla käyttäjälle hyvät ja helpot avustuksen säätömahdollisuudet, tarjoamalla käyttäjille ohjeita paloittain siten että ensin annetaan lyhyt tiivistetty versio jossa tarjotaan mahdollisuus avata tarkempi avustus; antamalla ohjeita jotka kertovat myös toiminnon käyttökohteista ja annetaan esimerkkejä.
Tutkimuksen tärkein tulos oli monimutkaisten asiantuntijasovellusten käyttäjien avunsaantia koskevien tarpeiden tunnistaminen ja ratkaisuehdotusten luominen näiden tarpeiden täyttämiseksi.

Avainsanat: käytettävyys, käyttäjän avustaminen, sisäänrakennettu ohjeistus, monimutkaiset asiantuntijasovellukset, asiantuntijat

Acknowledgements

This study was a part of my journey in the Helsinki University of Technology towards graduating to be a Master of Sciences in Technology. The study years have been good, and they culminated in making the Master's Thesis of this interesting subject.

During the making of this thesis, I couldn't have hoped for better guidance and advice than I received from my instructor Pia Nakari from Tekla whom I want to especially thank. Also thanks belong to the supervisor of this study, Professor Marko Nieminen, for his constructive comments and feedback. I also want to thank Marjo Lehtinen for reading the draft version of my thesis and giving valuable suggestions. Thanks belong also to Jorma Zielinski for excellent help in creating the web questionnaire and my father Pekka Tolvanen for checking the language of my thesis. I'm also grateful to all the members of different teams on Tekla who have helped me in various ways in making this study become ready. I also want to thank my parents for encouraging me during the years I spent in the University. Thanks also belong to my friends from the university who have given me also other things to think than the studies.

I also want to thank my wife Marja for her love and support, and standing of all the long evenings I spent with my thesis during the making of it.

Espoo, 17th of October, 2005

Osmo Tolvanen

APPENDICES

1 Introduction

1.1 Background

Everyone who has used computer software is familiar with the following situation: you are trying to use a command in a program, but can't figure out how it is supposed to be used. What must be clicked, what selected and how? You know that the answer might lie in the on-line help, but you don't have the time to start looking for it... Why can't the software just offer some help when trying to use the command?

Since users of complex applications for experts face these kinds of problems even more often than the average user, Tekla Corporation initiated the subject of this study in order to find an answer to the question: How to provide the user of a complex application for experts with help during the use of the software. The need for assistance is clear. In Tekla's own applications and many other complicated applications meant for expert's use, there are various places where the user should get some kind of assistance from the software. This would be so that he or she doesn't have to ask from a colleague, try to search from on-line help or manuals or call the service desk, since all of these mentioned methods of getting assistance require the user to stop their current task and start to look for the help. This in turn always slows down the task being done since finding the help takes time, and after finding the help, the user has to remember where and what about he or she was doing before starting to look for help.

1.2 Objectives of the study

In this study we want to find answers to the question of what kind of information the users of a complex application need, and how should the information be delivered. The information we are interested in consists of on how to use the software, hints and tips on using the tools in it, helping the users to get their tasks done more efficiently and with shorter learning times. Since this matter very closely concerns the users, it is more than natural to conduct the study in a user-centred manner. As Zubak (2000) puts it, developing good functioning user assistance is difficult because it requires a close examination of the users' way of working. Taking users into account in every stage of the design provides a sound basis for a well-functioning solution (Holtzblatt and Beyer, 1998). In this study we use Tekla's three dimensional (3D) modelling software called Tekla Structures as a case study program, and all the user data and the suggested solutions are based on this software. Tekla Structures is a leading 3D modelling software used by construction engineers around the world to design steel and concrete buildings and other constructions. It can be considered as a kind of Computer Aided Design (CAD) software even if it has several features of it's own compared to traditional CAD two dimensional (2D) software, because of the 3D environment. The software is described in more detail in chapter 3. The objectives concerning the Tekla Structures software are to lower the time needed for new users to start using the software and on the other hand helping the experienced users to start using the software more efficiently. Currently there is not much user assistance in the Tekla Structures. The findings about ways of assisting the users of complex applications can be generalized to other complex applications though.

1.2.1 Research questions

In order to crystallize the objectives of the study into a nutshell, two research questions were formulated. They are as follows:

1. What kind of assistance do users of a complex system for experts need?

2. What are the most suitable ways of presenting the assistance needed?

In addition to the questions above, we wanted to find out what are the situations where the users typically encounter questions that would need to be answered. Also we wanted to find with comparative tests what kinds of presentation types would be the best ones to offer the information.

1.2.2 Scope of the thesis

In this study the objective is not to find improvements to the user interface as such, but rather find ways to assist the user with the tasks they perform with the software. The borderline between developing and improving assistance methods or embedded help, and making changes in the actual user interface is sometimes difficult to determine. Since the embedded help is an integral part of the user interface, it is difficult sometimes to say whether some concept, functionality or just the text in the interface belongs to the user interface itself or to some kind of an embedded help or user assistance. Actually it may be gratuitous to try to distinguish these two parts, since as Zubak (2000) puts it, the embedded help is and should be designed as software, not as an afterthought. In the case study software of this study, the Tekla Structures, the user assistance is designed as an afterthought, after the actual interface has been developed. That's why in this study we have to somehow distinguish traditional user interface improvements from improvements in user assistance. The guideline in this division has been the thought of giving the user more information about his or her tasks. If an improvement incorporates more information given to the user about the software's functions, it can be considered as user assistance. Thus adding textual information into dialog boxes can be thought of as user assistance, but having a text of a button in a dialog box replaced with another text can not be considered one.

1.3 Organization of the study

This study is divided into two larger phases. The first one involves getting to know the user's way of working with the software, collecting data from them about their current way of seeking help or assistance, where the help is typically found and whether it provides value to the user. This phase follows the Hackos and Redish (1998) user task analysis suggestions expanded to suit our needs.

The second phase involves analyzing the data collected in phase one, identifying the users' needs in getting assistance during the use of the software, and identifying the most problematic areas that would benefit from an embedded help system. The other important area of the second phase was the testing of the solution ideas.

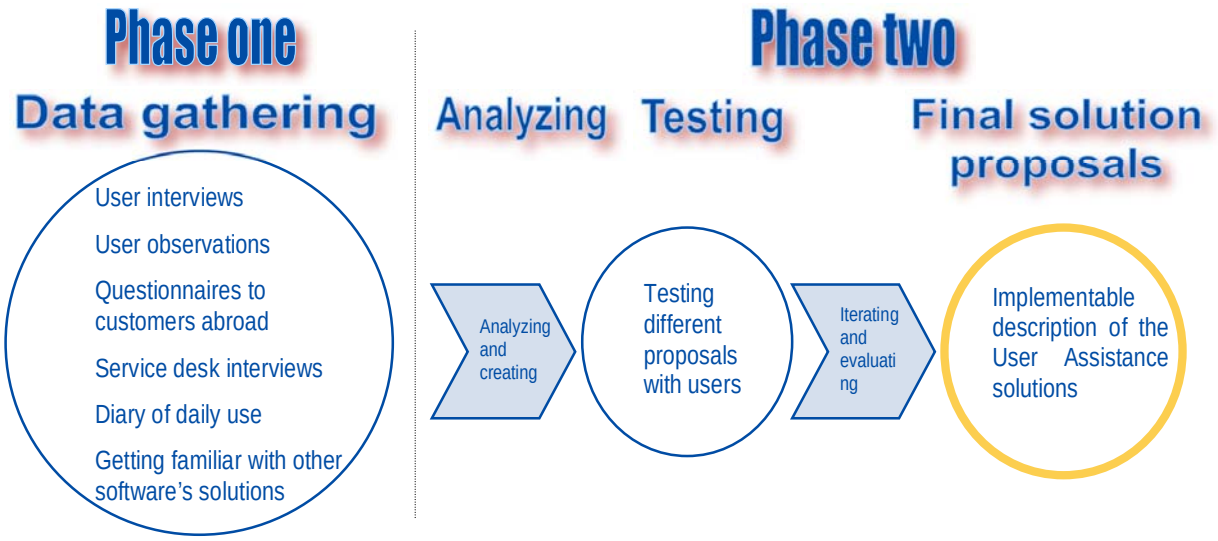


Figure 1: The organization of the study

In this thesis first the theoretical framework concerning user assistance and embedded help is presented in chapter 2. The complex applications and the case study of the software called Tekla Structures is presented in chapter 3. In chapter 4 all the methods of data gathering are presented with their results. After the phase of data gathering the data was analyzed and solution proposals were created based on it, and this process is described in chapter 5. The chapter 6 describes the testing process of the solution proposals and in chapter 7 all the final solution proposals are described in detail. The conclusions and suggestions for further research are presented in chapter 8.

2 User assistance: theories and studies

In this chapter a theoretical framework on the field of user assistance and developing it is described. First, an overall introduction to the field of usability is given, and next theory of user assistance from different aspects.

2.1 Terms and concepts

The terms *user assistance* and *embedded help* are used in the scientific studies and articles to refer more or less to the same concept of helping the user during the use of the software. In this study the term user assistance is mainly used to refer to this concept. The term *embedded help* has been also used in literature (for example Zubak, 2000) to refer to a similar kind of assistance provided to the user, but this term may be seen as a more restricted one. *User assistance* has also been mentioned to mean assistance that a user gives to the software, for example in an automated complex task (Lee, Huertas, and Nevatia, 2000), but in this study this definition is not used.

In this study terms related to the Tekla Structures are also used. *Custom components* are connections or other details in the model that the user creates him- or herself. *Connections* are a large group of pre-defined details that are used to connect different types of steel or concrete parts together. The term *modelling* refers to the process of building the model, including making all the details and larger parts.

2.2 What is user centred design and usability

When designing a house, the architect starts by deciding what the purpose of the building being designed is, what the needs that the owner of the house are. All the rooms must be designed so that there are no obstacles for the activities planned to be performed there, and that the space supports the tasks that it is for. (Holtzblatt and Beyer, 1998). For example, a room for carpenter's activities must have enough space for handling long boards or big plates. The designer doesn't start the designing task by deciding what wall materials he or she would like to use, or what kind of window frames will be used but rather thinking what the space is meant for.

The same principles as in the above example in apartment design, should apply to software design too. The design work should have its innovation in the user target group's needs rather than in the designer's technical choices. (Holtzblatt and Beyer, 1998) Hackos and Redish (1998) call this as the user and task analysis. The analysis should start in the very early phase of a design process and it should continue throughout the whole process. Since each product and its target group users are different, the analysis should be done at least partially with every single product (Hackos and Redish, 1998).

That's what usability is all about: Less time spent with recovering from error situations, more flexible ways of working or less time spent in learning to use a new software are just examples of ways that an organization can save large sums of money by paying attention to usability in the design phase of a new system (Nielsen, 1993). To some extent, usability can be considered also to be part of a larger issue of system acceptability which refers to the overall attitude towards a software by all its potential stakeholders, such as managers, actual users, users' clients etc. (Nielsen, 1993)

Nielsen (1993) defines usability as five different attributes: learnability, efficiency, memorability, errors and satisfaction. By defining these precise and measurable components as parts of usability, the abstract concept of usability becomes disciplined and not just a thing that can be argued about. The term User Centered Design (UCD) was introduced as a term long before Nielsen's definition by Norman and Draper (1986, in Faulkner, 2000) and it has come into use with the term usability. UCD can be accomplished by applying usability engineering methods (Faulkner, 2000).

2.3 What is user assistance

Good on-line help and user assistance calls no attention to itself, and asks the user do very little to get the information. This is a brief summary of the meaning of user assistance by Johnson-Eilola and Johndan (2001). In this chapter the user assistance gets described and reasoning for its existence are introduced.

The support that computer software vendors provide to the users has evolved quite a bit during the years. In the past times, before the era of computers people learned handicraft skills by apprentice-master relationship. This didn't involve any documents; skills were learned by watching, listening and imitating. After the literature became more familiar, people could also learn just by themselves, by reading books. The case was similar in the beginning of the computer era: first there were no books or manuals, people learned from other people. Then, software specifications and reference manuals started appearing. Actual printed user manuals followed only after that, and these were later accompanied with on-line documentation. Up until recent years learning by reading printed text or documentation provided on-screen has been the main method for getting information on how to use computer software. (Johnson-Eilola 2001)

While today almost all computer software provide the user with at least some kind of documentation, users still want to be taught by watching and imitating, and not reading. As we grow up, in real life we learn to survive independently without having to ask for help. People want to try to manage by themselves, with their own brains and asking for help is left only to critical situations. In computer world the same trend is living strong: users want to try things out themselves as long as they can, and asking for help, i.e. pressing the help button, is left to the critical situations when nothing else works out. It is also possible to assist the user while he or she is actually using the software. So that the help is not available only when the user decides to ask for it, but also when there is any need for it. This way the user doesn't even notice that he or she is being helped or assisted, and continues using the software without interruptions or errors. (Mueller, 2002). This is called user assistance.

User assistance thus means a novel way to provide the user of software with information on using the program. The assistance may have different kinds of representations but common among all of the different types of use assistance is the fact that it is provided during the use of the software and it gives information to the user. Examples of different kinds of assistance types are context sensitive help on fields, in-line texts, procedural help, wizards, balloon help, what's this links, tooltips, videos and audio. (Kalk, 2001, Johnson-Eilola 2001, "User Assistance", 2004) User assistance has also been referred to mean assistance that a user gives to the software, for example in an automated complex task (Lee et al., 2000), but in this study we don't use this definition

The fact that assistance is offered during the use of the software makes the user assistance different from traditional manuals and on-line help files; user assistance is context dependent and is offered to the user either automatically or with help of a simple call mechanism. It is highly granular, and it may in some places be seen as a special kind of computer-based training. (Zubak, 2000). Because the help provided via user assistance is not labelled as "Help" and it doesn't require the user to press the help button, also users that normally don't want to read help texts would most probably read this instruction (Mueller, 2002). The user has to have a high degree of control over the user assistance system though (Fenchel, 1981).

One big difference between the traditional on-line help and the user assistance which is embedded to the user interface, is the fact that the users don't have to break their work routine in order to get help in their current task. Traditional on-line help would force the user to open a separate window that hides by its appearance part of the task they are currently working on, and demands them to first look for the correct place in the help. After finding the correct place, they have to resize the help window to be able to see the task they are working on and simultaneously see the instructions. With user assistance, the problems of finding and seeing the instructions are solved already for the user. (Zubak, 2000) Johnson-Eilola (2001) summarizes the idea: "Everything important is visible, and the distance between idea and the result should be near zero".

However, user assistance doesn't directly replace traditional manuals. When a novice user faces a problem or encounters a situation he or she doesn't understand or that seems puzzling, he usually creates an explanation of his own to clarify the situation. This might lead to a situation that the user is thinking that he or she is doing something totally different than the software actually does. The explanations user makes about the behaviour of the software usually emerge from user's previous experience in similar tasks. (Mack, Lewis & Carroll, 1983) So, in the case of a steel detailer, if the previous experience is from a different modelling program or possibly manual drawing board, the user interprets the software behaviour according to his experiences about drawing board or the previous modelling software. It is up to the user interface and the user assistance in it to clarify the situation.

There are different kinds of solutions of user assistance on the market. Office software suites have introduced different kinds of active assistances that try to make suggestions and guess what the user might be trying to do. Complex software vendors have also introduced various solutions which include videos, animations, balloons etc. Many of these solutions are functional also in other software, but a general set of user assistance solutions is not a straightforward implementation of solutions in other applications. (Johnson-Eilola, 2001)

User assistance can also be seen as a way to bring task-centricity to an object centered software. When thinking about all these abovementioned properties of the user assistance, we can see that it tightly connects to the Nielsen's (1993) definitions of usability (see the previous chapter). It affects learnability and efficiency by giving the user help in using and finding new or unused commands, and it improves memorability by lowering the memory load of the user by giving instructions during the use of a command. Satisfaction of the user is improved also, since the user doesn't run into situations where he or she has to specifically start looking for help. Errors the user makes are lowered too, since the user knows better what he or she is doing.

2.3.1 User assistance vs. user interface design

Why not just design a user interface that is so intuitive to use that no any extra assistance is needed? Good user interface design should be the basis for a usable product and thus a good user experience.

In an ideal situation there would be no need for help or assistance: the user interface of software would be so well designed that a user learns to use the software just by using it. Unfortunately this is not reality. As Carboneel and Capobianco (2003) point out, the desire to build an application that is so transparent and easy-to-use that no any help or training is needed has been noted to be excessively difficult. It is possible that a software system has so many functions and the tasks that are made with the program are so complex by nature, that even with careful design the software becomes complex. Complex doesn't necessarily mean difficult, but it may easily be also that. It can be argued, that designing user assistance to a software product is like making a temporary fixes to problems caused by poor usability. Also Hackos and Redish (1998) point out that trying to patch up inadequate design with documentation, help and training are very far from being as good as good user interface design in the first place.

However, sometimes making the needed changes in the user interface are so big that it is better to assist the user by means of user assistance in order to make the product even a little more usable. Since user assistance is an integral part of the user interface, it can relieve some design flaws in the user interface even if it doesn't actually change the actual problem. Providing good user assistance should though never be considered as a replacement of good user interface design in the first place.

As mentioned earlier, most, or even all usability problems can be defeated with careful design, but the problems are not all that matters. Bhavnani and John (1997) studied the users of complex software and noticed that users often aren't using the capabilities of the software efficiently, even if the user interface was relatively well usable. They concluded that the users were missing good strategies in the use of the software. With strategies they mean capabilities to see and use generalizations of certain functions in several places in the software (see chapter 2.3.2). Good strategies can be taught to the user by means of training, but user assistance can also help in that and therefore proper user assistance is often needed even if the user interface was well designed, to improve.

Good user assistance is not easy to make. The difficulty relies in the fact that different users require different types of assistance and thus a solution that is perfect for someone is frustrating to another. Because of this, it also requires a close interaction with the users when creating it in order to get a clear picture of the user's way of working (Zubak, 2001).

2.3.2 Contents of the assistance

It is important to pay attention to the contents of the assistance provided. It is not all the same what kind of information the assistance gives to the user. It has been identified also in this study, that the users mostly want to have instructions in brief and crystallized form and because of this, it important to specify the type of information that is given to the user.

There have been various studies about what kind of information is best in supporting user's tasks. Lazonder and van der Meij (1995) studied users of a word processing software and noticed that the ones that were given manuals that contained information on possible errors

and instructions on how to recover from them instead of traditional manuals that mainly describe the tools of the software, had better performance in different types of tasks with the software. This should apply not only to the manuals, but also to other kinds of instructions given to the user, including user assistance, although it wasn't verified in the study of Lazonder and van der Meij. Giving error information to the user gives him/her more time to recover from the error, and gives a possibility to find out the reason for the error also on hectic situations when there's no time for anything extra.

Bhavnani and John (1997) studied the use of strategic information in manuals, and training led to better performance in completing typical tasks with CAD software. Bhavnani, Reif and John (2001) further investigated the effect of teaching strategic information to the users and found that the learning results were promising. They found that some of the good strategies enabling efficient use of software can be learnt by learning to use the commands of the software. Unfortunately some important strategies remain still unlearnt. By giving explicit instructions on these important strategies the users learn to use the software more efficiently.

The timing of the assistance is also important. van der Meij (1992) noticed in his study that information about possible errors should be given when actions that user performs are error-prone and when possible errors are difficult to correct. Carbonell and Capobianco (2001) noticed in their experiment, that a lot of the information users want while using an application is highly context-dependent. They thus want information directly related to the tasks they are currently acting with.

In "User Assistance" (2004) it is defined that user assistance can contain three different kinds of content: procedural, conceptual and reference. The procedural content means information on how to perform specific tasks, usually in step-by-step format. Conceptual content is a kind of wide expansion to the procedural help since it contains information about how the subject at hand relates to other subjects, where it can be used, background information and feature overviews. Reference help, as the name suggests, serves as an on-line reference index of all the functionalities of the software and their usage.

Johnson-Eilola (2001) makes an interesting discussion on the subject on what results if the software teaches the user to make complex operations without telling any reasons for why things are made the way them are made. He takes as on example the Microsoft Word's template wizards that allow the user create professional looking documents, for example a résumé, without telling any background on what information and why is to be used in different places. In other words, the user can easily produce good-looking documents that actually may not have much actual content in them. The same discussion could be made also with complex applications for experts: is it possible to have the users making documents or drawings with for example a modelling software that look good and reliable, but when taking a closer look, are just unusable crap. So, could a user assistance solution simplify a task too much? If the user would otherwise need to do some preliminary work before making it with the software, could he or she now do it without having to think about all the different aspects of it? Are we in the way of making good basis for second-class model design? These are questions that may be a little provocative, but they have an argued basis still and need to be considered.

2.4 Human factors concerning user assistance

The user assistance is aimed to help the user in his or her daily tasks and thus it is important to consider different factors of the user when developing a good solution. In this chapter some human factors issues are discussed.

2.4.1 The user needs help

Even small software can be very difficult to use, not to speak about the complex ones. In the case of complex software that is used for making operations that are complex by their nature, the user interface easily becomes more and more complex. As Holzblatt and Beyer (1998) say, when lots of people are working on the same product, there is easily a chance that the individual parts people design become overcomplicated. This is because for the designer the little piece of the whole software he or she is designing, is the most important part of it, and therefore he or she makes it as good as possible and adds more and more options and possibilities to it. Also, if thinking about different features of the software individually, these individual parts easily become larger than they actually should be.

In the design and implementation process of a software product often the analysis of the user's world, called user and task analysis, are unfortunately too often not an integral part of the process. (Hackos and Redish, 1998) The mentioned usability factors by Nielsen (1993) in the chapter 2.2 are thus too often left without the value they would really need.

These mentioned facts among others are reasons for the user interfaces not being as good as they could be, and as the result of it the users need help when using software. This help can be offered in various different forms.

Users can spend 30-50 % of their time with errors and recovering from them (van der Meij, 1992). When a user encounters an error when using software, he or she has two different strategic approaches to correct the situation. One is to apply a corrective method in which the user tries to correct the mistakes made, remove obstacles that prevent the completion of the tried command, try to undo the mistakes made or such corrective actions to get back to the situation before the error was encountered. Other possibility is to apply a reconstructive method in which the user simply tries to do the same thing again, and hoping that by paying more attention the error would not be repeated. Both methods are in most cases equally effective, but generally it is recommended to use the corrective method since reconstructive approach usually leads to sloppy work habits. (Lazonder and van der Meij, 1995). Being able to diagnose, prevent and correct errors is an important skill in order to master computer software (van der Meij, 1992).

Hopes in the increases in productivity are behind many investments in the computer hardware and software. A good example of a type of complex software are the Computer Aided Design (CAD) software used widely by engineering companies. Large sums are invested, but productivity growth for example in CAD systems in many cases is very small (Bhavnani and John, 1996, after PSMJ, 1994). This has been referred to as the productivity puzzle (Bhavnani and John, 1996, after Strassmann, 1999).

According to Bhavnani and John (1996), reasons for this are various, but when speaking about CAD systems it has been noted that efficient use of software is the fact that is missing. Users have levelled-off with their current skills using the software and are reluctant to learn new ones. More importantly, they are often using the software's tools in

suboptimal ways. Bhavnani and John (1996) state that users often don't know the right strategies when using CAD programs. Users translate their old working habits from their manual drawing boards and don't either want or know how to use more efficient ways to do the tasks at hand. Thus they use the tools in the software in the same way that they are used to work with the manual drawing board. This means for example that they are not aggregating functions efficiently since these don't exist in the manual domain. In computer drafting and modelling, aggregation means combining several parts in the drawing or model into an one entity that can be for example modified or copied as one. So, user details all the parts of an entity one by one and finally combines them into one. This strategy has no clear advantage in the manual drawing board, but with a computer software often reduces steps that must be taken to complete a task and thus is particularly useful in CAD. (Bhavnani and John, 1996) In Figure 2 an example on aggregation and a comparison between manual drawing and computer drafting is illustrated.

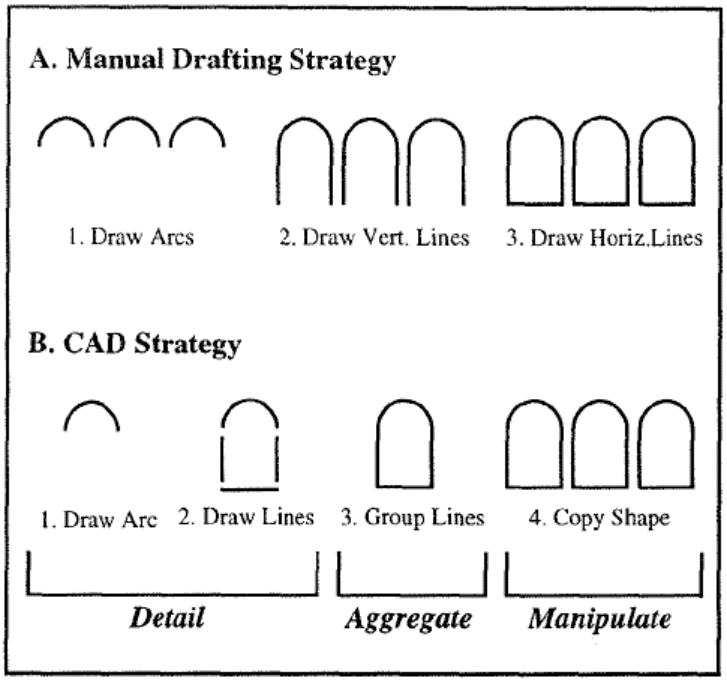


Figure 2: Use of aggregation as a strategy in CAD (Bhavnani and John, 1996).

Training and literature about CAD programs don't provide this kind of strategic information on where it is beneficial to use different kinds of functionalities and what kinds of new approaches there are to get the tasks done in a CAD software. In addition, since the outcome of the work, the drawings, is the same regardless the technique used to accomplish it, users often don't even notice that there is something to improve in their drafting technique and tools they use in the software. (Bhavnani and John, 1996)

Learning to use CAD software doesn't proceed to the state of strategic learning. The cause for this seems to be the fact that the flaws in the drafting technique are not visible in CAD use since the outcome, the drawings are more or less the same regardless how them have been produced. Thus ineffective use of the software doesn't reflect directly in the quality of the drawings. This is different in the world of manual drawing, where the user's technique is clearly visible in the quality of the drawings produced by him/her. (Bhavnani and John,

1996) Additionally, users don't usually discuss about their use of the software with their colleagues, neither do they watch over other people's shoulders to see how they are using the software. The users are also often in mood of just wanting "to get their job done" and thus not willing to explore the possibilities of the software. Thus offering assistance while they are "getting their job done" is an important aspect when developing the user assistance.

These same phenomena were also noticed within our observations at the user's offices: the designers discuss design issues and ask for help with some specific commands, but don't actually address the issue of improving efficiency in their discussion. This leads to the fact that often inefficient usage of the software persists since users don't get feedback from any source to get remedies for their ways of working.

2.4.2 Mental models

Mental models can be referred as knowledge of the components of software, their interconnection, and the processes that change them and actions that can be taken to accomplish the change (Carroll, 1988). There has been some other interpretations of the term mental model, like the system designers model his/her design (Carroll, 1988), but in this study we refer to it as the representation the of the user on how a computer software works.

The concept of mental model was introduced in the 1980's, and today it is being used widely by human-computer interaction experts.

When a user starts using a program, his or her prior knowledge on the software and his/her expectations concerning it undergo a change within the process of knowledge acquisition. In this process the user's mental models are restructured to fit the new information. It seems that the theory of mental models can be used to explain the changes that appear when learning to use a CAD software. (Bhavnani and John, 1996)

Mental models can be studied most often by means of an interview. Other methods include: user observations, free association, quizzes. In this study we didn't aim into revealing user's mental models about the Tekla Structures since Linja-aho (2005) has studied them earlier in more detail. However, from the user observations and the usability tests results some users’ mental models can be identified conflicting the design of the software.

Mental models need to be taken into account when designing the contents of the user assistance. Since the user's picture of the application's structure and relations is different from the ones in the actual application also in the case of Tekla Structures (Linja-aho, 2005), it is important to pay attention to making the best effort to correct these misunderstandings. Many of the places where the user's picture of how the software works and the actual functionality of the program differ are related to different tools and making modifications to objects. Many of the misunderstandings emerge from the mental models that users have about software they have used before (Linja-aho, 2005). This is natural, since the expectations on new software come from the software that the users have used before, which in this case most often refers to 2D drawing software.

The user assistance's task is to lead the uses out of these former expectations by providing instructions on using the functions of the software and thus replacing the wrong

expectations. When assistance is easily available, the users are more easily willing to look at it. It is also important to tell the users the differences between their further experiences in the 2D world and the new 3D modeling world. Even though this information is best offered via the on-line help, the user assistance solutions must provide the user with an easy access to this information when it is needed.

2.4.3 Visual layout: perceiving objects on the screen

The human perception process is a complicated process of neural activation and its interpretation. The retina of the human eye constantly receives stimuli which are transferred via the optic nerve to the brain's cortex. There the received data is pre-processed according to various schemes including similarity, edges, colours and movement, before being transferred to the conscious level in the brain. There a perception of the processed information is formed, which doesn't contain all of the information received from the retina. (Sinkkonen, Kuoppala, Parkkinen and Vastamäki, 2002). Because of this it is important to know the basics of human eye perception when designing objects for the user on the screen. Since not all data that comes to the retina is not perceived, the objects meant to be noticed must have proper properties.

Visual perception is not just seeing things. It greatly incorporates interpretations and attention aiming. Even the mental models discussed above can have an effect on the things the user perceives on the screen: all is interpreted according to previous experience.

To make the user perceive a thing on the screen the threshold for making a perception must be passed. The perception threshold is unfortunately dependent on many factors. For example, the mood of the user affects it, as well as his or her current aims.

The user's attention on the screen can thus be attracted with the following methods for example: The information is placed at a logical location where the user might anticipate it to be. The use of visual cues that easily pass the perception threshold, like flashes, small movement and sound can also be employed. Also the correct temporal placement of the desired object helps its perception. (Sinkkonen et al., 2002, pp. 94-120)

3 Complex applications for experts

In this chapter complex applications for experts and the complex application used in this study, called Tekla Structures, are introduced. An overall picture of the building modelling process that the Tekla Structures is used for is given.

Computer aided design in building modelling is nowadays handled in two different main approaches: two dimensional (2D) often referred to as CAD software, and three dimensional (3D) software that are used to provide similar kind of result as with 2D software, but with different means. The case study software Tekla Structures in this study belongs to the 3D group. Still today a lot of modelling with computer is made with 2D software (Linja-aho, 2005), and most of the users starting to use the Tekla Structures come from some 2D software background.

3.1 What are complex applications for experts

It can be said that there are mainly two types of software that people use: simple and complex. Usually tasks that are simple by their nature, can be implemented in a simple software, but simple tasks can also become complex with poor design. Complex tasks that require many steps, perhaps some engineering skills etc. are more often complex by their nature, and when implemented in a computer software, they very easily become complex as well. But this is not a necessity – with good design complicated tasks can be made simply to accomplish. By the term complex application we refer to the type of software that is used for tasks that are complex by their nature and require some educational background to the subject in addition to the ability to use the application's user interface.

Perhaps the biggest difference between the software that is used by non-professional computer user and a complex application for experts is the need to have prior education and knowledge about the expertise field of the software. In order to be able to use a complex application efficiently, the user has to have to have some knowledge about the subject the software is about, for example building modelling. These kinds of applications are used only by experts with the education to the subject and they use the application as their working tool. They are often used daily by their users. However, expert in this context doesn't necessarily mean that the user would be experienced in using the software at hand – he or she may only have the needed education for handling the subject.

3.2 Construction design, the building modelling process

Building modelling process is the process of designing and constructing a building or other construction. It encompasses all the tasks from gathering needed base information, selecting correct materials and profiles for the supporting structure to completing the calculations on loads and stability. It includes the work of several parties such as architects, electrical engineers, structural engineers and HVAC (Heating, Ventilating, Air Conditioning) engineers. (Linja-aho, 2005)

The modelling is today mostly made with computer software, but some designers still use the old traditional manual drawing board. The results of the modeller’s work are the drawings according to which the building construction workers build the actual construction. Building modelling not only covers houses, but also bridges, towers, masts

etc. The building modelling process is strictly regulated by law, and it consists of several phases.

