Framing the Reference
An Analysis of Contextual and Goal Oriented Design

Christopher Overall
Abstract

Framing the Reference

Christopher Overall

We are all in a codependent, dysfunctional relationship with the technology we use. We have grown to be utterly dependent on it but technology is often designed with its needs and not the users in mind. But is there a method of design that will create usable systems or are we left in the hands of design auteurs with a magical touch? By analysing and contrasting the methodology of Contextual and Goal Oriented design I applied a version of both to a piece of software in order not only to improve the experience of using it but also to determine how well the methods could be applied and how they could be combined.
Introduction
  Purpose
  Background
    The Company
      Tracy
  Problem Formulation
  Limits of Scope
Method
  Goal Directed Design
    Personas
    Goals
    Scenarios
Contextual Design
  Contextual Interviews
  Workmodels
    The Flow Model
    The Sequence Model
    The Culture Model
    The Artifact Model
    The Physical Model
Methodology
  Goal Oriented vs Contextual Design, a discussion
  Adaptation
Result
  The Contextual Interviews
    Initial Interviews
    Cultural Model
  The Personas
    Jocke the Assembly Worker
    Olof the Line Manager
    Tomas the Quality Control Officer
Goals
  Scenarios
  The Prototypes
  Reevaluation
  Discussion and Conclusions
Appendix
  Windows Presentation Foundation
  The Model-View-ViewModel Pattern
References
Introduction

Purpose
The field of usability and user centered design aim is to understand the user and her situation to bridge the gap of understanding between creator and user. Both in enterprise and consumer applications usability is important for several reasons. Companies are always looking to become more efficient and they don’t want employees to waste time with cumbersome software or that they make mistakes because of them.[1] In the consumer goods sector it is even more apparent that good usability is important as those companies that embrace design prosper and those who don’t loose their customers.

The purpose of this thesis is to compare and analyse Alan Cooper's Goal Oriented Design and Contextual Design outlined by Hugh Beyer and Karen Holtzblatt and to apply the methodology to construct a new interface. Both methods have merit but are at times in direct opposition, this thesis aims to clarify how the two methods correlate and if they can be combined. The aim is not to use both in parallel and create two designs based on two different methods but rather to explore the possibility of using both together and to forge a methodology where both compliment each other if possible or conclude why such a combination can not be achieved.

An initial round of interviews will be analysed using the methodology and the following design will be made in a new reworked prototype of the system. This prototype will then be the subject of testing and interviews to assert that the new interface is more user centered. The second round of interviews will be analysed and if possible new conclusions will be derived and new design ideas taken from those conclusions.
Background

The Company

Company-X is an international manufacturing company manufacturing a wide variety of Products. Company-X’s clients requests that all components the Products are comprised of are traceable in case of a component breaking due to a manufacturing deviation Company-X needs to be able to issue a recall notice for any Product using the same erroneous batch of the component.

Components for the Products are either sourced from a third party vendor or manufactured from raw materials. In the case of the components being bought the third party vendor provides a batch number that the components should be registered with. If a component is manufactured in house however an internal work order number is used as a batch number. The manufacturing of components is done by another department of Company-X and they deliver their components labeled with the work order number.

Company-X uses two different kinds of work orders, ones for the finished Product and ones for Assemblies. Assemblies are a part of the finished Product which require some assembly prior to being fitted to the finished Product. Products can be said to have a hierarchical structure consisting of components and assemblies, components are indivisible items whereas assemblies in turn consists of components. The work order for the Product mirrors this hierarchy by listing the work orders for the Assemblies it comprises of.

Tracy

To meet the needs of Company-X KMD has developed a software solution to track which components are used in the manufacturing of it's Products, aptly named Tracy. Tracy is comprised mainly of four different essential parts, a windows based client which is used in the manufacturing process, a database which stores the information and a website used to search the database and an administrative client allowing a superuser to alter some of the data in the database.

The Database is used in two ways, of course it’s used to store the batch numbers during the manufacture of products after all this is the main purpose of the of the system but in order to do this it also stores information about the components, assemblies, products and their relation to each other. This data is used by the windows client, named TracyIn to display the structure of the products and assemblies as well as displaying important information regarding such things as the components Storage Location.

The database currently has a number of issues, it has a very strict relationships, namely almost half of the columns are designated as key fields. This makes the database very rigid in terms of what data is necessary before it will allow to be written to.
From looking at the database diagram it is also clear that the tables have a poor naming, all tables contain “Workorder” and the diagram gives no real impression of what real world scenario it is supposed to model.

