Modeling Context;
A structured review and an industrial case

Master Thesis

Game and Media Technology
Graduate School of Natural Sciences
Utrecht University

Marthe M. Hegeman
Studentnumber: 5678455

Egon L. van den Broek
Wolfgang Hürst

June 2018

Contents

A Word of Thanks

A special thanks to my teacher and supervisor, Egon, for supporting, pressuring and talking things through whenever needed, for giving advice and keeping me motivated and always being quick to respond.

To my fellow data analyst, Thomas, for expanding our knowledge base by exchanging ideas and code, debating and examining together.

To the second reader, Wolfgang, many thanks for the feedback on the preliminary version of the thesis.

To Iwan from Stroomt Interactions (and his colleagues), for brainstorming, enthusing and aiming to get all the ideas on the table.

To Michael from Vanderlande and Merijn from Rabobank, for offering their data, knowledge and time

To my sister, Anna, for being an inspiration, always wanting the best, and forevermore the one that leads the way.

To my boyfriend, Daan, for being a go-getter, laughing about the ups and downs and keeping me positive.

To my parents, Ien and Peter, for loving and being there for me, always.

Executive Summary

Although we all have an intuitive notion of context, operationalizing this notion is strikingly complex. This thesis provides firstly a survey of such attempts, as done throughout the last decades, mainly from the Computer Science - engineering - perspective. This survey is reported in a structured review; hence, it can be replicated. The survey is followed by a case study, using the implementation of context and context-awareness.

Combining several definitions, the stipulative definition of *context* in this research is:

> "Context is any information useful to characterize the state and situation of individual entities and the relationships among them. An entity is any subject (virtual, physical, etc.) that is considered relevant to the interaction between the system and the user, including the user and system themselves."

And the stipulative definition of *context-awareness* becomes:

> "A system is context-aware if it uses context to provide, represent and deliver relevant information and/or services to the user, where relevancy depends on the user's task or situation. This relevant context information must be modeled in such a way that it can be pre-processed after its acquisition from the environment, classified according to the corresponding domain, handled and reasoned about to be provisioned based on the system's requirements, and maintained to support its dynamic evolution. This is done with the greatest possible accuracy."

In the Computer Science domain the categories most used to distinguish context information are:

- user (human),
- space (location),
- time,
- virtual (computational),
- type of activity, and
- devices (hardware).

Next to an appropriate categorization of information to make it usable, the challenges that context-aware systems deal with are that interactions are adaptive and constantly changing, and, thus, so does context. Furthermore, challenges are:

1. to determine what information should be sensed,
2. the integration of existing component technologies (and the large amount of heterogeneous sources), and
3. the design and the development, which requires a lot of engineering work.

From the viewpoint of the user it is challenging to exactly know what the user wants, to know its emotions, as some people do not even know their own at times. Moreover, some contextual factors cannot be easily be represented in a numeric value (e.g., tiredness). The list of requirements for context-awareness systems is long and, in reality, most context-aware systems do not meet all requirements fully.

In the literature the main discussed and implemented context modeling approaches (e.g. data structures for context information) are:

- Key-Value Models,
- Markup Scheme Models,
- Graphical Models,
- Object-Oriented Models,
- Logic-Based Models, and
- Ontology-Based Models.

This list is extended with Machine Learning Models and Meta-model-based Models, as they are deemed relevant approaches, recently receiving a lot of traction. The approaches differ in how well they handle complex information, how easy they are to implement and use, how well they scale, and on many other fronts. In sum, they differ in the way they handle requirements. Choosing the right modeling type depends mostly on the application that it is used for. However, in general the top three of models that score best on all requirements are:

1. Machine Learning,
2. Object-Oriented Models, and
3. Ontology-Based Models.

Context information is inherently related to uncertainties. Therefore, the quality of context is deemed very relevant in the literature, and the three context aspects that can (and should) be taken into account are: data validity, data precision, and whether or not data is up-to-date. There are several parameters that can be measured and that make up the three aspects.

We close the literature survey with a concise conclusion:

1. a variety of techniques exists; but, a lack of standards limits their impact;
2. real-world studies are either limited to a very narrow scope of context, are not mature, or are not replicated; and
3. a structured analysis of the field was largely absent.

The survey provides a remedy for the latter omission and suggests to take a pragmatic approach in model selection and requirement analysis.

Business processes benefit from contextual knowledge as they make clear what actions should be taken at what time. A context-aware system can reason on the data of a certain situation and knows which actions are most efficient and relevant given the situation. This makes the business process more adequate, self-managing, automatic, and demanding minimal administrator's guidance. In other words, the business process becomes more efficient when using a context-aware system.

The literature section followed a top-down approach and says: "This is all we can do! Model this, model that and make it work". However, in practice we are dealing with a certain dataset and this offers restrictions and limitations on what is actually possible, this is the bottom-up approach. In this thesis it becomes clear that these two approaches differ quite a lot from each other.

A model is constructed from data delivered by the company Vanderlande, a company that builds distribution systems for warehouses. Data is used from the factory Onninen in Hyvinkää, Finland, augmented with external, open, real world data, retrieved from several online sources. The factory Onninen is a warehouse, a distribution center, and handles orders including a range of electrical, heat and water, ventilation, air conditioning, and refrigerator products and tools. An automated system manages the production line in the warehouse, on which crates come and go, and a computer shows the operators working on the assembly line which action(s) they should perform.

A practical context-aware model comprises the data acquisition and storage, extension of the dataset with external, open data sources, and, subsequently, computations. The last component in a context-aware system would be the decision module, which takes action (or decides not to do so) based on the data. Such a decision module is not exploited here. The focus is on the information architecture, as it mostly deals with the heterogeneity of context data and the possible influence of external (open world) data. Future research could exploit the decision module, and the effect(s) the actions have on users.

The challenge with the dataset lies in what can be measured from it, as most variables provide information on events handled by the operators at the working stations and the codes that orders and products have. The sensors used are virtual sensors. The main objective is to incorporate external data and combine several variables (heterogeneous data) to see whether or not they add to spanning up an context model. The external parameters that are added to the internal dataset are: weather (e.g., precipitation amount and snow depth), world events (e.g., football matches and crime events), and economic data (e.g., CPI and annual change in %).

Given the dataset's characteristics, there are three factors that make it more difficult to use conventional statistical tests without pre-processing and summarizing the data beforehand: (i) there is a lack of data in general, (ii) we are dealing with a sparse dataset, and (iii) (most) variables in the dataset follow a non-normal distribution. Therefore, for pragmatic reasons, all the signified context factors have been investigated separately. Several statistical analysis showed strong significant effects. The analysis unveiled that world events, certain days of the week, and the *p*erceived workload play a role in worker competency. Dates on which world events occurred showed a lower throughput than days on which no world events occurred. The throughput is significantly higher on Thursdays and very low on weekend days. Production on Wednesdays showed a lot of variance, as a lot of Wednesdays in the days that covered the dataset were (public) holidays in Finland. Furthermore, when there are fewer operators at work on a certain day (i.e., < 7), the average throughput is lower for each operator, indicating that operators handle fewer events when they are confronted with less colleagues also producing output (the perceived workload influences throughput). The factory Onninen can use these findings to adapt their business process.

The amount of impact of the context factors and the interesting things touched upon here benefit from more research to understand and make use of fully: to know how to best act on this information, and if the information is acted upon, to investigate what effect these changes have (and if these are desired). A follow-up project could focus on extending the current context model, including a decision module for adaptations. Furthermore, testing the response of users to this decision module is an interesting aspect to study, especially with regards to user experience (e.g. is the system not too obtrusive?); A context-aware system should aid in the tasks that a user performs and not interrupt or obstruct the user in his or her tasks.

It would also be interesting to extend the dataset with more time, physical environment, user, and social information. As of now, the dataset mainly comprises of variables that are interesting regarding the logistics in the factory. These logistics are especially interesting regarding process optimization. However, extending the dataset with more data puts more focus on the user experience and further benefits the system. Moreover, extending the dataset with more context information and over a longer time period, would enable tests of robustness and substantiate the current results. Additionally, it would be interesting to measure a much wider range of scenarios, in the lab (controlled experiments) and real life (uncontrolled experiments), or both. An example could be playing music in the warehouse to see whether or not this influences worker performance. Also, future research can benefit from an open research culture (e.g., data sharing). Then, we can further extent our knowledge on context its executable model and introduce it to other cases, for process optimization, UX, or otherwise.

Prologue

This thesis is written for the master program Game and Media Technology (GMT) at Utrecht University. The research field of this study program is grouped under the Computer Science domain, and focuses specifically on games and multimedia. Hence it is devoted to the simulation of the real world through the use of different disciplines, e.g. physics, biology, and psychology, and the incorporation of elements such as drama, style, and emotions, with a strong focus on the technical aspects of such simulation and incorporation. The technical aspects involve the use of multi-sensory information to build a digital reflection (i.e. image). The integration of multimedia tools and gaming aspects into our everyday lives heightens the importance of research into this field, and stresses the value of such research.

To effectively research and build multimedia and gaming systems, it is essential to take the human into the loop; Interactions in game and media worlds and outside, the behaviour of human(like) characters, simulations, and sensory aspects, are all related to humans. The interaction between a technical component, in the form of a computer or digital system, and a human component, in the form of a real human (or its characteristics), is what the master program GMT is all about. This interaction is also the subject of this thesis.

The concepts *context* and *context-awareness* are crucial concepts in the study of interactions; The models and measurements used to build *context-aware* systems are very much focused on interactions between humans and technical systems. It is at this interaction that sensors can measure, data is gathered, and situations can be interpreted by a (computational) system. The aim of this research is to make use of *context* and to make systems *context-aware* in order to: improve the User eXperience (UX) and to optimize processes. This is decidedly relevant in the field of Gaming and Media Technology.

The specific name for field of study that deals with interaction is Human-Computer Interaction (HCI). Its central component is the research and development of User eXperience (UX). This is due to a recent trend in HCI that underpins the relevance of human needs that go beyond goal achievement but rather focus on the experience that users have (which go overtime and emanate from the interactions they have) [33].

The case study that will be handled in this thesis focuses on the HCI and UX. It deals with human performance, User eXperience (UX) and process optimization. It uses *context* factors to measure and enhance these aspects in the interaction(s) between human and computer. Such a *context-aware* system requires a "smart" adaptation, which means that the system should be modeled in a humanlike way (or understand what the human user wants and needs). It simulates the real world context into a system, that brings forth a desired interaction. This demonstrates how the case study coincides with the research field of the GMT master program.