With 2D software the drafting and modelling are made in two dimensional planes, and the result of the model, the building for example, is in the modeller's head. With 3D modelling software the actual shape of the building is clearly visible all the time, and all modifications, additions etc. are made directly to the actual shape of the construction. The modeller thus sees the result right away. In a way the 3D modeller thus doesn't have to use as much imagination as a 2D modeller has to in order to see the result of the work. (personal communication, March, 2005)

Regardless of the fact that most modelling is today made with 2D software, the transition to 3D world is going on with an increasing speed. The benefits of 3D modelling are inevitable and the computer capacities are also today enough for the rather heavy CPU load that the 3D modelling software require.

3.3 Tekla Structures building modelling software

Here the complex software called Tekla Structures and its users are introduced. The current means of providing the user with assistance in the software are introduced too.

Tekla Structures is the software that was used as a case study program in this study. Tekla Structures is 3D modelling software used for modelling steel and concrete buildings by Tekla Inc. Tekla is a software company whose main product the Tekla Structures is, but the company has also other products what are used by expert of different fields. The Tekla Structures software can be used in different parts of the building modelling process from calculating the rigidity of the structures to detailing the connections between parts of the building. It is being used mainly by structural engineers, but also other parties involved in the building modelling process are users of the software.

Tekla Structures has formerly been known as Tekla XSteel, but the name has been changed recently when the possibility to handle concrete parts was added to the software. Currently the user can purchase the software with different parts included, called configurations, the most important ones being the steel and/or concrete configurations.

Being a true 3D modelling software, Tekla Structures allows the design engineer to see the shape of the building being designed. The model can be zoomed, rotated and panned freely so that the designer can see different parts of it clearly. The main screen of the Tekla Structures user interface is illustrated in Figure 3. As can be seen from the figure, the user interface consists of numerous icons and menus. The icons on the left and top of the window are used for different kinds of operations in the software, and all of the actions that can be taken with the icons can also be performed via the menus. The icons on the right side of the screen are used for different kinds of connections, and there are several pages of those – the user can change the tool-icon page by pressing the arrows. These connections can also be found from a special component catalogue.

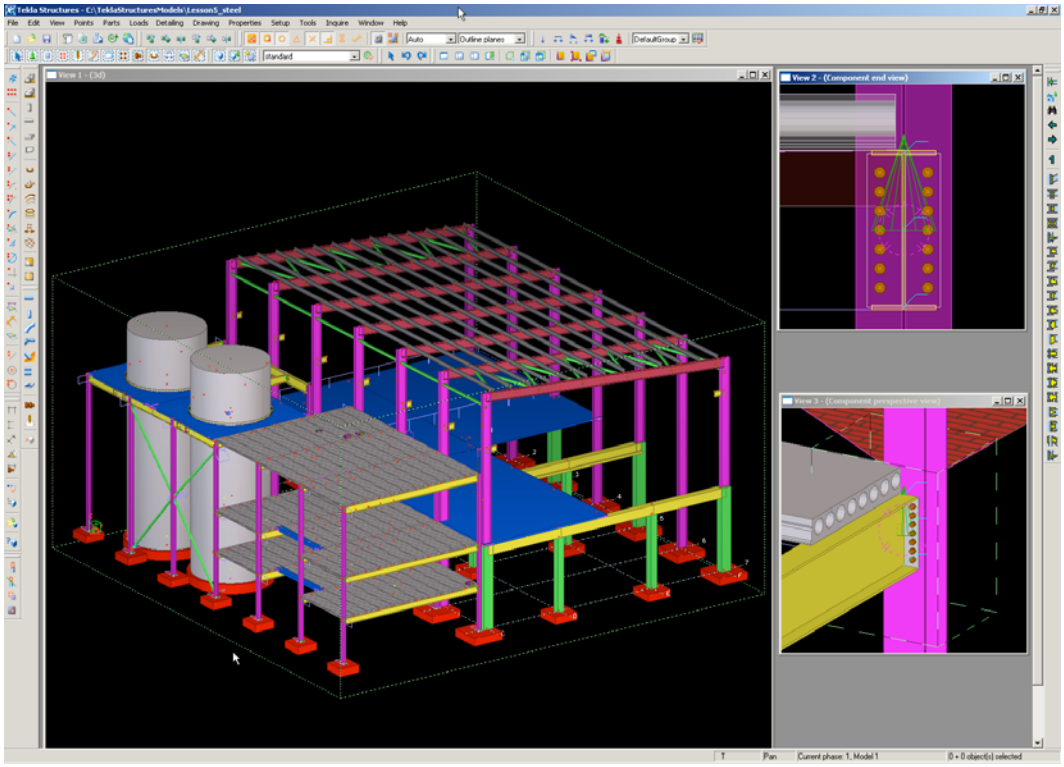


Figure 3: The Tekla Structures main window and user interface

Most of the users of Tekla Structures are of construction sciences background and are using the software as their main working tool. Thus they are familiar with the constructional terminology and processes and use the software daily, often all the time. Because the software has formerly been merely supporting steel building modelling, most of the older users today are experienced with the steel tools of the software, but more unfamiliar with the tools and commands related to concrete parts. There are still lots of users who are using the software only for steel detailing and modelling.

Currently the software does not provide the user with much help during the use of the it. The software is shipped with an on-line help which is quite comprehensive and has been lately improved to include procedural step-by-step instructions of some commands in the software. The user can evoke the on-line help from the Help menu in or by pressing the function key F1 in some of the dialogs in the software.

Tekla is also arranging new user's training that lasts for four days. During this session the users are taught the principals commands and functionalities in the software. Currently it is quite difficult to start using the Tekla Structures without attending the training.

4 Data gathering

In this chapter the methods used for gathering information from the users are introduced. Of each method the theoretical background and reasoning, the arrangements, the results and the discussion of the success of the method are described.

To collect data from the users about the studied subject there are various methods available. Mueller (2002) describes in his article the use of getting acquainted with the calls that have come to the technical support, user interviews, surveys and focus groups. Holtzblatt and Beyer in turn in their description of the Contextual Design process suggest the extended use of site visits consisting of interviews and observations. Hackos and Redish (1998) also suggest the use of interviewing and observing.

All these methods aim to the same target: to understand the users' tasks and the environment the user is working in when performing the tasks. To understand the work structure, what details affect to it, what is important and what is not. Hackos and Redish (1998) use the term task analysis for this activity.

In order to get a comprehensive picture of work of the users of this complex application, we decided to use a combination of different data gathering techniques suggested by different authors. We follow loosely the usability engineering methodology set suggested by Holtzblatt and Beyer's (1998) Contextual Design process. This process describes the user centred product design process and emphasizes the importance of following the user's real needs in the design of a product or software. Creating user assistance solutions closely involves users and their needs and the Contextual Design process is very good in involving the users in all phases of the design process. Contextual Design process' data analysis methods were also good in the type of this study where lots of unstructured data about an expertise field not very familiar to the researcher was to be collected.

However, we decided that the plain use of Holtzblatt and Beyer's process is not beneficial in this context since in our view it concentrates more on designing whole new software instead of making an improvement to an existing one. We were also interested in trying new techniques such as the use diary that had not been utilized formerly in Tekla. We thus left some small parts of the Holtzblatt and Beyer's process away, the most important being the storyboarding and added some other methods. The time constraints and manpower resources were also behind the reasoning when selecting the set of methods used for the background information gathering.

The methods chosen for the phase one of the study, the background information gathering and task analysis are described in the Table 1.

Table 1: The used data gathering methods in the first phase of the study

Method	Answers that the method gives
Tekla Service desk personnel interviews	• What kinds of problems typically exist with Tekla Structures • In what kind of situations the problems typically emerge
User interviews	• What is the current process of seeking for help? • What types of problems typically exist with Tekla Structures

	• What kind of guidance is preferred • Does the mental model of the user correspond the model of the application • The typical tasks, how hectic the work is • How new things in the software are approached
User observations	• What is the current process of seeking for help • What kinds of problems typically exist with Tekla Structures • In what kind of situations the problems typically emerge • The typical tasks, how hectic the work is • How full is users' screen
Use diary	• What kinds of problems typically exist with Tekla Structures • In what kind of situations the problems typically emerge • The typical tasks • How hectic the work is
Web questionnaire to the users abroad	• What is the current process of seeking for help? • How new things in the software are approached
Getting familiar with other complex software	• What kinds of ways to give assistance are there currently • What are the typical problems • When should the assistance be available automatically

The methods and reasoning for selecting them are described in detail in the following chapters.

In order to get information about the difficult tasks in the software we also considered the use of a few usability tests in the first phase, but since this method didn't seem to be much used in the early phases other studies of this field and because of the time constraints we decided to leave it to the second phase only.

4.1 Subjects of the study

Since the users of the case study program Tekla Structures are using the software for modelling of buildings of either steel, concrete or both, we had to first decide what kind of users we want to include in the target user group. After interviewing the Tekla Service desk personnel (see chapter 4.2), we got the impression that the differences between the overall types of problems and ways of getting help didn't differ remarkably from each other. The both user groups' basic tasks are quite similar and the command sequences they normally use in the program don't differ much from each other. Also, since the number of users who are using the software for both materials, steel and concrete, we decided that we don't leave either group of users out of the target users group. In addition, it is important to fulfil both user groups' needs when offering instructions and assistance since Tekla Structures is being used by both groups.

Both novice and experienced users were included in the target group, since it was important to find out the possible differences in the needs of these two types of users. Detailed descriptions of the selected to users participate can be found in chapter 6.2. The

participants of the interviews and observations are referred to as U + number, for example user one is U1.

4.2 Tekla service desk personnel interviews

Two interviews with Tekla service desk personnel were conducted as the first part of the empirical study part of this research.

4.2.1 Theory and related methodology

Interviewing the experts in dealing with the user's problems with Tekla Structures is an efficient method to get background information from the subject that is to be studied in detail. Since the researcher was new to the construction design field, it was good to start the information gathering by interviewing the persons who are professionals in assisting the users in their problems. We wanted to get good basis to the forming of the questions in other methods used and get to know an overall picture of the most problems with the Tekla Structures software with this method.

The interview method was chosen to be a semi-structured interview (Wood, 1997) since it seemed to provide the most efficient interviewing method of getting information on a subject the researcher wasn't very familiar with. Wood has developed and used this interviewing method in the early phases of software development projects and has adapted methods of ethnographers and cognitive scientists in it. In it the interviewer prepares a set of questions which are asked during the interview, but the discussion can flow to other topics too if the interviewee brings new topics to the table. The interviewer must frame the topics handled to the related subjects though. Wood emphasizes the fact the interviewer must try adapt him or herself into the language and terminology the interviewee uses about the tasks at hand, so that the interviewee does not start explaining things in a general and easier level in order to make the interviewer better understand. This would lead the interviewee to miss some important details about the tasks that are included in his or her work. Adapting is done by asking certain object identification types of questions that try to reveal the concepts and terminology of the user's work. Wood also suggests the use of object relationship type of questions in order to get better understanding of the distinctive characteristics of the organization and domain concepts.

4.2.2 Arrangements

The two interviews were conducted in Tekla's premises in the beginning of February 2005. The interviews took place in a meeting room, with only the researcher and the interviewee present. The venue was peaceful and good for discussing the subjects without disturbing the others working in the house.

The interviews were recorded with a computer recording software so that the researcher could later transcribe the interviews. The first interview lasted about one and a half hours and the second one about one hour. The researcher made notes constantly during the interviews with his laptop computer.

The interviewees were Tekla service desk employees. The first, younger of the two interviewees had worked in her current position for one year, the second one for three years.

The questions asked during the interview can be seen in the Appendix A.

4.2.3 Results

The interviews gave good overall picture about the types of problems that Tekla Structures users face and that what are the things that are mostly asked. The biggest problems that would benefit from improvements in assistance were identified to be:

- the import and export functions of the software
- the select filters
- creating reports and templates
- issues concerning printing, such as setting line stroke weight and the print area.
- adjusting the drawings so that you don't have to do it all yourself
- creating components
- picking order with new components
- making operations on empty groups; the user may think that operation was OK since no errors are reported but nothing was actually done.

Often the solution to the questions asked by the user is found directly from the on-line help. As it was expected, the problems and challenges distribute almost everywhere in the software even if it could be seen that there are more of them in the drawings editing functions and template editing. Thus it was decided that we don't concentrate especially on some specific area of the software, but try to handle problems with a large scope.

4.2.4 Discussion

The chosen method gave valuable information for preparing the other data gathering methods from the users of the software. The semi-structured interviewing method seemed to be a good choice since several topics that the researcher wasn't prepared to ask were covered during the discussion.

The results about typical user's problems, types of questions asked and types of help the users typically need seemed to be well covered by interviewing two employees from the Service Desk since partially same answers were given. Interviewing more persons might have given some more data, but it may have just gone into too detailed level. Some of the answers given in the interviews may have been biased by the employee's latest tasks, the ones that are currently open, but they could also remember things from the past when asked so this biasing wasn’t seen as a big problem.

4.3 Getting familiar with other software on the field

4.3.1 Theory and related methodology

A frequently used approach to get new ideas when designing new software or new features to existing software is to see what other suppliers of the same field or same kind of user interface have developed (Gould 1988). Since many software vendors have developed various kinds of user assistance solutions, and we also wanted to get a picture of the typical problems within complex applications, this method was chosen to be used.

4.3.2 Arrangements

Several different types of software were inspected in order to get a picture of the typical problems encountered within different types of them, and to get acquainted with different

types of solutions currently implemented in software. The researcher tested the total number of 17 different applications of several types: complex applications for experts' (professional's) use, complex applications for the non-professional users (like large word-processing software), small simple applications and games. The applications were test-used with the focus in finding what kinds of problems there exist and how the user is provided with assistance or embedded help during the use. Also different types of on-line help solutions were briefly verified. The features of the software were collected in a partially systematic manner to a prepared form sheet where different assistance types and their properties were listed. The assistance types put to the form sheet were collected from the literature and from the researcher own prior experiences. In addition to the pre-defined properties, all other forms of assistance provided were also listed to the form. The prepared form sheet was used in order to get the existence and type of the similar types of features checked in all the applications tested.

The reason for taking so many different types of software into the group of selected software was the intention to get a good picture of assistance types overall used in computer software.

4.3.3 Results

Several different ways of providing assistance to the users were encountered. Also the types of problems that are solved with assistance solutions were various. Here we present the most important findings, the filled form that was used to collect the data from different software can be seen in the Appendix H.

The use of tooltips was very wide. Some software used them in the traditional way with delay before presenting it, some made them visible immediately. Balloons of different forms were also used and this seemed to be a handy way of presenting information. Some applications, especially the complex ones used information panels that appear either when the user calls for it, or were constantly visible and changed only the contents according to the use. The What's this –type of function was also used and it seemed to be a good way of providing field-specific or dialog-specific information. Different types of tutorials were also used and they gave a nice feeling to the user because with the help of them it is possible to easily learn the important parts of the software. Context sensitive help was used extensively too. Introducing tools new to the user or tools with new properties in them, was used in one complex software.

The action games used a lot sound, like a little beep or short bell to attract the user's attention to the messages that appear from time to time. Some short messages were delivered to the middle of the screen, staying visible for a while and then disappearing. The longer ones had their own panel which attracted attention with movement in addition to the sound. Other games used a lot the method of getting more instructions with the right (the secondary) mouse button.

The problems and challenges that were found in the software had several typical properties. The visibility of the things going on currently in the software, and the visibility of the status of the system was often a problem. Instructions on how to recover from errors was a common issue also, which was solved with good error information dialogues in some cases, but often left without proper guidance. The visibility of the instructions provided also seemed to be a problem in many software applications.

4.3.4 Discussion

Overall the observations made by looking at other manufacturer's products, was a fruitful way of getting ideas and wider view to the problem field. The pre-defined form sheet that was used to collect the data was noticed to be a little too inflexible to this purpose. The problem was solved by adding free form comments to the end of the form fields. The good point in the form was that at least all the same things about each software were reviewed. This was beneficial in the sense that we could see how much each type of assistance was used among the selected software. It was a little difficult though to try to list comprehensively enough different assistance types beforehand. The list was updated during the inspection process.

Software types of a wide range were selected in order to get ideas and view from different types of solutions. The games were an interesting subject since in some of them, especially the action-packed ones, the information provided to the user is offered during very hectic situations where the user (the player) can't interrupt his or her current task but must be able to acquire the information quickly and easily during the task at hand. This can be sometimes compared to a situation with complex software when the user is producing something with a very tight schedule. Therefore interruptions in order to get instructions are not possible but the information must be presented to the user as apart of the task.

4.4 The user interviews

4.4.1 Theory and related methodology

Interviewing is an integral part of all usability research. It has been suggested by many authors, for example Hackos and Redish (1998), Nielsen (1993) and by Holtzblatt and Beyer (1998) as a part of their Contextual Design process. Interviewing users is an efficient way of getting information about them about their work and issues related to it. Many questions of usability issues can get answers just by asking from the users. This is especially true when talking about user's anxieties and subjective satisfactions, which are difficult to measure with any other method. (Nielsen, 1993)

It is important to notice however, that all design issues don't get answered by interviewing the users. Since the interviewing is an indirect method, it gives answers only about the user's opinions and not about their actual behaviour. Also, user's opinions about user interface elements they have not tried should not be taken literally. In a study by Root and Draper (1983) it was noted that there isn't always much correlation between the predictions of the users about features in the user interface and their opinions about them after trying them. Despite these downsides, interviewing users is a very good method in getting data from the users, especially if accompanied and supplemented with other methods. Also, because we were also asking questions about the ways of working and getting help, we thought that many of the answers wouldn't be too much biased by the predictions of the users.

When doing the interviewing, Hackos and Redish (1998) emphasize the important fact of keeping the questions neutral and non-leading. The questions asked were aimed to bee not leading, and in the interviewing situation the interviewee tried not to share any of the users' sometimes strong opinions about the software, but remaining neutral about all answers and

possible questions. We followed also the ethical guidelines Nielsen (1993) gives about the participants in the usability research.

4.4.2 Arrangements

The user interviews were conducted in February 2005, at the users' own offices. Two of the interviews were held in meeting rooms, and the rest in the users' own work desks, in the open office cubiculum or rooms. The intention was to make all the interviews in the users' own work desks as Holtzblatt and Beyer (1998) also suggest having the real working environment around, but in two cases the other workers in the same space would have been disturbed by the interview. In all cases the setting was peaceful enough to discuss freely the issues. The interviewing was estimated to last about one hour, and this was told to the users also. The actual length of the interview varied from about 50 minutes to one and a half hour. The interview was audio-recorded and the researcher made notes with a laptop computer during the discussion. The recording was done in order to be able later catch the exact words of the interviewees and complement the notes made during the interview.

As in the Tekla personnel interviews, we used the semi-structured interviewing method (Wood, 1997) as it was noted to be a well functioning interviewing style in the Tekla personnel interviews. The users were asked for their experiences in the software on how they seek for help currently, how they find it, where they face problems, what are their preferences in getting instructions, and other similar questions. The interview questions are available in the Appendix B.

Eight illustrations of possible assistance solutions were shown to the users at the end of the interview and they were asked for opinions about them. Each picture had a small scenario connected to them which was told to the user after seeing the picture. The solution ideas illustrated in the pictures were created according to the data gathered from the method of getting familiar with other software on the field (see chapter 4.3) and the ideas that the researcher had invented while getting familiar with the software. The scenarios and the illustrations are also listed in the Appendix B.

The participants in the interviews were chosen from the Tekla's customers in Finland. The researcher asked from the Tekla Service Desk connections of a few possibly interested Tekla's customers. The customers were contacted by the researcher and most of the ones asked agreed to participate. The total number of participants gotten to participate was six persons: three of them were novice Tekla Structures users and three experienced users. All of them were males. Detailed descriptions of the users with profiles of them are given in the Appendix D.

4.4.3 Results

The results of the interviews were analyzed with the affinity diagram, which is described in detail in chapter 5.1. A short summary of the results of the interviews is presented here.

All the users have been watching the status bar messages, but some of them have given is up more or less because of it's low information content. They would like to see more information the way the status bar it offers.

Difficult things in the software are editing the drawings, creating custom components and concrete tools in modelling. All the users have been reading the tooltips and most of them seem to be reading them still, at least sometimes.

Error messages are a problem. The current ways of providing the user with help haven't been obstructing the users, most probably because there are so few of them. The problems are more in the fact that there is no information available. Getting instructions is a problem, and finding it takes so much time that often the users don't bother to look for it and rather try to do things themselves even if the way wouldn't be an efficient way. An efficient way of getting instructions while using a tool or starting to use a tool seems to be something the users would like to have. The instructions they want to get should first be in a very brief crystallized form, and only after that more help should be available upon request. The instructions should contain also information on where the command can be used and examples of usage. The users like to plagiarize and look from other models how things have been done. It is common to ask for advice from a colleague, often because finding advice in other ways is so difficult and there often are no instructions available. If a user is working alone without colleagues close, he is in a much more problematic situation. The software must support also these types of users.

The users had a surprisingly positive attitude towards all of the presented illustrations of possible assistance solutions.

4.4.4 Discussion

The method used and the results gotten with it were considered satisfying by the researcher. A large amount of interesting data was collected, and the questions set used was working. The questions asked inspired the users to tell more and some of them started to tell a lot of their ways of using the software and means of getting help. In some interviews the subject of the discussion tried to move too much to listing different kinds of hopes and wishes concerning some problematic tools and functionalities in the software. Because of this some interviews took more time than was estimated, but fortunately the interviewees didn't seem to be bothered about this. Some of them seemed to very much like the possibility to say aloud some their frustrations with the software. Despite this the discussion was all the time in a positive tune, and the interviewer managed to set the focus back on the desired track after the sidetracks with new questions.

4.5 User observations

4.5.1 Theory and related methodology

Observing the users' normal work is also a very important part of the user task analysis and user centred software development. Various authors (Nielsen, 1993 and Hackos and Redish, 1998 for example) suggest it as a part of gathering data in the aim of getting familiar with the users' world and task. Observing is the simplest of all the methods in the usability field, since all you have to do is to just go to the user's work site and see what he or she is doing and do as little as possible not to disturb him or her. One advantage of observing is also the fact that it is possible to find unexpected ways of using the software while observing the users in their normal daily work. (Nielsen, 1993) Observing is also known by cultural anthropologists as an important method of getting data of unfamiliar subjects (Bhavnani, Flemming, Forsythe, Garrett, Shaw, and Tsai, 1996). Hackos and Redish (1998) suggest also a more participating way of observation in which the observer asks the user to think-aloud while working. Nielsen (1993) suggests that the observer should ask some questions in order to clarify some actions that are impossible for the

observer to understand. It is also important that the observer stays as an observer and doesn't start acting as a expert the user can ask questions about the software. In this study we used the approach suggested by Nielsen, so that the researcher was mainly observing.

4.5.2 Arrangements

The observations were carried out with the same users as the user interviews, right before the interviewing. The observation session lasted for about one hour. The observer introduced himself to the user and explained that he wants to see the normal way of working with the Tekla Structures software, to find out what kind of problems there are, how tools and different types of assistance provided by the software are used. The users were asked to act as if the observer wasn't there, but were told that if they want to say something or explain something, they can feel free to do so. They were also told that the observer may in some points ask the user to clarify or explain something they are doing.

The observer took notes while watching the user. The observer used a prepared form sheet to capture similar facts from each user. The categories on the form included the ways of seeking help, types of problems the user encountered and the properties of the user's working spot. The form can be seen in the Appendix C. In addition to the form, free formed notes were also written.

Although it was aimed that six users would be observed, it turned out that three of the users were not using the software at the moment in any project and thus they didn't want to be observed. Thus three users were only interviewed, and three observed and interviewed. One of the observed users were novice, two were experienced in the use of Tekla Structures.

After both the interview and the observation was over, the user was asked for his interest to participate the second phase of the study, the testing of the solution ideas. All of the participants promised to participate.

The users were rewarded with a small gift, and they were told about it after all the activities in the interviewing and observing session by letting them to choose a gift of their desire from a short catalogue. The gifts were delivered in the second phase testing meeting.

4.5.3 Results

The results of the observations were also along in the affinity diagram, which is described in more detail in chapter 5.1. Here a short summary of the results is given.

It was noted that the users tend to face a lot of problems with the Tekla Structures software, but they have developed ways to recover from them. An important way of getting help seemed to be asking a colleague. The users also used a lot the method of trial and error to learn and try new things. The feedback from the software was identified to be inadequate in many places and the users had to for example look from the Windows Task Manager to check whether the software has frozen or not during long operations. Other user had to check from another model how a component is used in order to make it work.

The users seem to be willing to have all the commands they use in the program in their control and see the results of them. It was noted during the observations that information about possibilities in the program would be appreciated. One user was very frustrated about the fact that he had been trying to use a same command for "a hundred of times"

without any success. In hectic situations the users seem to be willing to have information about what they are doing wrong, even if they admit that providing the correct information might be difficult. In a hurry non-specific instructions or even instructions on wrong subject are not welcome, according to the observations. Exact instructions are often received from a colleague.

The feedback from some of the commands was also noted to be inadequate, for example one user used to press apply and update buttons in the dialogs several times to be sure that the button press has been recognized by the program.

4.5.4 Discussion

The method was very good in giving a view of the users' tasks with the software, and the problems and recovering from them. A lot of valuable data was collected. The effort to arrange the session in view of the amount of data gathered was very small.

The use of a pre-defined sheet was not as nearly beneficial as the free form notes that were taken. The sheet gave some information about how much different titles were repeated, but it was a little too inflexible in the varying observation situations. Still the use of the pre-defined form can be considered to be good by using it the bias of the researcher could be lowered since the researcher had to define before the tests that at least the titles listed on the form are considered interesting and valuable.

There was a little problem with the notes, since the observer couldn't all the time keep up with all the happenings. This was especially the case with one user who ended up in some places explaining with enthusiasm the things he was doing and what problems there are with the tools. This particular user also went at times a little too much to the side of showing what kinds of problems he is encountering with the software instead of working on his normal tasks.

4.6 Web-questionnaire to users abroad

4.6.1 Theory and related methodology

A questionnaire published in the web was used to get same kind of information as with user interviews, about user's ways of getting help, their typical problems and their preferences and opinions about user assistance., from users outside Finland.

Questionnaires can be an efficient method of gathering data from several users. No other method can achieve as good coverage of the user population as questionnaires can (Nielsen, 1993). Questionnaires are good in producing quantitative data to analyze and they don't demand as much usability staff time as other methods, such as interviews. However, questionnaires are much more inflexible since they work best with closed questions, i.e. questions with readily given answer alternatives. Open questions in questionnaires are possible, but users often don't bother to write answers in them or if they bother, they often give so cryptic comments that them are hard to interpret. (Nielsen, 1993)

Attention must be paid to the forming of the questions. They must be formulated so that there is as little as possible room for interpretations. This also applies to the answer alternatives. If there are too many possibilities to interpret the meaning of the questions, the results can be quite unreliable. (Nielsen, 1993).

Although, according to Nielsen, there exists some sophisticated questionnaires such as the QUIS (Questionnaire for User Interface Satisfaction) method by Chin, Diehl and Norman (1988), we didn't use them in this study since they are more aimed at measuring the user satisfaction, which wasn't our intention. The number of questions should be kept relatively low in order to maximize the response rate (Nielsen, 1993).

4.6.2 Arrangements

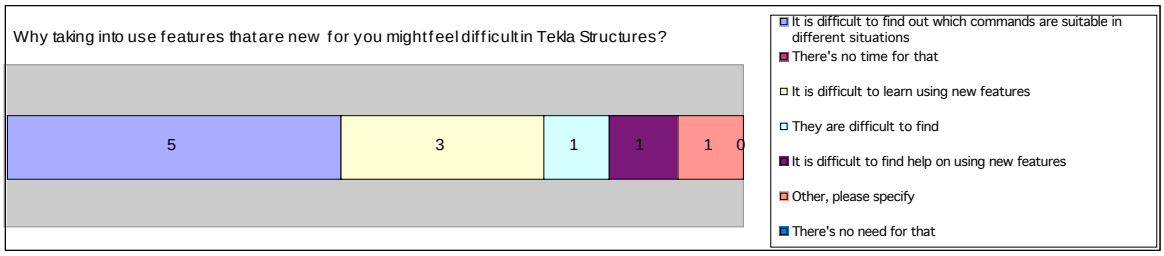
The web questionnaire was aimed to the customers of Tekla Structures outside Finland. The form consisted of 28 questions covering the same fields as the interview questions: current ways of seeking help, opinions about assistance and also problematic areas in the software. Most of the questions were closed questions so that the user could select one of the pre-defined alternatives. Many questions concerning the user's opinions used the Likert's 4 step scale having "I would like" at the other end and "I would not like" on the other end. The questionnaire questions can be seen in the Appendix F.

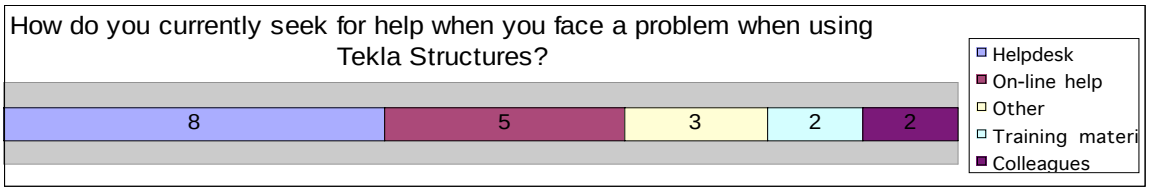
The participants in the web questionnaire were chosen by the local Tekla offices in the USA, the UK and Sweden. The participants were sent an invitation letter by email to participate the study. About 25 letters were sent, and 11 answers were received.

4.6.3 Results

The results from the web-questionnaire were mainly supporting the results gotten from user interviews and observations.

80% of the respondents had been using other 2D software before starting to use the Tekla Structures. 10% (one user) had been drawing by hand before Tekla Structures. This didn't seem to affect his answers when comparing to other respondents. 90% of the respondents used the software daily. Trying oneself was considered also among the respondents the most important way of starting to use new commands.





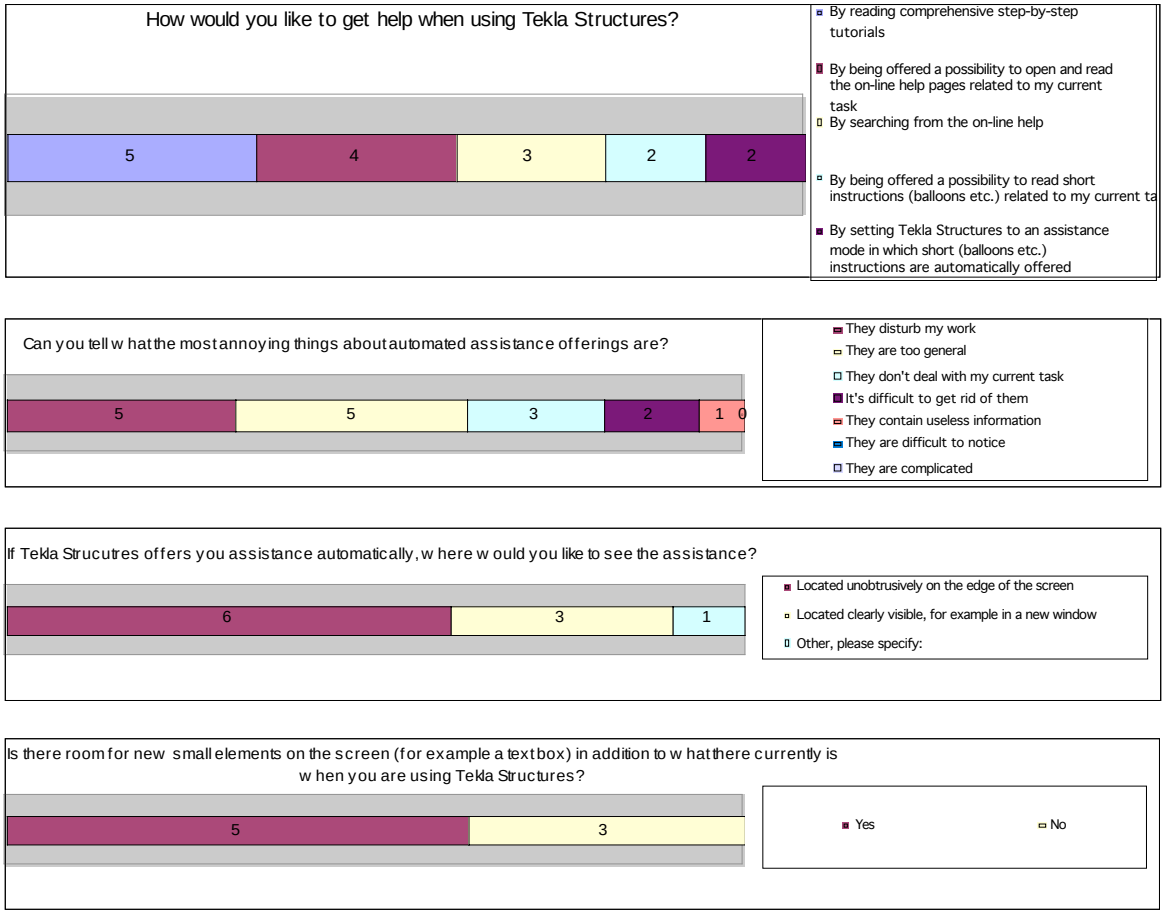


Figure 4: Six selected web-questionnaire results. The numbers in the bars indicate the numbers of answers to each category.