TracyIn as mentioned above is the windows client handling the input from the users, storing the batch numbers in the database. TracyIn is built using Windows Presentation Foundation (WPF) and implements the design pattern Model-View-ViewModel. The batch numbers are a random number sequence seven digits long considering a single product and be comprised of over a hundred different components manually inputting this data would be disastrously time consuming. Instead TracyIn utilizes a barcode/QR code scanner, all batches of components are marked with a sticker with a barcode that represents the batch number.
Barcodes also contain the article number and revision of the component and in some instances an additional lot number.

The whole point of this system is of course to track what components were used in what products and what their batch numbers were, and the way that this is made accessible from the database is through a web interface named TracyOut. TracyOut consists of a single page allowing the user to search for a number of qualifiers: article numbers, revisions and of course batch numbers and order numbers. The search results are then displayed in a simple table and also allows for the user to export the search results to Microsoft Excel.

**Problem Formulation**

This thesis will focus on studying all aspects of Tracy from a Usability perspective and evaluating using the methods described in *Contextual Design. Defining Customer-Centered Systems* by Beyer and Holtzblatt. This initial study is intended to show to what degree the existing system is user centered and hopefully identify where it is lacking to fulfill the users needs and wants.

The study will be the basis of a redesign of Tracy that incorporates the findings to produce a new working prototype of Tracy. The prototype will be developed also in adherence to the principles described in Eric Reiss’s *Usable Usability* and Alan Cooper’s *The Inmates are running the asylum*. This new prototype will in turn be subjected to a new usability study to determine whether it is an improvement on the previous version and how well it has adapted to the perceived demands of the users. To be able to compare the results between both studies

**Limits of Scope**

The thesis will not include developing any new functionality to the existing system but rather a revamping of existing functionality. It will also not include any integration of other production or logistical systems used by Company-X, the resulting prototype will be stand-alone. Nor will the prototype be acceptance tested, it is intended as a well functioning prototype and not a finished and verified product.
Method

There is no one way to achieve usable systems, several methods, that in part contradict each other have been proposed. The field of Human Computer Interaction is naturally as diverse as the possible uses for a computer and the results of user centered design are hard to gauge can we say that one method always produces more usable systems. Without a usability measurement it is of course impossible and we can't expect realistically to be able to objectively gauge a systems usability in the same terms we gauge it's speed, reliability or power consumption.

Goal Directed Design

Alan Cooper describes his method dubbed “Goal-Directed-Design”[6] in his book “the Inmates are running the asylum”. Before defining his method Cooper speaks at length about the current situation in system design to highlight why it is important refocus existing design methods or lack thereof and also tries to shed some light as to why so many systems are so hard to use. Cooper divides people into survivors and apologists, by apologists he refers to all with a keen sense of logic who after being confronted and mastering a complex and difficult system feels empowered. By overcoming a difficult hurdle they gain power over all who fails to do the same ie they Cooper dubs the survivors. The apologists have by gaining this power a vested interest in keeping this relationship, they apologize and forgives these unwieldy systems because of the position it gives them over the survivors. The survivors muddle through their computer aided tasks never being empowered by them, forced by necessity to use them rather than enjoyment or practicality. “The survivors are the vast majority of people who are not impressed by the newfound power, but who are mighty impressed by how stupid the interaction makes them feel.”[6]

This is of course not the industries way of looking at it’s users and customers most often all users are divided into three categories: the naive, the computer literate and the super users. Here Cooper makes the point of that the reason to aspire to create systems with good interaction design is not just about creating software that will sell or be efficient in the workplace but rather it’s a question of equality.

It creates a demarcation line between the haves and have-nots in society. If you must master a computer in order to succeed in America’s job market beyond a burger-flipper’s career, then the difficulty of mastering interactive systems forces many people into menial jobs rather than allowing them to matriculate into more productive, respected and better-paying jobs.[6]

To combat this and refocus the design process towards the user Cooper defines his process called Goal Directed Design[6], in essence this method of design rests on three pillars; personas, goals and scenarios.
Personas

When designing with focus on the user Cooper advocates that you explicitly define this user by creating what he calls a Persona[6][2]. The persona is a fictive person, a user, but completely fleshed out, a persona has not only a name, age and title but rather all the details of a real human being. Rather than having just a title a persona should be described as having a real job at an actual company, not just as having a car but rather a late model volvo sedan. The point of a persona is that it is specific this is to avoid the persona becoming a statistical average but rather a real person even though they are completely fictitious. When creating a persona Goal Directed Design urges you to make them as believable as possible but specific and avoiding statistical averages, as Cooper points out:

An average user is never actually average. The average person in my community has 2.3 children, but not a single person in my community actually has 2.3 children[6]

Personas should always strive to be precise rather than accurate, the reason for this to prevent the persona from becoming an elastic user that can take any shape or form and with a seemingly infinite different tasks to perform.[6] Intuitively we can say that a persona can never be absolutely accurate since they are fictitious people, what would be the odds of creating a persona that happened to exactly match a living person, the point is that they are believable and detailed.