1 Introduction

> "The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it" [78]

This quote from Mark Weiser was the start of his paper published in 1991 titled "*The Computer for the Twenty-First Century*". What made the research on this type of *invisible* technology all the more interesting was the transition from the mainframe computer, shared by many users, towards the coming of the internet and the widespread distribution of computers, to lots of computers sharing each of us, termed by Weiser: *ubiquitous computing* [78] [79]. Ubiquitous computing, as Weiser interprets it, means that computation is deeply embedded into our everyday world. It refers to the availability and constant access to information and computational capabilities, which requires the development of new interaction types with as (implicit) goal to assist everyday life and not overwhelm it [2].

Weiser envisioned the transition (from personal) to ubiquitous computing to be set around 2005-2020 [79]. Today, in 2018, we see that the vast amount of (distributed) computers get faster and faster in their computations and bigger and bigger in their storage capacities, allowing increasingly more information gathering and sharing. While remaining quite small in size and thus unobtrusive (to the user), this greater amount of available information leads to a greater knowledge base and thus computer intelligence (as information is knowledge, and knowledge of an event (the gathering and understanding of information) is intelligence).

In light of the trends that the internet, technological advancement and the availability of greater amounts of data (termed Big Data) brought along, *context* is a concept that demanded and received more and more attention. In essence, because a lot of things that humans do in a certain context can be done by a computer if the computer knows the right context. And more importantly, because humans seem to have an insatiable incentive to expand their knowledge and intelligence and reach beyond what was possible before. And what greater way to do this than to use machines that will outperform our brains in computation speed? To be in service of humans, computers can use information on the (internal and external) environment of a user and perform a lot of actions (automatically) that humans used to do themselves. What this means is that computer systems will recognize, i.e. perceive, the real world by using sensors and can react upon stimuli they get from the real world, to act and interact more appropriately [31]. In other words, computers will use context factors to become context-aware and behave in a context-aware manner. It is, in fact, what humans do all the time; Humans use their senses to perceive the real world and react upon stimuli they get from the world.

What Mark Weiser meant when he talked about technologies that weave themselves into everyday life is that technology becomes invisible, in the sense that you do not realize the technology is there, you just use it [78] [31]. In order for that to happen, computers have to implement complex algorithms and be inputted a lot of information. This information is essentially all that can be measured about a certain situation or event, and in effect this is what we call *context*. A context-aware system combines sensory information (acquired from from tracking sensors in mobile phones to EEG-sensors on the human body, and many more sensors) and reasons about it, this makes devices (i.e. computers) “know” which actions (i.e. adaptations) to make (or not make). In [31] Albrecht Schmidt claims that today almost all sensors are available, but that the sense-making process is the most difficult part. For example, humans are very good at distinguishing if a road is slippery due to it being icy; There are certain features in the way an iced road looks that lead a human to conclude that it is slippery. This is an observation that is much more difficult to make for computers (unless it runs a program that is trained in icy road detection, but then still it is a difficult task).

Computers “see” things in a much different way from humans. Computers are good at speed (increasingly so) and cracking numbers, but as the example above shows, they perform less well in distinguishing main from side issues (e.g. making a summary of a text). Additionally, context-aware systems have to deal with noisy and conflicting sensor data. If the complexity of a task (to be performed by a context-aware system) increases, then the context is harder to model, the context-aware system should behave properly in every possible situation otherwise the user can get annoyed, lose interest, and discard the system all together, while its goal was to be woven into the user’s everyday life! The solution that is mostly used to overcome this problem is to build large datasets that allow us to train and fine-tune models. However, as of yet, the issue remains that a context-aware system can not know every possible situation, this problem can be obviated by giving the user control over the situation again, to prevent the system to make unjust assumptions that annoy the user. But of course the aim is to build a solid robust system that is able to behave properly in any, even an unforeseen, situation. So, even if there are vast amounts of data at our disposal, and the possibilities of what to do with this data are thought of as endless, good techniques are needed to actually make that happen. To know which techniques to use where and how to use them, research is needed.

In the Computer Science and Artificial Intelligence domain, a research into the concepts of *context* and *context-awareness* of technical systems that incorporates a thorough literature survey as well as a practical implementation of the findings that arise from its literature study, is still lacking. The outline of this thesis is based on a thorough literature survey and a practical case study. The aim of this setup is to be able to learn from literature and from practice. What the literature learns us is implemented in practice to, in turn, add something to the literature again. More specifically, the aim of this thesis is to provide an overview of the concepts *context* and *context-awareness* in the Computer Science domain of today, and a framework of the possible applications, models and requirements of these concepts. This overview then serves as a foundation to build a practical system and test this system.

In a lot of software systems the complexity increases as they are dealing with services that need to be provided at runtime [10], dealing with imprecise and varied user expectations and needs [57], dealing with greater amounts of data to process, having to be versatile, flexible customizable, configurable etc. [21]. As software systems aid users in the execution of tasks, there is a growing interest in the optimization of these tasks, or in other words: processes. A way to do this is to make such systems self-adaptive, as underpinned in [21], which is another term for context-aware. for companies, target groups, or individuals that deal with (software) systems, it is interesting to adjust the process between the human user and a computer for manifold reasons: for a company it might be to maximize their business strategy, e.g. maximize throughput while keeping costs low. For a target group of the tax office, it might be to respond adequately when somebody reports the death of a relative (a mournful event). For an individual it might be to send personal mails to a person when he or she is done working and wants to enjoy leisure time. What is shows is that the assistance of computer systems that are aware of context can help in a lot of processes and optimizes them (they take less time, effort, or were not possible before). In the case of the tax office it reliefs the employee at the tax office of work, and it delivers the wished response to the person who has just lost a family member. In summary, it brings about a personalization of a process that was not personal before.

Another example is the self-driving car which has an algorithm that encompasses all types of context factors to be able to reason on the situation it is situated in. For this system to behave appropriate in every possible situation (if not, it is dangerous for its passengers), it has to know all possible situations. Or at least, to be able to reason on all possible situations.

According to [12] the Big Data era brings two main dimensions (or problems) for data: heterogeneity and contextual data. They state that data has limited value when it is not paired with its context, and usually internal company data is not connected to other, external, universal data. Furthermore, the data that makes up the *context* of a user is mostly heterogeneous (in source and data type), and thus there is a need for a good way to process and integrate this data to be able to infer something about the *context* of the user [20] [45]. It must be noted that there are other dimensions/problems distinguishable for Big Data: e.g. data capture, storage, searching, sharing, analysis, visualization, inconsistence and incompleteness, scalability, timeliness and data security. For a thorough analysis on these challenges one can consult [18].

In practice it is interesting to make the connection from internal company data to other, external, universal data. And also, to investigate if the processing and integration of several heterogeneous data sources gives interesting deductions. The aim of this research is to provide, next to a literature survey, a practical application that investigates the two problems as posed by [12] mentioned above. This study is limited to these two problems to make it more tangible, delimited and clear, as it serves as an explorative practical implementation and not an end product.

This thesis has the following structure: Section 2 provides the theoretical framework, i.e. a survey of the literature, of the concepts *context* and *context-awareness* in the Computer Science domain, and it proposes a general framework for dealing with building a *context-aware* system. Section 3 embodies the practical part of the thesis, it deals with real world data received from the company Vanderlande together with *open world* data, such as the weather, to see which context factors are interesting in this real world scenario, how to exploit them and which problems are encountered. And lastly, Section 4 encompasses the conclusion, discussion, and ideas for future research.

2 Literature Survey

"What if we can take anything repetitive and make ourselves a hundred times more efficient?"
(Sebastian Thrun, 2017)

2.1 Introduction

The question above is the one that Sebastian Thrun was asked in his interview at the TED2017 in April 2017. He envisions a future where an Artificial Intelligent (AI) system can do a repetitive job (which a lot of tasks are) more efficiently. This is an example of how AI can be applied. A related application is an interactive robot that is able to interact in a humanlike way via various input data. Or a self-driving car that is able to interact in a desireable way to its surroundings. In a nutshell, what these applications have in common is that they have some type of input, they do something with it, and the output is artificially intelligent behaviour. What happens within the black box between input and output are computer calculations. The collection of input data to such a system is what one can call *context*, and - in this regard - we can call such a system *context-aware*.

The meaning of the term *context* is - in its literal notion - "that which goes with a text". A *text* in this notion can refer to a literal text, but also any other message that is delivered by a medium. So all inputs to a context-aware system are messages delivered by a medium. Such media can deliver all sorts of signals: from biosignals to audio signals, and images to sonar signals. A collection of all these signals in a certain situation is called a *context*. And in this definition of the word, it can be used in computers; As signals can be processed, computers can work with these signals and try to *make sense* of them.

In the AI field the perspective on context and context-awareness is historically very much related to text (linguistics, literary theory) [6] [11], logical theories (to provide a formal notion of context) [59] [6] [11] and knowledge representation and reasoning (very much related to philosophy and logics) [11]. As such, research on *context* is mainly theoretical in the AI domain.

Another sphere in the Computer Science domain (where AI is part of) that considers *context* and *context-awareness* is the engineering perspective, which focuses more on - the above mentioned - *computer calculations* and is considered under different names; [11] make a distinction between pervasive computing and communications, ubiquitous computing, the internet of things, ambient intelligence, and intelligent environments. However, such a distinction is arbitrary as the terms are used interchangeably in a lot of literary works. What they have in common is that they focus on the technological implementation of context-aware systems [8], or as explained by [14]: they strive for a formalization of context information modeling and reasoning. The engineering perspective (chosen as a general term to all these notions) has come into being by building practical context-aware systems (or applications) that use an infrastructure composed of sensors, actuators, networks, interfaces and intelligent software to deliver - re-active but also anticipatory - services to its users in a satisfactory manner [11]. From these practical implementations the requirements for context models and reasoning techniques were acquired [14]. So, the engineering perspective on the research of context-aware systems developed itself by empirical study, rather than theory.

The engineering perspective is especially interesting in light of the fact that we are collecting tremendous amounts of data today. This data is collected from all types of sensors and can be transformed and combined to make inferences and reason on this data. Reasoning on this data means that computers are able to deliver, take action and respond to users in multiple situations. Not only does this increase the efficiency of repetitive tasks, it can also - when having an expert context-aware system - make non-repetitive tasks many times more efficient, improve human-computer interaction considerably [51] and enhance the user experience [7] [8].

Today, a study of a system that considers all subjects related to *context* ánd has a good practical implementation, is still missing. The aim of this research study is to provide - from the engineering perspective - a generic model that can gather data, transform it, reason on it (i.e. score it according to relevance), and use it in a middleware layer. From this middleware layer the data is fed to an application that can choose to either react on this data (i.e. change a virtual or physical environment) or decide to do nothing, In this sense this research follows a closed-loop model.