An important finding is that the users feel that the most important reason for having difficulties in taking new commands into use is that it is difficult to find out which command is suitable in different situations. Another important factor was the lack of time. However, when asked if they have time for learning new or unfamiliar features in the software, the answers were divided half and half: half fully agreed and other half fully disagreed. With user assistance it is possible to give the users examples on where the new commands could be used.

The users' attitude towards automatically offered assistance divides the respondents opinions completely, half would like it, other half would not. This is a clear indication having such feature in the software favoured, but it must be controllable. The assistance should be located in the edge of the screen, according to 60% of the users. The use of sounds gets users' careful support, 30% saying that they would like it, 20% that they would not and the rest being between these opinions on the Likert's scale.

62% of the respondents said that they have space for some additional elements related to user assistance on their screen.

Six selected question answers are illustrated in the Figure 4. The detailed answer to each question in the questionnaire can be found from the Appendix G.

4.6.4 Discussion

The web questionnaire gave usable results even if the number of responses was left under the target. Even if most of the results followed the same trend as the interviews conducted in Finland, there were some interesting differences too. The use of on-line help was rated surprisingly high when asked for what the user does when starting to use a command that is new for him or her. In the interviews the use of on-line help seemed to be remarkably lower. It is an interesting question whether the users answering the questionnaire about user assistance have tuned themselves into a mode where they considers the on-line help valuable, to indicate that it is being used and it should not be removed. Or is the difference due to differences in the social habits at the work place: do the Finnish workers get more easily help from their colleagues and thus are more eager to ask than to look from help. Or is the reason the fact that in this study we had only one user from an organization with no colleagues to ask from - unfortunately in the web questionnaire this fact was not asked from the answerers.

The making of the questionnaire was more difficult and time consuming than was expected. The difficulty mainly wasn't in the technical implementation but in the fine-tuning of the form of the questions. The layout and publishing of the questionnaire took of course time, but the most unexpected time consumption was in the forming of the questions. Since we wanted them to be as accurate as possible, it took time to think about all the possibilities.

4.7 Use diaries

4.7.1 Theory and related methodology

Cultural probes or just probes have been used in many studies about finding information target groups daily life and work. For example, Gaver, Dunne and Pacenti (1999) used cultural probes to get familiar with their project's target group whom they didn't know much about, the elderly people. They used different kinds of probes, including cameras and media diaries delivered to the participants. The probes are delivered to the participants, left for them to use and fill, and asked to be returned to the researchers after a certain time period. They are like the probes used in medical or astronomical purposes: sent to the unknown, and waited to return some day with lots of fragmented data in them. (Gave et al., 1999). Use diaries can be seen as one kind of cultural probe. Them are also sent to the participants, and asked to be returned after a period of time. The users fill the incidents that they encounter during their normal working days. The value of the probes lies in this fact: the data collected is not from a special setting, but from the normal day. In this study, the participants, the users are asked to fill the diary each day, and finally send it back to the researcher.

4.7.2 Arrangements

Two users were asked to participate this method. They were sent an empty diary template which contained a few subjects that the researcher especially wanted to get data about. The template can be seen in the Appendix E. The subjects concerned the situations when the user didn't know how to proceed during that day, the places from which they sought for help concerning this situation, the ways how they solved these situations, things that they felt were slow or inefficient with the software, and things that they found satisfying or

unsatisfying in the program. There was also room for general thoughts and other things the user might want to share. The user was asked to fill the diary at least once a day, preferably several times. As a reward, a small gift was promised to be given.

4.7.3 Results

Unfortunately other of the diaries was not received back, so only one diary results were in our use. The findings from the user diaries were analyzed with the affinity diagram, which is described in detail in chapter 5.1. Here a short summary of the results of the user diaries is presented.

It seems that the users are confronting problems in finding commands from the user interface. They are asking help from the colleagues, but in this case the user received hint for a command which he couldn't find. Only after looking from the help he succeeded in finding it. It seems that the users like to create custom components of details that they have to use several times, but if the custom component created doesn't work in all places it was intended to work, they don't have time to try to fix the custom component. Rather they make the malfunctioning parts by themselves by just modelling them piece by piece. They also might copy some existing connection or detail, and modify it to the new place. This can often be slower than creating a functioning custom component, but it seems that creating a custom component is a rather difficult task. Picking problems seem to emerge every now and then with new commands or commands that haven't been used in a while. The user also complains about the need to make changes to different text files in order to change certain settings.

4.7.4 Discussion

The use diary probing method was a new method to be used in Tekla, and in this study it had a kind of experimental position. The number of participants was small, but it doesn't necessarily reduce the value of the data collected with this method. Several interesting findings were made one of the most important being that the findings from the observations were in line with the findings from the use diaries and thus we can argue that the observations results are not much affected by the presence of the researcher. Other interesting finding was the user's need to use the on-line help to locate a command he heard from a colleague.

All the findings can be considered to be of special value since them were encountered during the normal day of the user. Use diary can be considered to be a special case of user observation with the different that in the case of the use diary one can be sure that the presence of the researcher in the user's office doesn't contribute to the things happening.

When thinking about the results afterwards, it would have been good to include more participants to this part of the study. This is because the results were well usable and the effort to get them was rather small. The downside of the method of course is that enthusiastic enough users must be found, and one can argue whether these enthusiastic users present the typical user of the software. Also we can't know how the user reports the incidents, what is left untold and what is perhaps expressed too much. Despite these facts this method is well worth using in other studies too.

4.8 Overall success of the data gathering

Because of the nature of the study, we didn't collect quantitative data nearly at all. This decision seemed to be quite good since with the qualitative methods we managed to get information for our needs.

The choice of data gathering methods was quite good. An interesting setting would have been to use a controlled experiment of having the users using the software and asking for help from an expert through a textual chat system and analyze the types of questions they have and the answers the expert gives. This would have given a good view of the types of questions users need answers to, and the typical problems. We however already had a quite good picture of the problems with the Tekla Structures so we wanted to concentrate more on the ways of getting help and opinions on it.

5 Data analysis and creation of the solution proposals

In this chapter the analysis methods used to analyze the data gathered and the process of creating and composing the solution proposals are described.

After the information gathering part of the study there was a large amount of unstructured data about the user's typical tasks, their typical problems and overall the user's way of working. To make good use of all this collected data, the data needed to be organized and analyzed in order to make to be a good basis for innovating assistance implementation ideas. Hirsjärvi, Remes and Sajavaara (1997) suggest that the analysis of collected data should be started as soon as possible after the data gathering period and thus right after the last interview the analysis phase was started. Since the gathered data was mainly of qualitative form, we decided to use methods that aim at understanding the data, in other words methods that aim at making deductions and qualitatively process it. Quantitative methods were only used with the web-questionnaire results since it was the only part of the study that produced enough responds for feasible quantitative analysis methods.

It was decided that the researcher invites other members of the usability team and the implementation teams to participate innovating. Since other members of the usability team were going to take part in the implementation ideas generating, a clear presentation form of the collected data was also needed. To meet these needs we chose to use the methods explained in the next chapters to structure and visualize the data: affinity diagram, and the work flow models within the affinity diagram. The same method set was suggested by Holtzblatt and Beyer (1998) as a part of the Contextual Design process in the phase when there is data collected from the users. Brainstorming or innovating session with other team members was used after structuring and visualizing.

We decided to leave the storyboarding part of the Holtzblatt and Beyer's (1998) Contextual Design process since the data we were using concerned so much things of the computer screen that physical drawings seemed not to serve their purposes as well as they would in the Holtzblatt and Beyer's cases.

5.1 The affinity diagram

To analyze and structure the data gathered from the background information gathering part of the study, i.e. the interviews, observations and diaries we chose to use the affinity diagram. Holtzblatt and Beyer (1998) suggest the use of an affinity diagram in handling unstructured data from interviews and observations. Affinity diagram is a bottom-up method to group and analyze data collected from the users, their tasks, environments and problems. It is a good basis for later generating solution ideas based on the collected data. It offers a method to transfer the issues and notes collected from the users to design ideas. (Hackos and Redish, 1998)

The use of affinity diagram was chosen since there was a need to make a clear representation of the data collected. The researcher was also interested in trying this method with the relatively large amount of data, since he had only once used the diagram before and at that time the amount of collected data was significantly smaller. On the Tekla's point of view the use of an affinity diagram was also interesting since the method

had not been previously used in the usability team in the house. Thus in this study the usability team members of Tekla had the possibility to see this method in use.

Also it was good for the other teams in Tekla to see what kinds of usability research methods there are that can be applied in Tekla's business scope. This reason was an important point of view since the usability research is a relatively new topic to many of the employees in the house. The use of new concrete usability research method was seen as a good publicity for the usability team as a whole and the usability issues overall in the house.

5.1.1 The process of creating the affinity diagram

First all the data from the interviews was transcribed. Hirsjärvi et al. (1997) point out that complete transcribing of the interview recordings is not always reasonable and the level of transcribing should be chosen according to the further analysis methods chosen. In this case we decided also that we don't need a word-by-word transcription since the affinity diagram doesn't need transcribing in word-by-word level - the ideas, phenomena and other observations are sufficient detail level (Holtzblatt and Beyer, 1998). With this in mind, we decided to transcribe word-by-word only those phrases of the users that were considered especially important or that might enlighten or proof something important.

A separate room for creating the affinity diagram was reserved. This was done because it was expected that a lot of wall space is required for the diagram and the open office cubiculum where the researched normally worked didn't have much empty wall space. Also Holtzblatt and Beyer (1998) suggest the use of a separate design room where all the data is visible during the work. The use of a separate room was a very good decision since lots of wall space was truly needed. The researcher first took into use 2/3 of on one wall in the room but ended up in using most of the wall space in the whole room which had about 12 square meters surface area.

First all the interviews, observations and diaries of use were read through and all the events, opinions, phenomena etc. were written on separate post-it sticky notes. All the notes were put on the wall so that they were clearly visible. All notes were numbered so that the origin of the note could be easily tracked: the user number and an ascending numbering. The researcher used note papers of the same colour for the all data sources but used different colour coding in the numbering of the notes to mark the origin of the individual note. This way it was easy to distinguish notes from interviews, observations and diaries of use. About 350 notes were written and the wall space that was at first thought to be enough for the whole affinity diagram was overfilled already at this phase.

When starting to sort the notes into groups there was a problem with the space. More space to attach the notes was added on the wall, but at first we didn't want to use two separate walls in the room for the affinity diagram. This is because it felt that it's easier to see the whole picture if you could do it just with one glance. Taking another wall into use would have meant that part of the notes are behind your back when looking at the other half. It became clear after a while though, that more space was needed. Too little space at times introduced the problem of not being able to add lots of small groups instead of larger groups. This is because every single group needs a space around it and thus larger groups need less wall space. Especially in the beginning of the affinity process this led to too large groupings. The researcher noted that it was often more difficult to later on divide a large

group into smaller groups than immediately create enough small groups. According to Holtzblatt and Beyer (1998), the maximum number of notes in each group should be four, since larger groups easily get a name that is on a too common level. Finding out the real reason behind words of users or their actions was sometimes very difficult. It was much easier to just find a common denominator among the notes on a common level than to figure out the reasons. It was decided that the group titles should be concrete statements as Holtzblatt and Beyer (1998) suggest. This was a good decision since with this kind of titles it was easier to decide which notes belong to each group.

The number of notes being relatively large and since the researcher was doing the task alone, it took much longer to complete the diagram than expected. The researcher spent almost three full days of work just to group all the notes, and several days were spent to first write all the notes up. The creation part of the affinity diagram was done alone because it would have been difficult to get people from other teams to participate the work for a whole day since all of them are quite busy with their own work.

We used different colours for different subject levels: green notes represented the highest (first) level titles, cream-white notes the descending next level (second) on the hierarchy, and red notes the lowest (third) heading level. Not all white notes got a green level heading and even some red headings were left without white heading because some of the groups were so small.

Figure 5: The affinity diagram, the other half of it

5.1.2 Results and discussion of the affinity diagram

After finishing the grouping there were 10 high level group titles, 41 second level group titles and 87 third level group titles. All the group titles are described in the Appendix I. Group titles identified from the data were very valuable source for later processing of the data in producing the solution ideas in and finding and revealing reasoning and justification for all the solutions. In the solutions chapter 7 the groups have been referred to as reasoning of the suggested solutions.

The most important findings made with the affinity diagram can be considered the first level group titles which are listed in the Table 2. The number of sub-titles also indicate the importance of the title The most important findings are also described in the chapter 4 within the user interview, user observations and use diaries results chapters. For example

the title “My experiences with instructions” has the biggest number of sub-titles and it was one of the most important sources of solution ideas.

Table 2: The first level group titles in the affinity diagram and the number of sub-titles

First level title	2nd level titles	3rd level titles	Total no. of sub-titles
I want to keep all the wires in my hands	2	4	6
The past time is interesting	3	7	10
I want to get information about my possibilities	2	4	6
My attention is needed	1	3	4
When facing a new command	5	12	17
My experiences with instructions	11	21	32
Typical difficult things	2	10	12
We learn…	4	6	10
When I face a problem	2	6	8
I use the software like this	6	14	20

Even if the use of this method took very long to finish, the result was seen worth the effort. With no other method familiar to the researcher could the large amount of user data be structured and grouped as well as with the affinity diagram. The diagram was used extensively when finalizing the solution proposals in finding reasoning for them and in fine-tuning them to meet the user’s needs.

The use of this method also caused good publicity to the concept of usability in the Tekla since, as mentioned, the method wasn't earlier used and it was a really visible in the research room. When “walking the data” as suggested by Holtzblatt and Beyer (1998) all the participants of the brainstorming sessions said that the data was very interesting because they could see what the users actually are thinking about the software and how they are using it. Also some other Tekla people came to see the diagram after the researcher had announced that people can freely come and see it.

5.2 Creating the solution ideas

It was decided that the researcher doesn't try to innovate solution ideas alone, but with the help of other Tekla employees. Holtzblatt and Beyer (1998) suggest use of a design team through the affinity process, but in our case this was not possible. We therefore introduced other team members to the innovating process at the stage when the affinity diagram was ready. As Holtzblatt and Beyer (1998) suggest, we used a "walking the data" method to familiarize the other team members with the data on the diagram. In this method the team members walk the walls covered with the affinity diagram in the design room and read the group titles and findings from the users to get a good picture of what has been found from the user's world. During the walking each team member had a stack of notes in their hands and used them to write immediately down all ideas that they had when reading the notes. After each member had read enough the notes we continued discussing the design ideas

and developing some of them. It was stressed to the participants of the session that filtering ideas at this stage is not recommended and that the wilder the idea, the better it could be.

Two different brainstorming sessions were kept. Two different groups consisting of Tekla employees from production, documentation and usability teams spent one hour each reading the diagram and writing ideas about user assistance and discussing about them. These brainstorming sessions produced a considerable amount of ideas and were thought by the participants to be a good way to get familiar with the customer data collected. It was noted however, that one hour is quite a short time to familiarize oneself with the data and then start creating innovations based on it. Otherwise the brainstorming sessions were considered successful. The researcher got 123 design ideas to evaluate, and the other Tekla employees who participated had a good chance to get familiar with the user's way of working. Since this kind of research had not been much done before in Tekla, it was a very valuable opportunity for example to the production team members to read what the users have had to say.

We didn't use drawings to visualize the ideas during the brainstorming like Holzblatt and Beyer (1998) suggest. This is because the researcher found it quite difficult to make drawings about the computer screens and possible interactions on it. This felt like a right decision since the ideas that were innovated, were mostly ones that firstly didn't need any illustration to clarify the idea, or secondly would have been difficult to draw sufficiently clearly.

5.2.1 Selecting the solution ideas

The third meeting was held a few days after the two first ones. Four Tekla employees from different teams were invited but unfortunately two of them couldn't finally make it to the meeting due to some urgent matters. Other participant was from the usability team and other from the production team.

The ideas that were created during brainstorming sessions were sent to the participants before the meeting and they were asked to select three best from them. This way we collected eight different solution ideas from the total number of 123 ideas. These favourites from each participant were discussed in detail first, for example what they consist of and why them seem to be important. Then all the eight ideas were written on the wall and of each idea at least two positive and two negative sides were discussed and written on the wall. This way we got to think about the problems about both sides. The eight ideas with the positive and negative sides discussed are described in the Table 3. The pros and cons of each solution idea were collected so that the researcher led the participants to discuss good and bad sides about the ideas.

This method of analysis was chosen since that way we were able to concentrate on the most interesting solution proposals from the total large number of ideas. It would have been a very long task to go through all them in a similar manner and the people from the other teams would not have had the time needed for this kind of selection. The method seemed to work quite well. We succeeded on having a vivid conversation about the subject and managed to find the required at least two positive and negative aspects of each subject. The ideas were at the same time developed further.

Solution idea	Positive sides of the idea	Negative sides of the idea

A history log	The user gets to know what has been done.	The history log easily becomes too long for easy reading.
	This gives the user feedback about the command that he or she uses.	The readability can be difficult in a long list.
Novice/experienced modes	The amount of instructions and assistance is easily controlled.	It might be difficult for the user to know in which state the software currently is.
		The appearance of the software may change and cause confusion when changing the mode.
		The user might not want to rate oneself to novice and therefore miss some important assistance.
Instructions offered in a detected error situation	When there is a clear error situation, the user usually is ready to read instructions.	There is a possibility to give wrong instructions if the user is doing something different than the software thought.
	A good possibility to offer suggestions to use the functions more efficiently.	The instructions can easily be on too general level and thus irritating.
Following the user's actions in order to find out inefficient usage of tools and offer for example shortcuts	A good way to enhance and promote the efficient use of the software's tools.	Automation may be irritating especially to advanced users.
	A possibility to teach the user how things should be done with the software.	The level of automation is difficult to determine.
	The user might get a positive experience if the software offers a better alternative to accomplish a task.	Wrong offerings may cause frustration.
A direct way to access the command from the on-line help	Novice user finds commands more easily.	Some users might start using this way to access tools.
	An easy way for the user to start learning by trying.	Most users probably look for the commands in the user interface rather than from the on-line help.
Wizards	Eases the user's memory load	It is not possible to do all the things the user might want to do with the pre-defined wizard.

	The users make less errors	Advanced users may not want to use a wizard.
	The need for a separate help becomes smaller	The wizards should be short enough.
Separate message fields in addition to the fixed status bar	Continuously visible	May cover something important or disturb the user
	Easy to look at	User may not notice changes in a message box and a suitable way to attract attention may be difficult to find.
	Fits more text than status bar.	
Tutorials	Learning from examples	Tutorials are easily too long.
	Good for beginners	It may be difficult to find the answer to the problem at hand.
	The user may learn small tricks from the tutorials	Experienced users may not want to see the things they already know that there are in the tutorial.
	The user learns the correct ways to use the tools	

Table 3: The solution ideas selected in the meeting and their strengths and weaknesses.

In addition to the meeting, all the ideas were compared with a method of priorization listing created by L. Lehtola based on the article by Lehtola, Kauppinen and Kujala (2004). The method consisted of listing all the solution proposals into one table. Nine different aspects considered being important in the view of the Tekla Structures where chosen and all of the proposals were evaluated according to the aspects. They were given a grade from 1 to 3 for each of the aspects. Thus all the proposals got a grade from 1-3 for each of the nine aspects. The grades were summed for each proposal to get a total ranking for the proposal. After all the solutions were evaluated this way, we had a list which we could sort to see the most important solutions. The aspects the proposals were graded against are listed in the Table 4.

The list of the most important solutions was not usable as such since there were so many solutions with similar points and because the evaluation according to the chosen aspects couldn't give an absolutely correct answer. We decided to use the ranking given by this method as a guideline in determining which solutions are the most important. We chose solutions with the highest ranking to the testing phase, but also a number of solutions from lower rankings.

Table 4: The grading aspects of the priorization method

Aspect	Upper category	Definitions
Novice users	User types	• How many customers it will affect? • Effect on the easy start
Experienced users	User types	• How many customers it

		will affect? • Effect on the efficiency of use
An effect on pragmatist customers	User types	
Sales argument	Sales	• Beating closest competitors
In how many places can the solution be applied	Solution type	• How well it links to other products • The maturity of the solution
Unobtrusiveness of the solution	Solution type	
How much problems does the solution solve	Solution type	• The effect on the basic tasks
Criticality of the problems that the solution solves	Solution type	
Technological issues supporting the solution	Technology	• Compatibility with the current technologies • The artificial intelligence needed • Workload

6 Testing the solution ideas with users

The functionality of the user assistance or embedded help in a complex application is a combination of its usability and its pleasantness and non-obstructiveness. A good assistance solution with poor usability is not of help to the user and an obstructive solution with easy-to-use control is not either. The question of obstructiveness can be considered even more important than the usability of the ideas. Because of this we chose to test the assistance solution ideas with users with a prototype, to get their impressions and feedback. After choosing the ideas with the methods described in the previous chapter for further processing, a prototype with these ideas implemented was created.

6.1 The method

Holtzblatt and Beyer (1998) mention that it is often quite hard for the user to figure out how a product fits his or her needs, only after seeing a screenshot and a verbal presentation of the product. With scenarios and use cases the users can comment the solutions at the level of telling whether they like is or not, in other words it is possible to get an answer to the question what matters to the customer. But it is difficult to get an answer to the question how the system should be structured according to such customer data. We decided that the case is similar in this case when a new larger functionality is introduced into an existing software product and decided to try the user assistance and embedded help ideas with users. During the testing and the post-test interview the researcher tried to keep focus on finding the areas in which the suggested solutions fail, and not only to validate the solutions, as Holtzblatt and Beyer (1998) suggest.

Nielsen (1993) points out, that designing a product according to early user interface designs is not beneficial since it is much more effective to build first a prototype and try the functionality with it. Naturally it is possible to build the actual user interface directly and test the usability with it, but it is easily a lot more expensive to make changes at the phase when big parts of the user interface are already implemented. With a prototype on the other hand, the implementation ideas can be easily tested and changed according to the results of the testing. This is why we also chose to try and test the user assistance implementation ideas with a prototype. Nielsen (1993) also stressed the fact that without a prototype it is very difficult for a user to understand what the usually very abstract plans about a product or software actually mean.

Since, as mentioned earlier in chapter 4.4.1, the plain interviews can give a biased picture of the user's needs. Therefore we decided that user testing is very important to have reliable results.

Within the scope of this study it was not possible to build an actual functioning user assistance to the target application Tekla Structures. Thus prototyping the implementation ideas was the only feasible way to get feedback from the users about the solution ideas. We chose to use prototype which is a combination of horizontal and vertical prototypes, called prototyping with a scenario (Nielsen, 1993). This means that we cut both the number of functions and also the depth of the implemented functions. This was done since it was not feasible to start implementing more realistic user interface for this kind of testing. Since the user assistance implementation ideas consist not only of functions that a user should be

able to use, but also functions that the user must feel comfortable with, it seemed sensible to build prototype that just functions on a pre-planned path and does not allow the user to do everything with the software.

Because we wanted to get the user's impressions more clearly visible in the tests about the assistance ideas, we also chose to use a technique called "forward scenario simulation" introduced by Cordingley (1989). In this technique the experimenter asks at times the user questions of type "What would you expect would happen if…" or "What if the interface did this…". This way we expected to get more data about the user's feelings towards the user assistance solutions.

Scenarios (Nielsen, 1993) were used in creating more realistic situations for the test tasks. In a scenario the user is given an imaginative situation where and according to which he or she is supposed to do the test task. The users were given the brief scenarios before the test tasks.

The tests with the implementation ideas innovated from the user data were conducted with the same six users that also participated the interviews. It was considered important to conduct tests with the implementation ideas with the real users before making any further or detailed proposals about user assistance implementation. With the testing we wanted to find out how well the users can use the proposed user assistance solutions. One of the main points of interest is the user's ability to find the proposed solutions from the user interface. Also, to find out, what are the users' reactions to them are, was an interesting point of view.

We chose to use the think-aloud protocol during the tests (Salter, 1988). This method is suggested by many authors to be used in usability tests (for example Nielsen, 1993) and it is considered to be a good way to get to know what the user's real intentions are when he or she is working with the test tasks.

6.2 The participants

The users were asked to participate the tests already during the interviews and observations and all of them accepted to participate also the testing part of the study. All of them were chosen to participate the study after suggestions from the Tekla Service desk personnel. The researcher asked Service desk personnel to provide customers who might have time and/or interest to participate this kind of study. The selection was done randomly from a list of possibly interested or persons given by the Service desk so that three novice and three experienced users were chosen.

Half of the users were experienced Tekla Structures users hand the other half novice users. All of them were males and they were using the software as their main working tool at least in part of their projects. Three of the participants (Ut4, Ut5 and Ut6) were also using other software every now and then, depending on the project they are working with. One of the users (Ut6) was working in an organization where he was the only one using Tekla Structures and thus didn't have colleagues to ask for advice. The participants are referred to as Ut + number in this chapter.

Short descriptions of the users, user profiles roughly in the format that Holtzblatt and Beyer (1998) suggest, are presented in the Appendix D.

6.3 Test arrangements

We chose not to videotape the tests since it seemed that it doesn't provide enough added value compared to the efforts needed for arranging it. As Holtzblatt and Beyer (1998) point out, videotaping is not necessary unless you are looking at problems in detailed problems in the user interface. In these tests, the aim was not to pinpoint all small design flaws of the prototype. Besides, as Nielsen (1993) mentions, the analyzing of video tapes of a usability test take time from 3 to 10 times the actual duration of the test, and it was decided that the time constraints in this study don't allow using this much time into it.

Instead of videotaping, all of the tests were audio-recorded for later reference and part of them were also screen-captured with a special application that records all the movements of the cursor on the screen.

The tests were conducted in the user's own offices with the researcher's laptop. The user was using the prototype with mouse and the researcher acted as a test conductor and note-keeper. In some of the tasks the prototype required a key-press from the test conductor when the user wanted to use the secondary mouse click.

Nielsen's (1993) ethical guidelines on testing arrangements were followed. The user was told in the beginning of the test that the test can be interrupted at any time if the user wishes to do so. Permission to audio-record the situation was asked orally before turning on the recorder. The fact that the prototype is being tested, not the user, was expressed. The user was made clear that all findings and errors that the user makes are valuable data for the study and thus any problems with the prototype should not be considered as a failure. In Figure 6: The test setup at user's office. a test setup from one user's office is illustrated.

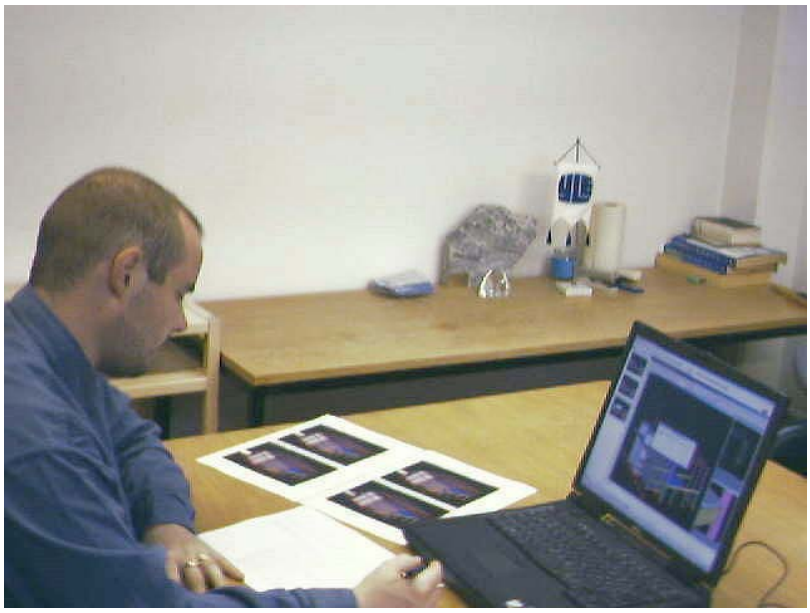


Figure 6: The test setup at user's office.

6.4 The prototype

The prototype was produced with Microsoft PowerPoint software so that it was possible to use it with a computer and click it with a mouse. This software was chosen to be the

prototyping tool since the researcher wanted to get a "clickable" prototype for the users to test to provide better feeling of how the user assistance implementations would affect the user's work. Other tools for prototyping are on the market, but for this purpose it wasn't realistic or feasible to purchase a license for such a program. PowerPoint was already available on the researcher's computer. Besides, the researched was doing the testing alone and it would have been more difficult to act simultaneously as the test leader and the paper prototype operator. A series of screenshots from the prototype can be seen in the Appendix J.

Nielsen (1993) points out, that when building a prototype, the designer has to make decisions of many different aspects of the system, ones that have not yet been thoroughly designed. These include for example font or colour selections. This was the case also in this study, and thus not all of the prototype's design decisions were directly adapted to the final implementation proposals even if them were not noticed to be unsuitable.

6.5 The tested solution proposals

With the methods mentioned in the previous chapters, we finally came up with a list of assistance solutions that we wanted to get more information about from the users. All the solutions are based on the data collected from the users. In this chapter the solution ideas that were taken into the testing and interviews are presented. After the tests the solutions were refined and revised and the results are presented in chapter 7: Final solution proposals. There also detailed descriptions on the basis of the solutions are given.

Table 5: The solution ideas that were tested or interviewed in the post-test interview

No.	Solution description	Task no. where tested
1	Easy to adjust the level, novice/experienced modes	3
2	Multi-level assistance	1
3	A possibility to get instructions on commands with the secondary mouse button click	1
4	A separate place to show instructions, and results of the commands; both constantly visible.	1
5	A larger status window instead of the status bar from which it would be possible to check what was the result of the operation. There's also a "more" button that tells why things happen. Visible all the time.	1
6	Quick inquire into status bar: information about selected object constantly visible in status bar	post-test interview
7	What's this -function in dialogs	6
8	History window, which can be used to return to a certain point in history. Also provides links to corresponding help or smaller guidance about the corresponding command.	2
9	Result of each addition comes to a log or message box.	2

10	Video or animation clips about new commands	post-test interview
11	Tip of the day into the splash -starting screen: the user must read it!	post-test interview
12	Providing hints and tips during the wait times the there's an operation going on. Waiting time's instructions related to the task currently at hand	post-test interview
13	Tekla Structures could suggest according to the result of an operation a better of more efficient way to perform the operation. A separate module that follows the user's actions and tries to offer help when it notices errors or non-optimal ways of doing a task.	5, post-test interview
14	An information box, bubble or larger tooltip opens when using a tool or a command for the first time	3
15	Top 10 component list, automatically updated	post-test interview
16	A search function which could find commands in the user interface. It would highlight the places where searched things are located.	4
17	Wizards on different subjects, such as creating/starting a project, drawing, template or report, importing	6
18	Short tutorial modules	post-test interview

Some of the solution ideas considered to be worth developing were left out of the testing because them would have been difficult to test with the prototype technology chosen or we already had asked the user's opinions about them in the first user interviews. The solution ideas left out of testing were:

- An instruction that is applied to the cursor
- Cursor appearance changes to give assistance: shape, very short instructions applied to the cursor, according to the selected tool.
- A sub-tooltip to the tooltips, available after clicking.