An advantage gained by creating these personas is that you develop a more nuanced and realistic view of their skill level. The three tiered pyramid so often used is rarely an accurate description of an actual person and their skill level.[6]

Cooper makes a point of not using actual people rather than personas, actual users can provide raw data but should not be the basis for the design. This is due to the fact that real people have idiosyncrasies, traits that are more or less unique to them or at least not representative of the majority of users. Reversely a particular person might grasp a concept or problem to solve easier than others effectively masking a problem. Users can describe a problem but not necessarily offer the solution to it “being the victim of a particular problem doesn’t automatically bestow on one the power to see its solution”[6]

Personas themselves serves different purposes and during the design process several personas will be developed, the personas that are the focus of the design are called Primary Personas[6]. The Primary Personas are those you determine to have unique goals, and if different personas are developed with the resulting same goals these should be aggregated into one Primary Persona. You might develop more than one Primary Persona and for each of these a different interface will be designed, as a rule of thumb if there are more than three Primary Personas the scope of the problem is too large[6]. If there are too many personas the
The purpose of creating them will be defeated, the idea is to have precise individuals and not to design for the average of a group. It is also important to create the personas of the individuals that you are not designing for to be able to separate out where the focus should not be placed, these personas are called Negative Personas[6].

Goals

The goals refer to the goals of the Personas, the Goals and the Personas are intrinsically linked you can not have one without the other. An important distinction to make is that a Goal is not the same thing as a task, a task is the process that you perform to achieve your goals. Programmers have a tendency to think of tasks as isolated processes and not to focus on the users end Goal[6]. Tasks change with technology whereas goals do not, the goal of an accountant running an agency have been the same for as long as the concept has existed; to keep track of their clients finances but the technology has moved from abacus and clay tablets to computers and iPads and so has the tasks performed also changed.

When programmers create software based on the tasks to be performed they might ultimately work however they will only work well if tasks are performed in exact sequences and by exact ways with exact data, this often fails to satisfy the users own goals[6].

The Goals of the Persona can be divided into four different categories[6], Personal Goals, Corporate Goals, Practical Goals and False Goals. Cooper defines the Personal Goals at the core to be:

- Not feel stupid
- Not make mistakes
- Get an adequate amount of work done
- Have fun (or at least not be too bored)

A person will always strive to complete their own personal goals before that of the corporation that they work for[6]. The Corporate Goals can be summarized to simply maximize profits and market share these are goals that of course need to exist but they utterly fail to motivate users. Cooper borrows a term from Saul Gellerman; *Hygienic Factors* defined as “prerequisites for effective motivation but powerless to motivate by themselves” without the Corporate Goals there would be no motivation for using the software at all but it can not motivate the user by itself.

Practical goal are what lie in between Personal and Corporate goals these are e.g taking a customers orders or summarizing expenses, these are the type of goals that programmers and executives find appealing, concrete and simple to grasp rather than the human personal goals or abstract business goals. But ultimately these goals are also hygienic in the sense that they have to be fulfilled but for example a software that allows you to take a customers order but does not adhere to your personal goals will fail and will not motivate a user but rather discourage.

Lastly Cooper identifies False Goals[6], usually these are goals that ease the creation of the
software rather than eases the use of it. False Goals are for example:

- Save memory
- Run in a browser
- Look great

A user's primary goal is never to save memory or use some new technology, they want to get some work done without being made a fool of, saving memory might be the developers task to accomplish but it will never be someone's goal. Similarly a user never wants to use something purely on it's esthetical merit, software is not art and beautiful software is nice but it's looks does not imply that it will fulfill the users personal goals.

Scenarios

Following the development of Personas and their goals are to put these into scenarios. The scenarios are derived from observation of actual users performing tasks and interviews and can be described as the process of a persona accomplishing a goal. One aspect of creating scenarios is to identify any unnecessary task but the scenarios need to describe the interaction in full rather than with great detail in every step[6]. The process of creating scenarios can be compared to how film or television is written and in particular method acting, this means trying to envision how a character would act in a certain situation and what motivates them to act in a certain way.

Scenarios can be divided into three categories: Daily, Necessary and Edge Case Scenarios. Edge case Scenarios are however never developed as they do not contribute to furthering the design, rather these are the kind of scenarios that programmers will seek out since they focus on the edges to make sure the program never crashes. An important distinction is however that the program should be able to handle these edge cases but the design should in no way focus on them.