The general human-machine closed-loop as proposed by [70] is used here in an adjusted manner; The altered model is shown in Figure 1. This adaptation to the model is done because [70] focuses specifically on bio-sensors and -signals, which we will not do here, and because what we will call the *middleware layer*, was termed *machine* by [70] and was made up of different components than the ones relevant here. However, the general idea that the signals measured from sensors (virtual and physical) in the human environment are fed to a system (e.g. middleware layer) and that adaptations are then fed back into the human world, is the view that we will also follow in this research.

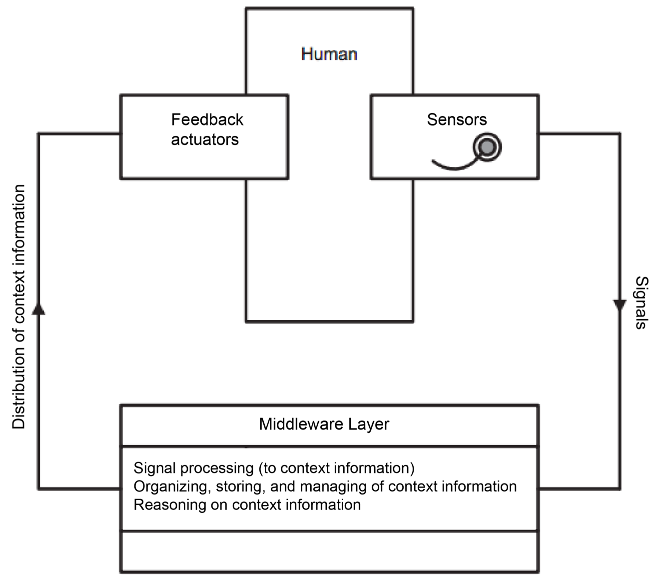


Figure 1: The adapted human-machine closed loop model as proposed by [70].

Firstly, in Section 2.1.1 and Section 2.1.2 the way the studies (articles, journals and books) that are used in the literature study are gathered is set out. Secondly, in Section 2.2 the different context(-aware) views and techniques, as they are used in related work are discussed, which ends with a stipulative definition - the definition of terms as they will be used in the context of this particular research. Section 2.3 and Section 2.4 define respectively data types commonly used and a set of requirements valuable when modeling context. In Section 2.5 eight modeling techniques are discussed. Subsequently, Section 2.6 presents a structured approach on how to choose between these modeling techniques. Finally, this part ends with a conclusion in Section 2.7.

2.1.1 Literature Search Methods

The literature search was performed using Google Scholar, with the focus on articles published in the Artificial Intelligence (AI) domain between 2007 and 2017. Additionally, the Psychology discipline was used as search area for the body of literature; The time-window used for the Psychology discipline has no lower boundary and is set up to 2017. This was done because the articles on *context* in the Psychology domain could be narrowed down more, with the extension of the right search terms particular for this field, and, therefore, could all be included without having the problem of an overload of information, i.e. studies. Trial and error was also used to try out certain search terms, because the different terms used for the same concepts in different research fields could not be known in advance.

In Table 1, the different search terms together with their number of articles found (*Amount*) and from this the number of useful articles found (*Useful*) is shown. It was chosen not to include theses to the search, therefor for both disciplines the negation of the term *thesis* is used. Furthermore, for the *Artificial Intelligence* discipline the negation of the term *home* is used. This was done to exclude studies that focus on the development of smart homes, since we do not wish to focus on that subject in this study. For both disciplines the search terms *context dependent OR context aware OR context-based* is used, and the focus on the discipline (*artificial intelligence OR AI*, *psychology*). Also, for both disciplines a few search terms that are specific for their domain are used, this is: *techniques*, *user*, *data*, and *model* for *Artificial Intelligence*, and *emotion OR mood OR mental state*, *ontology OR mental model*, *behavior OR behaviour*, and *task OR activity* for *Psychology.* As you can see the focus in the AI domain lies more on models that use data and certain techniques; The AI domain is more computational focused. While the focus in the Psychology domain is more on mood or emotion, and the performance of, and behaviour with regards to, tasks and activities, i.e. the interaction of humans within their context.

Table 1: Search terms for the different disciplines (Artificial Intelligence, Psychology) and the amount of (useful) articles found, using *Google Scholar*.

Discipline	**Search terms**	**Amount**	**Useful**
Artificial Intelligence	-(negation) thesis AND -(negation) home AND context dependent OR context aware OR context-based AND techniques AND user AND data AND model AND context modeling OR context modelling AND artificial intelligence OR AI AND intitle: context	340 (since 2007, without citations and patents)	...
Psychology	-(negation) thesis AND context dependent OR context aware OR context-based AND emotion OR mood OR mental state AND ontology OR mental model AND behavior OR behaviour AND task OR activity AND psychology AND intitle: context	119 (total, without citations and patents)	...

After gathering all the studies, a selection was made based on the reading of the abstract of the article and/or a short scan of the book, article, journal (etc.). Then the selection was based on so called *inclusi/exclusi* criteria; If the study under review (i) did not have *context* as main topic (e.g. the term context was rather used to put another topic into its context), (ii) was research from a totally different domain than listed above, (iii) was focused too much on a side topic , (iv) was too specific (e.g. an understanding of too much very specific domain knowledge was required), or (v) originated from an unknown or unreliable source, it was not included in the body of literature used for this research report. That is why the number of *count* differs from that of *useful* in Table 1.

2.1.2 Scoring the relevant articles

In Table 2 you can see an overview of how the articles that are selected are rated. They get rated by an number between 1 and 3 for a category ranging from D_1 to D_α. Their end summation (Σ) determines their total score. The first n articles with the highest scores are deemed most important and are read thoroughly and included in the Literature Study. The articles after that are not included.

Table 2: Example of article overview with their ratings according to the categories D_1 to D_α and their sum Σ (overall rating).

Article	D_1	D_2	D_3	$\cdots$	D_α	Σ
Article 1	$\{1,2,3\}$	$\{1,2,3\}$	$\{1,2,3\}$	$\{1,2,3\}$	$\{1,2,3\}$	$[5,15]$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Article n	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

The criteria on which the articles are rated (D_1 to D_α), are in this case: (D_1): Theory, (D_2): Real cases, (D_3): Computational model, (D_4): Empirical, and (D_5): Type of Data.

It must be noted that whenever it was necessary, certain concepts, ideas models etc. were studied additionally, in order to obtain a better understanding or global picture of those elements. But this was only done if the selected body of literature was deemed, by the author and her supervisor, not extensive enough and this shortcoming influenced the understandability and readability of the literature review.

2.2 A Definition

In the domain of Computer Science (where AI is a part of) there is no general agreed upon definition of *context*. Traditionally, there was a focus on *location* as the most important dimension of context [25]. However, it quickly became evident that context is rather a process in which users take part [81]. The pursuit of a good definition of *context* is mostly done in studies on interactive systems [41]. An interactive system can be a product or service, and it can be digital or analogue (or both). It can be an explicit system, notable by the user, or it can be 'pervasive', meaning that the presence of the system is masked from the user [41]. Consequently, the interaction with the system can be explicit and direct, or interaction situations are 'sensed' by the pervasive system without [37], or with minimal [77], direct user interaction with the system. A third possibility is a combination of the two (e.g. a system that has both direct and indirect interaction components).

The word itself, *context*, stems from the Latin *con* meaning *with* or *together*, and *texere* meaning *to weave*. So a *context* is, according to [15], not just a design or a blueprint, but an active operation that deals with "the way humans weave their experience within their whole environment, to give it meaning". Context greatly influences the way humans and machines act, interact and interpret things; A change in context changes the lived experience [15]. This notion of context assumes that context is an abstract process [81], while other authors underline that context is never all-encompassing but rather defined relative to a concrete situation [81]; *Context* "defines an individual's current prevailing state and as such it is inherently complex and domain specific" [44]. In other words, *context* is cause and application specific, which compels the identification of functions and properties typical for each domain [41] (or application) [17]).

In [50] the definition of *context* by Fischer is given as: "the 'right' information, at the 'right' time, in the 'right' place, in the 'right' way to the 'right' person". While in [13] *context* is defined with regards to how it is used; Context is used to model and describe the environment wherein a given product or service is to be deployed and executed. This complies with the rather abstract definition of [81] who assert that context can be defined as all things that "surround a user or device and give meaning to something". This last definition is still abstract because what does "give meaning to something" entail? Another possible definition of context with a more technological focus is posed in [25] cited from Schmidt: "A context describes a situation and the environment a device or user is in. A context is identified by a unique name. For each context, a set of features is relevant. For each relevant feature a range of values is determined by the context". This can be extended with the notion of [43] who believe that "almost any information available at the time of an interaction can be viewed as contextual information". So, the relevant features are determined by a certain time and interaction component. However, by far the most recited definition of *context* in AI (but often used in other disciplines as well) that tries to encompass all the above-mentioned ideas, is the one posed by Abowd et al. [56][74][43][52][12][27][4]:

> "Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves." [1]

This definition is extended by [41] with the claim that an object can refer to physical or computational objects. In the definition of Abowd et al. *context* is described as a set of situations and actions that are determined by dynamic and frequent change, where the states of the involved entities are a part of [81].

According to [56] there are two categories of context distinguishable: a) the context of the user (that requires information), and b) context of the information itself. The definition of Abowd et al. is according to [56] the definition for the user's context, while the counterpart of this user context, the information context, can indicate important features of the information itself (e.g. location of a supermarket), or it can label the information. This labeling is done according to the context of the user requesting it (e.g. a nice restaurant for people that enjoy Thai food). [77] also underpin this distinction when they claim that context can be retrieved from human users (e.g. emotions, preferences, social interactions, activity) or real world entities, i.e. things (e.g. sensed contexts, descriptions of RFID-tagged things). The system senses the current state of these entities and reacts autonomously to events (changes in states of humans and things) [77].

The boundaries of what *context* entails exactly are not evident. This is underpinned by [60], who state that the above-mentioned definition of Abowd et al. of *context* is not specific enough and is applicable to (almost) any application, thus it needs an upper and lower bound to be able to use it when building applications (or systems). [60] therefor state that, following the definition from Hartmann et al. (2008): *context* defines a certain situation in which the application is used. The situation is established with the usage of information that distinguishes the actual utilization from others. This is done by looking in particular properties of the user (location, task at hand etc.) and examining physical or virtual objects (noise level, nearby resources etc.). Only information that can be processed by the application at hand is referred to as *context* (relevant information), but that is not essential for it's normal functionality (auxiliary information). According to [81] *context* cannot be defined or determined beforehand, since its scope is dynamic, changes often and is unpredictable; *Context* arises in the course of action and is is not just "existent", it is an outcome rather than a premise.