6.6 The test tasks

To find out the functionality and suitability of the innovated implementation ideas, a set of test tasks were created to be completed with the prototype. The prototype was created to support the test tasks. Six test tasks with small scenarios were created. From the large amount of possible implementation ideas were chosen the ones that have something functional or have possibly something that could disturb the use. The ones that had just something to show were mainly left out of the testing. This decision was made because the prototyping tool, PowerPoint was quite heavy and inflexible in this purpose and thus all the

ideas would have been difficult to test with the prototype. In addition, with usability tests it is better to test features that actually have something to test and not just showing something.

6.6.1 The 1st test task

The test assignments are in Finnish because the test persons were Finnish-speaking.

Assignment:

"Olet juuri asentanut uuden version Tekla Structuresista ja käytät uutta versiota ensimmäistä kertaa. Työskentelet uudehkon projektin kanssa jolla ei ole vielä kovin tiukka aikataulu joten sinulla oli aikaa asentaa uusi versio ja nyt hieman aikaa tutkia ohjelman uutta versiota."

"Huomaat uuden komennon, Create polygon surfacing, ja haluat tietää miten sitä käytetään ja sen jälkeen tehdä sillä laatan puoliksi peittävän kolmion muotoisen pinnoitteen. Tiedät että uudessa ohjelmaversiossa on uusia aputoimintoja ja päätät koettaa etsiä apua jostakin mahdollisesta sellaisesta on-line helpin, kollegalta kysymisen tai pelkän kokeilun sijaan."

Summary in English: Find instructions on how you can use the new command Create polygon surfacing.

Solution: right-click the Create polygon surfacing –icon. After getting the instruction, the user follows the instructions to complete the command. While executing the command, the status bar shows info about the actions that have been done and the message box shows instructions about the next step.

Tested ideas: Right-click info, information box in the upper right corner of the screen, history lines in the bottom of the screen, multi-level help, separated places to show instructions and results of actions, result of each addition comes to a visible place

6.6.2 The 2nd test task

Assignment: (continues directly from previous task)

"Huomaat erehtyneesi pinnoitteen muodossa, sen olikin tarkoitus peittää koko laatanpala. Haluat palata taaksepäin ja tehdä neliön muotoisen pinnoitteen kolmion sijaan."

Summary in English: You want to change the shape of the polygon

Solution: Use the history lines in the status bar to go back two steps, i.e. undo the creations of the surface and the third corner pick. Pick the new third corner and the fourth corner, complete with middle click.

Ideas tested: usage of the history lines, possibility to see individual picks

6.6.3 The 3rd test task

Assignment:

"Olet aloittanut ohjelman käytön versionpäivityksen jälkeen. Haluat luoda pinnoitteen Create surfacing –työkalulla mallissa olevaan pystyelementtiin. Työkalun kuva: 

Olet aloittanut juuri kiireellisen projektin ja haluat säätää pois mahdolliset ohjeistukset ruudulta, koska luulet osaavasi käyttää tarvitsemiasi työkaluja riittävän hyvin."

Summary in English: Set all the assistance off

Solution: Use the Assistance menu to set the assistance level to low.

Ideas tested: Novice/experienced modes, information bubble opens when using a command for the first time and suppressing it, multiple level help, the type of the second level help: colour, shape, position: this is asked also in the post-interview, easy to adjust the level of the assistance

6.6.4 The 4th test task

Assignement: "Haluat katsella model history log –lokia, ja tiedät että sellainen löytyy ohjelmasta. Et kuitenkaan muista tarkkaan että mistä, etkä halua lähteä selaamaan kaikkia menuja läpi."

Summary in English: You want to find "model history log" function, but don't want to search all the menus through to find it.

Solution: The user can use the help and search from there, of use the "Locate a command in the UI" assistance function. The command is situated in the monkey wrench icon that appears on the lower right corner of the window. The user must click the icon, select Locate a command from the menu that opens and type the desired command in the search field or look for the command from the alphabetical index.

Ideas tested: a search function or possibility to find commands or functions in the UI, a help icon constantly visible in the UI

6.6.5 The 5th test task

Assignment:

"Toteat että äsken muokkaamasi betonilaatta onkin vääränlainen ja haluat poistaa sen."

Summary in English: This task was designed to test how users react to the information message given to them after a simple command, in this case a deletion command which has an effect on the drawings.

Solution: The user selects the slab, and presses the delete button. He or she gets an information message to the bottom of the screen about consequences of this deletion.

Ideas tested: Reminders about consequences of actions in other places in the software

6.6.6 The 6th test task

Assignment:

"Tahdot tuoda malliisi FEM tyyppisen tiedoston sisältämät osiot."

Summary in English: The sixth test task was designed to try how users can use wizards and how they feel about them in Tekla Structures. It was also interesting to see how this particular task was accomplished with this kind of wizard.

Solution: Use the File-menu to find Import, and from there FEM model. Enter desired values in step-by-step dialogs.

Ideas tested: Wizards and especially their length and complexity of the individual steps. Suitability to experienced users, what's this function in dialogs

6.7 Usability metrics

We defined a set of usability metrics before the tests to evaluate the quality of the prototype as Nielsen (1997) suggests. We selected the percentage metrics since with the assistance functions it is most important to find out how well the concepts of the assistance are understand and that the users can sufficiently easily control the amount of assistance.

Task 1

1. 4/6 users find the right-click help without guidance

2. All users should find the right-click –guidance without excessive hints.

3. 5/6 users understand what happens when pressing the more button.

4. 6/6 of the users should notice the upper corner instructions while using the command.

Task 2

1. 3/6 of the users find the history listing and are able to click it without guidance. 2/6 of the users find the history listing and are able to click it with a little hint. At maximum 1/6 needs several hints to find and use the history.

2. 5/6 users understand the meaning of the history listing after clicking it.

3. All users notice the color coding

4. 2/6 should figure out the meaning of the colours.

Task 3

1. 4/6 users should find and use the assistance level adjustment menu without hint.

2. 6/6 should be able to close the assistance bubble without help

3. 5/6 should understand the concept of assistance level.

Task 4

1. 5/6 users should find the monkey-wrench icon on the screen without any hints.

2. 5/6 users should be able to click it without any hints after finding it, and after clicking they should be able to click the right command.

3. 5/6 users should be able to use the locate command dialog correctly.

Task 5

6/6 should understand why the information dialog appears.

Task 6

6/6 should be able to import the file without hint from the operator.

6.8 Test results

The results of the test tasks are described in this chapter. First, the results according to the pre-defined usability metrics are presented.

6.8.1 The results according to the usability metrics

The pre-defined usability metrics are described in chapter 6.7. In this chapter short description of the success of the test task is given, according to the pre-defined usability metrics. Overall picture of the results is promising. Only a two tests failed according to the pre-defined metrics, namely the noticing of the upper right corner instructions in task one, and the usage of the history listing in task two. These failures were taken into account when designing the final solutions described in the chapter 7

Task 1

Metric	No. of successful users	Number of unsuccessful users	Target level	Remarks
1. Users find the right-click help without guidance	5 (Ut1, Ut2, Ut3, Ut4, Ut6)	1 (Ut5)	5/6 successful. Target met.	
2. Users find the right-click – guidance without excessive hints	6 (all)	0	All successful. Target met.	
3. Users understand what happens when pressing the more -button	6 (all)	0	5/6 successful. Target met.	All users pressed the more -button, some expected the bubble to expand
4. The users should notice the upper corner instructions while using the command.	4 (Ut1, Ut2, Ut3, Ut6)	2 (Ut4, Ut5)	6/6 successful. Target level not met.	One user noticed the instructions when he tried the task the second time.

Task 2

Metric	No. of successful users	Number of unsuccessful users	Target level	Remarks
1. Users find the history listing and are able to click it	2 without help (Ut5, Ut6)	3 users need a little hint (Ut1, Ut2, Ut4) 1 user needs several hints (Ut3)	3/6 find without guidance, 1/6 with little hint 1/6 with lots of	Problematic area. Old conventions rule. The prototype was changed to the last test to ease the use of the history listing by changing the layout of the lines, and it seemed to be

			hints. Target level not met.	better.
2. Users understand the meaning of the history listing after clicking it	5 (Ut1, Ut2, Ut4, Ut5, Ut6)	1 (Ut3)	5/6 successful. Target met.	
3. Users notice the colour coding	5 (Ut1, Ut2, Ut4, Ut5, Ut6)	0	6/6 successful. Target met.	One user (Ut3) was not asked about the colours.
4. Users figure out the meaning of the colours.	2 (Ut1, Ut2)	3 (Ut4, Ut5, Ut6)	2/6 successful. Target met.	One user was not asked about the colours. One user (Ut6) is on the right tracks about the use of the colours.

Task 3

Metric	No. of successful users	No. of unsuccessful users	Target level	Remarks
1. Users should find and use the assistance level adjustment menu without hint.	5 (Ut1, Ut2, Ut3, Ut5, Ut6)	1 (Ut4)	4/6 successful. Target met.	
2. Users should be able to close the assistance bubble without help	6 (all)	0	All successful. Target met.	
3. Users should understand the concept of assistance level	5 (Ut1, Ut2, Ut4, Ut5, Ut6)	1 (Ut3)	5/6 successful. Target met.	One user understands the concept after a short explanation. The problem was with the high-low term.

Task 4

Metric	No. of successful users	No. of unsuccessful users	Target level	Remarks

1. Users should find the monkey-wrench icon on the screen without any hints.	5 (Ut1, Ut2, Ut3, Ut4, Ut6)	1 (Ut5)	5/6 successful. Target met.	One finds the monkey-wrench –icon with a little hint.
2. Users should be able to click it without any hints after finding it, and after clicking them should be able to click the right command.	6 (all)	0	5/6 successful. Target met.	The menu was easy to use.
3. Users should be able to use the locate dialog correctly.	5 (Ut1, Ut3, Ut4, Ut5, Ut6)	1 (Ut2)	5/6 users successful. Target met.	One users uses the Browse -button instead of the Locate -button

Task 5

Metric	No. of successful users	No. of unsuccessful users	Target level	Remarks
1. Users should understand why the information dialog appears.	6 (all)	0	6/6 successful. Target met.	

Task 6

Metric	No. of successful users	No. of unsuccessful users	Target level	Remarks
2. Users should be able to import the file without hint from the operator.	6 (all)	0	6/6 users successful. Target met.	

6.8.2 The test results task by task

In this chapter results of each test task are presented. Detailed descriptions of the paths that the users took and other performance are described in the Appendix K.

Task 1 results

The right-click help balloon seems to be good. The users didn't much comment its text contents. One user pointed out that a picture would be clearer, possibly because the instructions were in English and it is not the user's mother-tongue. All the users that noticed the upper right corner instructions followed them, and seemed to be satisfied with them. The more –button seems to be good, but opinions vary on what the users expected to open from there: mostly they expect to get a help window on the current task

Task 2 results

It seems that the users didn't realize that it is possible to actually click the history lines in the history panel (the status bar area). After realizing this possibility, all except one understand the idea of being able to go back with them, and also say that this functionality would be valuable. The prototype was modified to the last test to better express the possibility to click the lines by changing the layout of the lines, and it seemed that the change was to better.

The meaning of the colours is not identified very well. A good suggestion from one user was to use the same colours as in the components are used: green for a completed command.

Task 3 results

Many users would start to look for the adjustment from options in the tools or the setup –menu. The assistance menu is a logical place for selecting this. One user was first looking for a toolbar icon or similar switch to select "big help or small help".

An information bubble appearing from commands that are new for the user is mostly considered as a good thing. As one user puts it "It is a lot easier to turn the assistance on and get help on new commands than to open help and search from there".

Task 4 results

All users except one find the monkey wrench icon easily. After finding it, it seems to be intuitive to click it and open the locate command from the menu. All except one are able to use the locate dialog correctly. The users seem to like the possibility to find a command like this. One points out that this kind of functionality should be constantly available.

Task 5 results

Four users would like to have this kind of reminder in this kind of situation, and the two with different opinion would like to have this kind of reminders in for example numbering. It is also pointed out that it is important to be able to switch this kind of feature on and off according to the situation: when finalizing the drawings reminders of the effects of modifications or deletions are more than welcome, but in the editing phase you don't want to see them. A good idea might be to include a possibility to set this feature on from the drawings list.

One user would like to have a cancel button on the dialog that would undo the actions made. He also points out that with undo you can easily delete something you didn't mean to delete, for example when undoing a connection which has failed.

Task 6 results

The ones that haven't used the import command before, liked the wizard form of the function. The one, who has done importing more, says that he would prefer the old way of doing it. The "What's this" -pop-up balloon is accepted well, but the contents of it should be more precise: for example, does the software take care of the revisioning by itself?

6.9 User's opinions and expectations probing: post-test interview

In addition to the testing, we decided to probe the users' opinions and expectations with the help of an interview and a brief questionnaire after the usability tests. The users were shown pictures of different kinds of situations and asked their reactions about them. Since it is very important to know how users want to get instructions and what kind of instructions support their tasks the best way, we chose the interview and questionnaire approach. We decided to show them pictures and ask first how they feel about the illustration and more importantly, ask how they would suppose the situation would continue, what would be the next step for the user and for the software. This way we could get information on how the user anticipates the assistance to operate.

6.9.1 The post-test interview

The interview with the users was conducted directly after the testing, on the user's working desk. The environment was in all cases sufficiently calm for discussing the interview questions. The interview was again in a semi-structured format suggested by Wood (1997). The questions covered users' opinions and expectations of different assistance situations. They were for example shown an illustration of a tool help bubble and asked that what kind of instruction they expect to open if they click the More –button in the bubble. Also questions concerning reminders, the history panel and the "top 10 components" list were asked. For several questions the users were shown an illustration to clarify the situation. The interview questions and the illustrations showed are in Appendix L.

The post-test interview summary

The collected data was analyzed by categorizing it as suggested by Taylor-Powell and Renner (2003). The answers were categorized to each question's subject. The categorization was good in identifying the most prevailing themes and extracting important issues from the answers. The results are described below.

It was identified from the collected data that the users want to have instructions given to them in small steps: first short instructions, and them more if needed. We asked the users what they would expect to get when requesting more information after reading a brief instruction bubble. The answers supported clearly the use of a larger window with pictures and/or animations. Three users mentioned the help window, and only one would have expected to have only a larger bubble. As one user (Ut6) puts it: "When you want more information than the bubble gives, it's not only more text that you want but also a picture or animation." It seems that there is need for both information on individual fields in the dialogs and also overall information about different tabs and other entities. It would be beneficial to offer both. The most favourite components used in the mode divided interviewee's opinions half and half; others would have very much liked it and others didn't see it as an important feature. The history lines were noticed by all expect one user right

away when doing the first test task and the users said to have understood the difference in the contents of the instruction panel and the history panel.

The use of the waiting times seems to divide opinions. Half of the interviewees would like to use the waiting time for reading instructions about new or unused commands and the other half would just be satisfied with information about how long it takes to complete the command. Automated messages from the program have surprisingly high support: all the users would like to have them at least in some cases. The important thing seems to be the possibility to control the appearing of them. Information when doing something repetitively wrong, modifying something that affects other things that are not visible, suggestions about more efficient tools or ways to find tools all are all messages that the users would like to see. In the case of difficult and rarely used commands four users would like to have more control from the software in the form of wizards. Only one didn't see wizards beneficial. A possibility to skip the wizards was however supported. Controlled tutorials were something the users would like to have, especially the connections and components were mentioned as good tutorials subjects.

From the data collected in the first phase it was identified that the users need to see information about the object that is currently selected. What this information should be was asked in the post-test interview and the answers are illustrated in the Figure 7.

Answers to the questions by each user can be seen in the Appendix M.

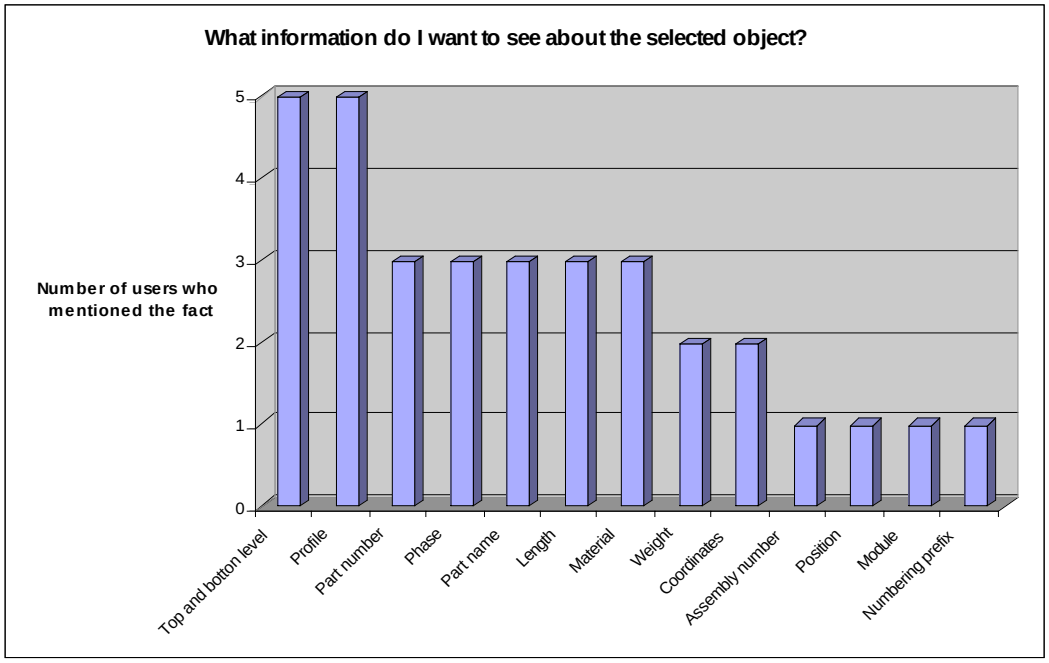


Figure 7: Information that the users would like to see constantly about a selected object

6.9.2 The questionnaire within the usability tests

The questionnaire was filled by the user between the tests and the interview in the presence of the experimenter. The questionnaire form consisted of three questions. Two were probing users' opinions on the lay-out of the information panels and one asked how the user would rank different kinds of assistance solutions to high assistance level category or

to low assistance level category. With the first two questions we wanted to get direct feedback from the users about the layout of the information provided. It was planned according to the facts identified from the collected data in the first phase of the study, that assistant modes with different sets of assistance solutions would be good. Thus we added the ranking question to the questionnaire, and with it we wanted to get the users' opinions in order to define a good default set of assistance solutions into each category.

Questionnaire results

The questionnaire results show that the users would like to see the assistance provided by the software clearly, i.e. presented with a bright colour. This fact was already identified in the first phase of the study, and with this result we got prove for it. The users want se clearly see when there are instructions available so that they don't miss them. The form that was used is presented in the Appendix N.

The contents of different assistance modes divided the user's opinions in some places, but with many of the asked solution there was still a clear trend visible. The solutions considered to be the most valuable were selected to be either always visible or in the low – set. The solutions that were seen either valuable only in certain situations, or more rarely, were put into the high level set. The answers of the users are illustrated in the Figure 8.

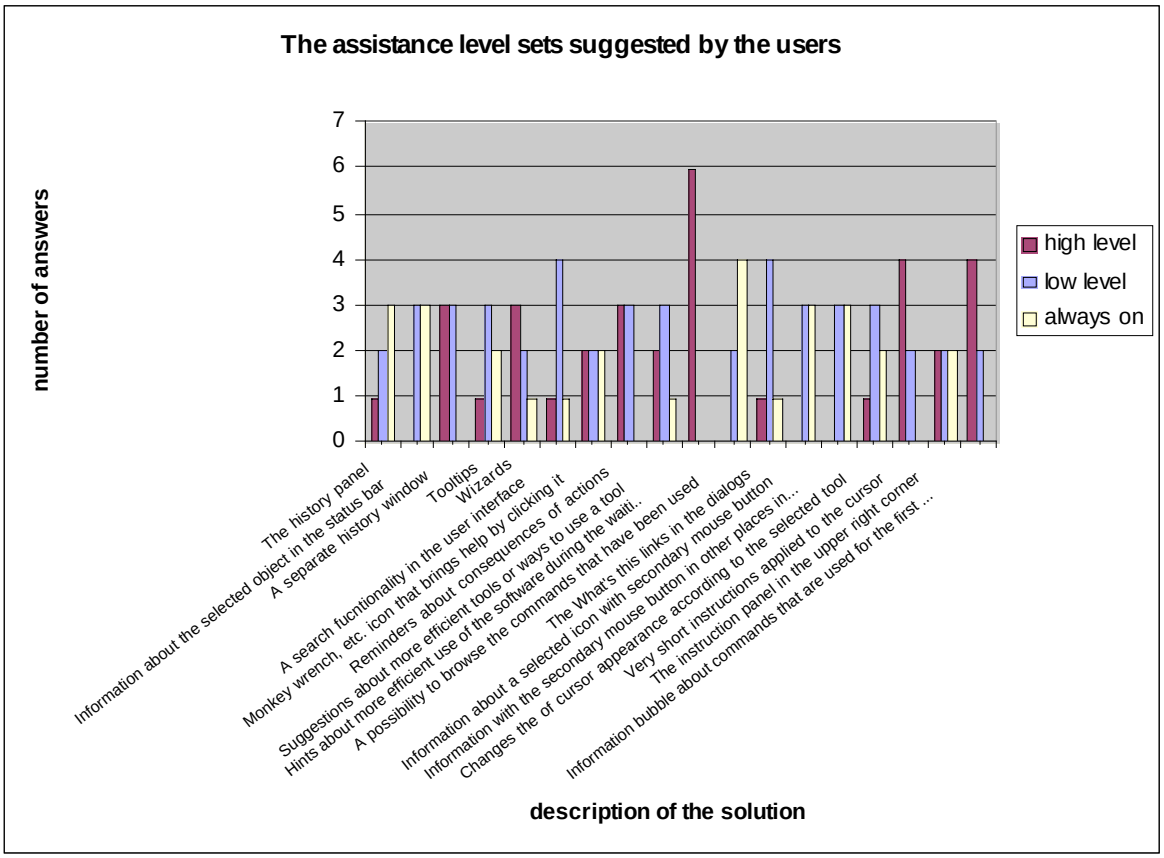


Figure 8: The assistance level suggestions by the users according to the questionnaire within the testing

6.10 An expert evaluation of the iterated solutions

After the user tests were conducted, assistance solutions that were tested were modified according to the test results. These solutions were reviewed with an expert evaluation method called cognitive walkthrough evaluation (Lewis, Polson, Wharton and Rieman, 1990). This method was used to ensure that the changes after the tests with users would not result in any obvious usability problems that could be corrected with the Nielsen's heuristics. Only the solutions containing something that the user has actually to use were inspected with this method. Not big changes were needed according to the result of the evaluation. The main modifications were related the visibility of the solutions, and tooltips were added to a few places. The changes made to the solutions are reported in the Appendix O.

6.11 Discussion on the testing

The audio recording of the tests seemed to provide enough accuracy about the happenings in the tests for our purposes and only in one or two places there would have been use of a video taped recording. Since the transcriptions of the tests were made soon after the tests itself by the researcher itself, it was quite easy to remember what happened during the test with the aid of the audio recording. The screen capturing technique used in one test added some value to the audio content, but the benefits of it were considered small in comparison with the disadvantages of it. The synchronization of the voice from the recorded files and the screen capture "video" file was a small problem in places. This was mainly due to the fact that while transcribing the voice, the output has to be stopped or slowed down once in a while in order to keep up typing with the voice and thus the screen capture video easily gets out of sync since it doesn't stop with the same key-press as the voice. Other disadvantage was the fact that the capturing software slowed down the performance of the laptop computer used to run the prototypes and the response times at times grew too long.

In the beginning we aimed at having visible all the time all the functions that are used in all test tasks, so that in the beginning of each task there are not always additions which would lead the user immediately to the correct path even if he of she otherwise wouldn't have used that path. This aim wasn't completely met. This may have been a small cue to some of the users on where to start doing the task: looking at new things on the screen.

7 Final solution proposals

In this chapter the final user assistance solution suggestions are presented. Each solution gets described in detail, and the reasoning based on the gathered user data for the selection basis and the form of the solution is given. Also, an illustration of the solution is presented, if one exists. In the text there are numbered cross-references in parentheses to the group titles identified from the gathered data with the affinity diagram. The titles and their numbers are listed in Appendix I. The solutions are all applied to the case study software Tekla Structures. Many of them can be generalized to various different types of complex applications for experts.

After testing the different solutions with the users and discussing with them about different possibilities, the final solution proposals were composed. The researcher went through the user interface of the Tekla Structures with the list of assistance solutions proven to be good and listed all the places where assistance could be needed and what kind of assistance in different places should be used. All the tested solutions were taken into this final suggestions list, many of them being slightly modified according to the test-, post-test interview and expert evaluation results. To see the exact test results, please see chapter 6: Testing the solution ideas with users. All the solutions are listed in the Table 6.

Table 6: Summary of the suggested user assistance solutions and their priorities

No.	Solution	Priority
A	Three assistance modes: High, Low and Off	1
B	Assistance mode can be easily adjusted from several places	1
C	The assistance instructions are provided in two levels	1
D	Help with the right (secondary) mouse button	1
E	Introduction to new features in commands or unused commands	1
F	The instruction panel	1
G	The history panel	1
H	The expanded status bar	1
I	Quick inquiry information	1
J	What's this –links in dialogs	1
L	Information about the new features presented while opening the software	2
M	Information about new features and unused features during the waiting times	2
N	Suggestions of more efficient ways of working	2
O	Recognition of erroneous usage	1
P	Top 20 component list	2
Q	Reminders about consequences of certain actions, error messages	2
R	A search function in the user interface	2

S	Clear showing of the progress of long operations	1
T	Wizards on rarely used functionalities	1
U	The right-hand side of the status bar	1
V	The user assistant icon	1
W	Instructions scroll back window	1
N	Suggestions of more efficient ways of working	1
Y	Tutorials on various subjects	2
Z	Suggestions on the contents of the on-line help and release notes	2

A. Three assistance modes: High, Low and Off

Summary

The user can adjust the assistance level quickly according to his needs. There are three different levels available: high, low and off.

Description of the solution

The assistance level can be selected according to the user's selection. The user can also turn all the assistance methods off. A pre-defined set of assistance methods is included in each set, and the user can also customize the high and low setting. Assistance solutions included in each set, high and low, are described in the

Table 7: Solutions included in high- and low –set

Solutions included in the high -set	Solutions included in the low -set	Solutions in use always
E: Introduction to new features in commands or unused commands	F: The instruction panel	D: Help with the right (secondary) mouse button
L: Information about the new features presented while opening the software	J: What's this –links in dialogs	G: The history panel
M: Information about new features and unused features during the waiting times	N: Suggestions of more efficient ways of working	I: Quick inquiry information
Q: Reminders about consequences of certain actions, error messages	V: The user assistant icon	P: Top 20 component list
O: Recognition of erroneous usage		R: A search function in the user interface
T: Wizards on rarely used functionalities		S: Clear showing of the progress of long operations
		W: Instructions scroll back

		window
		Y: Tutorials on various subjects

Background and reasons for the solution

User experience findings

This solution is based on the fact that users want to be able to efficiently control the amount of assistance they get. This need was identified from the data collected in the interview and use observations (1.2.2).

This solution was selected because of a clear need for easy control of the assistance was identified from the user data. Another possibility would have been to let the users control the assistance level individually by selecting each assistant option from a customize dialog. This solution was discarded because it would not respond so efficiently to the need for an easy and quick way to adjust the assistance. According to the user data, it seems that users don't have time or energy to go adjusting items they don't see necessary. Thus a default set is used, and the user can go and modify it if he or she specially wants to.

More importantly, it was identified that the users don't need more assistance only when they are novices in using the software, but rather depending on the situation: Sometimes experienced user too uses a new command or otherwise is unsure about what he or she is doing. The use situation can change quickly and many times.

Test and interview results

The tests show that users are familiar with, and are able to use different assistance modes. Since 5/6 six users could adjust the assistance level without any help and also understood the concept of assistance levels, this can be considered as a successful solution.

User comments on the solution

"I could use a tourist-mode with things that I haven't done before" (Ut5)

Priority:

1. Must be implemented in the first phase, affects all other solutions.

B. Assistance mode can be easily adjusted from several places

Summary

The assistance level can be adjusted from three places: the assistance icon, the Help –menu, and the Tools –setup dialog. Customize option is available too.

Description of the solution

The assistance level adjustment can be made from three different places, all of them affecting the same content. The first place is by clicking the assistance icon (see the solution V), and selecting the appropriate level directly from the choices in the menu. Second way is to open the help menu and select directly from it the appropriate level from the similar options list. Third way is to go to the options dialog. The user can also customize the assistance level, but only from the Help –menu and the Options dialog.

When selecting the customize option, a dialog is opened. In the dialog the user can select which assistance solutions are included in each set.

Background and reasons for the solution

User experience findings

This solution is based on the fact, that there is a need to be easily able to control the amount of assistance (1.2.2). It was identified, that users don't categorize themselves as novices or experienced users when talking about assistance needs, but they rather see themselves in different states of usage in different times: sometimes you want to, or have to, start using features that you have never touched before, and in this case some assistance is very welcome. Also, when using commands that are rarely used and don't contain lots of repetitive steps, assistance and suggestions by the software are accepted and welcome. However, when the user is performing a task that consists of some very familiar steps and the user knows what he or she has to do, there's no need for assistance and it may be considered obstructing. Especially commands that, require repetition of similar steps, are places where only information that is welcome is the information strictly related to the task at hand.

It was identified from the interviews and the observations that users' mode of operation changes according to the situation. The same user, no matter how experienced in using the software, may need assistance in different places of the software, and the situation changes. The novice user may be, or most probably will be more in the mode of needing more assistance and thus not willing to change his mode so often, but such user also is sometimes in a situation where no instructions are needed. Therefore is necessary to be easy to changes the assistance level according to the changing situation.

Test and the post-test interview results

The users were satisfied and not confused with the possibility to change the assistance level from both the assistance icon and the Assistance menu. One of them mentioned that he would have expected to find a toolbar icon about the assistance level.

Priority

This solution should be implemented in the first phase.

C. The assistance instructions are provided in two levels

Summary

The assistance that is provided to the user is offered mainly in two steps: first a short crystallized version and after that a longer detailed and illustrated version upon user's request.

Description of the solution

When users get assistance from the software, whether upon request or automatically, the assistance should be provided in two steps. First the user gets a short, very crystallized version of the assistance information available which can be read quickly. The idea is that the user can quickly read it and decide whether he or she wants or needs more information. Within the short version of the assistance information a button for getting more

information is provided. By pressing this button, the corresponding on-line help page in a new window is opened.

Background and reasons for the solution

User experience findings

This assistance solution is derived from the users' identified need of getting information quickly and especially being able to rapidly read through it to decide whether the command or functionality at hand is good for the current situation. (6.4.4) By being able to first read a short version of the instruction, the user can easily start using the command just by trying, or can choose to read and/or see more information before starting to use it. As trying was notified to be one of the most important ways of learning new commands, it is important to support it by providing instructions that can be rapidly glanced through.

Users pay a lot of attention to the quality of the contents of the information provided. They seem to be very critical in their every-day use of the software on what kind of text is provided to them. Especially the start of the instruction is important – in a hasty use case, which is usually the case for an average user, he or she doesn't want waste time in reading non-useful information. The judgement about the usefulness of the instruction is made very quickly, usually according to the first couple of sentences (6.3.2). Thus it is good to have first a brief crystallized instruction and a possibility to read more if the user considers it necessary.

Test and post-test interview results

The users liked the possibility to get first brief information and the possibility to get more upon request. All except one were hoping to get something more than just text after pressing the more -button in the assistance bubble. There would be need for at least a picture which would clarify the often complex situation significantly. The pictures especially help those who don't get the instruction in their mother-tongue. Animations were also considered an informative and helpful medium.

Priority

This solution should be implemented in the first phase.

D. Help with the right (secondary) mouse button

Summary

Clicking the right (the secondary) mouse button on toolbars brings information about the clicked command or tool. Right-click functions also from the Component Catalog, from the dialogs and the menus.

Description of the solution

The user can click the right mouse button on any of the toolbar icons or component catalogue. It is also possible for the user to right-click the items in the dialogs of all the commands. Right-clicking brings a bubble next to the clicked item. The bubble contains short instructions, and a possibility to view more information via a "More…" –button. The layout and approximate positioning of the bubble are illustrated in the Figure 9. The colour is light yellow.