Daily use Scenarios are the most important and are the actions a user will perform the most frequently. Generally users have no more than one or two such tasks to perform regularly and these need to be the focus of the interaction design. Ideally these need to be supported by help from the software so that users can learn them quickly and at the same give the intermediate user access to them quickly and even be customizable so they can be made to suit each individual. This is in contrast to Necessary tasks, these are performed infrequently and due to this their interaction requires the same user help as the Daily Scenarios however due to their infrequent use this help should always be provided. Users won't also need the customization they require for their Daily Scenarios nor the quick access.
Contextual Design

Contextual design stems from the belief that systems can be designed based on data and that we do not need to rely on design gurus but rather gives us a systematic approach[5][3] to collecting, interpreting and modeling data collected from users. The basis for Contextual Design is performing Contextual interviews and then interpreting them, creating different workmodels.

Contextual Interviews

Contextual Interviews take place in the users workplace while they work. The idea is to get the user to explain how they perform their job while they are doing so, the reasoning behind this is that this will bring unconscious behaviour into the conscious realm. Beyer and Holtzblatt describe thoroughly different techniques to accomplish this and points out various pitfalls, by letting the user work instead of retelling how something works you avoid the user summarizing the process [3]. By summarizing detail is lost and if you instead observe the process unfolding you can record in detail what happens instead of getting the summarized, abstract version from a persons memory.

When interviewing and observing the user work Beyer/Holtzblatt advocates forming a partnership with the user, becoming collaborators [3]. Traditionally when we think of interviewer and subject all power is really in the hands of the interviewer, the interviewer controls the questions and by that the content and in some respects the results. The focus of the interview is the work and switching between watching it unfold to discussing it between designer and user as equals. During these discussion breaks from observing the user work designers are encouraged to vent spontaneous design ideas, this in order to get immediate feedback and to get user involvement cemented as early as possible in the process rather than getting feedback on a prototype with limited possibilities of re-design or alteration.

Workmodels

After the interviews have been conducted and data has been collected this is interpreted in group sessions with the entire design team, the goal of which is to produce workmodels. Workmodels are as the word suggests models of different aspects of the users work converted into generic models. Beyer/Holtzblatt defines five different basic workmodels: the flow model, the sequence model, the cultural model, the artifact model and the physical model [3].

The Flow Model

The flow model describes how an organisations have distributed different roles and responsibilities between individuals and how information flows between these individuals. By trying to answer how roles are distributed and what roles a person takes on to get work done as well as how tasks are distributed, communicated and how their completion is ensured and by whom a picture emerges of the flow of information in the organisation. As Beyer/Holtzblatt puts it: “Work flow represents every phone call between two people, every document passed for review,
every email message, every conversation between people in the hall” [3]. A document or email is a static tangible artifact that can easily be appropriated into the flow model but equally important is the informal conversation in the hallway and just as important to observe and interpret.

*The Sequence Model*
Based on the reasoning that people do things not in a random order but rather in a rational sequence, the sequence model tries to determine the process in a task but also to expose the users intent. Beyer/Holtzblatt recognizes that simply doing something doesn’t necessarily show the users intent, this is left to the designers to interpret [3]. The sequence model could be said to simply be a flow diagram of events but with the addition of intent to highlight a users motivation for performing certain tasks. If the intent of the action is deduced then the design can cater to fulfill the users actual intent rather than focus on the performance of a certain task.

*The Culture Model*
To define expectations, desires and values Beyer/Holtzblatt tries to create a model of how the culture of the organisation [3]. Compared to the other models this is perhaps the most abstract and most difficult to define but by listening to tone and how users describe their and others group or role as well as the more concrete policies and hierarchies within an organisation influencers and influences can be defined. The following image is an example of a culture model of a developer:
The Artifact Model

Just as archaeologists search for artifacts so should a designer collect all items that are produced or altered during the process of work. These could be anything from invoices, emails or blueprints. All these artifacts can tell the designer of the users actual intents, assumptions and strategies although not implicitly rather the design team have to deduce this from interpretation.

The Physical Model

All that we design have ultimately reside in the physical world even software is dependent on such things as screen resolution, battery life or just simply the distance to the printer. If the physical surroundings are not taken into consideration can cause the design to fail. Alerts and alarms might be useless if the user's surroundings are noisy and rather a visual alarm might be of more use.
Methodology

Goal Oriented vs Contextual Design, a discussion

The method section of this Thesis describes two alternative methods of design which differ, contradict and sometimes compliments or agrees with each other. The fundamental difference in the theory behind these two methods is Beyer/Holtzblatt positivistic view of the world, Conceptual Design suffers from the rigorous detail needed to fulfill this positivism to the point of being almost unusable. Contextual Design creates huge amounts of data[4][5] and everything needs to be analysed in detail; every piece of paper, the distance between the computer screen and the printer even coffee break discussions. Just producing five workmodels, consolidating and appending these would consume all time allowed for this master thesis, leaving no time to actually produce a prototype.