With all these notions there is no consensus on what context is, however according to [81] there is consensus on what context is about: "context is concerned with an evolving, structured, and shared information space that is designed and utilized to serve a particular purpose".

[74] state that to make full use of the possibility of making services and interactions smart, a operational definition of context should lead the way into the identification of relevant features of the context, so that it context modeling and context management can be exploited and the corresponding requirements can be set up to build the actual context-aware system. Furthermore [74] claim that there are three aspects of context that are most important in the building of context-aware systems (i.e. making a context definition operational): (1) the interaction between user and system and the information that characterizes this interaction or situation of relevant subject to it, (2) the categorization of the design space of context models i.e. the models required to present such information, and (3) the dynamic nature of context information and the management and change necessary across its life cycle i.e. contextual framework. This last aspect does not only involve the change of the individual situation, but also the new information that emerges due to the interaction among different subjects [74].

In accordance with their viewpoint [74] make use of the following (extended) operational definition:

> "Context is any information useful to characterize the state of individual entities and the relationships among them. An entity is any subject which can affect the behavior of the system and/or its interaction with the user. This context information must be modeled in such a way that it can be pre-processed after its acquisition from the environment, classified according to the corresponding domain, handled to be provisioned based on the system's requirements, and maintained to support its dynamic evolution." [74]

The next sections will explicate the three aspects proposed by [74] further. But, first, the next paragraph deepens the notion of *context-awareness*.

2.2.1 A Stipulative Definition

The starting point for a definition of the concepts *context* and *context-awareness* is, as in many other studies, the definition of the concepts as posed by Abowd et al. [1], starting with *context:*

> "Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves." [1]

As already mentioned, this definition is extended with the comment that objects can refer to virtual and physical objects. Secondly, it is extended with the idea that the situation of an entity is dynamic and thus should be considered as "the current state of an entity". This then influences the definition of *context-awareness*, in that current and also past context(s) should be used to reason on relevant information and services.

The viewpoint of [74], as mentioned earlier, is also interesting as an extension. In their definition *context* is - primarily - "any information useful to characterize the state of individual entities and the relationships among them", where *state* and *relationships* are what was termed *situation* by [1]. However, these three terms together provide an even better clarification of what *context* entails, and so all three are incorporated here.

The next sentence in the definition of [74] is: "An entity is any subject which can affect the behavior of the system and/or its interaction with the user", this is a broader definition than the one posed by [1], stating that "an entity can be a person, place or object" (virtual or physical, as already established). This broader definition of *any subject* will be used here, since it does not put any restriction on what the possible categories for *subject* are, and this might be desirable in certain situations or systems (e.g. an *event* can also be an entity, and there are entities that cross a line between being object or person, for example a robot or an other system). However, the second part of the sentence by [74] states that such subjects "can affect the behavior of the system and/or its interaction with the user". What is missing here, is what [1] did involve: the inclusion of the user and the system itself. Furthermore, the emphasize in [74] is on *affect*, while it is one *relevance* in [1]. To go into depth on the semiotics of these two terms is beyond the scope of this research, however the term *relevance* is deemed more accurate here, since it is a bit broader and less focused on an effect or change but rather a connection (between subject and system).

What [74] then add to the definition is very relevant for the employment of *context* in Computer Science or AI:

> "This context information [*the information on the state and relationships of entities*] must be modeled in such a way that it can be pre-processed after its acquisition from the environment, classified according to the corresponding domain, handled to be provisioned based on the system's requirements, and maintained to support its dynamic evolution."

This definition nicely delineates the application of *context*. What can be added for clarity is that - mainly in Computer Science and AI - the *reasoning* on the context information is important. However, the application of *context* refers more to context-aware systems and thus shall be integrated in the definition of *context-awareness*, rather than the definition of *context*. The stipulative definition of *context* in this research then becomes:

> "Context is any information useful to characterize the state and situation of individual entities and the relationships among them. An entity is any subject (virtual, physical, etc.) that is considered relevant to the interaction between the system and the user, including the user and system themselves."

Then the term *context-awareness*, this is explained by [1] as:

> "A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task." [1]

As stated this definition can be extended with:

> "This context information must be modeled in such a way that it can be pre-processed after its acquisition from the environment, classified according to the corresponding domain, handled and reasoned about to be provisioned based on the system's requirements, and maintained to support its dynamic evolution."

And also with the idea that context-awareness frameworks supports "meaningful data representation, delivery of service and reaction" [62].

Furthermore, the set of relevant (*context*) features is identified and used to describe the user's task or situation with the greatest possible accuracy [81].

In total the stipulative definition of *context-awareness* becomes:

> "A system is context-aware if it uses context to provide, represent and deliver relevant information and/or services to the user, where relevancy depends on the user's task or situation. This relevant context information must be modeled in such a way that it can be pre-processed after its acquisition from the environment, classified according to the corresponding domain, handled and reasoned about to be provisioned based on the system's requirements, and maintained to support its dynamic evolution. This is done with the greatest possible accuracy."

2.3 Data types

To control and manage context information in context-aware systems it is useful to characterize context information types and management categories [74]. Contextual information is context data that is processed into information that is useful in context processing. It is a central part of context-aware applications and systems. It can be said that contextual information describes the context and thus creates the context definition. When we take a look at the definition of *context* from Abowd et al., we see that entities include (1) places, (2) people, and (3) things (physical or computational objects) [44]. The context of any entity can be described with the usage of context dimensions (i.e. context parameters [12] or context attributes), which consist of context properties and their *Literal Values*) [44]).

Context information helps the interaction between user task completion and the system by acquiring situation-specific information [15]. The current state, i.e. situation, of an entity can be described by using context information that observes and characterizes this current state. Examples of information sources are sensors, persons, places, networks, smart gadgets, RFID information and other objects. The context information can be operationalized in several ways, and helps in the modeling of complex (or easy) situations [54].

Which context properties to describe, depends on the point of view that is used to describe the context. One way to go about this is to use the point of view as it is perceived by the user, another way is the application point of view, where the user then is part of the context [15].

An interaction context helps us to describe context properties that are relevant to a specific user's interactions with the system [32]. When this interactions are regarded from the viewpoint of the user then we wish to have a personalization of information. This means that the context-aware system is able to adapt to a users' context [24]. However, as stated by [32] a generic interaction model should be posed that makes use of "common characteristics of contextual factors regardless of their specific manifestations".

Context information can be formulated along several axes. From static to dynamic, internal to external, nonvolatile to volatile, non-transient to transient, and many more [74]. The dimensions for classification vary from domain to domain, and application to application.

Context properties can be classified into general dimensions. An overview of these dimensions in (a selection of) the literature can be seen in table 3. What is most important when viewing the table is that the determined context dimensions are dependent on the application for which context is modeled; The type of application influences the context dimensions to be perceived and modeled.

Table 3: Possible context categorisations from literature.

Reference	**Year of Publication**	**Context Dimension**
Bolchini and Bellatreche [15]	2007	space, time, absolute/relative space and time, history, subject, user profile
Moore et al. [41]	2008	spatio-temporal (physical situation), personal, mobile and computational device(s), infrastructure and connectivity constraints, resource(s), tasks, social setting
Faraone et al [26]	2010	location, identity of people near the user, objects around, time, season, temperature, physical status, conceptual status, emotional status
Yu et al. [80]	2010	user, physical, computational
Villegas et al. [74]	2010	individuality, time, location, activity and relational (i.e., social, functional and compositional relationships among the first four types)
Agrawal et al. [3]	2012	user, type of activity and domain, social, spatial, temporal, resources available, devices and interfaces
De Lourdes et al. [22]	2014	physical, social, computational
Jovanovic et al. [32]	2014	user, interaction, devices, environment

Continued on next page

Table 3 – *Continued from previous page*

Reference	Year of Publication	Context Dimension
Barkat et al. [12]	2015	location, time, user, identity, activity, computational, physical, history, networking, things (this is a summation of previous works)
Moore and Van Pham [44]	2015	identity, location, status (or activity), time
Esseynew et al. [25]	2016	computational, user, physical, time
Surve et al. [62]	2017	computational, environment, history, social networking, spatial (profiled user identity, location, time), activity based, sensor based and cognitive

Initially, studies in the field of Computer Science mainly focused on the context dimensions *location* and *time*. However, this notion is enriched by many studies and can include many more context dimensions [44][12]. These extended lists of context dimensions are mainly caused by the different possible perspectives to look at *context (information)* [12]. It must be noted however that contexts have infinite dimensions and cannot be described completely [39]. So, any aim at a categorisation of context is a try and never an universal solution. Furthermore, it is important to know that different papers address *context* and *contextual information* differently. For example in [46] *context information* differs from a user profile, interests and preferences, while e.g. in [19], [41] and [81] the user context (where user profile is part of) is part of the overall contextual information. A possible reason for this distinction is the difficulty in obtaining the user context; the *why* and *how* of a user's activity is hard to obtain [19]. Here, we will follow the assumption of (among others) [81] that *contextual information* is "any information that may be used for describing the situation the user or a device is currently operating in".

Context Information Elements, Dimensions and Features. When we take a look at the dimensions of context as posed by [25], with reference to Schilit and Theimer, in table 3, we see that the discerned context dimensions are *computing context*, *user context*, *physical context* and *time context*. The context properties that belong to *computing context* are claimed by [25] to be: "network connectivity, communication costs, communication bandwidth, nearby resources such as printers, displays, and workstations". Furthermore, user context consists of the properties: "user's profile, location, people nearby, the current social situation", *physical context* has the context properties: "lighting, noise levels, traffic conditions, and temperature", and *time context* has the context properties: "time of a day, week, month, and year". However, these context dimensions are not the only ones, as you can see in table 3, different authors and studies have different ways to define context (dimensions) and categorise context properties. Furthermore, each context category can have several context properties that belong to that dimension. When we take an overall look at table 3, it is notable that a lot of the studies use the same context dimensions. And even if they do not use the same terminology, there are some dimensions that overlap with others or that comprise or are a part of other dimensions. According to [60], who follow the study of Jun-Zhao and Sauvola, "context entities (i.e. a person, place or physical or computational object [41]) can be structured into three domains: the user domain, the computer domain and the environment domain". While [80] and [17] state that context information can be distinguished in physical context information, user context information and computing context information. These two notions are the same when we denote *the environment domain* as being equal to *physical context information*. In an application the three domains are interacting with each other and this delivers context data, which is the totality of context data for each entity. As the entities are devised in domains the context information for each entity can be represented more easily [60]. These entities can have different interactions and situations at different times and thus the application or system should be adaptive and should be able to handle an evolving context. This means that a model should also take into consideration the adaptation of the data and system [46]. Adaptation in the system can be done passively or actively. In a passive system the system monitors, continuously or discretely, updates the system and proposes the user with appropriate options for her actions. Whereas in an active system, the monitoring is the same, but the option selection is done autonomously by the system, it acts proactively on the profiled user preferences [62].