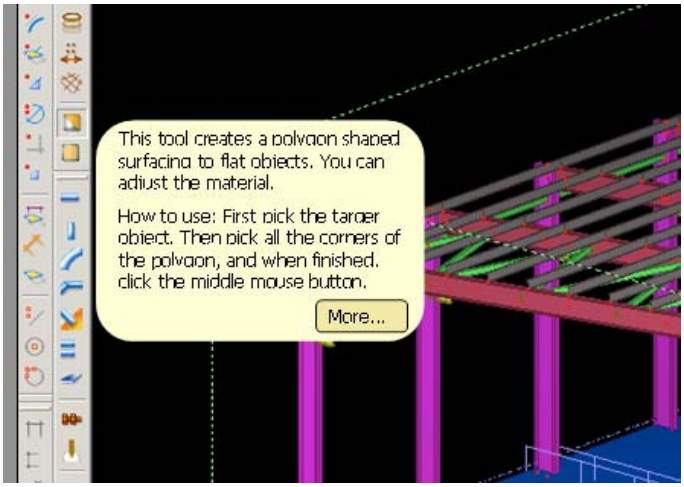


Figure 9: The layout of the right-click balloon

The contents of the information text in the bubble consist of the following elements (in this order):

1) What is this command for, where it can be used. One or two sentences.

2) How is this command used. Brief description of the usage of the command.

3) (optional) Very brief summary of what can be adjusted.

It is very important to keep the bubble very short. The maximum length of it should not exceed seven lines.

When the user clicks the "More…" button, he or she gets the normal on-line help page opened from the correct place: the beginning of the command. The help page contains information in textual as well as illustrated and/or animated form.

Background and reasons for the solution

User experience findings

Users of a complex application such as Tekla Structures have a large amount of commands or functionalities that they don't know how to use (Bhavnani, 1996). This was identified also from the collected data in this study: 20% users reported in the web questionnaire that there are lots of features they don't know how to use, and 50% informed that there are "not much" such features – none of them postulated that there aren't such features. The trend was also visible in the interview responses: users know that there are features waiting to be used, but are reluctant to start trying to use them. The problem lies mainly in the fact that the command and functionalities are often complicated by their usage and also that there is not easy and fast instruction available – the users simply can't use them. The users don't generally want to go and search instructions from the on-line help – it just takes too much time (6.3.1). They also have difficulties in finding the target of usage of a particular command. They might find a command, but give up using it since it is too difficult. And since they don't want to use any of their valuable time for looking for instructions, they rather leave the command unused and continue using the roundabout they have found, even if it was inefficient. Also, the Service desk is often asked questions that have direct answers in the on-line help – the users just don't find them and/or don't have the energy to start searching for the correct place from the help.

In the web questionnaire the users were asked for reasons why taking new features into use in the software might feel difficult, and the most significant reason according to the answers was the difficulty to find where different commands should be used. A clear need for getting information easily on the command at hand was identified (6.4.4).

"The tooltip doesn't tell enough about the functionality of the command, I have to try it myself" (U3)

Test and post-test interview results

In the tests 5/6 users found the possibility get information about a specific command with the right mouse button. When pressing the "More…" –button, 5/6 users would like to get something else than just more text, three of them especially mentioning the on-line help pages. They would very much like to see animations or at least an illustration of the command's use.

The users would like to have the instruction clearly visible, so when we asked for the colour and shape of the balloon the light yellow colour with rounded shapes was clearly the most popular alternative. See chapter 6.9.1 for details on the interview answers.

E. Introduction to new features in commands or unused commands

Summary

Commands that have new features in them get an information bubble when being used for the first time. Commands that the user has never used also get an information bubble when being used for the first time.

Description of the solution

When the user selects a command from the toolbar that has new features in it, an information bubble is presented. The bubble contains a text that says that there are new features added in the command and that the user can get information about them by clicking the bubble. The bubble contains a closing "X", but the bubble closes also just by clicking outside it. If the user clicks the bubble, a similar kind of information bubble opens as with the right-click (see the solution D). The bubble also says that the user can later see the information by using the right-click on the icon, or the menu command.

If the user selects a command that he or she never has used before, a similar kind of bubble is presented, the only difference being in the contents in the beginning which in this case contains information that the user hasn't used the command before. See Figure 10 for illustration of the suggested layout of the bubble.

The bubbles are showed only once.

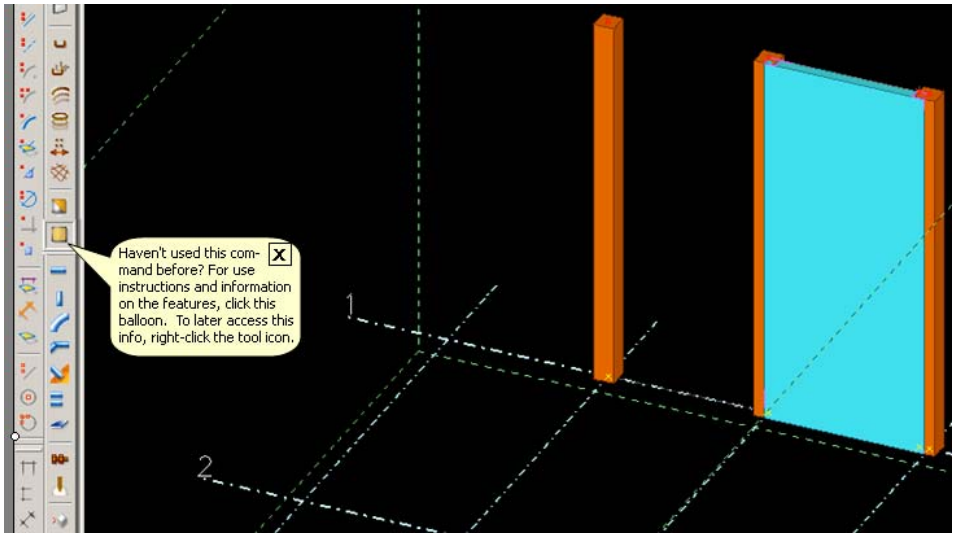


Figure 10: The new feature introductory bubble

The user can also see all the new commands and commands with new features by choosing the "Show new commands and features" –option in the assistance icon menu (see the solution V). After selecting the option, all the commands in the toolbars that are new and/or have new features in them, are pointed with a bubble. The bubble very briefly describes the new functionalities.

The difference between the secondary mouse button -click information (see solution D) and this solution is that the right-click information gives the user a chance to view instructions on using the command at any time. This solution is to promote new features in existing commands and to ease the step needed to be taken to start using a novel feature.

Background and reasons for the solution

User experience findings

There seems to be a need for this kind of solution, since it was identified that the users don't always notice the new features in existing commands after version update. It became also clear from the data collected that the users don't want to start using new commands because they find it difficult to learn using them. The difficulty is present because of two subjects: firstly the commands are often complicated by their nature and user interface, and secondly, because it is too difficult to find information on using them. (5.2.1, 5.3.1) If the user gets information offered to him or her when starting to use a new functionality, it is a great relief to get information offered on the usage of the functionality. Moreover, users often are reluctant to try new functionalities even if they know about their presence and their possible benefits because they don't want to waste their time in trying to figure out how it is supposed to be used. Or how they could find information on how it should be used.

A similar kind of solution was also used in some other complex applications to introduce new functionalities. According to the web questionnaire, 4/9 users just glance through the Release notes. This leaves a good possibility for not seeing the important new features and thus leaving them intact just because of ignorance. It is also notified in the service desk that often answers to the questions asked from there can be found directly from the release notes.

Test and post-test interview results

The users in the test seemed to like the information bubble that appears.

Comments from the users

"In other programs I usually skip the presentations of the new features, but Tekla Structures might well be a difference" (U6)

"I cannot miss commands that I know nothing about" (U3)

"If you haven't been waiting for a new command, you don't start using it because it is not guided how to use it" (U1)

"It is a lot easier to turn the assistance on and get help on new commands than to open help and search from there" (Ut2)

F. The instruction panel

Summary

An instruction panel in the upper right corner shows short instructions for the current phase of the command.

Description of the solution

The user gets information from two places while using a command. In the place of the current status bar there is the history panel, and in the upper right corner of the screen there is the instruction panel. The history panel shows what the user has done, and the instruction panel shows what he or she should do next.

The upper right corner instruction panel contains only information concerning the current phase of the active command. It contains no information on the purpose of the command, just what the user has to do next in order to complete the command. The instruction panel is used also to provide information about certain common errors. When user tries to select an object by clicking it, or by making a selection with the selection square and nothing gets selected because of a select filter, the instruction panel shows a message telling about the about the filter being on. The instruction panel is illustrated in the upper right corner of the Figure 11.

The panel also contains a "More…" button which enables the user to open the on-line help pages of the currently active command.

The panel is by default visible only when there are instruction messages to show. When an instruction message appears, the panel quickly fades in, and when the instructions end, the panel makes a quick fade-out after a while. If the user places a cursor on the panel, it doesn't disappear until the cursor has moved away. The numbers that are used should be presented as digits, not written.

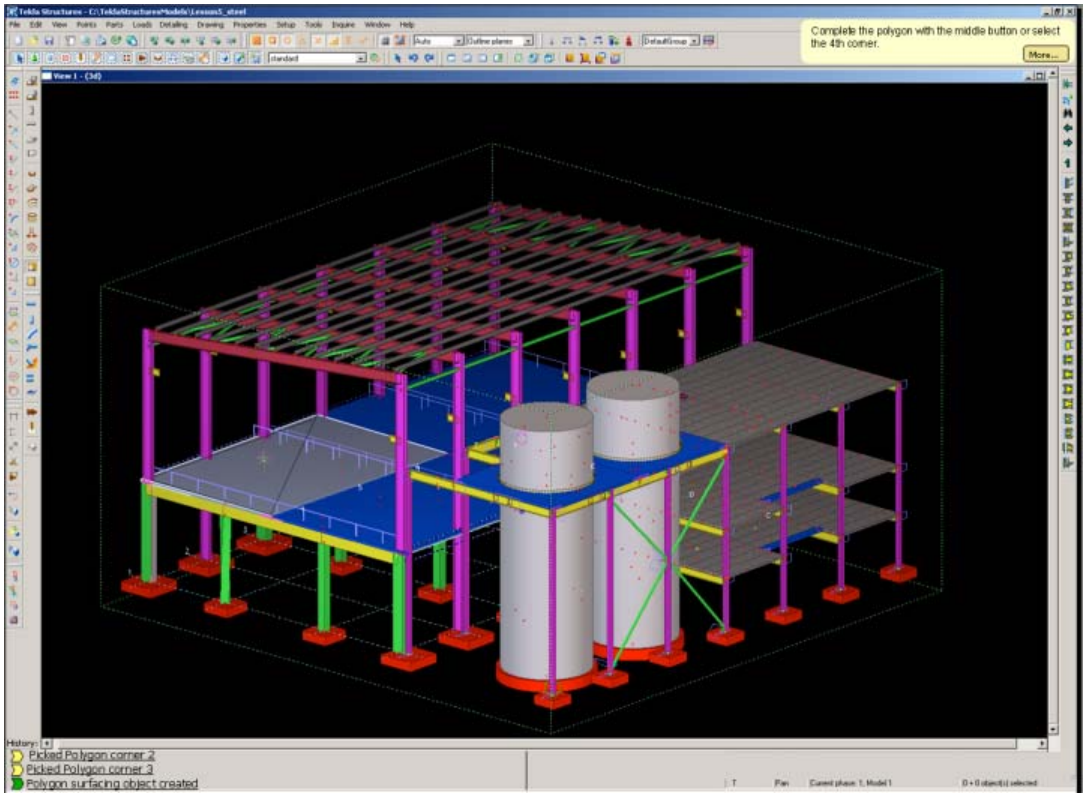


Figure 11: The instruction panel in the upper right corner of the screen and the history panel in the lower left corner of the screen.

Background and reasons for the solution

User experience findings

It was identified that users tend to use the software with the method of trial and error (8.2). It seems that they like this approach even if it remains a question whether they have learned it because there has not been much assistance available or because it just is so good way to do things. Anyway, it is important that the software can support the users in this way of using and learning. Complex commands sometimes require several steps that the user can't remember even if he had just read instructions concerning the command. Even if users of this kind of software are experts and many of them use the software daily, the memory load of the users is quite large since they have to remember how different commands work. The instruction panel should relieve this memory load.

It was noted that the users don't always notice the texts in the current status bar. Some of them pointed out that the status bar is so small that you don't actually see if there is anything new or not if you don't especially look in there. They wanted something that the software would somehow attract their attention to the status bar instructions. Almost all users were not satisfied with the contents of the current status bar and would have liked to see more information on using commands. Therefore, a larger panel to show instructions is needed. The positioning of the panel to the upper right corner of the screen is based on the fact that most users preferred being offered instructions in a visible manner, if it still is un-obtrusive and can be skipped without actions if needed. The fading of the panel also enhances the user's ability to see the new instructions appearing in the panel.

It was also noted that users like to try things out themselves (8.2, 9.2) and providing instructions during the use of a command is a good method to make this way of using more efficient.

Test and post-test interview results

In the tests 4/6 users noticed the upper right corner instruction panel right away when starting to use a command. One of the two who didn't notice it made the task twice because of a prototype error and on the second time he noticed the instruction panel.

User comments on the solution

"The information value of the status bar is lousy" (U5)

"I rather try than read long instructions" (U4)

"Yes this is better than if it [the instruction] was there in the left lower part of the screen, status bar" (Ut3)

Priority

This solution should be implemented in the first phase.

G. The history panel

Summary

The history panel is situated in the lower left corner of the screen (see Figure 11). It shows all the actions the user has taken. It also enables him or her to undo actions by clicking the lines. The panel is resizable but it cannot be turned off completely.

Description of the solution

The history panel is meant for providing the user with information on what he or she has done, and giving a possibility to step-by-step undo the actions the user has done earlier. Every action the user takes is logged to the history panel as an individual line. All actions that the current undo button can undo are logged into the history panel. However, the actions are logged in more detailed level: every single pick when using a tool is logged as an individual line. In addition, all errors are also logged here. The logged information is detailed: it contains for example the material or profile of the object that was created, the number of copied objects, the direction the copying was done, result of a measurement, number of objects that were used with an operation, for example a clash check, etc. This means that lines also include information on results of the actions, such as how many objects were just welded or deleted. The numbers that are used should be presented as digits, not written. The word order of the history lines is: object, verb. For example: "Corner 1 picked". The font face of the history lines is the same as is used in the menus. The clickable area's height is a little less than the area's height is in the menu titles (about 17 pixels), resulting that the total height of the three lines is 50 pixels.

The logging is growing so that a new line is always added to the bottom of the panel. An illustration of the panel is shown in the Figure 12.

The user can use the lines to undo the actions he or she has taken. This is done by just clicking the appropriate line with the primary mouse button. When clicking, the clicked line turns grey, and the corresponding action is undone. All action lines downwards from the clicked line are also undone and greyed. For example if the user clicks the middle line (Picked Polygon corner 2) in Figure 12, Tekla Structures would turn the line itself, and all

the lines downwards from that middle line grey. More importantly, it would undo the greyed actions, i.e. return back in time to the point where the user had picked the polygon corner 2 and was ready to pick the corner 3. If the user changes his or her mind and would like to just undo lowest line, i.e. the action where the middle button was pressed to finalize the surface creation, for example to add one corner more to the polygon, it would be possible to click the now greyed middle line again. That would redo the clicked action and turn it to normal colour again. If the user after undoing takes any action that is logged to the history lines, all the greyed lines are wiped and the new action is added in place of them.

The lines have colour coding on the left-hand side of them. The colours are as follows:

- Errors have red colour (code R: 241 G: 30 B: 0, HTML: #f11e00)
- Completed commands have green colour (code R: 31 G: 174 B: 0, HTML: #1fae00)
- Others actions (picking etc) have light yellow colour (code R: 255 G: 255 B: 102, HTML: #ffff66)

In addition to the colours, there are small differences between the colour shapes: the red ones have a small exclamation mark (!) in them, the yellow ones are empty and the green ones have the American v-symbol of correctness.

The colours should have sufficient contrast with each other so that they can be easily recognized with just a glance. Always when a red error line is displayed, it is accompanied with a bell sound.

The history lines may be saved, but this is not the default setting. The setting can be changed from the assistance customize dialog.

Figure 12: The history panel

Actions that can be applied several times with same values to different selected objects, such as copying and moving, get a redo –icon (see Figure 13) next to them. The user can click the redo –icon and in that case the corresponding action is re-executed with the same values and applied to currently selected object(s).

Figure 13: The redo -arrow that appears next to re-executable commands

Actions that have related information available get a question mark (?) -help link next to them, to the same column as the redo arrow. These actions include all the selecting and unselecting of tools and errors that have some information available. When clicking this link, the corresponding one-line help page is opened. It is important that all the links take

the user to the correct place in the help, not to any general help page. If there is not command specific help available, the help link is not provided.

The user can adjust the height of the panel, but the default size is three lines. When there are more than three lines to show, a scroll bar is added to the right-hand side of the lines. (See Figure 12)

The user can configure the help panel to be a separate window, but the default setting is that it is included in the status bar area.

The history panel contents can be saved with the model. The size of the panel can be adjusted by dragging from the top of the panel.

Background and reasons for the solution

User experience findings

From the collected user data it was identified that the users have a clear need for seeing what they have done. This applies both to actions they have just done, and to actions they have done some time ago. (2.1, 2.1.2, 2.2.1) The user's tendency to use the program by trying and seeing what happens is also supported with the possibility to see what was just done and the possibility to undo actions easily step-by-step. It was identified that especially long series of picks cause problems to users since one single faulty pick forces the user to start over from the beginning. The logging that includes all the picks too is a relief to this problem. The users currently seem to be using the undo button to check what they have done and especially, check whether the operation they just tried to do did anything. Often it happens than when using the undo functionality like this, something that was not meant to be undone, is accidentally undone. The error may not be noticed immediately, but only later when it causes problems.

In other software similar types of history windows are implemented which seem to be of good value to the user.

The possibility to re-execute commands is meant to fulfil the need to make repetitive steps more rapidly. It would enhance the efficiency of the software's use since the user can use same values over and over again. This was identified to be a way of working among the interviewed users.

The users need to continuously see what they are doing and what the results of their actions are (2.3.1), so a separate history window would not be as good as this solution. This is because users don't normally want to have any extra windows blocking their view to the model (10.4). Also because of this all the details of the action taken must be visible here, for example the number of parts welded together or the number of parts deleted or copied. This way the user can see that the action that was taken applied to the objects that it was meant and not any extra objects. This was identified as a problem with the users: they don't know what they just did. (2.3)

The colour coding is meant to make it easier and faster identify different actions from the list while scrolling it. Since the human eye can detect much quicker shapes and colours than text, it is faster to be able to follow the colour coding instead of having to read all the lines when looking for a specific action. The colour coding also makes it easier to see changes in the history panel without actually having to look at it. The colours used derive

from the colours that are currently being used to indicate the successfulness of the connections.

The shapes inside the colour coding next to the lines are provided so that the colour-blind users can also easily distinguish the colours from each other.

The bell sound that comes along an error is meant to attract the user's attention to the error. It was identified that the users want to get their attention attracted to errors (9.1.2). It is not always good to have an error message that must be closed with a mouse click since sometimes there are errors coming a lot and the users don't want to be closing the error messages all the time.

The links to on-line help pages are offered, when appropriate, since it was identified that the users often have problems with tools or commands that have answers in the on-line help. Users just don't have the energy to go and search for it, and there's currently no way to quickly access the on-line help about a specific command.

The possibility to save the history panel contents with the model is a requirement based on the users' way of learning by plagiarizing. They like to sometimes go and see other models, how some things are done there, and use them as an example. If the command history is saved in the history panel, it is easier to see what has happened in the model. It also happens that users have to jump in the middle of a project into a new model, and in that case a possibility to see there has been happening and what tools have been used.

Since some of the users are concerned about the work space they have (10.4) the size of the history panel can be modified.

Test and post-test interview results

The users were not able to use the history panel as well as it was thought. The problem was mainly in the fact that they could not come to think of a possibility to use the history panel for undoing actions. All except one realized that the panel shows actions that have been taken, and after figuring out that it is actually possible to click the panel, all except one got right away the clue of what the lines are for and what happens when clicking them. The appearance of the prototype was changed a little to the last test: the lines where changed to underlined font type to get a similarity to the hyperlink appearance frequently used in web applications. There was also a grid around the lines which was removed when the underlining was added. The changes seemed to be to the positive direction since the last tested user was able to click the lines faster than the others.

Comments from the users

"There are several places where you think you haven't done anything, but still you actually have done something" (U6)

"This history is something that you really need and have been longing for in the software" (Ut6)

"I wasn’t able to think of this… but this would feel really good" (Ut2)

"I’m not sure that to which command [or pick] you have returned with the lines" (Ut5)

Possible future implementations

When the user undoes a pick, the last pick that is left is highlighted for a moment. This could happen with mouse-over tool

H. The expanded status bar

Summary

An expanded status bar is available, if the user turns off the instruction panel (solution F). The expanded status bar is located on the right-hand side of the history panel. See also the solution U.

Description of the solution

The expanded status bar shows shortened versions of the information that the instruction panel would show. If the same instruction as in the instruction panel does not fit the status bar area, a "…" button is offered to the user at the end of the text. The "…" -button expands the status bar upwards so that the whole text becomes visible. The expanded status bar information is presented in the same space as the solution I. The possible quick inquiry texts are replaced by the expanded status bar instruction content. Most of the time there are no quick inquiry data and instruction data to be shown simultaneously, but in case there is, the instructional expanded status bar data overrides the quick inquiry data.

Background and reasons for the solution

There are users who are very concerned about the workspace they have, and for them it is important to have a possibility to suppress the information panel.

I. Quick inquiry information

Summary

The quick inquiry information is information about the selected object. This information is provided in the right-hand side of the status bar.

Description of the solution

Quick inquiry is constantly visible in the status bar's right-hand side. It provides the following information: Top and bottom level, profile, part number, the phase of the part, name of the part, length and the material. When there are no parts selected the space reserved for quick inquiry information may be used by the solution H.

Background and reasons for the solution

User experience findings

Users of this complex software were noted to appreciate more information about the objects on the screen. Users often need to know something about an object, but aren't willing to always use the object inquiry function.

It seems that getting automatically this kind of information would have a positive effect on the efficiency of the users' usage of the software.

Test and post-test interview results

The users were asked to mention things that they frequently need to know about objects in the model. For details, see the Figure 7 in chapter 6.9.1. The most often mentioned facts were selected to this solution.

J. What's this –links in dialogs

Summary

The various dialogs of the program provide the user with a possibility to get information about each part of the dialog by clicking the what's this link. By clicking the right mouse button on an item on the dialog, the user gets information about that specific item. There's also a help link which is a gateway to the on-line help page about the dialog.

Description of the solution

The dialogs have a what's this link visible close to the upper right corner, as illustrated in Figure 14. The link brings visible a set of balloons that describe the meanings of different groups of editable fields in the dialog. The balloons are not descriptions of individual fields, but rather field groups. The bubbles disappear when user clicks anything on the dialog. There is one balloon in the upper right corner of the dialog which tells briefly where this connection can be used, mentions the possibility to right-click the individual fields, and finally offers a more –button that is a gateway to the related on-line help page. The related on-line help page should contain a 3D picture a possible use of the connections. The length of the balloons must be kept short since the longer explanations can be given in the on-line help page with a picture.

Background and reasons for the solution

The user can also use the right mouse click to get information about individual fields in a same kind of balloon as is offered from the tool icons.

The user can also click the help –link which opens the on-line help windows with the correct page open.

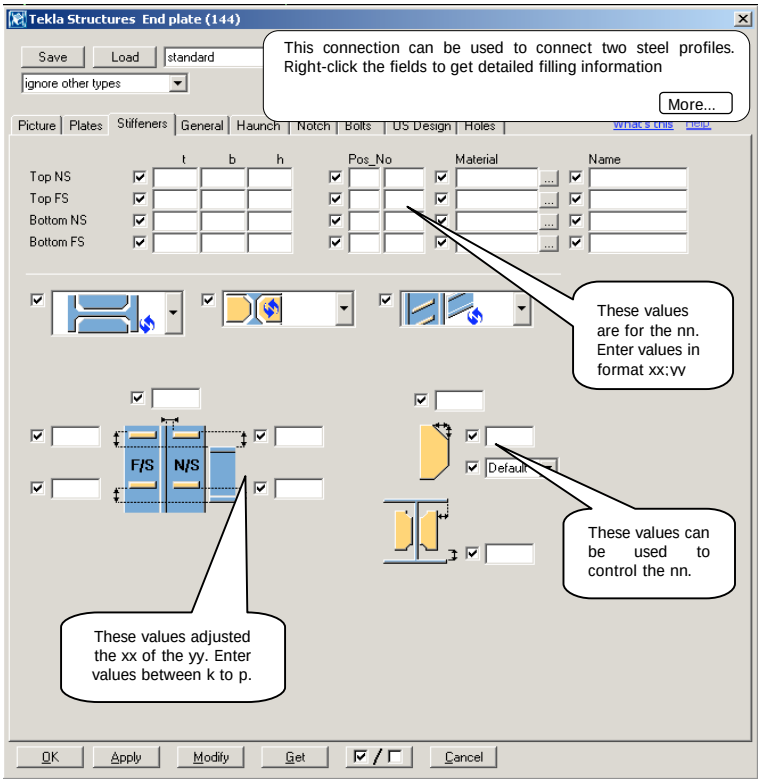


Figure 14: The positioning and layout of the what's this –link, the help -link, and the information bubbles that appear after pressing the link

Background and reasons for the solution

User experience findings

The idea of the what's this –links has been used widely in a variety of other software. Many spreadsheet and word processing software utilize it, and it has been recognized to be a good method of providing the user with information about things on the dialogs.

The use of balloons describing areas of dialogs was used in one other complex software and it seemed to get support from the users of Tekla Structures too. It was recognized that the users have difficulties in deciding what to enter to different fields in the dialogs, especially component dialogs. The most important information they would like to see is what kind of values it is possible to enter to different fields. The users like to try themselves out and see what the result is, but if they don't know the format in which the values have to be entered, they cannot start trying.

The balloons that are not modal (don't require the user to close them in order to continue work) are good in the fact that they don't break the user's task at hand. They just provide information that can be omitted by clicking just somewhere.

Test and the post-test interview results

In the tests the users were satisfied with the possibility to get quickly more information about the connection dialogs. A picture that would clarify the using of the connection and a possibility to get more help when needed were often mentioned.

K. Introductory dialog when running the software for the first time

Summary

The software displays an introductory dialog when running the software for the first time after installation or version update. The dialog offers the user a possibility to go through tutorials introducing different parts of the software (see the solution Y) and to see an introduction of the new features in the new version.

Description

The dialog offers the user a list of tutorials with short descriptions of their contents. The new features in the current version are offered to be viewed through the on-line help window.

The introductory dialog has a "Don't show this dialog any more" checkbox, which is checked by default.

Background and reasons for the solution

The users want to get information about their possibilities (3.1.3). In several other software these kinds of introductory screens are used.

L. Information about the new features presented while opening the software

Summary

The advertising animation that is presented to the user while loading the software is replaced automatically by an animation or a brief textual representation of the new features in the new version of the software.

Description of the solution

The introductory advertising animation that shows while loading the software is replaced by an informative presentation about new features in the software. The introductory advertising animation is replaced by the feature presentation after it has run two times. The user can still choose not to view this presentation by ticking the checkbox in the animation. After installing a new version, the presentation introducing the new features is shown even if the user has selected in the earlier version not to show it. The presentation shows new features one or two at a time, and shows a different feature each time.

If there are no new features to show, the presentation starts to show information about features that the user hasn't used in the software.

Background and reasons for the solution

User experience findings

When the software is making long calculations, drawing something etc., some of the users seem to be willing to use the waiting time into something advantageous. Some of them don't see this as a benefit, some on the other hand would very much like it, and the usage of the waiting times seems to split user's opinions.

The users are often in a hurry and don't want to waste their time into reading or trying to find more efficient ways to work (5.2.1). Thus a method to provide them information about

new features without wasting any of their time is needed. Exploitation of the waiting times is a method that should be an answer to this problem.

M. Information about new features and unused features during the waiting times

Summary

While the software is processing a longer operation (longer than approximately 7 seconds), it shows information about either new features in the program, or information about features or commands that the user hasn't used before.

Description of the solution

The software starts to show brief textual information in a new window about new features, or features that the user hasn't used before. The user may select from the assistance customization which ones he or she would like to see, but the default setting is that new features are shown.

When the operation finishes, the help window becomes inactive and the focus changes to the software itself. This way the user doesn't have to take his or her focus away from their current other task.

Background and reasons for the solution

The same reasons apply as mentioned in the solution L.

N. Suggestions of more efficient ways of working

Summary

The software follows the user's actions and tries to identify tasks that the user is doing. When a task is identified and if the user's way of doing it is different from the optimal way to do it, the software gives the user a hint.

Description of the solution

The software is equipped with a module that is different from the other software. This module is following the users actions, that is, the clicks, key-presses, typing etc. In the module there are saved characteristics of different kinds of tasks that the users typically do with the software. When the module recognizes a pattern in the user's way of using the software that matches to some task, but differs from it in a way or another, it provides the user with an informative hint on how to perform the task more efficiently. The hint is provided through the assistance icon (see the solution V), which flashes briefly and leaves a small bulb icon to the top right corner of the icon. The hint consists of textual information supplemented with a picture and/or animation. The hint is presented in a bubble, which opens when the user clicks the assistance icon and selects the option named "A hint from the assistant", which is highlighted while the bulb icon is visible, or clicks the bulb icon.

The module also identifies other places where the user might need assistance. These include the situation when the user tries to perform some task without succeeding in it, i.e. makes the same procedure all over again and again without actually creating anything. In such a situation the information provided contains information about the most common mistakes that are normally made with the selected tool or functionality. The number of

messages is limited to the maximum number of 4 messages / day. The same message is not shown more than once within one work day (8 hours).

Background and reasons for the solution

User experience findings

It was identified that all users are sometimes wondering whether the tools they use are the optimal and/or whether there would be better ones available. They may know that there are features that could be utilized, but don't have time or courage to start trying to use them (5.2.1) or even more often don't even know about features that would be efficient in their tasks (10.3.1, 5,4,1). Especially the novice users seemed to long for a possibility to get information about common mistakes when they are performing some task all over again without success.

It is important to offer this kind of information to the users so that it doesn't necessarily obstruct their current task. The short flash of the assistance icon can be easily omitted, but was still noticed by 5/6 users noticed it in the tests. Thus those who are willing to check the information offered can do it, and those who don't want to break their task, can continue without any disturbance.

This kind of assistance was tested in an experiment made by Bhavnani et al. (1996) and it was seen as a functional way to improve users' efficiency of use. In the brainstorming session it was also considered to be a good method to teach the users the right ways to do things and to enhance their efficiency of use

Test and post-test interview results

When they were asked their opinion about information that is offered to them in a situation when there might be a more efficient way of performing a task, they all had a positive attitude. This is clearly because, according to the data collected in the user observations and interviews, users have problems in finding tools and places where to use them.

Comments from the users

"I cannot miss the features that I don't know anything about" (U3)

"I would like to have hints about things that could be done otherwise too" (U3)

O. Recognition of erroneous usage

Summary

The user's actions are tracked with the same module as described in the solution N and when an error is detected, the user gets a hint.

Description of the solution

When repetitive steps are taken on same object with no any progress, a list of the most common mistakes with the current tool is provided via the user assistance icon (see the solution V). The hint consists of the most common errors made with the tool the user is using and instructions on how to use the tool. The most common errors are given first in the instruction. The instruction is provided via the assistance icon which flashes and gets the bulb icon to indicate the possibility to click it. When clicking the icon, an instruction bubble containing information about the most common errors with the selected tool is given, and when pressing the more button in the bubble, a help window about the tool is

opened which explains the errors in more detail, accompanied with the step-by-step use instructions.

Background and reasons for the solution

It was identified that the users tend to do errors that could be recognized by the software by following the repetitiveness and result of the actions the user is doing. These kinds of errors and problems seem to be common especially among novice users. In these situations it is beneficial to the user get information on the typical errors since most probably the error is due to some of them. The software can suggest the user this information automatically according to Bhavnani et al. (1996).

P. Top 20 component list

Summary

A list of the most popular components in the model is composed automatically. The list is visible in the Component Catalogue.