Here Beyer/Holtzblatt worldview differ remarkably from Coopers, for programmers it seems natural that the solution to bad interaction would be an algorithm, a generic process that if followed in great detail and with enough data would always produce a system with good interaction. Cooper on the other hand would term this kind of behaviour as the work of “Homo Logicus”, the logical man. To strive for control and thereby accepting a huge increase in complexity, for “Homo Logicus” this is desirable, mastering a hugely complex system leaves them feeling empowered. Beyer/Holtzblatt method might aim to create user centered systems and interactions but the method itself is anything but, just as a programmer feels empowered by sitting in a cockpit surveying a huge array of systems and controls and mastering these so it seems Beyer/Holtzblatt enjoys creating wall sized charts of interactions and cataloging every minutia of office life.

*The inmates are running the asylum* seem to be almost a call to arms, even the title is belligerent and so the book has a focus on describing the climate and people that produces such poorly functioning software rather than being a pure how-to-manual of user experience design. There are some absolutely glaring omissions in Coopers description of the design process, chiefly concerning the initial stages of gathering information from the actual users. You are hard pressed finding information other than “conduct interviews” before Cooper leaps into describing the Personas he created and their goals he designs for. Cooper could have any number of reasons to omit this, goal oriented design is not focused on the actual users but rather personas and Cooper therefore skips descriptions of user interviews as not to throw the focus back on actual users. But regardless of why the omissions in Goal Oriented Design are there they make it appear almost magical; personas might be easy enough to define, but to identify primary personas and deriving their goals seems at times to be plucked from thin air.
Adaptation

I agree with Cooper and his world view and design ideas to a large extent but as I stated his method is somewhat lacking in detail whereas Contextual Design is excruciatingly detailed. I propose to use some of the techniques described in Contextual Design to conduct the initial interviews and gather data. I will use contextual interviews and interview the users while they are using the current system and observe them in the environment in which the system is used. I will not make a Physical Model but I believe the physical surroundings of this system will impact it's design. With this as raw data I will create several personas and identify which are primary and my design targets. I also propose to create a Cultural Work Model and see if this can shed some light as to the Goal of a persona or at least to more depth into their motivation. And lastly I will define some Scenarios and identify which are everyday use and which are only necessary or edge cases.
Result

The Contextual Interviews

Three different interviews were held one with the quality control officer and two with the assembly workers. All interviews were held at the users actual workplace while they were performing a typical task. The quality control officer used the improved prototype of the web interface TracyOut which was re-designed based on Goal Directed Design. The interviews with the assembly workers were held on two separate occasions, once to gauge the existing interface and once to evaluate the prototype. No contextual interview was done with the Assembly line manager but rather a normal interview since the existing Administrative interface has very limited functionality.

Initial Interviews

When visiting the assembly line one thing became abundantly clear and that was the un-ergonomic position of the terminal, placed on top of the storage units and at shoulder height.

Terminal with barcode scanner next to the assembly line
Next to the terminal the assembly workers kept the work orders they were in the process of manufacturing. The Assembly Line manager would receive orders to his desk besides the assembly line and print these along with work orders he created for pre-assemblies. Pre-assemblies are parts of the product that require to be assembled before being mounted on the final product. The assembly workers always start with creating the work order for the product and then completing one pre-assembly at a time, each with it’s own work order number and sheet.

The initial view, on the top right are all products and below are the pre-assemblies

All orders are created by scanning the article number on the order sheet and then indicating the amount in the resulting dialogue. When asked about the amount the workers said that each product was always ordered in the same amount and if you were unsure the amount was stated on the order sheet. Alternatively they can be created by double clicking the article in the list (pictured above on the right). To open a already created order the article in the list to the left are double clicked.

After a blank order is created the work order number is scanned from the sheet and the serial numbers are punched in. The worker explained:

Worker: The first two numbers are the year, 13 since it’s 2013 then the number of the week and the last four are from the sheet.
**Me:** Ok so the first half is that from when you start working or when the order comes in?

**Worker:** It’s from when we start on the order but we change it when we are done. It has to be the week it’s shipped.

A half completed work order (all names have been altered for anonymity). The Product is at the top marked in yellow since it is started but not complete and below all the Pre-Assemblies and Components. Completed Pre-Assemblies are light green and completed Components dark green. Pre-Assemblies with no information are white and components grey.

Each Pre-Assembly has it’s own order view and list of components that are created the same way as the main product order. They are all tied together by scanning the order number for the Pre-Assembly in the main product order view.

At one point the assembly worker stopped, and stated that the label he was about to scan was incorrect and proceeded to enter the batch number manually.

**Me:** What was wrong with the label?

**Worker:** Sometimes when a shipment comes it’s missing the revision and the logistics guys just leaves the revision blank since they don’t know.