A context-aware system should be able, in our case, to both sense and react, and thus provide services based on context information. The total context information space is termed *context world* [80], where "the formal semantics of the goal and services and the facts are stored and updated dynamically" [80]. Context elements are the signals that the application needs to start functioning in order to become aware of the context [60]. According to [41] these context elements are: (1) spatio-temporal, (2) personal, (3) device(s), (4) infrastructure and connectivity constraints, and (5) resource(s). Furthermore, [41] state that there are three context dimensions: (1) spatio-temporal, (2) identity, and (3) activity. From these three primary context dimensions several secondary contextual information can be distracted. Secondary contextual information comprehends: "property values that describe context factors, such as: location, device characteristics and identity information" [41]. It must be noted that primary and secondary contextual information can also be interpreted slightly differently, as in [62]: primary context information is "retrieved directly without performing any kind of sensor data fusion operations". While secondary context information is "processed or computed using primary context elements by using sensor data fusion operations or data retrieval operations". In short, there are different ways to subdivide context (e.g. into context elements, domains, context dimensions etc.). And the one proposed by [41] is definitely not the only one, as you can also see in table 3.

Additionally, a distinction can be made between internal and external properties, or features, where "internal features are used to describe characteristics that exist inside the entity or its domain and the external features are those which describe the context information that can be retrieved from the interaction of an entity with other entities" [60]. So the user identity and preferences are typically internal properties. Another way of looking at internal and external parameters stems from [12] following the idea of Pitoura et al.: "(1) Internal context parameters that concern attributes stored in the database and (2) External context parameters that involve attributes outside the database". And in [4] the idea of *internal* and *external* parameters is also explicated. Their idea is that context can be categorized as: internal, external or boundary context. Or more generally, into general and independent domain context (open world) or specific domain context (closed world). And proposed in [4] is that context consists of six contextual classes:

1. User: information about the user's profile, situation, and preferences (internal context);
2. Activity: describes the different sets of activities that can be developed in the context (external context);
3. Time: describes the notion of time in the context, which can be used to define the chronological situations in the context (external context);
4. Device: describes the device of software and hardware in the context (external context);
5. Services: describes the characteristics of the service required by the user, Quality of Service (boundary context), and;
6. Location: describes the location of the context, its indoor and outdoor space, and the property of the environment (external context).

Another categorization is proposed by [32] who make a distinction between two categories – simple and complex. Simple context parameters can be internal or external when we take the viewpoint of the user. Internal context parameters are then the factors used to model users given a certain situation i.e. the contextual factors described from a users (human) perspective. External context parameters are the factors that describe environment and device properties that influence the situation. The complex parameters are then a combination of these internal and external context parameters [32].

This complexity is underpinned by [12] who state that context information is not just a set of parameters, there is a dependency relationship "between contextualized attributes whose values depends on context and contextualizing attributes (also called context parameters) whose values impact on the values of the contextualized attributes". For example, the age of a person depends on the time context; Then *time* is a context parameter, whilst *age* is a contextualized attribute. Or in other words: the value of time impacts on the value of age.

What we undeniably can distinct from the literature is that there is such a thing as user, personal, or identity context class. Secondly, there is a technological or computational component, e.g. a system and/or device (physical or virtual) that gathers and provides information, and that has a possible interaction with a user. Thirdly, there is a space and time component, arguably combined. Fourthly, there is an interaction, situation, activity, and/or status to be observed. And lastly, there is a social and/or relational component. These general components are still quite abstract.

With the aim of encompassing a lot of components in their context classification [74] propose an extensive framework, see fig. 2.

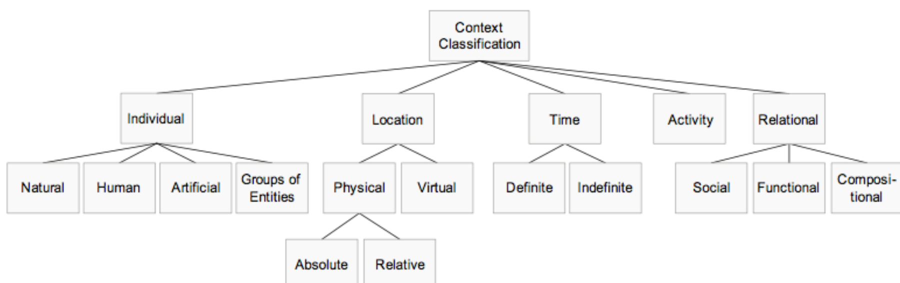


Figure 2: The classification of context information as posed in [74].

The first category is *individual* context. Individual context information encompasses "anything that can be observed about an isolated subject (i.e., the state of the subject)" [74]. The subject can have more than one individual context if it plays more roles. The sub-elements of individual context are natural, human, artificial, or groups of entities. Natural context is associated to living and non-living entities that are not directly the outcome of a human activity, for example: the weather. Human context refers to user behaviour and preferences, such as user profiles, and captures the way he or she interacts with the system. Artificial context describes the status of entities from human activity or technical processes. Examples are hardware and software components and their deployment. With the context of groups of entities the context elements that emerge from the common characteristics shared by a group are meant. This does not necessarily mean that the entities in this group interact with each other. Very important for this aspect is that membership to groups may emerge dynamically at run-time. Examples of such dynamic interactions are: "social interests, computing power, cultural background, software architectures, and network topologies" [74].

The second category is the *location*. This location can be physical or virtual. Examples of physical location are exact location (GPS coordinates) (absolute), places, and distances (relative). Examples of virtual location are IP-address or mouse-clicks on a website.

The third category is *time*. Since not all interactions happen at once, the time component is very important. From [66] it follows that the historical context is very important when modeling context. This is underpinned by [15] as well, as they state that the current context state could depend on previous ones. Examples of the time category are not only date and time, but also holidays, working days and meeting schedules [74][35]. Besides, the historical component of all contextual information can be very important in a context-aware system; The time component can be used for inferring secondary context (via context reasoning), for example user preferences might be different at different times of a day, relating these two can then increase the level of context-awareness. Another example is deciding of future behaviour based on past experience in the same type of activity (a recommender system is a good example of this). The definite time component represents time frames with a definite duration (begin and endpoints). While the indefinite time component represents a recurrent event that happens while another situation is taking place, its duration is not know in advance. In real life, interactions mostly take place concurrently and their duration is, generally, not known in advance.

The fourth category is *activity*. This factor represents future, current and past goals, as well as actions and tasks of an object.

Lastly, all context categories are related to each other (or can be related to each other) based on the three relational subcategories: (1) social, (2) functional, and (3) compositional. The dependencies of context information instances should be modeled and operationalized. Social context describes the interrelation among individual users and groups of entities (e.g. connections, colleagues etc.). Functional context describes the use that an object can make of another (e.g. personalized information on a website). Compositional context is the aggregation and association of context factors. The authors stress that a combination of the context categories in fig. 2 can produce new context information and enrich the body of context information. These combinations are based on "social, functional, or compositional relations among relevant objects for the interaction between users and systems" [74].

The set out ideas from [74] are, as said, extended. However, their explanation of the *individual* context category is not that extensive and comprehensible, and needs further explanation.

The *natural* factor can be understood as the ultimate *external* parameter. These are the elements that might influence the system and user but are not a part of it. Great examples are: the content of news broadcasting (negative or positive with regards to certain aspects related to the user or system), the weather, and so called "world events" like Christmas or Hanukkah, or a football match (these factors are related to culture, country and religion).

The *human* factor encompasses the modeling of the human user; The user behavior can be monitored, the preferences, user profile (age, gender, occupation etc.), expectations, but also emotions. The user profile aims to identify individual users and accommodate users' evolving preferences [41]. In [48] belief, desire, intention (BDI) logic is mentioned as a way of inferring human emotions. According to [51] the user context can be determined along four dimensions: "physical context, user activity, health and preferences". Furthermore, they underpin that user context changes dynamically. [44] stresses that *emotion*, or emotional response, is also an important element in describing individuals. Examples of contextual information on emotional response are: (1) the physical and social situation, (2) spatio-temporal data, (3) physiological information, and (4) cognitive and abstract information [44]. Cognitive and abstract information is said to include e.g. "emotional responses, intuition, feelings, and sensibilities expressed in terms of linguistic and semantic terminologies" [44]. However, [44] state that the results of incorporating human emotion as context factor remain limited to date (2015). A way to overcome this according to them is to implement context processing with an Open World Assumption (OWA) (a lack of knowledge does not mean that something is untrue), instead of a Closed World Assumption in context processing (everything that is not proven to be true is false) [44]. Furthermore, [43] deeply discuss eLearning applications and how user characteristics are modelled in such applications. Even though our focus is not in eLearning applications, their findings are interesting; They state that there are different levels of observation and computation of users possible:

- Individual – outlines the user characteristics of a specific individual user and assumes every user to be different;
- Stereotypical – assumes that any user can be classified into several stereotypes. This assumption makes personalization or adaptation strategies easier to implement since just a couple of stereotypes need to be distinguished and determined for each user;
- Role based – intends to base user characteristics (individual or stereotypical) on the role of a user in an e.g. company, society, or activity etc., and infers user characteristics based on it;
- Group based – this approach is considered to overcome conflicting recommendation or adaptation of a context-aware system when interaction or situations take place involving a group of users.

The main form of modeling is that one of the user levels is used in combination with an expert, by using an overlay model (this provides more options or paths in a linked network) [44].

The *artificial* factor consists of the technological and/or device components, its hardware and software components and its status, current and past.

The *groups of entities* factor describes the common characteristics of entities. For example, users with the same cultural background respond the same in a given situation. As said, most of the parameters of this factor emerge dynamically at run-time. Following the example: we can not know beforehand that people with the same cultural background will respond in the same way, but we measure this, and thus can adapt the system to this observation. Another example could be the measurement that a people with an age above seventy will respond slower to pop-ups in a certain interface.

Static and Dynamic Context. Furthermore, contextual information can be *static* or *dynamic* [81][41][38]. *Static* context information is e.g. a person's birth date. A static context describes a state that is user-driven, as in the user decides which content to provide to the system. *Dynamic* context information is e.g. a person's location [81].