Description of the solution

The user is provided with a possibility to view what are the most popular components used in the model. This is done through the Component Catalogue which contains in addition to the existing categories there two new categories called "Top 20 components in this model" and "Top 20 components with this licence". The first one shows the most popular components, details, custom components etc. sorted by their popularity and showing 20 first. The list is automatically updated. The second one shows a similar kind of list with the difference that it calculates the popularity between the components within all the models that have been built with the current licence. The list is illustrated in the Figure 15

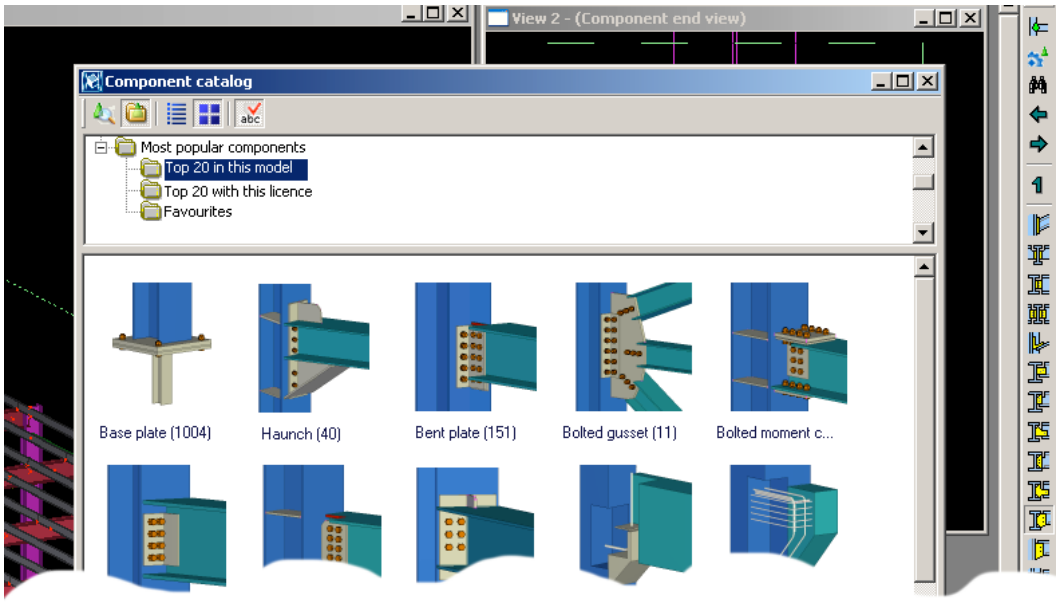


Figure 15: The top 20 components list

Background and reasons for the solution

User experience

The users like to plagiarize and learn from what the others have done (2.2.2, 8.5.1), and being able to quickly see what kind of components the other have used would be good information.

Test and post-test interview results

In the interview the users were asked whether they would find this kind of feature usable, and their opinions varied between full support and dubious attitude. The number of items to be shown was changed from 10 to 20 according to the feedback from the users.

Comments from the users

"When I come to a new model, I don't know what (objects) there have been used" (U5)

Q. Reminders about consequences of certain actions, error messages

Summary

A reminder to the user about consequences of his or her actions in the software is presented. Error messages remind the user of what he or she needs to do. The error messages contain a link to the corresponding on-line help page about the error.

Description

The reminders are made from the following actions: modifying connections so that they become "red", deleting parts that have drawings of them, and modifying numbering. When these actions are made, the user gets a message which tells him or her, what other places the action taken affects and offers a possibility to directly go to the place affected in addition to the close button. Also an undo button is offered, which undoes the action that caused the reminder to appear. See Figure 16 for a sample of a reminder.

The error messages always tell the user what he has to do in order to get over the error, and what the program does for the user. There is always a link button that opens the corresponding page in the on-line help about the error, reasons for it and how to recover from it.

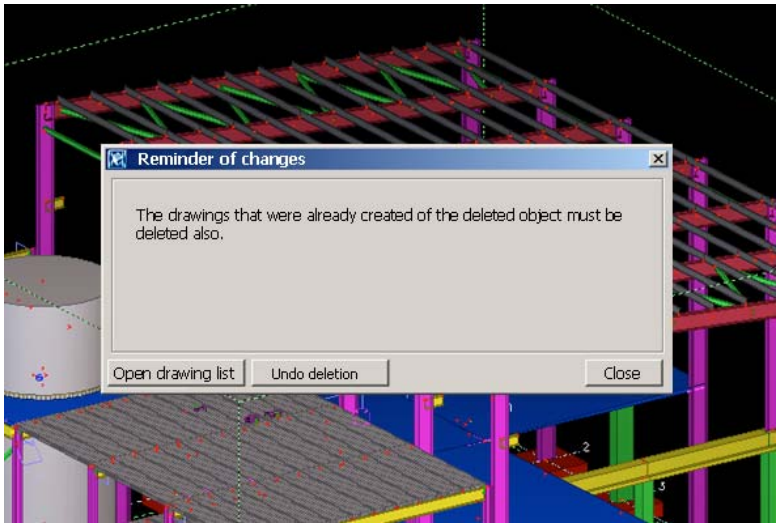


Figure 16: A sample of a reminder of changes

Background and reasons for the solution

User experience findings

From the information gathered from the users it was noted that users sometimes make changes to the model that they either didn't mean to make or don't understand of remember the consequences of. For example, the user would change the properties of parts connected by several connections and not notice that all the related connections become "red" after the modification.

Users also want, that they don't have to remember everything themselves (10.6). They like it if the software would remind them about things that aren't self evident from what they are doing. Especially changes that happen outside the view they are currently using are difficult.

Error messages also tend to cause problems to the users. The user interface should assist the user to recover from the error and tell the user exactly what must be done, and what the program does itself (9.1.2). For example with a numbering collision, the user may not get a clear picture whether he or she has to go and correct the collision by him- or her self, or whether the software makes the corrections itself. The ability to diagnose, prevent and recover from errors is an important skill in mastering computer software, and providing good error information in the right time is efficient in tackling this (van der Meij, 1992).

Test and post-test interview results

In the tests the users seemed to like getting a reminder to their face. It was pointed out that it is important to be easily able to switch on and off this kind of assistance feature according to the usage situation: when the model is in the finalizing phase, these kinds of reminders are very welcome, but in the drafting phase they aren't wanted.

R. A search function in the user interface

Summary

A possibility to locate a command from the user interface. The functionality is provided from the assistant icon. (see the solution V) User can type in a command or browse from

an alphabetical list of commands to locate the placement of a certain command in the user interface.

Description of the solution

The user can locate a command from the user interface with this functionality. The commands are found by typing in a name of a command or a partial name into the dialog (see Figure 17). It is important that also partial command names make a match since users don't remember the complete names of the commands. If several hits are found, the user is presented with a list of possible matches. From the list of matches the user can either select a suitable command, or go back to search with another keyword. When the user finds a right command name, the command is shown by opening the corresponding menu and highlighting the command, or highlighting it with a circle if it is found from the toolbars.

The user can also select the command to be located from an alphabetical list of commands. The alphabetical list of commands is illustrated in Figure 18. The list offers the user a possibility to return to the first dialog.

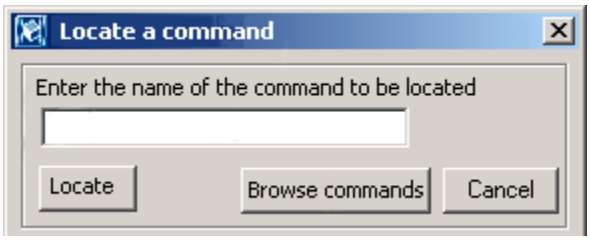


Figure 17: The search functionality's dialog

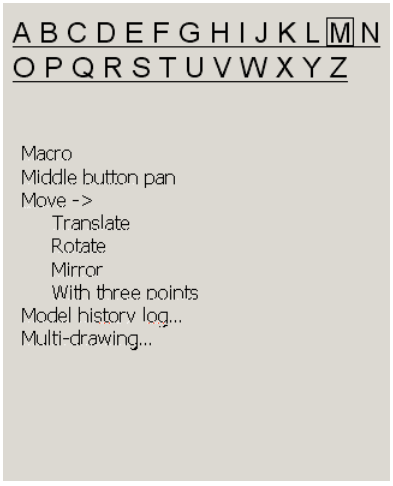


Figure 18: A sample of the style in which the alphabetical list of commands should be presented

Background and reasons for the solution

User experience findings

Especially in the drawings the users have to "fish for" the right place to make the desired modification to the drawing. They know what they want, and they can use a keyword search to find the correct place to make the adjustment. The users seem to remember parts of the command names rather than the complete command names and thus the search must be able to locate also partial name matches.

There seem to be situations when the user remembers a command but doesn't recall where it is located in the user interface (7.1.1). Currently the users have to browse the menus and toolbars in order to find it, and it takes lot of time. It is good to have the locate function easily available for the user since searching from the on-line help is such a big process that users don't seem to be willing to do it.

Comments from the users

"It is an idea of some greater wisdom, and it would be good" (Ut5)

S. Clear showing of the progress of long operations

Summary

The software shows clear signals of processing while performing an operation that takes a long time.

Description of the solution

A progress bar is used always when a command takes time more than 10 seconds (Nielsen, 1993). The progress bar should be moving as continuously as possible. The progress bar is illustrated in the Figure 19.

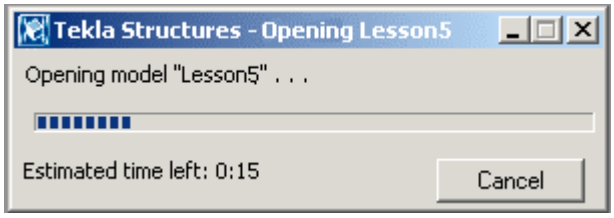


Figure 19: The progress bar

Background and reasons for the solution

User experience findings

It was identified that the users seem to be dubious whether the software hangs always when it performs a long operation. They tend to use the task manager to check whether the process of the software is still responding or not. This is a clear signal of the fact that they don't get enough feedback from the software about the progress of the operation initiated. They also would like to get an approximation of the time it takes to complete an operation. Nielsen (1993) has noticed that with longer delays than 10 seconds the user loses his or her attention to the operations and wants to do other things while waiting. A graphical progress bar is better than just numbers since it provides the user something to look at during the painfully long waiting time, Nielsen points out.

T. Wizards on rarely used functionalities

Summary

Wizards are used to give guidance on complex functions that are rarely used.

Description of the solution

Wizards that have already been implemented in the software should be used more widely. The wizards consist of small parts tied together so that the user can leave the wizard in many places and make the desired adjustments without the wizard too.

Background and reasons for the solution

The wizards are being used extensively in other software to guide the user through difficult operations or ones that require a lot memory load to the user. Since the user interface of the Tekla Structures is highly system-oriented because of the nature of the tasks done with it, there is very little task orientation. This is mainly good in the view of the modeling task complicatedness, but some single operations that the user has to take in the software would seem to be better performed with a more task centric approach. The wizards bring this to the places where it is needed.

U. The right-hand side of the status bar

Summary

The right-hand side of the status bar contains information:

The activated command, information about the selected object, and information that is constantly visible regardless of the choices.

Description of the solution

The bottom of the screen is divided to the history panel and the status bar so that the history panel occupies 55% of the horizontal space and the right side of the status bar 45%.

The activated command is listed with the same yellow background colour as the instruction panel. When no tool or command is activated, the yellow background turns into the normal background grey.

The following information is shown about the selected object: top and bottom levels, the profile, the part number, phase of the object, the name of the object, the length and the material. The information is displayed so that the place where each data is displayed stays always the same regardless of the length of the data fields displayed.

Information that is continuously visible is the number of logged users in multi-user mode, the coordinates of the picked point, the axis locks and the orthogonal selection.

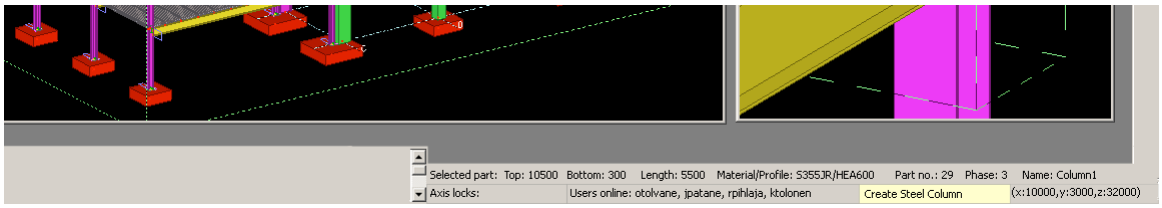


Figure 20: The right-hand side of the status bar

Background and reasons for the solution

It is noted that the users have often problems with activating and deactivating the commands or tools. (SI2) They can be accidentally turned off, and user doesn't notice that it has happened. The background colour of the activated command field adds visibility to this.

The users want to be able to easily, with only a glance, view information about their objects and thus the object inquiry data is always visible about a selected object. The same information is always on the same location of the status bar so that it is quick to move one's gaze to the correct position to see for example the material of the selected object.

V. The user assistant icon

Summary

The user assistance icon is located on the lower right corner of the screen and it provides the users a way to access several assistance functions. When clicking the icon, a menu opens.

Description of the solution

The user assistance icon menu contains the "Show new commands and features" option. There are also options to control the assistance level (see the solution A). The Locate a command function

The icon has a tooltip which appears as the user moves the cursor on the icon. It describes the functionality of the icon with the following phrase: "Access the Assistance menu to control and use Assistance functions and Views hints when flashing."

When there are hints to be shown to the user, i.e. after flashing and when the bulb is visible (see the solutions N and O), there is a highlighted option first in the menu "A hint from the assistant" which opens a bubble containing the hint.

The instruction scroll back window is also accessed via the assistance menu (see the solution W). The icon is illustrated in the Figure 21 and the menu that opens from the icon in the Figure 22.

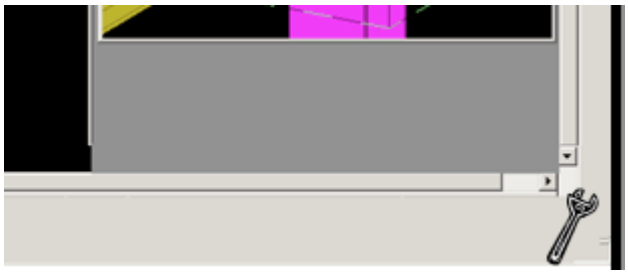


Figure 21: The user assistance icon in the lower right corner of the screen

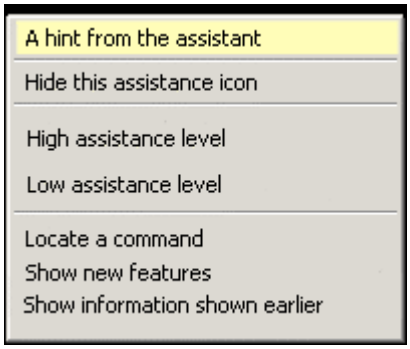


Figure 22: The menu that opens by clicking the assistance icon. The up most option is only visible when the assistance icon has flashed to indicate a hint.

W. Instructions scroll back window

Summary

A window that can be opened from the Help –menu of the Assistance icon –menu that shows all the instruction messages and hints that have appeared during the current session.

Description of the solution

The user can open a separate window from the Help –menu, or the Assistance –menu which shows all the informational messages that have appeared during the current session. These messages include all the Instruction panel instructions, all the balloon contents and all the hints that have been given via the Assistance –icon. The window is opened from the Assistance icon menu.

Background and reasons for the solution

It was identified that the users are sometimes in a situation when they would like to read a message they receive from the program, but don't have time for it at the moment. Thus it is beneficial for the user to have a possibility to easily later return to the instruction. (2.2.3). This also enhances the user's possibility to control the instructions given to him or her – the possibility to return at a later time to the instruction given gives freedom to choose the appropriate time when to view to instruction.

X. The axis lock selections and the select filters

Summary

The axis lock symbols X, Y and Z that are located on the right-hand side of the status bar, get a bubble when some of them are activated for the first three times. The cursor gets an axis-lock symbol while the lock is active. The cursor gets a symbol when there are selection filters on

Description of the solution

When the user presses X, Y or Z to lock one of the coordinate axes, the right side of the status bar indicates the selection in the Axis locks –section with a corresponding letter. When the user activates the command for the first three times, he or she is reminded about the symbol in the status bar by presenting a bubble showing the letter. After three times of viewing, the bubble is not viewed any more.

In addition to the letter and the bubble, the user gets an indication of the lock in the form of an axis lock symbol in the cursor. The lock symbol and its appearance place are illustrated in the Figure 23.

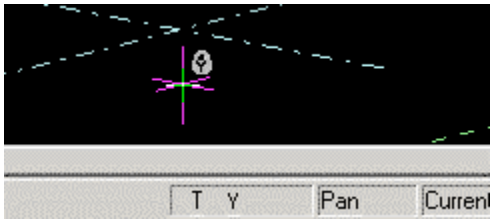


Figure 23: The axis lock symbol in the cursor. In this case the user has activated the Y -axis lock.

A lock symbol indicating the presence of selection filters is also used in the cursor. It is visible when the user has any other selection filter setting on than the default setting of the filters is (i.e., only the "Select single bolts" is unselected). The symbol is illustrated in the Figure 24.

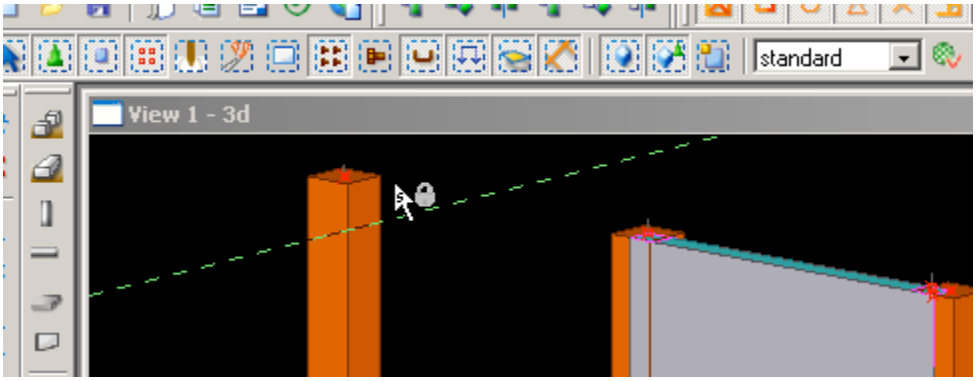


Figure 24: The symbol indicating active selection filters in the cursor

Background and reasons for the solution

User experience findings

The users were shown a picture of a short instruction applied to the cursor in the interview, and it was considered to be informative and non-obtrusive. The users claim that sometimes they don't see all the things happening on the screen and thus the axis locks may be accidentally left on (4.1.1). The symbol on the cursor should remind the user and on the other hand let him or her know that the function actually is active without having to look to the bottom of the screen.

Y. Tutorials on various subjects

Summary

The user is offered a possibility to view tutorials that consist of introductory lessons to the basic tasks of the user interface. In addition to this, some of the most difficult complex tasks are covered

Description of the solution

The tutorials about basic tasks in the software introduce in small parts how to make each part of the modeling process from starting the project to the printing of ready-made drawings. The tutorial consists of small sub-parts that are tied together so that the user can follow and read the whole process through, or view just one small part of it.

The complex tasks tutorials include at least template editing, report creation, defining own custom components, how to change the most common things in the drawings. Also a tutorial about the differences in 2D and 3D modeling should be given.

The tutorials consist of procedural instructions including step-by-step instructions, but provide the user also with conceptual information i.e. information about strategies what one could apply to the commands, examples where the commands can be used, what other things are related to the command etc. The use of animations is preferred.

The tutorials are accessed via the Help –menu.

Background and reasons for the solution

Despite the users most of the times are in a constant hurry, there are sometimes situations when they have a possibility to take time to learn something with the software. Since many of the users aren't so curious that they would start investigating the software on their own, a tutorial that guides the user through different features is a good way to get them started.

The users seem like it very much if the instructions contain pictures and animations especially on difficult subjects.

The suggested tutorials subjects are based on the opinions received from the users when asked, and the results of getting acquainted with the requests the users have made to the Service desk of Tekla.

Z. Suggestions on the contents of the on-line help and release notes

Description of the solution

The release notes that are used to introduce new commands and new features in the existing commands should contain examples on where the new features could be efficiently used.

The on-line help pages that open when pressing the "More…" button from the assistance bubbles should contain concrete examples on where the command could be used. Pictures should be used to clarity the text and animations when the subject can be considered difficult. The on-line help window that opens by pressing the more-button is opened to the right edge of the screen so that the Tekla Structures main window is diminished. This way the help window doesn't hide anything in the model. When the on-line help window is closed, the window is reduced to its original size. Also, if the user moves the help window, it moves and the main Tekla Structures window is reduced to the original size.

The on-line help should also contain information on strategies using the software. See chapter 2.4.1 for further information on the strategic content of on-line help.

Differences between 3D and 2D modelling should be made clear to the user in on-line help.

The help should also contain error information. This information tells users reasons of different error types and ways to get over them. Also typical reasons for getting into there error situations are described.

Background and reasons for the solution

The users are having troubles in finding where they could use different commands (Web Questionnaire). A command or feature may be left unused if the user doesn't figure out a proper use target for it. The users especially mentioned that examples of the use would often be beneficial. Also a typical method of learning how to use a command is to look from an existing model (8.5.1), and providing an example in the on-line help would save the user from opening a different model. Pictures help users who don’t' get the user interface and the instruction in their mother tongue. The animations are good in clarifying complex operations, but they should not be used in the simple basic tasks – in those the effort in making them goes in vain since the users seem not be willing to watch animation on simple tasks. On the other hand, animations on complex tasks were identified to be welcome. (6.9.4)

Most of the users who start using Tekla Structures come from another 2D software and thus aren't familiar with the benefits and differences of 3D modelling. Especially good usage strategies that are involved in the 3D modelling should be written out in the on-line help in a separate chapter dedicated to this subject.

Users need to use the help window simultaneously with the model in order to follow the instruction given in the on-line help window, and that's because the help window is initially placed to the edge of the screen and the main window is diminished to make room for the help window. If the user is using two monitors, he or she can move the help window to the other screen.

If an operation is done to an empty group of a very small group with no results, information about the selection and, suggestion on doing it on bigger group should appear.

7.1 The textual contents of the assistance

The assistance or embedded help bubbles' contents are discussed also in chapter 2.3.2. Here some examples of the contents of the assistance bubbles are given.

The most important thing to be considered when thinking about the contents of the assistance is to remember that the bubbles should be as short as possible. As mentioned earlier, the users want to get quickly information first and decide then whether more information is needed. Another important issue is the context dependency, as identified from the collected data and suggested by literature, for example Carbonell and Capobianco (2001).

The contents of the instruction text in the bubble consist of the following elements (in this order):

1) What is this command for, where it can be used. One or two sentences.

2) How is this command used. Brief description of the usage of the command.

3) (optional) Very brief summary of what can be adjusted.

The maximum length of it should not exceed seven lines.

When opening the help window with the more-button, the user should have immediate access to the step-by-step instructions of the command, information about typical errors with it, the ways to recover from them, some information about strategies, other commands related to the currently open command and examples of where the command could be used.

Step-by-step instructions are important to the user if he or she has decided that he can't figure out how the command is used based on the right-click bubble.

The error information has been noted to help the users more efficiently find solutions to their problems (Lazonder and van der Maij, 1995). The strategies help the user to generalize the use of the command to different situations and to find more places where to use the command. (Bhavnani et al., 1997). Examples of where the command could be used were identified to be important information to the users in this study.

7.2 Improvements not related to the user assistance

During the study usability issues within the software were encountered that are not directly in the scope of this study. As discussed in chapter 1.2.2., the borderline between improvements related to user assistance and other improvements in the user interface is sometimes not easy to determine. Thus some of the issues identified during the study that are close to the actual scope of the study, are presented here.

Default values in the dialogs should be chosen to be such that the user can create some kind of a draft version of the component and start making it better by the method of trial and error. If the user gets nothing created, it is much more difficult to see which values have an effect on which parts.

When a connection dialog is open and the user selects a part of the existing connection already in the mode, the relevant (corresponding) parts on the dialog are highlighted. The reason for this improvement is the fact that the users don't often know what different values in the dialogs change in the model.

8 Conclusions and further research

In this study we wanted to find out what kind of assistance information expert users of a complex application need, and that how this information should be offered to the user. We used eight different methods in all to find answers to these questions. All the methods chosen present user centred approach to the problem. The combination of methods suggested by different authors was chosen to best support finding answers to the research questions within the time and resource constraints.

With the results we got from the study we could identify the users' needs in getting assistance during the use of a complex application and provide solution proposals to address these needs.

8.1 Experiences of the methodology used

We gathered a large amount of background information to encourage the innovation of the assistance solutions with user and Tekla's Service desk personnel interviews, user observations, user diaries and web questionnaire to the users. The solutions of assistance included in some other software applications were also studied. Based on the data collected an affinity diagram was built to structure the data. With six other employees of Tekla, solution ideas based on the collected user data presented with the affinity diagram were created by brainstorming. Based on these solution ideas six small prototypes were built in order to test the ideas with the users. With the help of the test results, 28 concrete user assistance solutions to the case study of the Tekla Structures software were created.

The most valuable of the methods used turned out to be the interviews and the user observations. Getting familiar with other software resulted in lots of design ideas and understanding of the problem field within different types of applications.

The use diaries were an interesting method, which had a kind of experimental position in this study. Unfortunately only a small number of answers were received, but the results were still valuable. It still was an experiment that is well worth re-using in later studies.

Testing the assistance solutions with prototypes was a good way of finding out how the assistance solutions work. Some of the solutions proposed, for example the right-hand side of the status bar contents could not be prototyped without a more functional prototype than we used. Thus we complemented the testing with interview questions concerning the solutions that couldn't be tested. The time left to the testing of the assistance solutions was, however a bit too short. It would have been useful to spend more time on the testing phase and reduce the time spent on the data gathering phase. After the tests, some interesting issues such as the changes of the history panel, could have been tested in further tests, such as the changes made to the history panel, but this a subject for further research.

As discussed in chapter 1.2.2, the guideline in determining whether a solution is user assistance or other user interface development, has been the consideration whether the solution provides the user with more information or not.

8.2 The most important findings

Users want to be able to control the things that take place in the software. This applies to both the commands they are using and the user assistance. User assistance or embedded help solutions easily become irritating if the possibility to control the amount of assistance is not readily available. They are considered important as long as there is good control over them. We expected that the novice users would want to use the novice assistance mode in the software and experienced users the expert's mode, but this turned out to be a wrong expectation. The users need assistance with the software depending on the use situation: novices sometimes don't want instructions when they are doing something they know well, or especially repetitive tasks, and experienced users often run into situations where they want to view more information about a new command or a command they haven't used in a long time. This applies especially to different reminders the software gives to the user. The use situation changes thus often, even many times during one day. Assistance modes seem to be a good solution for the easy control for the amount of it. Because both the experts and novices use both the modes, they should be labelled as high and low assistance levels, not expert and novice modes. A pre-defined set of assistance solutions belonging to each mode should be carefully considered since it seems that the users typically don't have the time to customize them.

The need to see what has happened in the software beforehand is clear. Users want to get feedback of their actions, see what they have done and check whether the result was what they intended it to be. Thus the results of all actions must be visible to the user. A need to re-use information or executed commands was also identified.

The users like to use the software by trying themselves. Since by trying the users can get visual and concrete direct feedback of the command, this method of learning and using the software is preferred. Thus the software must support trying by providing instructions of commands during the use of commands. The instructions provided during the use of them must be available so that it doesn't interfere with the task at hand. It is also important that it can be ignored if the user chooses to do so. It must not be invisible though. The need to have one's attention attracted to the information messages provided was identified.

Because of the need for visibility and the trying tendency, a good method to undo the actions taken must also be provided. The more accurate the 'undo' feature is, the better it is. It must show what is being undone, since the users appreciate a possibility to see what they are undoing.

The instructions must be given to the user in stages. The users want to read a short crystallized version first, and then decide whether the command or function is good for them and whether they need more information about it. Overall, the information and instructions provided must be easily available and highly context dependent, i.e. closely connected with the task the user is currently doing. The users must not be forced to spend their time on searching for it; the software must do it for the users based on the users' actions.

Learning by following the various examples e.g. from colleagues or on-line help, was identified as an important method of learning new things and remembering the forgotten ones by both the novices and the experienced. This can be supported by providing easily accessible examples and directions of where the commands can be used. Also by giving

the users a chance to view how other users have used the software gives them a good way to share and get information of the usage. The software should automatically share out the accumulated information, since users seldom have the time or energy to document their actions if they don't have to do it.

Users of complex applications also want to get information about the various features of the software. They have may typically learned one or two ways of performing different tasks and have stopped learning new ones. Since there are often lots of commands that the users don’t know how to use, the software should present these features to the user. Longer waiting times connected with the loading of large files or executing complex commands can be used for introducing new or unused commands or new features in existing familiar commands, but the automatic introducing of these is also considered desirable. The software should keep track of what commands the user ha used and introduce the unused ones.

8.3 Validity and reliability of the results

The acceptability and usefulness of different user assistance solutions are easily prone to individual differences. Some people want to get suggestions and instructions from the software as if it was an experienced colleague: they listen to the ideas and suggestions and decide whether they are good or not. Others, on the other hand, don't want the software to try to be any kind of guide: they want to get information from it only when they want it. If trying to categorize himself into one of the categories mentioned above, the author belongs most probably belongs to the former one, the group with a positive attitude to assistance. While every effort was made to be as objective as possible in the study, when creating user assistance solutions, the researcher's bias according to his preferences could not be completely removed. The use of pre-defined forms in observation sessions and other software inspections, the use of several data gathering methods to complement each other all contributed to increasing the objectiveness of the study, but when analyzing and interpreting the results some effect of the author’s attitude is inevitable.

The number of test users was not very large. The use of six users in the tests and the interviews and only three in the user observations adds to the possibility to have biased results according to the selected users’ attitudes towards assistance. However, different types of users from different organizations participated in the study, and the differences between their backgrounds and the working environments suggest that results obtained were those of a typical group of users, rather than those of a special group.

8.4 Applicability of the results to other complex applications

Even though the data was collected from users of only one complex application, the Tekla Structures, the result can be applied to other complex systems, too. This conclusion is based on the fact that the in related research literature the same types of challenges and needs with complex software have been identified as in this study. These challenges and needs include for example the need to have context dependent information (Carbonell and Capobianco, 2001), error recovery instructions (Lazonder and van der Meij, 1995), or information about strategies and generalizations (Bhavnani, 1996) Also the challenges and problems found in the other complex applications examined within this study seemed not to differ much from the ones identified in Tekla Structures. Based on these facts, it can be argued that the needs of users of complex applications, as far as user assistance and

embedded help are concerned, are basically the same across the different types of complex applications. Of course not all of the solution proposals in chapter 7 can be directly applied to an application of totally a different field of expertise, but the users' needs which led to the solutions proposals, remain the same. With modifications the proposals, can however, be generalized to various types of applications.

8.5 Further research

It would be interesting to test more different kinds of assistance solutions with the users in performing some complex tasks and to find out if the user assistance solutions created in this study, have a positive effect on the efficiency of the use of the software concerned and other complex software applications.

A comparison between different kinds of user assistance solutions, the ones proposed in this study and others, would be an interesting study subject. It was assumed in this study that the user assistance solutions presented would enhance the users' ability to utilize the features of the software more efficiently, and this is clearly a subject for further study.

It was also assumed in this study that the user assistance solutions would reduce the users' time spent on learning to use the software. It would be interesting to test whether the user assistance solutions created in this study would reduce the time the users spend on learning how to use the software efficiently. Since the Tekla Structures is already an application that is in use it would be quite easy to arrange a test comparing the updated version with the user assistance solutions and the earlier version without them.

Observing the users with the suggested user assistance solutions implemented would give valuable information on how well they are accepted and how much them actually help the user, and how they are used. This information would be valuable in the further developing of user assistance.

The subject and contents of the tutorials were not a major point of interest in this study, however, they are well worth studying in more detail. In this study, the users of Tekla Structures were asked for their opinions on how to develop the tutorial features of the software, but a controlled experiment in finding out their real needs concerning tutorials would be beneficial.