When conducting the interview it was apparent that when assembling products the workers were not close to the terminal and they always had the scanner or a component in one hand and the other were holding a tray (pictured on the left in the image of the workstation) and placing the
components in the tray for later assembly. Typing would therefore need to be kept to an absolute minimum and TracyIn needs to handle this situation without resorting to manual input. The revision is a key field in the database table and so has to match what is scanned.

Cultural Model
Following Contextual designs process the contextual interview is the basis for interpretation and the raw data used to make work models [3].

The cultural model focused on the Assembly Worker

We can now more clearly see and be aware of what influences the Assembly Worker in their daily work. The work place of an assembly worker seems quite stressful when looking at the work model. His line manager wants quality while customers wants the orders as soon as possible. And the assembly worker pushes for the logistics department to give them correct information.

The Personas
After the Contextual Interviews and some general interviews resulted in three Personas: Jocke the assembly worker, Olof the Line Manager and Tomas the Quality Control Officer. All these personas are primary personas in the respect that all three require their own interfaces that differ wildly since they have completely different goals.
Jocke the Assembly Worker

Jocke is a part of an almost extinct species here in Europe an assembly worker in a factory, 28 and living in a small town. The factory is one of the big employers around where he grew up and he got his job right out of school. He likes his work, it’s concrete and he feels working with his hands is rewarding and he takes pride in the fact that he does “real work”.

Jocke has almost no training to use a computer, they were not part of the curriculum of his school in the 90’s apart from writing reports on and they were never really a part of his work in the noughties. But Jocke likes tech, not what we would call an early adopter but his home cinema and Blu-ray collection is impressive. He uses technology as a status symbol, somewhat unconsciously, his phone is new, huge and shiny, just like a Rolex watch, something that he doesn’t wear, “you have a watch right there on your screen!” Just like someone who admirers luxury sports cars Jocke knows which are great and what specs they have but has absolutely no idea of how a tiptronic gearbox actually works.

Olof the Line Manager

Olof is 58 and has worked for this production company for a long time and he’s seen a lot of changes over the years and how computers slowly seeped into the manufacturing process. He started as an assembly worker and made the move up to be responsible for old and new co-workers. Olof also has a big new shiny phone but he did not choose it himself, it’s the company issued smartphone. But Olof is not using it as a smartphone he uses it as his old feature phone, he will barely send text messages never mind installing new apps. But just like Jocke, Olof takes great pride in his work and in the department he is responsible for.

Needles to say computers are not a big part of Olof’s life and he has had no formal education and has had no personal reason for engaging in them. He has however worked with the long lived logistics system for almost a decade. Even though he might not take to new systems like a fish to water and doesn’t think that a new system will improve his life he will learn how to accomplish his work and he does not know computers in general but boy does he know that difficult logistics system in and out.

Tomas the Quality Control Officer

Tomas is 43 and has not been working for the production company all that long, he was headhunted not long ago from another company in the area but he has plunged into his new position head first. He is making quality control his baby and is eager to have a tracing system in place.

Tomas is the only one that has any official training regarding computers and he views computers as his day to day too. When Tomas works he is behind a desk interacting with a computer, but as Jocke he doesn’t really understand the inner workings. He mostly works as a businessman does, with spreadsheets and presentations.
Goals

The reason for creating a cultural model was as to more easily define Jocke’s goals. But after reflection we can clearly see that the cultural model more represents Jocke’s needs for others and others needs for him. It doesn’t represent what Jocke is trying to accomplish, looking at Jocke’s persona it’s almost ludicrously obvious what his goal is, apart from the basic all encompassing personal goal of not wanting to be made a fool of and not make a mistake causing production to cease, Jocke want’s to produce. It’s his job but more than that it’s why he feels he is there and what he feels is important about his job. He views the system as an annoyance it’s not really aiding him in any way to accomplish his goal, he knows how to put together these products and doesn’t really understand the great importance of being able to track every single nut and bolt. What Jocke does with the system is important to everyone else but him, it’s just another non-essential part of performing his job just like clocking in every morning or sweeping the assembly room floor. Realizing this the goal of the design is to minimize Jocke’s need to interact with it. If the system was perfect Jocke would not realize it was there and simply go about assembling products as the system cataloging everything for him, but as it stands we still need Jocke to scan the barcodes.

Olof on the other has other goals, he doesn’t scan the barcodes or assemble the products but is responsible for it being done and what he want’s is smooth running, no hiccups, no stops in production and want’s to be kept abreast of what’s going on. Olof has an administrative interface to the system which allows him to change data about products and components. He updates the storage place when they move where they keep a component and he adds a new one when an old is replaced. These are all tasks he performs and they are all necessary but they do nothing to fulfill his goal of smooth running and overview. Olof want’s to know how many orders are being process at one time and for how long an order has been worked on. He want’s to be able to see that production is on track and to know if someone is having trouble with an order. We need to design an interface that meets those goals rather than just the necessary tasks.