Another explanation of the terms *static* and *dynamic* context is given by [41]. They state that a dynamic context describes a state where the user is passive, or less in control. The system monitors and analyses the user situated role and actions. According to [41] these two types of context reflect the two major ways in which context is used: "(1) as a retrieval clue (a static context) and (2) to tailor system behaviour to match users' system usage patterns (a dynamic context)". This idea corresponds to the distinction is made by [81] who state that contextual information can be distinguished in how it is acquired: (1) Explicitly acquired contextual information is information given by the user, it can be information such as established social relationships or fields of interest. (2) Implicit acquired contextual information is acquired via physical sensors and hardware. They "capture specific aspects of the surrounding context by using sensing technologies or by monitoring user and system behavior" [81].

[81] further explains that static context descriptions are unable to deal with unknown context information at run time. They do, however, require interaction between different context vocabularies to be particularised at design time. A fundamental requirement therefor is the ability of context-aware systems to dynamically handle and integrate new context information in existing structures.

Context Information Sources. Context information (i.e. data) is obtained from various, heterogeneous sources with variable quality [54] i.e. input sources to the context-aware system [51]; The technologies to obtain the input data, as well as the type of input data, differ. According to [22] the types of context information that are modelled in the literature differ in terms of persistence, level of abstraction, content and quality. Therefore, it is very important to manage and interpret *context* data, i.e. information, correctly [51].

How contextual information is acquired is a critical issue when modeling context [49]. It can be very hard to obtain contextual information directly, and therefor sometimes contextual information is inferred from existing uncontextual data. What is meant is that context data is not always provided directly (e.g. a user lets a company know that she moved), but can be inferred from multiple data sources (e.g. the location and time of day, from which is inferred that the user has moved) [49]. It is important that the data obtained from the multitude of sources that make up a context element, are reasoned about in the right way, especially in mobile contexts [51][24].

According to [46] and [19] lot of studies put the focus on sensor data when talking about acquiring *context* information. What is interesting is the variance in types of sensors. [41] report that the multitude of sources from which contextual information can be obtained are, for example, sensors. They then give examples of sensors as being: sensors that monitor computer networks or status sensors for human users or computing devices. So, the sensor sources physical as well as abstract [45]. This idea is supported in [81], where context information is said to be distinguishable between: "virtual or real-word aspects that have a specific relationship to the current task at hand". So, on the one hand there are the *physical sensors* i.e. real world aspects [81], these are, for example, the sensors that can be found in a mobile smartphone, including: accelerometer, gyroscope, light sensor, temperature sensor, camera etc. [51]. And on the other hand, there are also the so called *virtual sensors* i.e. virtual aspects [81]. These are, for example, user profile information (i.e. identity), (clicking behaviour on) web services [51], user preferences and interests [45] etc. or it could be the system state.

In [22] they follow the assertion of Henricksen et al. that assess four classes of context information are the most useful: integrated sensed data, static, user-supplied and derived. Integrated sensed data is processed raw sensor data (i.e. data unprocessed and retrieved directly from the data source); Processed raw sensor data means that it is checked for consistency and meta data is added [62]. The data is dynamic (run-time) and, for example, collected from user interaction with the system [24]. Static data is, as the word says, information that is fixed, non-dynamic [22]. It can be collected, for example, from software repositories or other static information sources [24]. User-supplied data is e.g. social context, emotions, and activities, it does not change much over time but validity is an important aspect to consider. Level of abstraction are the different layers of semantic context interpretation. The lowest level is the sensor data, followed by reusable semantic interpretation of low level sensor data, and situation and relationships as the highest level of semantic abstractions. Situations can be learned from data or provided by the user [22].

The different data types are according to [22] distinguishable in: types of sensor data (binary, numeric and featured values), domain knowledge, and different relationships between situations. So, when modeling context information it can be "extended as set of interrelated events by means of logical and timing relations among them", these events can be continuous or discrete [62]. Furthermore, context information can vary in quality since it is dynamic and heterogeneous; "Context information can be out of date, imprecise, incomplete and contradictory with other information" [22].

In summary, context information differences in persistence, level of abstraction, content and quality must be taken into account when trying to match context requirements and context information [22].

2.4 Requirements

With context categorized and data types identified, the requirements to model context need to be determined. This section aims to do so and, in parallel, identifies remaining challenges.

The literature describes quite some challenges for context-aware systems. Firstly, as interactions are adaptive and constantly changing, so does context. [43] state that a context can be highly dynamic and must reflects a users current dynamic state, which can vary from moment to moment. And since context is required at run time [58], this also entails some technical challenges. Secondly, understanding which information should be sensed or determined, which information is deemed relevant, is not always straightforward, as well as the granularity of the information [29]. Thirdly, the design and development of context-aware systems still requires significant engineering work, since there is no general development method determined [58]. This might be due to the different interpretations in the field of what exactly *context* is. Fourthly, and arguably most importantly, "the development of intelligent agents implemented in intelligent systems" [41]. And lastly, the integration of existing component technologies [41].

From the viewpoint of the user it remains challenging to implement a good service provision based on relevance and in accordance with users' situated roles, and meeting the expectations, needs, objectives, beliefs, desires and intentions of users [41]. These last aspects are challenging if only because they exist within the user i.e. they are internal context information. A major challenge to overcome when dealing with systems that model context information and users' mental attitudes is that there is (or can be) an immense gap between the system behavior and user's expectation [25].

Even though context information seems very accessible, it remains a challenge to take advantage of the large amount of open and heterogeneous knowledge collected from different resources [22], and to make this information reusable and shareable across different applications [24]. Flexible solutions that deal with new types of context information are needed [22]. Furthermore, [22] state that, following thought of Euzenat et al., a context information management framework must be be open, dynamic and minimal:

1. Open: An open framework allows for new devices and applications to participate in the interoperability process. This requires accepted standards to represent context information, as well as possible extensions and the evolution of context information.
2. Dynamic: In order to deal with heterogeneity in devices, applications and context representation at runtime the framework needs to be dynamic and adaptable. To match established representations with the new ones, semantic web technologies can be used.
3. Minimal: The computational resources should be kept to a minimum, with as few intervention of the application and device developers as possible. This is important to maintain clearness and openness.

In such a dynamic environment, with changing users, devices and sensors with unknown capabilities, a very flexible context model is needed that allows sharing and reusing of information [22]. It is also necessary to constantly check the validity of previously integrated elements, so that computational costs stay at a minimum; As [22] state: "information and devices frequently become old and out of date".

[24] state that collecting context information and integrating context-awareness at the application level is expensive. Therefore, reusing and sharing of context

information across applications is deemed necessary, and this must be done from the beginning of the development cycle. This requires that well-defined, shareable and reusable context models are at hand. It furthermore requires that context can be integrated across application and domain boundaries.

Another limitation posed in [47] is that a lot of context-aware applications and systems today are build by designers, and also the modeling of context information is done by the hand of the designer. However, [47] foresee that this might cause a fail in the system if a context is encountered that the designer did not anticipate. Therefore self-learning and self-adapting methods can be implemented. These methods make use of an iterative approach when searching for the best possible action if the system is confronted with an unknown context. The downfall in this approach is that, when there are a lot of actions to evaluate, it takes a lot of time for the system to find the best action and this delays the adaption time of the system. However, in [47] it is claimed that it also takes a lot of time for a database administrator (DBA) (i.e. the person to respond to an unforeseen context in the old scenario) to find the best action and respond. In this case the solution to the problem can be incorporated in the system and this will help future situations.

Furthermore, some contextual attributes cannot be satisfactorily represented as a numeric value, e.g. tiredness, or affection [36]. In order to be able to model mind-states, so called BDI-models are used, based on belief, desire and intention as mental states and actions [48].

Obtaining high-level context data, that really says something about an activity or interaction, remains challenging; Some sensors just display information without defining or interpreting what it means in terms of high-level context. Another example of high-level context is obtaining the user's current task. Because of sensor uncertainty this also remains a challenge. Some solutions are to check the user's calender for extra information, use sensor data fusion, or rules or machine learning [51].

[29] state that there are seven issues to be considered when modeling and building context-aware frameworks or applications. Those issues are:

- Privacy and Protection of Personal Data. This is an important issue when regarding the collection of large amounts of context data; Contextual data could be sensitive e.g. when private documents or interaction histories are included. This information can be abused, misinterpreted, or even sold to marketing agencies. So methods that protect users' privacy need to be incorporated into the system and acceptable trade-offs might need to be assessed.
- Efficient Instrumentation. To retrieve data, several (physical and virtual) sensors need to be instrumented. If we want the context-aware system to be unobtrusive, then the user's activities should not be interrupted. The context-aware frameworks should then be able to work independently of the user's workspace, so that the system is independent of their domain. Therefore it should be explored which sensors to use and how to integrate them into underlying frameworks, GUI libraries, operating systems,

middleware, and execution environments.

- Heterogeneity. The challenges regarding this aspect have already been discussed. What [29] underpin however is that when building a context-aware framework, context should be regarded independently from the application and the usage domain. A universal model should be build with an abstract notion of context (information), so that the model can be used in different scenarios. Therefore it is necessary to investigate what should be observed and what should not. Also privacy plays an important role here when including data into the model.
- Scalability. Models and processing techniques should be able to handle large amounts of data. It depends on the application how much context information will be extracted and then processed and modeled.
- Richness and Quality of Information. This aspect refers to the variation in data provision over time; The richness and quality of context information will vary and context models should be able to adapt to these changes.
- Incompleteness and Ambiguity. Information acquired from raw sensor data can be ambiguous and incomplete. A possible solution to incomplete data is their interpolation on a instance level.
- Level of Formality. When there is a consensus and there are protocols in place it makes it much easier and quicker for developers to build effective context-aware applications and systems. This shared understanding is achieved by, for starters, represent contextual facts and their interrelationships in a precise and traceable manner.

The main requirement for building a context-aware system is to have a good model [42]. An overview of the context modeling requirements from the literature, in different domains, is shown in table 4.

Table 4: Context modeling core requirements.