References

Bhavnani, S. K. & John, B. E. (1996). Exploring the Unrealized Potential of Computer-Aided Drafting. In *CHI April 1996, Proceedings of the SIGCHI conference on Human factors in computing systems: common ground* (pp. 332-339). New York, NY; USA: Academic Press.

Bhavnani, S. K., Flemming, U., Forsythe, D. E., Garrett, J. H. Jr. Shaw, D. S. & Tsai, A. (1996). CAD Usage in an architectural office: From observations to active assistance. *Automation in Construction, 5 (1996),* 243-255.

Bhavnani, S. K., John, B. E. (1997). From Sufficient to Efficient Usage: An Analysis of Strategic Knowledge. In *CHI 1997, Proceedings of the SIGCHI conference on Human factors in computing systems* (pp. 91-98). New York, NY; USA: Academic Press.

Carbonell, N., Capobianco, A. (2001). Expert Contextual Online Help Strategies. *Ercim News*. Retrieved June 2005 from http://www.ercim.org/publication/Ercim_News/enw46/carbonell.html.

Carbonell, N., Capobianco, A. (2003). Online help for the general public: specific design issues and recommendations. *Universal Access in the Information Society, 2003: 2,* 1-15.

Carroll, J. M. (1988). Mental Models in Human-Computer-Interaction. In Helander, M. (Ed.) *Handbook of Human-Computer Interaction* (pp. 45-65), Amsterdam, The Netherlands: Elsevier Science Publishers B. V.

Chin, J. P., Diehl, V. A. & Norman, K. L. Development of an instrument measuring user satisfaction of the human-computer interface. In *Proceedings of ACM CHI'88 Conference.* (pp. 213-218) New York, NY, USA: ACM Press.

Faulkner, X. (2000). *Usability Engineering.* Palgrave Macmillan, Houndsmills.

Fenchel, R. S. (1981). An Integral Approach to User Assistance. *ACM SIGSOC Bulletin, Volume 13, Issue 2-3, 1982*. 98-104.

Gaver, B. Dunne, T., Pacenti, E. (1999). Cultural Probes. *ACM Interactions, January-February 1999.* 21-29.

Gould, J. D. (1988). How to design usable systems. In *Handbook of HCI* (pp. 757-790). : Yorktown Heights, New York: IBM Research Center-Hawthorne.

Hirsjärvi, Remes, Sajavaara. (1997) *Tutki ja kirjoita*. Jyväskylä, Finland: Gummerus Kirjapaino

Holtzblatt, K., Beyer, H. (1998). *Contextual Design*. San Francisco, USA: Morgan Kaufman Publishers.

Johnson-Eilola, J. (2001). Little Machines: Understanding Users Understanding Machines. *ACM Journal of Computer Documentation, 2001:25,* 119-127.

Lazonder, A. W., van der Meij, H. (1995) Error-information in tutorial documentation: Supporting user's errors to facilitate initial skill learning. *International Journal of Human-Computer Studies, 1995 (42),* 185-206

Lee, S. C., Huertas, A. & Nevatia, R. (2000). Modeling 3-D Complex Buildings With User Assistance. *Applications of Computer Vision, Fifth IEEE Workshop* (pp. 170-177).

Lehtola, L.,Kauppinen, M., Kujala, S. (2004). In *Proceedings of 5th International Conference on Product Focused Software Process Improvement, April 5 - 8, Kansai Science City, Japan*, (pp. 497-508). Berlin, Germany: Springer Verlag.

Lewis, C., Polson, T., Wharton, C., Rieman, J. (1990). Testing a Walkthrough Methodology for Theory-Based Design of Walk-Up-and-Use Interfaces. In *Conference on Human Factors in Computing Systems, Proceedings of the SIGCHI conference on Human factors in computing systems: Empowering people.* (pp. 235-242). New York, NY, USA: ACM Press.

Linja-aho, Minttu. (2005) *Improving and evaluating the learnability of a building modelling system.* Master's thesis, TKK.

Mack, R. L., Lewis, C. H. & Carroll, J. M. (1983). Learning to Use Word Processors: Problems and Prospects. *ACM Transactions on Office Information Systems* Vol 1, No. 3 (July), pp. 254-271

Mueller, P. (2002). Developing an Embedded Help Solution. *Technical Communication, Volume 50, number 1, February 2003.* 24-31.

Nielsen, J. (1993). *Usability Engineering*. Sand Diego, USA: Academic Press.

Norman, D., Draper, S. (1986). *User Centered Design*. Mahwah, NJ, USA: Lawrence Erlbaum Associates

PSMJ (Professional Services Management Journal) (1994). *CADD application and user survey*.

Root, R. W. & Draper, S. (1983). Questionnaires as a Software Evaluation Tool. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems* (pp. 83-87). New York, NY, USA: ACM Press.

Salter, W. J. (1988). Human Factors in Knowledge Acquisition. In Helander, M. (Ed.) *Handbook of Human-Computer Interaction* (pp. 957-968), Amsterdam, The Netherlands: Elsevier Science Publishers B. V.

Strassman, P. A. (1999). *The business value of Computers*. New Canaan, CT, USA: The information Economics Press

Taylor-Powell, E., Renner, M. (2003). *Analyzing Qualitative Data*. Retrieved June 3rd from University of Wisconsin-Extension Web-site, http://cecommerce.uwex.edu/pdfs/G3658_12.PDF

User Assistance. (2004) Retrieved 25.5. 2005, from http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnwue/html/ch13c.asp

van der Meij, H. (1992). A Critical Assessment of the Minimalist Approach to Documentation. In *SIGDOC 1992, Proceedings of the 10th annual international conference on Systems documentation* (pp. 7-17). New York, NY, USA: ACM Press.

Wood, L. E. (1997). Semi-Structured Interviewing of User-Centered Design. *ACM Interactions, March-April 1997,* 49-61.

Zubak, C. L. (2000). What is embedded help. *Intercom, March 2000,* 18-21.

Appendices

Appendix A Tekla personnel (Service) interview questions

A semi-structured interview. The original questions were in Finnish, but them are translated here into English.

Types of questions the users ask

1. Are there contacts daily? Mail or telephone? Same customer often?
2. Please describe a typical contact from a customer
3. Do the questions often concern bugs in the software or problems with doing something with the software? In this interview we concentrate on the ones that are not about bugs.
4. When user asks about a command, has he or she typically tried is first without success?
5. Has the user typically looked from help before contact?
6. How do you log the questions from users?
7. How do the problems in different parts of the software differ, if they do so?
8. Are there any types of problems that exist throughout the software?
9. In what kinds of situations the user often needs help?
10. Are there lots of impromement proposals from the users?
11. Are there differences between steel and concrete users' problems?

Properties of the customer

1. Are there more often questions from experienced or novice users?
2. Are there differences between the question types of experienced and novice users?
3. Are there differences between question types of users who have attended the training and who have not?
4. What are the general computer skills of the users typically? Is there a lot of variance?
5. What are the typical tasks with the software?

Other help

1. Do you know in what kind of problems users look for help in other places than the Service desk?
2. What other places?
3. Have the users commented the extent of the current on-line help?
4. Do you know how extensively the on-line help is being used?
5. Have the users commented anything about the current status bar/tool tips?
6. Are there questions about the training material? What is good/bad?
7. Do the users use only mouse of moth keyboard and mouse?
8. How busy are the users typically?
9. Have the users wished for any special type of help?
10. Have the users wished a special mode for experienced and novice users

Appendix B The user interview questions

1. How long have you used Tekla Structures
2. What is your educational background?
3. How long have you used computers in addition to Tekla Structures?

The current help and assistance

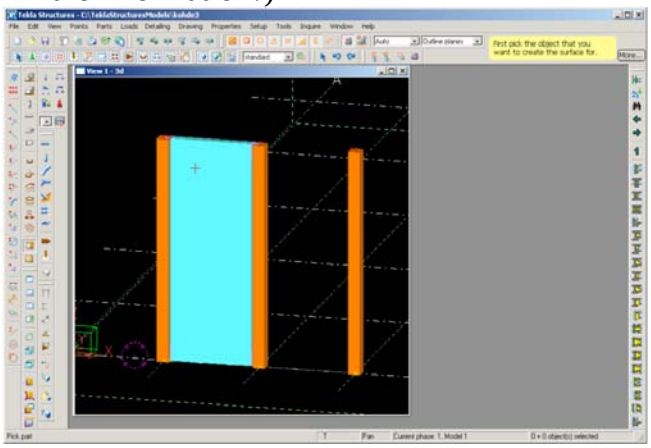
1. What comes first to your mind if you think about difficult tasks in the software?
2. Do you often follow the instruction texts in the status bar? Do them easily pass without noticing?
3. If you notice, have them been beneficial? Why?
4. What about the tooltips?`
5. Have you possible read them earlier more? Why?
6. Is it easy to find into the right place in on-line help?
7. Is there anything in the current help or assistance features that would have irritated of hindered your work? What?
8. If you are trying to use a command you haven't used before, and face a problem, how do you start to look for help? Imagine that there are no colleagues available.
9. If you face a problem or an issue you don't know the answer to, from where do you start looking for help? Or where would you like to get help?
    - If you know what you want to do, but you don't know how, and aren't even sure whether it is possible or not. For example, you want to modify a system connection in a special way
    - You know what you want to do, but don't know how, but you still remember that it is possible. For example, you remember that in the training you did that thing, but you don't remember where the command is.
    - You are in the middle of using a command, but you arten't sure whether it can be used the way you want to use it. For example, you are creating a connection but it always becomes slightly wrong.
    - You are in the middle of using a command, but don't know whether it is the right command for the place at all. For example, you are trying to make a connection with the tool you found, but aren't sure whether it fits to this point.
10. In which things the biggest problems occur when you are looking for help?
11. Do you feel that you can use the commands of Tekla Structures adequately well when thinking about your tasks?
12. Do you have a feeling that there are features or commands in the software that you haven't been taught, but that would be beneficial in your work?
13. If you think about these unused features, can you tell why them are left unused? What would make you to use them?
14. Do you have time for learning new features in the software? How do you typically learn?
15. Have you had problems in finding commands after version update?
16. Do you feel that there are happening things that are difficult to notice?
17. Would it be good to attract attention to small things on the screen?

Ideas and opinions

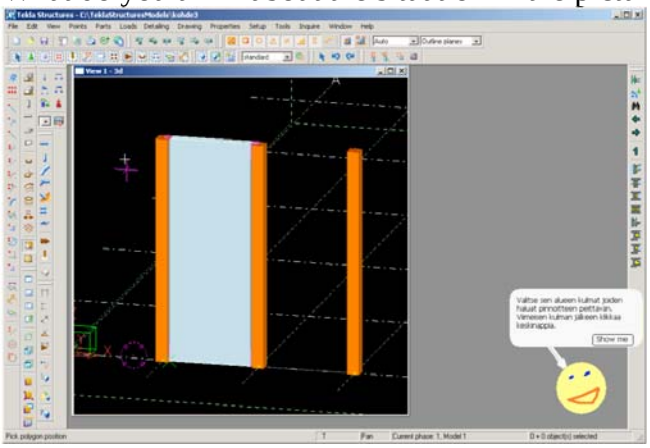
1. Would you like to have new features presented to you when you use them for the first time?
2. When using the software, is there room for something related to assistance on the screen?
3. If the software offered textual instructions, where would you like to read them?
4. How long instruction texts you normally want to read from the screen?
5. Do you often miss information about what it is possible to do with different tools? For example with examples?
6. Have there been situations when you would like to know what it is possible to do to a certain object in the model or what alternatives you have to do a certain thing?
7. What about scrolling back instructions received?
8. What do you think about videos or animations in the software? Sounds?

The illustrations

1. Scenario: "You are modelling a simple wall and want to create a surfacing with the create polygon surfacing –t ool."
   What do you think about the situation in the picture? (do you think you would read the information?)

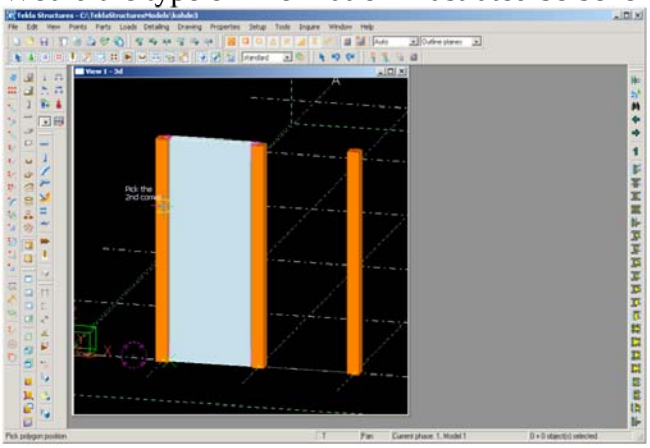


2. Scenario: "You are continuing the previous task, and have now selected the panel you want to create the surface for".
   What do you think about the situation in the picture? Disturbing?

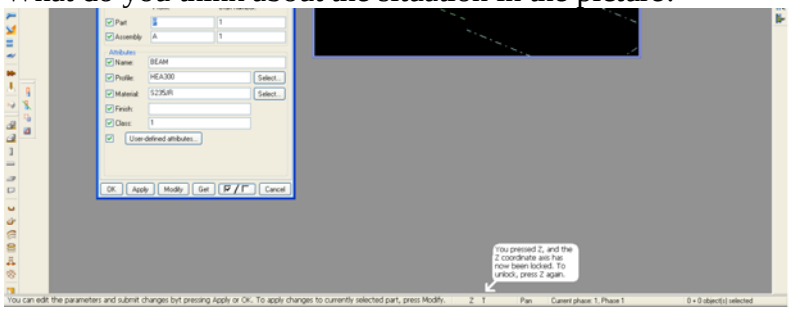


3. Scenario: "You have chosen with the Create polygon surfacing –tool the object and the first corner."

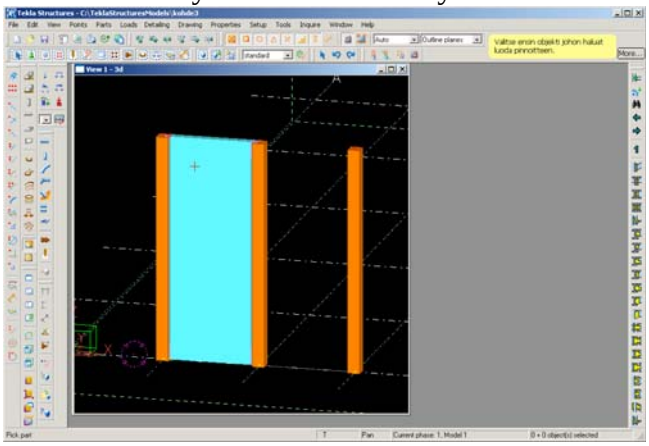
Would the type of information illustrated be beneficial?

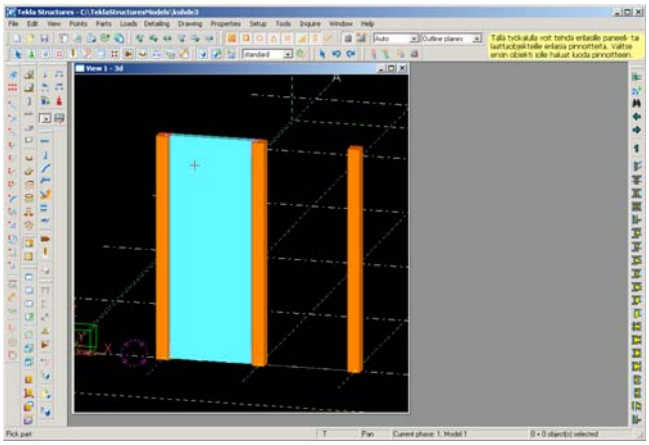


4. Scenario: "You are modelling a building and have pressed the Z-key in order to lock the Z –axis.
   What do you think about the situation in the picture?

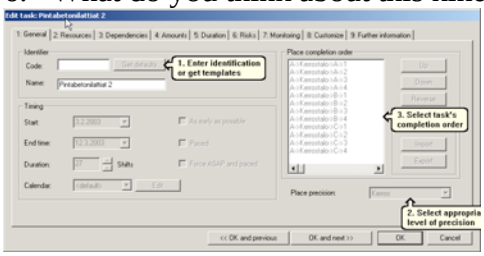


5. Scenario: "The same situation as in the first picture"
   Which one do you like more? Why?

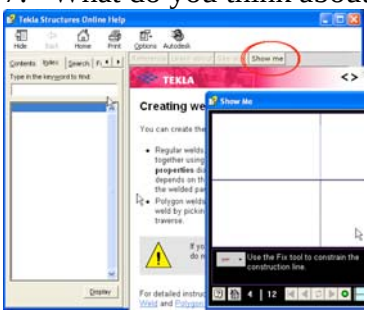




6. What do you think about this kind of assistance in a dialogue?



7. What do you think about this kind of help window? Would you use the button?



Appendix C The user observation form

Havainnointilomake	
Avunhaku	
Käyttää helpiä dialogista	
Lukee statusbaria	
Lukee release notes:eja	
Etsii helpistä ohjetta	
Kysyy apua	
Kokeilee toimintoa useamman kerran	
Kohdatut ongelmat	
Piirustukset	
Mallinnus	
Liitokset	
Tulostus	
Tiedostojen lukeminen	
Tallennus	
Mallin avaus	
Muu	
Muu	
Muu	
Ikonin klikkaaminen	
Dialogin painikkeet	
Tyypilliset tehtävät yleisesti	
Tyypillisen tehtävän osat: (täytä)	
Työpisteen ominaisuudet	
Tilaa	
Ahdasta	
Rauhaisaa	
Melua	
Häiritseekö ääni muita	
Kuuluvatko äänet	
Kuulokkeet	
Kiire	
Aikaa tutkia ongelmia	
Kiinnostusta tutkia ongelmia	
Aikaa lukea ruudun viestejä	
Hermostuu ylimääräististä klikeistä	
Näyttöpääte täynnä tavaraa	
Näyttöpäätteellä tilaa	
Hiiri	
Näppäimistö	
Tarkentavia kysymyksiä	
Miksi teit äsken noin?	
Miksi et tehnyt?	

Appendix D The user profiles

User 1

User one (U1 in the interviews, Ut2 in the usability tests) is a youngish male person who works at a big building engineering company. He is an experienced Tekla Structures user and has been using the software actively for two years. Despite the relatively short Tekla Structures use period he has become so experienced that he is a one of the so-called contact persons in his company. He has studied construction engineering and has been using computers since childhood. He has also some programming experience, mainly Visual Basic programming.

He is using Tekla Structures as his main work tool. He could be called a heavy-user: he utilizes four screens and three mice attached to two computers. He currently uses the software much to create custom connections, custom details and custom components.

He builds custom components that other modellers in the company also use.

He likes to read the message box to get along with the functions. He is irritated by the fact that there is so little assistance available in the software. He would like to get active assistance for example with error situations. He would also like to get answers whether something is possible or not. When creating custom components, he would like to have a possibility to add himself instructions for example to the status bar. Overall, his attitude towards active assistance was quite positive.

User 2

User two (U2 in the interviews, Ut1 in the usability tests) is a young male person who works in a large building engineering company, the same as Ut1. He is a novice Tekla Structures user and has been using the software for about one month. His education is a construction engineer. He has relatively good experience in using computers and has been using CAD software before Tekla Structures.

He is a person who likes to try things out with computer – he usually is not scared to try things in the software. When he faces a problem, he likes to ask from his colleagues for help. He has several colleagues that most of the times can help him. He thinks that by asking you can clear your problem more quickly that by any other way.

He says that he cannot use the features in the program efficiently. Basic functions are OK, but more specialized things must be asked from colleagues or tried to "dig" from the software. He isn't using on-line help at all, the help is not even installed on this computer. He doesn't like to read long text from the screen and it feels like he is so socially oriented person that he would not like to spend time with reading the screen in search for help. But short instructions get his support, it sounds like he has missed sometimes such things but hasn't use time to look for such things.

User 3

User three (U3 in the interviews, Ut3 in the usability tests) is a middle aged male person who works in a middle-scale building engineering company. He is a senior Tekla Structures user, he has been using the software for about five years. Before that he had

been using another 3D modelling software called Steel CAD. He uses Tekla Structures as his main work tool.

By education he is a machine engineer and he has a long experience in using computers. He feels that Tekla Structures use in general is flexible and not very difficult. He explains that you can always proceed by staring from a suitable macro, explode it, make the needed modifications. If you need same kind of component elsewhere, you make a custom component out of the modified one, otherwise you're done.

He seems to be using help, both by pressing F1 from dialogs and from by opening it from the menu. But he mentions that first he tries always to figure out functions by testing and it feels that he's not afraid of trying and clicking around in the software. He seems to be somehow interested in solving problems with the software and mentions that sometimes if there is time he finds himself from trying to figure out how to use the function that caused a problem. If there's no time, he doesn't hesitate to call the support though.

He says that there are some favourite connections that he is using most of the time. Even if he sometimes works in a hurry, he finds also time for learning by practicing. He likes examples as instructions and short texts.

User 4

User four (U4 in the interviews, Ut4 in the usability tests) is a male around 40 years old designer who works in a mid-size building engineering company, the same as U3. In his workplace there are several Tekla Structures users. U4 is a novice Tekla Structures user: he has been using it in one project which was completed a few months ago. By education he is a construction engineer. He says he has quite long experience with computers. At the time of the interview, he wasn't using the software in any project and thus the observation part of the meeting couldn't be done.

He says that he is using the statusbar quite a lot in the software since he thinks that it is important to know what is going on in the software. He doesn't have on-line help installed so he doesn't know about it. He says that in some situations he would look at the help if it was installed, for example when trying a new command. He is learning new commands by trying and asking from a colleague. He is quite busy with his projects and says that there is no time for learning new commands – he tries to complete his projects with the commands and functions he is familiar with. He doesn't like to read texts, but rather would look at a picture. When using a tool in the software, he would like to get information whether the tool he is using is a good one for the purpose. If he was trying to get help from the software he would rather use an index than typing keywords to a search function.

He doesn't like the creature giving help, but he would still like to see the information it gives.

User 5

User five (U5 in the interviews, Ut5 in the usability tests) is an experienced male Tekla Structures user. He has been using the software for about ten years as his main working tool. He has a constructional sciences educational background. His computer using skills originate from the 80's from the times of PC/AT 286 machines.

U5 looks at Tekla Structures' status bar occasionally since he has noticed that it's information value is close to zero. He is irritated by the fact that there are some

inconsistencies between the functions in the model and in the drawings. He doesn't like to use the on-line help much, but says that when he goes to look for something there, he usually finds the information he's looking for.

He likes to try out things himself, especially when speaking about joints. He says that in them it is more convenient to just try different values. Sometimes if he isn't sure about something he likes to call the support to get a confirmation to his on experiments he has done. He admits that he is kind of stuck into his old habits, meaning that he is quite reluctant to take new commands into use. He rather uses to good old way even if it was not the most optimal way of accomplishing the task at hand. He says that this is partially due to the fact that there is so little information available. He would be more interested in new functionalities that change the way that the outputs look.

There are places in the software where he would like to see more information. For example, he would like to get more information by moving the mouse on different objects. He seems to have nothing against more help or assistance as long as he can control it. It sounds like he would turn all the help off and enjoy his old way of working, but would still like to get some information in some difficult places.

User 6

User six (U6 in the interviews, Ut6 in the usability tests) is a male person working in a small building engineering company. He is a novice Tekla Strucutres user, he has been using it intensively for about 8 months. He is quite experienced computer user in general; he is for example the one who at least partially takes care of the company's local area network. He has been using CAD before starting to use Tekla Structures. He has both steel and concrete parts of the software but has been mainly using the steel modeling part of it.

He is interested in learning to use the program, but says that there is always the shortage of time. He likes to learn by doing, plagiarizing ready-made examples is a good way of learning according to him. He is not afraid of testing and trying things with the software. He finds that one of the most difficult thins in the software is to make the drawings look what you want them to look like. He also likes short crystallized instructions where there is everything you need to get started. This kind of instructions he would preferably read on paper. Also a possibility to get more detailed information after the first short description would be good, he says. He also uses the on-line help when he faces a problem. Wasting time in searching for information is one of the reasons that prevent taking new features into use, he says.

He is concerned about if possible instructions would cover something important on the screen. Otherwise his attitude towards more instructions is very positive. He says that in fact there are surprisingly little toolbars and icons in Tekla Structures. In other programs he normally skips all introductions and tutorials, but says that Tekla Structures could well be an exception in this matter.

Appendix E The use diary template

Background information

For how long have you used Tekla Structures?

For how long have you used computers in general?

What other software do you use in your work (or have lately used?)

The issues covered each day

Tell briefly what you did with Tekla Structures today?

Describe situations occured today in which you right away didn't know how to proceed? *For example "When making a surfacing with the Create polygon surfacing –tool, I didn't know what all points I have to pick with the mouse"*

Tell where did you look for solutions for these issues?

Tell where did you get answers or solutions to these issues?

Did you feel that something you did today was slow with Tekla Structures?

Did you try to find a more efficient way to do this thing? How did you try to find?

Tell what irritaded you or pleased you today in the software.

Today was (please tick)

hurry	
time to explore	
time to learn new	
there was time but not energy	

Thoughts about the software today

Something else

Appendix F The web questionnaire web form

Tekla Structures User Assistance Study Questionnaire

Improving the usability is one of of our important objectives in the development of Tekla Structures. you.
This questionnaire collects information from users of Tekla Structures about the use of the current o the user in using the software. This questionnaire is a part of the Master's Thesis study of Osmo Tolv

The answers will be handled anonymously and will be used only in the development of Tekla Struct

Filling instructions:

Please choose the alternatives that best suit your opinion in each question, or write down your answe When finished, please click the "Submit answers" -button on the bottom of the page.

Your answers will help us greatly in developing Tekla Structures.
Thank you very much for your time and answers!

Background information

What is your educational background?

Computational sciences

Construction sciences

Other

Which age group do you belong to?

18-25

26-40

41-55

56-

Years of experience with Tekla Strucutures (or former Tekla Xsteel)?

Less than a year

1-2 years

3-7 years

8 years or more

Detailing background before starting the use of Tekla Structures

Drawing by hand

Using a 2D modeling software

Using other 3D modeling software than Tekla Structures or Tekla Xsteel

No previous detailing experience

Frequency of Tekla Structures use currently

Daily

1-3 times a week

Few times a month

Infrequently

How would you describe your skills in using computers in general?

Novice Expert

>Current use of Tekla Structures

How do you currently seek for help when you face a problem when using Tekla Structures? Yo

Training material

On-line help

Colleagues

Helpdesk

Other, please specify:

Why do you prefer this or these way(s) of seeking help?

Which of the following do you prefer first when starting to learn using a command you haven'

Trying myself

Asking a colleague

Asking from helpdesk

Reading the on-line help

I try to avoid new commands

Other

Do you have a feeling that there are features in Tekla Structures that you don't know how to u

Yes, lots of such features

Yes, but not much

Not generally

None

It's difficult to say

How do you agree with the following statement:
In my work I have time for learning existing features in Tekla Structures that are new for me.

I fully agree	I disagree

Why taking into use features that are new for you might feel difficult in Tekla Structures? You

They are difficult to find

It is difficult to learn using new features

It is difficult to find help on using new features

It is difficult to find out which commands are suitable in different situations

There's no time for that

There's no need for that

Other, please specify:

What kind of problems do you have more often?

Ones that must get answer right away

Ones that can wait answer for a few hours or even days

In which areas of the program do your tasks concentrate in your team? You may select several

Modeling

Detailing

Drawigns

Templates for reports/drawings

Other, please specify

In which areas of the program do you mostly face problems? You may choose several alternati

Drawing editing

Creating connections and details

Creating primary structures

Creating custom components

Template editor

Other, please specify:

How do you read the Release Notes that come with a new version of Tekla Structures?

I read them through carefully

I glance them through

I glance only the possibly interesting parts

I don't read them

>Opinions

How would you like to get help when using Tekla Structures? You may choose several alternat

By reading comprehensive step-by-step tutorials

By searching from the on-line help

By being offered a possibility to open and read the on-line help pages related to my current task

By being offered a possibility to read short instructions (balloons etc.) related to my current task

By setting Tekla Structures to an assistance mode in which short (balloons etc.) instructions are

What is your general attitude towards the idea that a program tries to figure out situations wh

I like it	I don't like it

If you answered that you don't like it in the previous question, can you tell what the most anno

They disturb my work

They contain useless information

They are difficult to notice

They are complicated

They are too general

They don't deal with my current task

It's difficult to get rid of them

If Tekla Strucutres offers you assistance automatically, where would you like to see the assista

Located clearly visible, for example in a new window

Other, please specify:

Do you feel that text messages or other information pass on the screen without your noticing in

Often

Sometimes

Never

What do you think about using animations or videos as part of Tekla Structures?

I would like it	I would not like it

What do you think about using sounds in Tekla Structures?

I would like it	I would not like it

What do you think about using speech in Tekla Structures?

I would like it	I would not like it

Which of the following do you prefer when using Tekla Structures?

Use only mouse if possible

Use only keyboard if possible

Use both as much as possible

Do you share your screen space with other programs when using Tekla Structures?

Most of the time

Sometimes

Never

Is there room for new small elements on the screen (for example a text box) in addition to wha

Yes

No

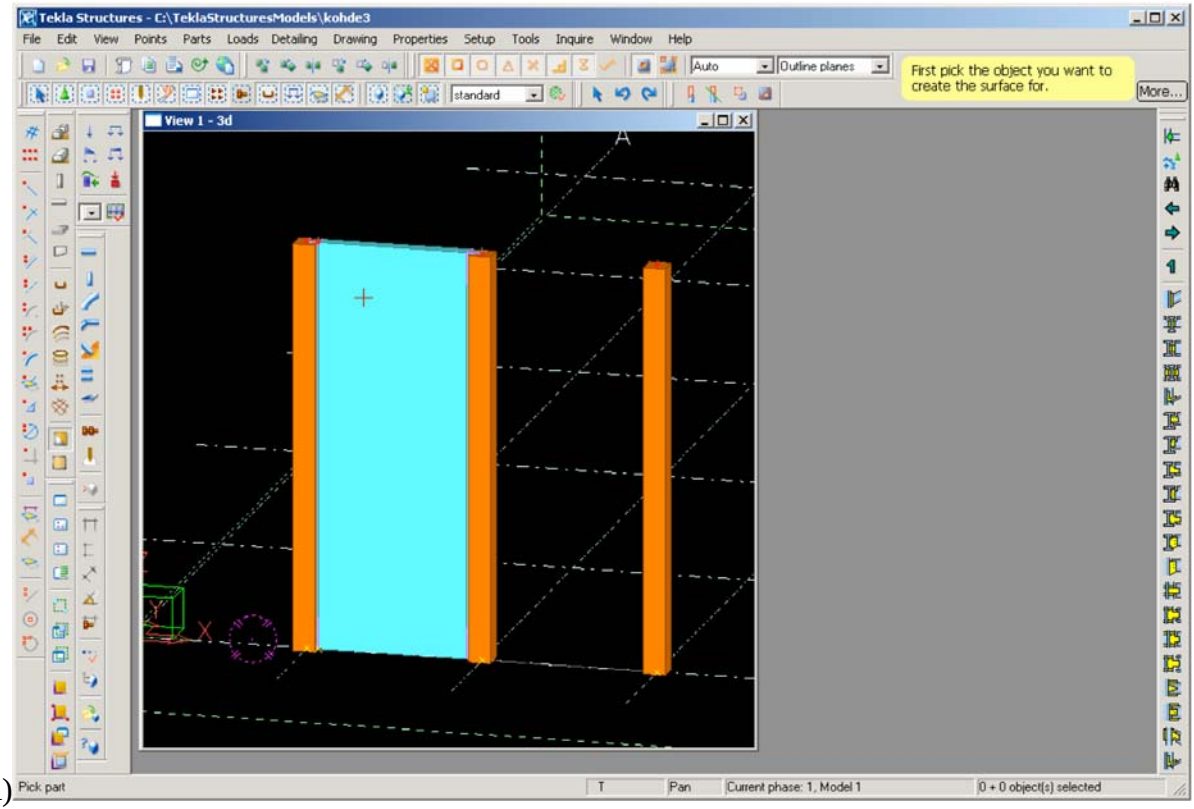
Imagine that you have started to use a command called "Create polygon surfacing" and have s do you like more?