Tomas might have the most invested in the system, he is ultimately responsible but he is also the furthest from the system. If you talk to Tomas about it you would get the impression this was the most important thing the company did and the biggest step towards meeting new demands. Demands placed by the customers, this is Tomas goal, he want’s to show his customers and superiors that he is ontop of quality and have precise information at their disposal at any time. He wants to be able to present this and to work with the information if need be in tools that he is familiar with.
Scenarios

Jocke has really only one thing he needs to do create orders and complete them, there are some odd cases switching serial numbers or order numbers between to orders, these are necessary and we can accept that Jocke has to go up to the terminal and use the mouse and keyboard to select and manually alter records, after all this is something he would do only once i a while and he would view this kind of action as abnormal and accept the fact that manual work needed to be done. But creating orders and scanning barcodes must be done without having to resort to manual input.

Olof has the daily task of supervising production and sometimes he has to manually alter data via the administrative interface. We should design an interface that allows Olof to know what’s going on at just a glance. Something he can have open and throw a glance at now and again to see if someone on the floor needs his attention. The seldom performed tasks need to have a lot of helpful information and support him those few times that he does perform them.

Tomas as stated earlier thinks the system is the most important but uses it the least. He only has necessary tasks performed once in awhile, he won’t search batch numbers on a daily basis and when he does he won’t actually work with them in the system, he wants the results and then work with them in a environment he is used to. We should not focus all our effort on designing for Tomas but simply give him the necessary tools and some shortcuts.
Knowing that we had to minimize Jocke’s manual input and interaction with the system the main redesign of TracyIn was to include all orders in a hierarchy to eliminate one having to be created for each Pre-Assembly and switching between these as you worked. Much of the interface was kept however, there was no need to completely revamp the UI, there was nothing wrong with colors indicating what type of component each part was and the information displayed. The problem was in the interaction.

The new hierarchical structure

Now Jocke only has to create one order, he can scan the order numbers for the Pre-Assemblies as he goes along. To make sure he did not have to enter batch numbers manually in case the logistics department had supplied an incorrect barcode the requirements for revision was dropped, this did however result in having to scrap the function of creating a new order by scanning the principal products article number and a button to create orders had to be introduced. In a sense this introduction is a good thing, if you don’t know that scanning a
particular barcode on the order sheet will create a new order, and without instruction how on earth could you know, you would be lost. If you look at the original initial view nothing there tells you how to create an order. If you have missed this information or forgot exactly which of the five barcodes on the order sheet to scan you are completely lost, nothing seems to point you in the right direction and we have already on the first step broken the most important personal goal: I don’t want to look stupid.

When creating an order we, instead of just creating a blank one ask for some basic details required to store the order. We had to get rid of the manual input of the serial numbers which proved easier than initially thought. The system knows what year it is, it knows which week of that year it is and it knows the last serial number used, all information was there yet the user was prompted to collect and input this data manually. Just in case and to make sure we covered all eventualities a manual override was provided.

The new create order dialogue

What Tomas really needed was to get the search results into a format where he could work with it, he was used to searching for information online and then working with it offline. We could have completely reworked the interface to more closely match the process of searching first by one term and then adding additional search terms until the search had narrowed down to relevant results but this would have been overkill. We designed a number of shortcuts but did not try to steer Tomas to using a predefined sequence of searching.

He has an array of search terms but he will most likely start with the serial number, find the component he’s looking for search for all components of that type and the same batch number
to identify which orders contain components from an erroneous batch. If he would have conducted dozens of searches a day then that’s what we would have designed for and left this as an advanced search option to cover all eventualities but he might on average perform one search a week so instead we added shortcuts for entering search terms and adding search terms from a result.

auto complete functions help entering search terms and clicking a term of the search result adds it to the array of search terms

To aid clarity columns can be hidden by unchecking them in the search dialogue but looking through as little as 136 records with 20 on every page is a struggle. We added a function to export the search result to the spreadsheet application Tomas uses every day and is trained and familiar with. The spreadsheet application lets him sort, copy, alter, comment and create graphs and pictures to show his customers which is Tomas final goal. He does not mind going between two applications to get his work done as much as he would mind having to learn how to do all the things he needs one more time and it saves us the trouble of having to create all the functions Tomas already has and knows how to use.
For Olof the main point is to keep him from using TracyIn to keep track of what’s happening on the assembly floor. We designed a overview tab in his admin interface which displayed all orders being processed and their progress. We also added alerts in the form of orders changing color when they have not progressed in a certain amount of time, Olof will by glancing at the screen see something might be wrong and who is processing the order and go to the assembly floor to check why there is a hold up.