Requirement	**Description**	**Source**
Scalability; Extendability	The system or model deals with highly unstable, changing environments, where context information can change, increase, be enriched, added, adapted etc. Therefor the model needs to be extendable and scalable. This also is important to make it usable for future technologies and standards, and to be able to implement new scenarios.	[61] [40] [50] [29] [28] [44] [25]

Continued on next page

Table 4 – from previous page

Requirement	Description	Source
Security and Privacy	The system or model should be secure and compliant with privacy regulations. Critical information should be protected; Information identifying a specific user should be made anonymous and the user should authorize the monitoring of his/her interaction on the different applications.	[61] [40] [29] [28] [44]
Heterogeneity	The system or model should be applicable to versatile and heterogeneous environments, and make use of different types of context information and a wide range of devices.	[40] [13] [29] [24]
Timeliness	The system or model should be real- or near real-time for a dynamic service configuration and execution. It furthermore should support asynchronous and ad-hoc communication; Data production and consumption should be possible at different times (time decoupling), and sinks and sources do not have to know each other (space decoupling); in other words communication should be asynchronous and anonymous among context producers and consumers. And lastly, it should use context history as an information source.	[61] [15] [74] [40] [13] [24]
Adaptive	The system (or model) should be adaptive to context changes. Contexts are dynamical and changing in nature, therefore the system should support context-triggered action. The context system should observe which changes occur to which properties of which entities, and adapt to these differing contexts.	[61] [60] [13] [50] [29]

Continued on next page

Table 4 – from previous page

Requirement	Description	Source
Reasoning; Decision Support; Inference	The model (or system) should be able to interpret context data. Furthermore, it should be able to aggregate and infer on the data, check for consistency, context adaptation and new context information inference. The approach has to give a solution for making the semantics of manipulated data explicit.	[15] [74] [29] [50] [12] [44] [24] [25]
Quality; Imperfection	The system might encounter incomplete, contradictory or uncertain data. It should thus support the measuring of the quality of the data and manage it in the right way.	[42] [15] [74] [13] [29] [28] [24]
Incompleteness; Fault-Tolerance; Ambiguity	The context data has to be unambiguously interpretable even in heterogeneous systems. If the system does perceive ambiguous, incoherent or incomplete context information, it should reason on the possibility to interpolate or mediate the context information somehow and construct a reasonable context.	[42] [61] [15] [50] [29] [28] [44] [25]
Generic	Systems or models should not be domain specific, and therefor a closed system, but rather an open system that could be used to address a wide range of problems and applications in different domains. It should be as general as possible and should be able to tackle as many problems as possible (e.g. multi-context modeling).	[15] [50] [28] [12] [25]
Learning	The system (or model) should be designed for evolution and should provide the means to evolve through the input of users, as well as by itself. Furthermore, the ability to infer new facts from acquired data or pre-existing world knowledge is desirable; This can be done by e.g. observing the user behavior, individual experiences of past interactions with others, or the environment.	[15] [61] [50]

Continued on next page

Table 4 – from previous page

Requirement	Description	Source
Efficiency; Performance	Access to data has to be efficient; Fast access to large amounts of context information and data objects should be supported (when necessary). Furthermore, the user should not face any performance problems while doing his or her activities.	[29] [28] [24] [25]
Aggregation	In order to be able to reason about data, high-level data should be aggregated from low-level data, This means that low-level data gathered from context sensors is aggregated into higher-level units of information.	[29] [50] [74]
Relations	Relations between context entities should be modeled, to be able to aggregate more accurately and also to make collected information accessible to other components. Representing relationships and dependencies helps in the expressiveness and support for reasoning for the model (or system).	[74] [40] [50] [29] [24]
Formality	There is a need for a description of context information and interrelationships in a precise and traceable manner, this is necessary to be able to - for all entities involved - share a common understanding and interpretation of the contextual data exchanged. The level of formality is then the existence of a formal definition and whether the formalization well expresses the intuition. The type of formalism is the class of the conceptual tool used to capture the context (key-value-, mark-up scheme-, logic-, graph-, ontology-based). [42] [15] [29]	

Continued on next page

Table 4 – from previous page

Requirement	Description	Source
Context Management	On a meta-level the management of context is important, which entails: context acquisition, classification, modeling, handling, exploitation, maintenance and evolution. The system should acquire information by means of sensing events and actions and transforming raw sensor data to be usable in the model. A context model is needed to represent the characteristics of the context at different levels of detail (granularity), furthermore it represents the entities and situations of a context. The modeling of context information should be done at a conceptual level to keep independence from any specific implementation. Depending on the field of application, there are significant differences in the processing and modeling approaches. The handling and exploitation of context involve several techniques and the derivation of context facts, but also context sharing and actions that the system should undertake. Maintenance and evolution involve context structuring, data management, scalability etc. It is noteworthy to mention the requirement for context management, however it is less relevant here as this table mostly discusses specific requirements of a model.	[15] [74] [29] [50] [12] [28] [25]

Continued on next page

Table 4 – from previous page

Requirement	Description	Source
Distributed; Flexible	The system (or model) should be distributed and flexible in order to perform context related operations within dedicated context information domains (source, transport, distribution, consumer, etc.). The model should easily adapt to different contexts, it can however be application-domain bounded if it is specifically focused on a single domain. In most context-aware systems the context is gathered from a set of partners that reach an an agreement about the description of the current context at run-time, this means that it is distributed and has a decentralized architecture. The characteristics of a distributed system should be supported.	[42] [61] [15] [61] [40] [29] [44] [25]
Technologies; Lightweight	The system (and model) need to be interoperable over the whole ecosystem including the networks, supporting Information Technology (IT) nodes, storage and a myriad of customers' devices. The technological part should be lightweight, meaning that the infrastructure of the system (or model) poses as few and low requirements as possible on soft- and hardware components. Furthermore, platform independence is required. The approach has to offer dedicated techniques to exploit the proposed model, management overhead should be as low as possible and elicited context should be persistently stored in order to enable reproducing and understanding errors.	[74] [40] [50] [29] [44] [28] [12] [25] [24]

Continued on next page

Table 4 – from previous page

Requirement	Description	Source
Usability; Re-usability; Natural-Projection	Real world concepts should be mapped, i.e. naturally projected, onto the context model. This mapping should be simple, direct, immediate and comprehensible. Furthermore, context information can be manipulated at run-time. Additionally, the context information should be re-usable, meaning that the same context data can be used for different context information types, but also historical context information can be re-used and combined with new knowledge.	[42] [29] [50] [28] [12] [24]
Reliability; Robust systems	The system and model should be reliable, e.g. deemed trustworthy and not prone to errors.	[50] [28] [44] [25]
Validation	The model (or system) should have the ability to validate partial or full data content. This is important given the potential for errors in defining contextual relationships between entities and context acquisition and usage.	[42] [74] [24]
Constraints; Preference Compliance	Constraints should be satisfied and the user preferences (which can also be constraints to the system) should be guaranteed.	[15] [44]
Visibility	The system should focus on the need of the user and not on the system itself, therefore the system itself should be invisible. It is desired that the system performs its activities in the background and does not require input from the user, however this is not always possible.	[13] [50] [29]
Delivery	Choosing the right moment to deliver information is important. It may involve prioritizing between interruptions and user discretion, in other words: it should be context-sensitive.	[61] [50] [25]

The concepts mentioned in table 4 are requirements and criteria to context-aware systems and their modeling approaches. Their applicability depends on what exactly we wish to model, and so for a context-aware applications not all of the criteria might be relevant.

2.5 Modeling

Why is it that we are modeling context and what exactly do we want to model? In reference to the section on context information, section 2.3: “a context definition is made up from sub-contexts that describe and define entities” [42]. Context modeling approaches then aim to define the sub-contexts of the involved entities and the relationships between these contexts. All of this is done with the ultimate goal to effectively implement and make use of context [42].

A context model is thus a way of representing the context we want to utilize and operate [49]. Context modeling has been applied a lot in research studies, especially over the past decade. Models generally make an abstraction and conceptualization of different (sub-)contexts (or systems, or applications) and collect these into a unified model. Different models have different abilities and competencies [24], and according to [24] these they differ in their "expressiveness, usability, interoperability, and support for specific application domains".

In many research fields, such as artificial intelligence, ubiquitous computing, or ontologies, contexts are represented by context models and used to organize and structure knowledge [16]. In the AI domain, a lot of research shows the same list of context modeling approaches (while some extend this list a bit); An overview of these context models is shown in table 5.

Table 5: The most commonly used context model types from the body of literature.

Model	**Reference**
Key-Value Models (KVM)	[42][41][60][74][13][37][77][29][24][25][62]
Markup Scheme Models (MSM)	[42][41][60][74][13][37][77][29][24][25][62]
Graphical Models (GM)	[42][41][60][74][37][77][29][24][25][62]
Object-Oriented (OO) Models (OOM)	[42][41][60][74][13][37][77][29][24][25][62]
Logic-Based Models (LBM)	[42][41][60][74][13][37][77][29][24][25][62]
Ontology-Based Models (OBM)	[42][41][60][74][13][37][77][29][24][25][62]
Machine Learning Models (MLM)	[42][41]
Meta-model-based Models	[74]

The models ranging from Key-Value Models to Ontology-Based Models all follow a bottom-up approach. This means that the context features are modelled before the contextual information is gathered, and the gathered information will fall into the categories as set up beforehand. While Machine Learning Models, on the other side, follow a top-down approach, meaning that from the gathered contextual information context features are extracted. In this case the features are not known beforehand, but rather found by means of clustering or grouping techniques; The features and model come into being after data processing instead of antecedently; Features are detected. A totally different type of context model in this aspect are the Meta-model-based Models. This type of model is rather a set of guidelines for different model types than that it is a model itself. In the next sections a summary of the most important aspects of the different models is explained.

2.5.1 Key-Value Models (KVM)

The Key-Value Model approach is the most simple, basic data structure possible for modeling context [42][60][77] and that is why it has some popularity [13]. In the approach, Key-Value pairs represent context information as a list of attributes [74], or in other words: they are used to describe the capabilities of e.g. a service [60]. A *key* is then the attribute name, and the *value* represents the value corresponding to this attribute [74][13]. Different formats for Key-Value pairs are possible, such as text files or binary files [62]. The values are provided to the application environment variables [25].

According to [60] an object's (or non-object) properties (or attributes) are accessed in an indirect way, since strings are used to identify them. They state that a more direct way would be to use an accessor method or access them directly through instance variables. The mechanism of KVMs can use matching algorithms for easier lookup [25].

The disadvantage of the KVM approach is its inability to model sophisticated, complex information [77] and efficient context retrieval [25]. The approach also lacks capabilities for structuring context data, and mechanisms to check data validity [13]. And according to [62] the model does not scale well. There can be reasons to use the KVM approach however, as is underpinned by [29], who state that KVM's are lightweight and relatively efficient. Furthermore, they are useful for building prototypes (who still are sensitive to a lot of change) and to run on devices with low computing power and storage [29]. Also, they are easy to manage [25]. If the model is not applied for the system's architecture, it can still be useful for the representation of context data and context meta data [13][29] or to tackle simple and easy to manage small amounts of data [62].