You may see an enlarged version of the picture by clicking it

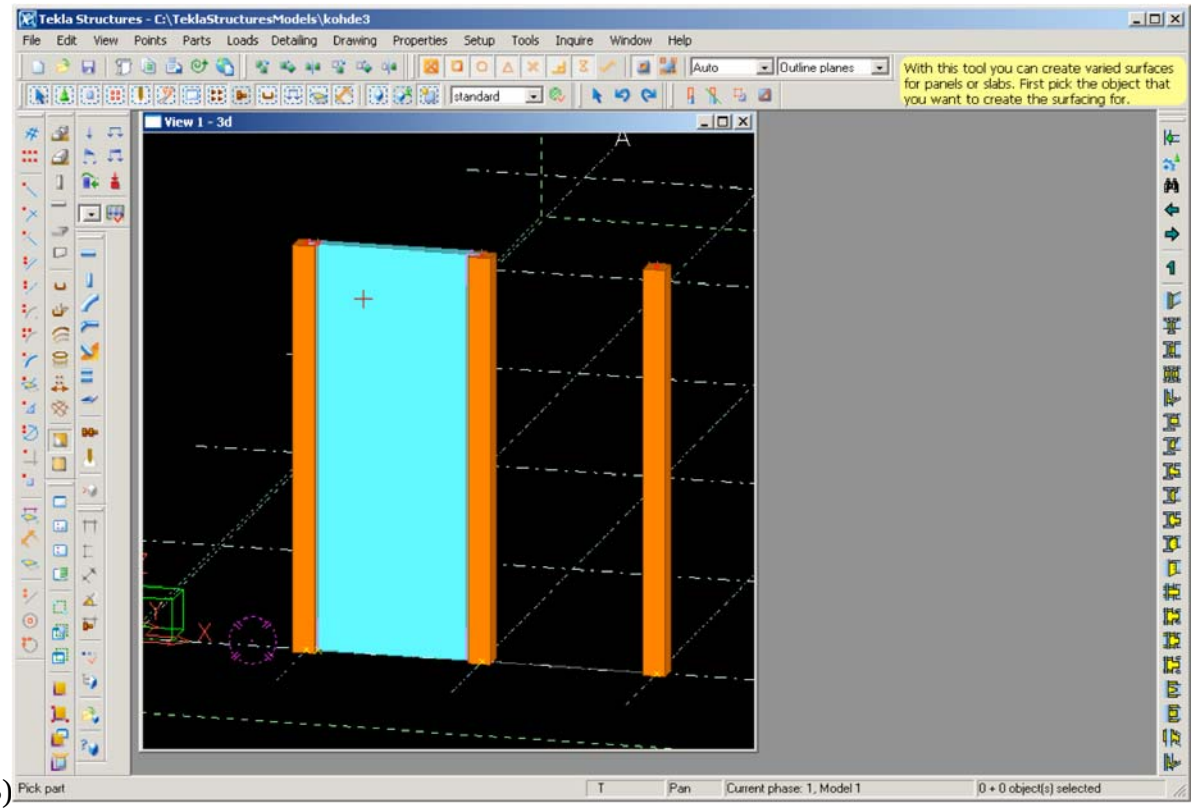
I like more the first picture (A)

I like more the second picture (B)

A)



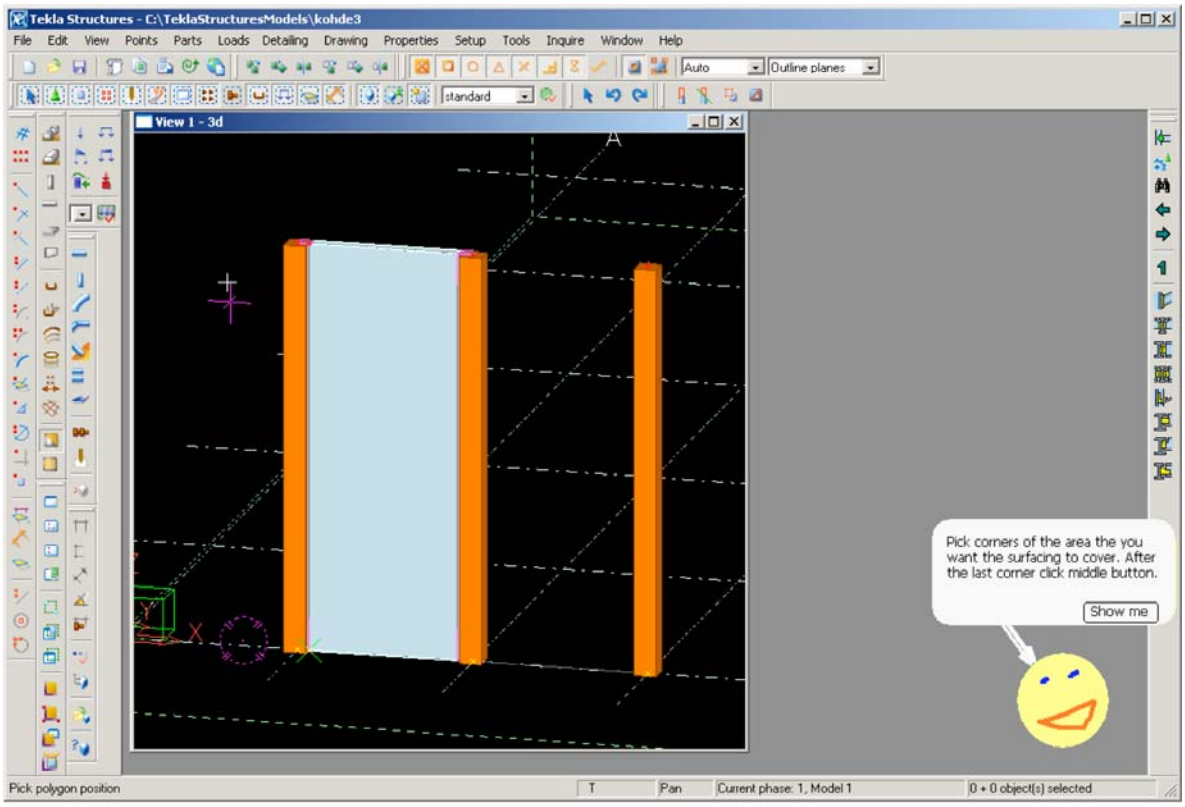
B)



Possible comments on the pictures

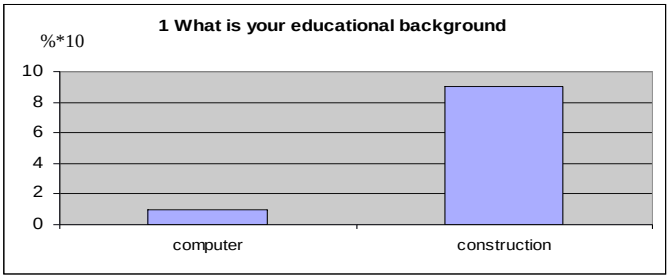
In the following picture you have started using the create polygon surfacing tool. You have pic
You may see an enlarged version of the picture by clicking it

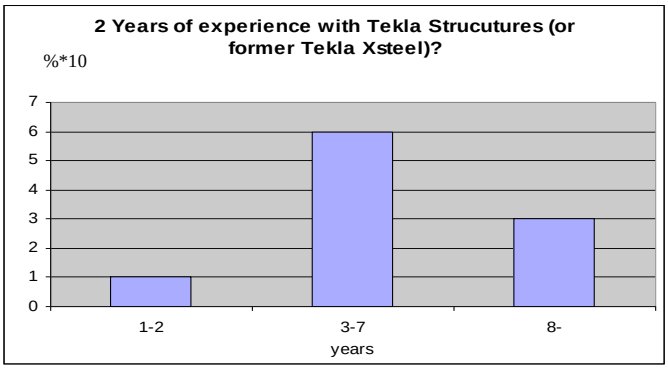
I would like it	I would not like it

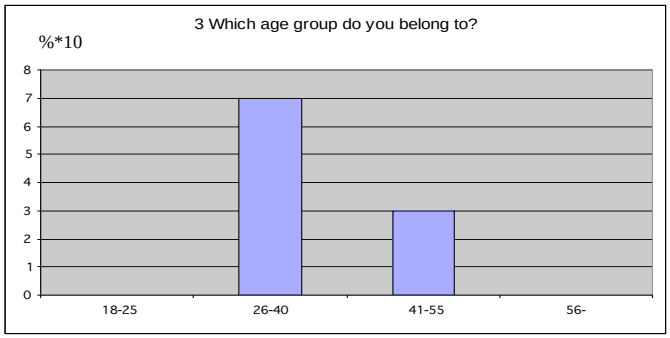


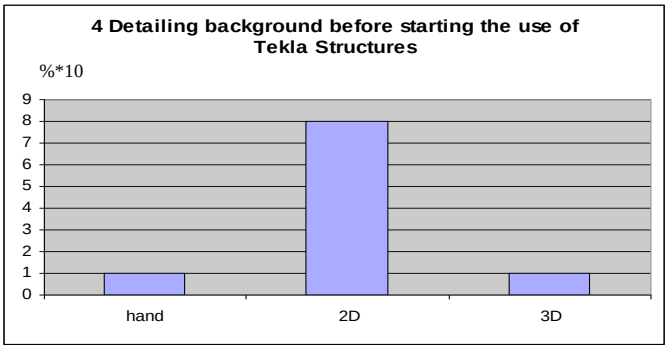
Possible comments on the picture

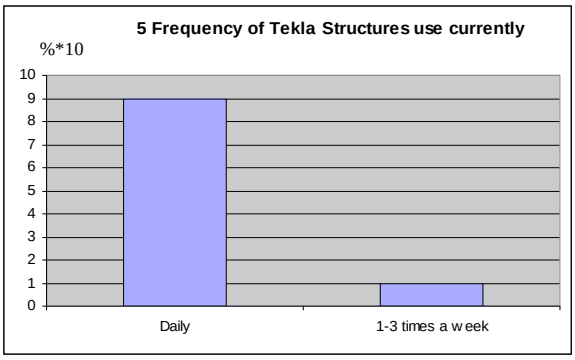
Appendix G The Web-questionnaire results

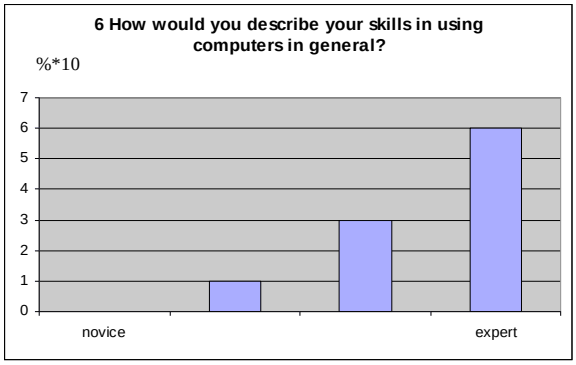


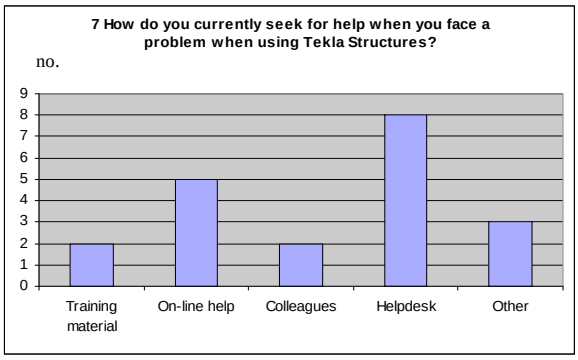


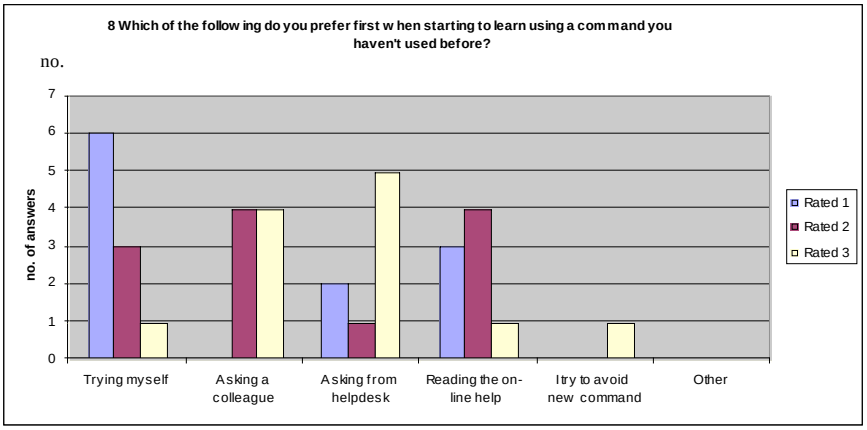


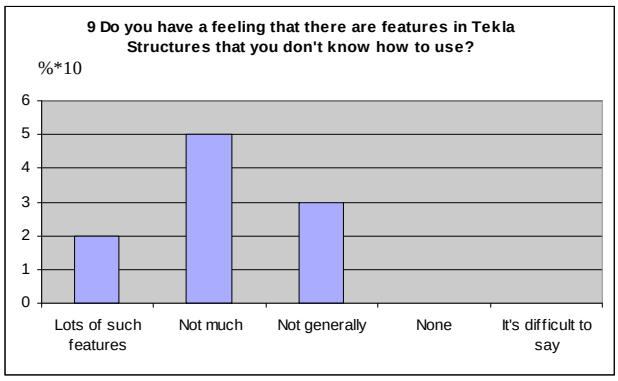


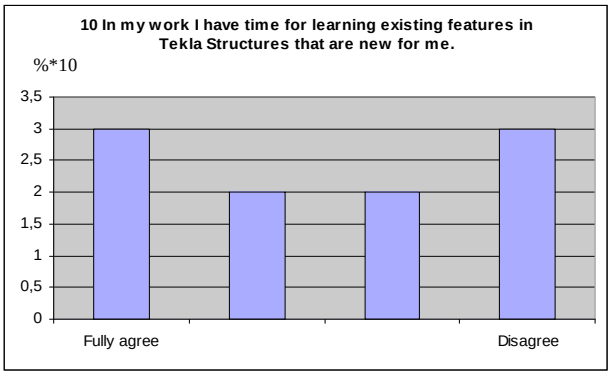

5 Frequency of Tekla Structures use currently
%*10
10
9
8
7
6
5
4
3
2
1
0
Daily
1-3 times a week

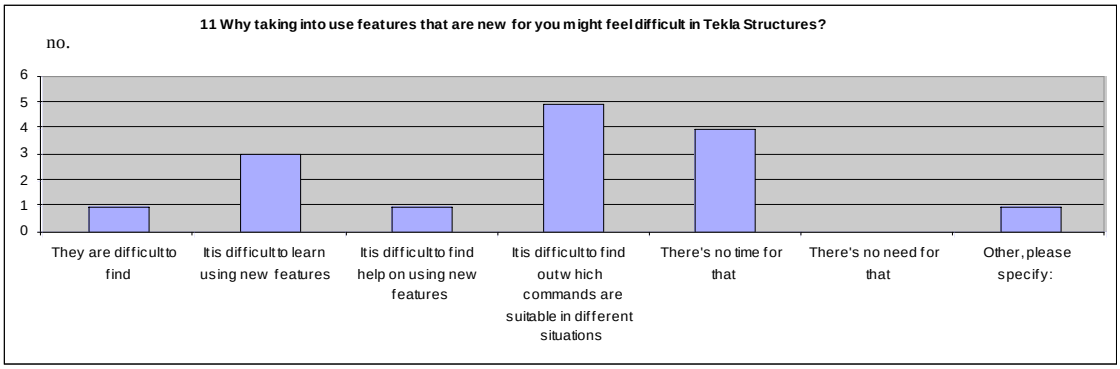

6 How would you describe your skills in using computers in general?
%*10
7
6
5
4
3
2
1
0
novice
expert

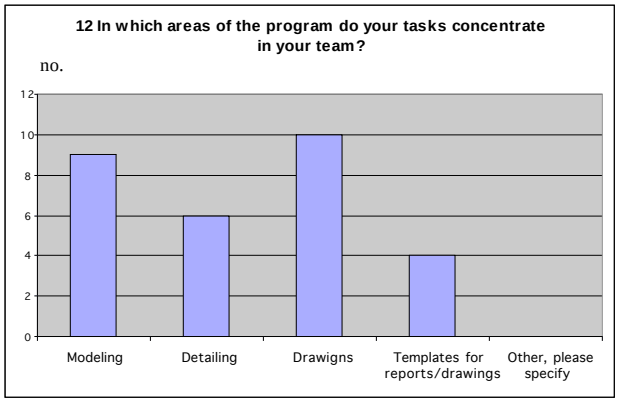

7 How do you currently seek for help when you face a problem when using Tekla Structures?
no.
9
8
7
6
5
4
3
2
1
0
Training material
On-line help
Colleagues
Helpdesk
Other

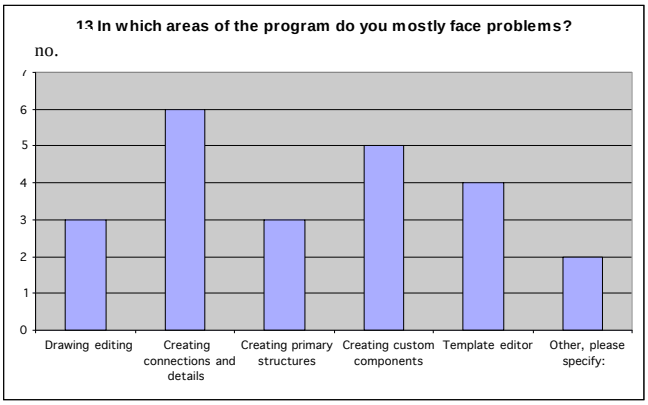

8 Which of the following do you prefer first when starting to learn using a command you haven't used before?
no.
no. of answers
7
6
5
4
3
2
1
0
Trying myself
Asking a colleague
Asking from helpdesk
Reading the on-line help
I try to avoid new command
Other
Rated 1
Rated 2
Rated 3

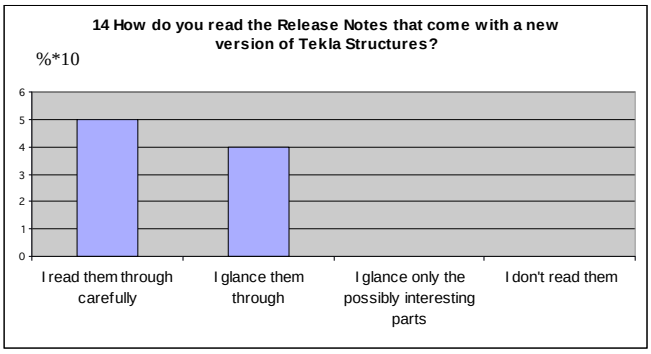

9 Do you have a feeling that there are features in Tekla Structures that you don't know how to use?
%*10
6
5
4
3
2
1
0
Lots of such features
Not much
Not generally
None
It's difficult to say

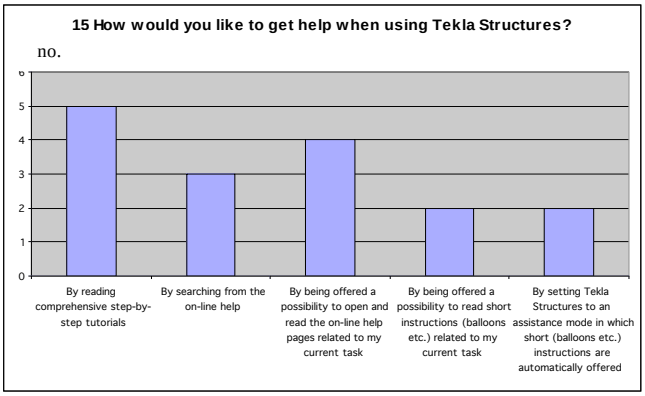

10 In my work I have time for learning existing features in Tekla Structures that are new for me.
%*10
3,5
3
2,5
2
1,5
1
0,5
0
Fully agree
Disagree

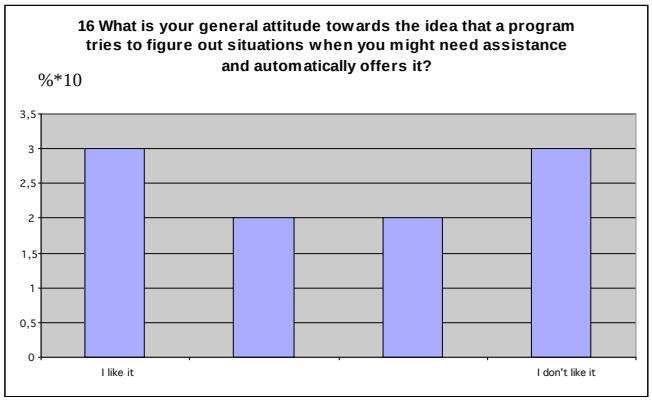

11 Why taking into use features that are new for you might feel difficult in Tekla Structures?
no.
6
5
4
3
2
1
0
They are difficult to find
It is difficult to learn using new features
It is difficult to find help on using new features
It is difficult to find out w hich commands are suitable in different situations
There's no time for that
There's no need for that
Other, please specify:

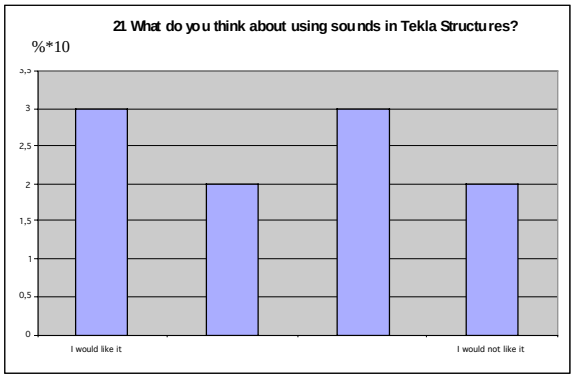

12 In which areas of the program do your tasks concentrate in your team?
no.
12
10
8
6
4
2
0
Modeling
Detailing
Drawigns
Templates for reports/drawings
Other, please specify

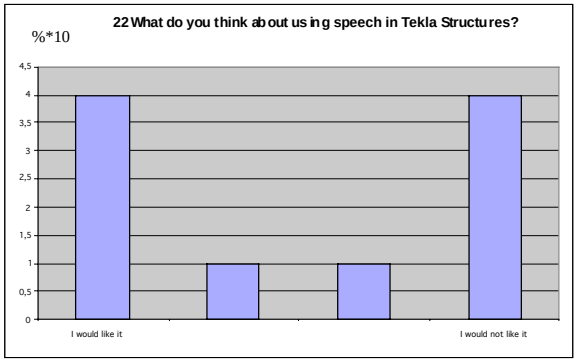

13 In which areas of the program do you mostly face problems?
no.
7
6
5
4
3
2
1
0
Drawing editing
Creating connections and details
Creating primary structures
Creating custom components
Template editor
Other, please specify:

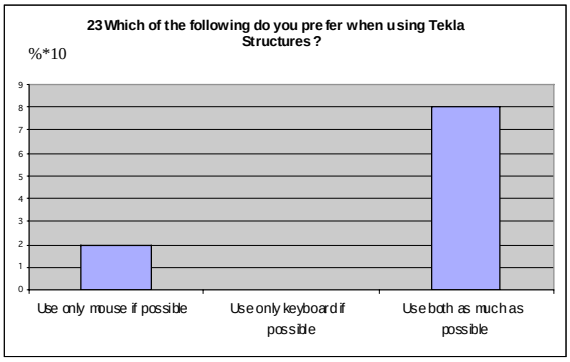

14 How do you read the Release Notes that come with a new version of Tekla Structures?
%*10
6
5
4
3
2
1
0
I read them through carefully
I glance them through
I glance only the possibly interesting parts
I don't read them

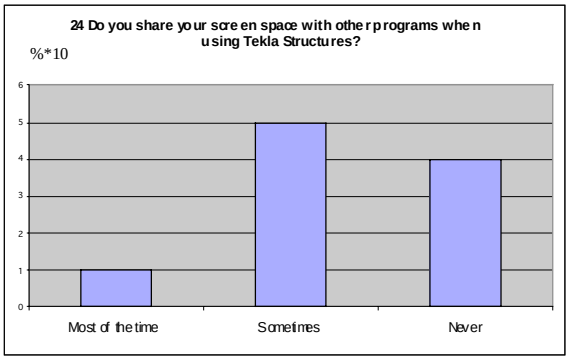

15 How would you like to get help when using Tekla Structures?
no.
6
5
4
3
2
1
0
By reading comprehensive step-by-step tutorials
By searching from the on-line help
By being offered a possibility to open and read the on-line help pages related to my current task
By being offered a possibility to read short instructions (balloons etc.) related to my current task
By setting Tekla Structures to an assistance mode in which short (balloons etc.) instructions are automatically offered

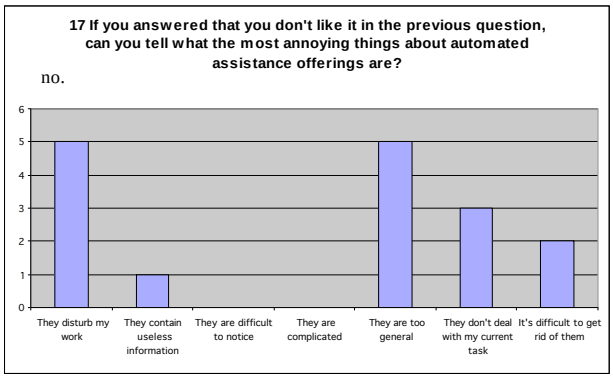

16 What is your general attitude towards the idea that a program tries to figure out situations when you might need assistance and automatically offers it?
%*10
3,5
3
2,5
2
1,5
1
0,5
0
I like it
I don't like it

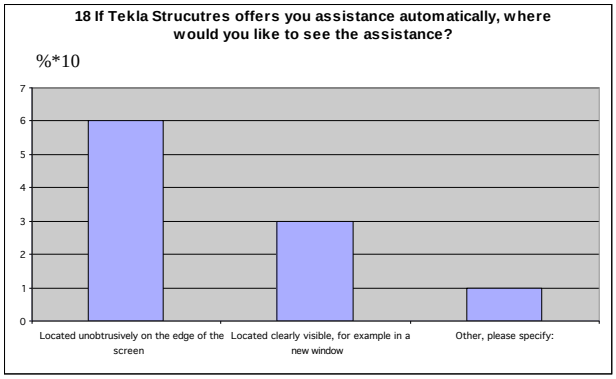

21 What do you think about using sounds in Tekla Structures?
%*10
3,5
3
2,5
2
1,5
1
0,5
0
I would like it
I would not like it

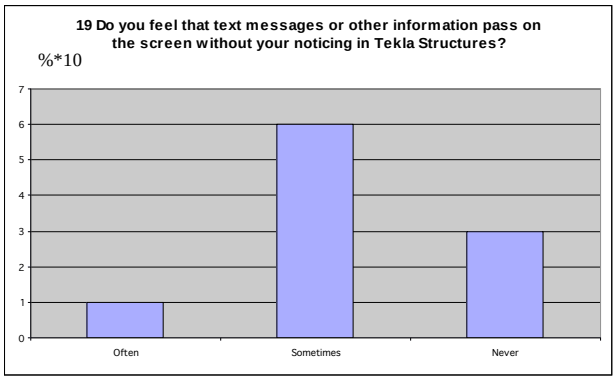

22 What do you think about using speech in Tekla Structures?
%*10
4,5
4
3,5
3
2,5
2
1,5
1
0,5
0
I would like it
I would not like it

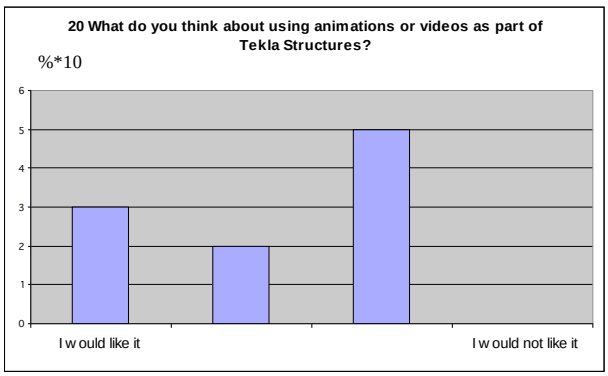

23 Which of the following do you prefer when using Tekla Structures?
%*10
9
8
7
6
5
4
3
2
1
0
Use only mouse if possible
Use only keyboard if possible
Use both as much as possible

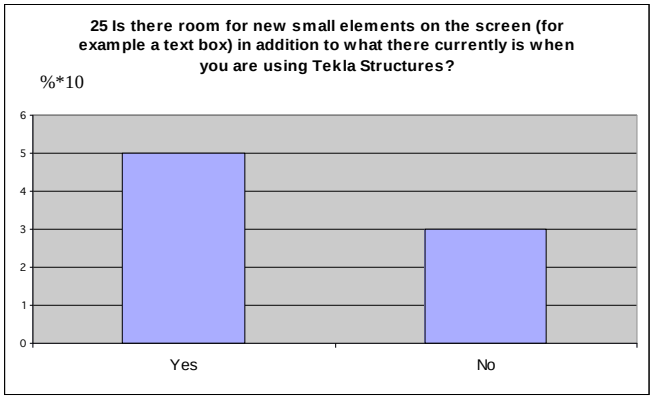

24 Do you share your screen space with other programs when using Tekla Structures?
%*10
6
5
4
3
2
1
0
Most of the time
Sometimes
Never

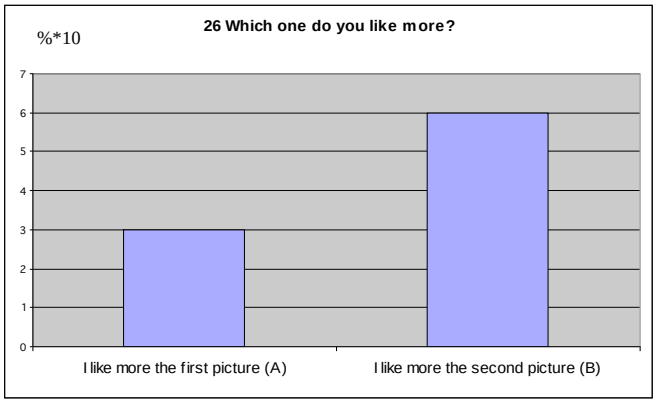

17 If you answered that you don't like it in the previous question, can you tell what the most annoying things about automated assistance offerings are?
no.
0
1
2
3
4
5
6
They disturb my work
They contain useless information
They are difficult to notice
They are complicated
They are too general
They don't deal with my current task
It's difficult to get rid of them

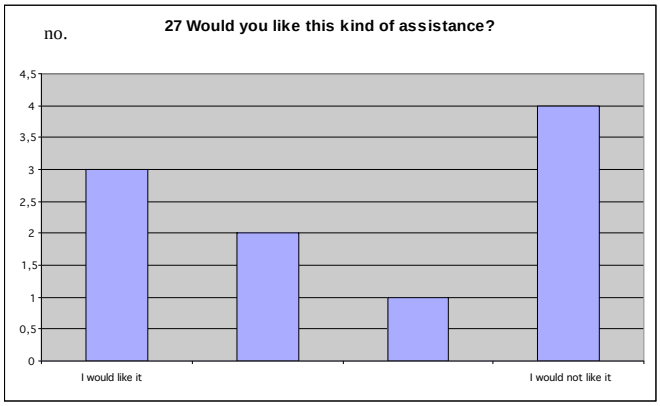

18 If Tekla Strucutres offers you assistance automatically, where would you like to see the assistance?
%*10
0
1
2
3
4
5
6
7
Located unobtrusively on the edge of the screen
Located clearly visible, for example in a new window
Other, please specify:


19 Do you feel that text messages or other information pass on the screen without your noticing in Tekla Structures?
%*10
0
1
2
3
4
5
6
7
Often
Sometimes
Never


20 What do you think about using animations or videos as part of Tekla Structures?
%*10
0
1
2
3
4
5
6
I would like it
I would not like it


25 Is there room for new small elements on the screen (for example a text box) in addition to what there currently is when you are using Tekla Structures?
%*10
6
5
4
3
2
1
0
Yes
No


26 Which one do you like more?
%*10
7
6
5
4
3
2
1
0
I like more the first picture (A)
I like more the second picture (B)


27 Would you like this kind of assistance?
no.
4,5
4
3,5
3
2,5
2
1,5
1
0,5
0
I would like it
I would not like it