Reevaluation

To gain feedback an early version of the prototype for TracyIn was subjected testing by an actual user in a second contextual interview.

When introduced to the new interface the assembly worker immediately understood how to create a new order the function to automate serial numbers however was not intuitively understood and when explained the question followed: “Oh ok, but I don’t see it until I’ve created the order?” An obvious shortcoming of the interface is to not show the serial number so the user can have an opportunity to verify the number with that on the order. The products are also almost always ordered in the same amounts three or four pair depending on the product yet after scanning the article number the user was still forced to punch in the amount manually. All of this data is known by the system but not offered to the user.

TracyIn now supported scanning batch numbers that were missing their revision however a warning had been added in the form of a popup the only way to remove the popup and continue their work the user had to use the mouse to dismiss the warning message. This caused the user to move back and forth from where he was picking up the component to the terminal to dismiss the message. The warning was added with good intent to make sure data was correct and that the user was aware of the inconsistency but clearly this happened too often and the warning was essentially unnecessary.

At one point the user had to retrieve a component from where he could not observe the screen and yelled back: “Did you get that?” Feedback to the user is only visual when a component turns green but the user can not always see the screen so an audible feedback to indicate whether or not an error had occurred should be added.

When the user was processing the order he moved back and forth between the drawers in the stockroom.

Me: Does it matter in what order you scan the components?
Worker: No, not really. I just follow the list on the screen because I can’t remember where each little thing is stored.
Me: And how do you tell them apart?
Worker: You just read the name I only have to look at the article number for the washers since they all look alike.

The list of components had been sorted by Article number and revision since these were the key
fields in the database but in reality for the users these were not as important as their names and storage location which the programmers viewed as just data. If you sorted the list according to storage location the users would not have to walk back and forth but rather work them selfe from one end of the stock room to the other.

**Discussion and Conclusions**

The purpose of this thesis was to design a new more usable interface to the Tracy system but also to compare Goal Oriented and Contextual Design. In the Methodology chapter I discussed their differences and what my approach to the two quite different design methods would be and how I would try to combine them.

My main concern with contextual design is that it relies so much on positivism in that everything can be measured, collected, processed and consolidated into readable charts that are the basis for the design. Not only do I consider the method far too costly in terms of how much time and resources are needed to implement the method but also in the basic assumption of the method as too the fact that good design can be achieved solely by gathering information. If you look at the cultural workmodel of the assembly worker it contains information but go go from that to understanding the assembly workers environment still requires insight and not just information.

Coopers method of design changes utterly how we think about the users. In Contextual Design several interviews are the basis for the workmodels and the user is always an abstracted person based on several interactions and so they are viewed more of an amorphous blob than actual persons. It seems utterly contradictory that creating a fictional user and focusing on that person would be more intuitive and less abstract than actual users but personas mask human idiosyncrasies while focusing your efforts.

Contextual Design does however outline good practices and techniques when conducting interviews which Goal Oriented Design does not. Goal Oriented Design does not really specify at all how to conduct interviews which is still the basis for creating personas why Cooper has omitted any mention of the interviews might be because of placing focus on designing from personas and not from actual people. I feel that context is crucial and contextual interviews are a superb way of gathering initial information but Contextual Design overall has far too much overhead and does not necessarily give you the right insight to the users.
Appendix

Windows Presentation Foundation

Windows presentation Foundation (WPF) is a system for presenting graphical interfaces on windows clients. WPF is based on the .NET platform. The main benefits of WPF is a separation of design elements and logical elements. All controls such as panels buttons and so forth are described in Extensible Application Markup Language or XAML and their behaviour is described in the code behind. This makes for a separation of appearance and behaviour.

The Model-View-ViewModel Pattern

As the name of the pattern suggests it comprises of the division of data, logic and interface into separate entities. This has several advantages such as clarity, maintainability and modularity. As stated this is simply a design pattern it is not connected to any specific technology. This is accomplished by dividing the code into three different categories with different functionality, the Model, the View and the ViewModel.

The model typically encapsulates the data used and is responsible for interfacing with whatever repository hosts the data as well as business logic and should also be responsible for the validation data before storing it.

The View is simply concerned with displaying the data everything the user will see is defined in the View which is typically a window, page or user control. Any logic in the realm of the visible interface should also be encased within the View namespace.

The ViewModel should not reference the view directly however it does expose a certain amount of properties that are used by the view. The ViewModel will typically handle what takes place in the view, any action performed (clicks, mouse movement and so on) is handled by the ViewModel none of the actual functionality should be hosted within the context of the view.
References


[3] Contextual design