2.5.2 Markup Scheme Models (MSM)

In a Markup Scheme Model (MSM) a hierarchical data structure is used [60] [74][13] which consists of (markup) tags, attributes and content (i.e. values) [42][60][74][13][25]. A markup language is the representation of a Markup Scheme Model and it is a combination of text and additional descriptive information. Additional descriptive information can be e.g. the presentation structure for the text, and the markup is then intertwined with the primary text [42]. The most well-known markup languages are the Hypertext Markup Language (HTML) [42] and the Extensible Markup Language (XML) [42][74][25], which fall, among others, under the umbrella of the Standard Markup Language (SGML): the super class of all the markup languages [60].

The content of the markup tags is (typically) defined recursively by other markup tags in a nested structure [42][60]. Mostly, the markup schemes are used to collect information for user profiles. In this research the attributes would be context parameters, for example the context parameter "User Name" could then have the value "James Jones" (e.g. name="James Jones") [42].

The advantages of MSM's are that they can handle heterogeneity and incompleteness [25], context data can be validated by means of validation tools such as XML-schemas, and data can be structured via nested XML structures [13], and, thus, MSM's allow the expression of complex relations. Furthermore, MSM's allow for an efficient data retrieval [62].

An example of complex relations modeling is the usage of profiles. Such profiles are based on a serialization of a derivative of Standard Generic Markup Language (SGML) [25]. Examples are: Composite Capabilities/Preference Profile (CC/PP), Friend of Friend (FOAF)[77], and User Agent Profile (UAProf) (Open Mobile Alliance) [60]. These models are based on Resource Description Framework Schema (RDF-S) syntax [25].

The main disadvantage of MSM's is that they do not allow reasoning; There is a lack of design specifications, which means that context modeling, data retrieval, interoperability, and re-usability over different markup schemes can be difficult [62]. Furthermore, MSM's lack expressive structure, have weak formalism, are inadequate for capturing context information, relationships, dependencies, timeliness, and quality of context information [25]. This means that, just as KVM's, there is a lack of support for more sophisticated demands, while at the same time MSM's are easy to implement and understand, and flexible.

2.5.3 Graphical Model (GM)

Graphical models are used to model context with relationships [62]. Contextual information is represented using graph data structures and richer data types [25]. According to [42] there are two approaches when it comes to the context modeling of GM's: 1) Diagrammatic modeling, using e.g. Unified Modeling Language (UML) diagrams [25][62], and 2) Entity relationship diagrams (models) (ERD).

Fact-based Models are mentioned by [74], but also defined in [77] and [25]. Object-Role Modeling (ORM) is a part of to Fact-Based Models, and also a graphical modeling approach. In fact, ORMs are a context extension of Graphical Models proposed by Henricksen et al. [25]. ORMs arose because of the necessity to use formal expressive models that support query processing and efficient reasoning [74][25].

The advantages of GM's are that they are more expressive than Key-Value and Markup Scheme Models, as relationships are captured into the model [25].

Furthermore, they have a good balance between expressive power and efficient reasoning, have a good support for software engineering [77], and easy to learn and use (since it is well-known).

However, the disadvantages of GM's are their lack of support for hierarchical context description [77], lack of formalism for online automated access, and lack of support for distributed context modeling [25].

2.5.4 Object-Oriented (OO) Models (OOM)

Object-Oriented Models (OOMs) are based on the object-oriented approach, where the model is able to handle the representation of dynamic characteristics of context [74]. The benefits of object-oriented programming are *encapsulation* and *reusability* [42][60][13][25].

Context parameters are defined by classes and have related access functionalities. Type-checking and data validity can be done at both compile- and run-time, and Quality of Context (QoC) components can be mapped to other objects [13].

Context processing consists of the encapsulation of contextual information as the possible states of an object. These can be accessed and adapted by calling the related methods of the object [77] e.g. by the usage of specified interfaces [42][60][25]. As this encapsulation of the details of context processing is done on an object level, this is hidden from other components of the system (or service) [60][25].

The most important advantage of OOMs is that the interactions between context data and the services (e.g. context-aware systems) are easy, because the same abstractions as in object-oriented programming languages are used and this simplifies the implementation of context handling code [13]. Also, raw sensor data is processed in a satisfying way to be able to infer high-level context information from it [29]. Furthermore, the OOM approach is flexible [29]. However, high computation power is required to be able to handle the complexity of the object-oriented context model, this might not be supported by low-end hand-held devices [25]. Furthermore, the OOM approach does not support built-in reasoning capabilities, and the validation of the object-oriented design remain a challenge, in that there is a lack of standards and specifications [62].

2.5.5 Logic-Based Models (LBM)

The field of *logic* is concerned with deriving an expression or fact(s) from a set of expressions or facts, by means of a reasoning or inference process [42].

Systems based on Logic-Based Models (LBMs) use inference rules as reasoning techniques. Mostly, a context is defined using a set of facts (context parameters/properties), and relationships and constraints, are defined by expressions and rules [42][77][25]. New contextual information can flexibly be added, adapted or removed (managed) in the system in terms of facts [42][60].

New knowledge (facts) can be derived by inference [13] i.e. a reasoning process to derive new facts based on current rules in the system [25].

Logic-based systems have a high level of formality [42][60][74][77][25], this is their biggest advantage. LBMs also provide a lot of expressivity (this quality is intrinsic to the logic formalism[13]) compared to the other models [13][62][25].

Generally, adequate formalisms are used to establish inference rules. However, they do not offer simple functionalities to deal with data validity [13]. Validation can be provided, but "associated rules are not straightforward to specify and depend on the adopted type of logic" [13]. Reasoning is possible up to a certain level [62].

In LBMs heterogeneity and incompleteness are still lacking [25]. [62] state that a lack of standardization diminishes the re-usability and applicability. Furthermore, [29] state that LBMs are more heavyweight than, for example, Key-Value or Markup Models.

2.5.6 Ontology-Based Models (OBM)

The previous mentioned modeling approaches can, according to [42], all be seen as precursors to Ontology-Based Models (OBMs), since OBMs make use of many of the properties that are used in the other modeling approaches. [41] underpin this by stating that the modeling categories are not independent, but have an element of interdependence. OBMs are great ways to structure and model relationships and constraints between entities.

The ontology modeling approach is a representation way to specify concepts (i.e. contextual information) and their relationships [42][60][77][62], with the usage of semantic technologies [62]. Contextual information is modeled using ontologies by representing the contextual information in a machine readable form in a data structure [42]. Examples of a possible data structures (i.e. standards; semantic technologies; representation schemas) are Resource Description Framework (RDF), Resource Description Framework Schema (RDF/S) and Web Ontology Language (OWL) [42][24] [62].

Ontologies are widely adopted [13]; Several development tools and reasoning engines are available [62]. This capacitates the reuse of previous works, data, and the establishment of common and shared domain vocabularies [13]. In addition, a domain specific ontology can, for example, be combined with a more generic one [13], or (in another example) the ontology of the model's core concepts can be combined with domain specific concepts [24]. These examples support the claim that ontologies are very suited to model complex relationships and systems.

Ontologies are very appropriate when one wants to map every-day knowledge "within a data structure easily usable and manageable automatically" [13]. Ontologies have a high formal expressiveness and specification of context parameters and relations between context entities in a particular domain [60][74][25]. Therefore, ontologies are able to specify and express complex relationships in the model [13]. Furthermore, OBMs make use of powerful reasoning techniques [60], which can be applied per requirement [62], knowledge sharing and reusing capabilities [77], and they are able to handle heterogeneity [25]. Data validity is (usually) guaranteed through different strategies: the requiring of ontology constraints [13], data-type validation, consistency checking and range specifications [24].

Even though OBMs seem very useful in modeling context for context-aware systems, their biggest disadvantage is (just as for LBMs) the memory and CPU power they require. Because in some applications they should be able to run on resource-constrained mobile devices [13], which do not have enough memory and CPU power. Also context retrieval can be computationally intensive and time consuming as the amount of data increases [25][62]. This can be a reason not to adopt these models. Furthermore, in [77] and [25] it is claimed that OBMs are not beneficial in the representation of context that is e.g. incomplete, uncertain, unavailable, noisy or has a temporal nature [77]. There are, however, examples of research that tries to overcome these drawbacks of OBMs, such as a mix of an LBM with an OBM. In these type of models, it is possible to reason about uncertainty [77].

2.5.7 Machine Learning Model (MLM)

The Machine Learning Model is introduced by [42]. Even though the authors state that it is not necessarily a context model approach, still it has the same objectives as the other context modeling approaches (dealing with a lot of information and technological limitations (e.g. of devices)). The claim of the authors is that, following the ideas of Flanagan (2003), ontologies do not deal thoroughly with real world context(s). They then claim that Unsupervised Machine Learning provides a better resolve for an effective personalized service provision. Machine Learning refers to the study of algorithms that improve automatically through experience [9].

In machine learning a training sample is fed to a learning algorithm. The learning algorithm then builds a model - it categorizes the data in a certain way - and for this it needs enough training sample examples in order to gain in accuracy. Depending on the task at hand the meaning of 'enough examples' differs [76]. There is a distinction between supervised and unsupervised machine learning approaches; For the supervised machine learning approach the input is a labeled data-set, where the input data is unlabeled for unsupervised machine learning [76]. In unsupervised learning a structure is found without knowing the 'right' labels. In effect what the algorithm does is cluster, e.g. find useful classes or features (or labels) [34], based on the data-set and as such it is not possible to know if these labels are correct. This is however, when dealing with real-world data, the most straightforward approach because the data gathered from the (virtual or physical) context is often not labeled. However, it might be possible to infer labels from the data by means of implicit feedback for example. Or use a small set of labeled data to infer the labels for the bigger data-set [76].

According to [42] the utilization and operation of context is performed to assist in making systems that are personalized, based on a user's profile (termed context). And, as mentioned, the authors claim that the unsupervised machine learning approach is preferred to enable effective personalized service provision. The focus in [76] is on user modeling using the machine learning approach, and they claim that the objectives for user models can be: "to describe (1) the cognitive processes that underlie the user's actions, (2) the differences between the user's skills and expert skills, (3) the user's behavioral patterns or preferences, or (4) the user's characteristics". The conclude their paper with the observation that in order to exploit effective user modeling with the usage of machine learning, there is a need for large data-sets, labeled data, concept drift (attributes that characterize a user are very dynamic and subject to change, learning algorithms should adjust to these changes quickly), and computational complexity [76].

As the ideas posed by [76] are specified for user modeling, they are also applicable to other parts of the context. Since, in general, context is time dependent, dynamic and there is a need for detecting useful categorization of features. Therefore, the (unsupervised) machine learning approach can be very useful.

2.5.8 Meta-model-based Models

The notion of Meta-model-based Models is introduced by [74]. It differs from the context models discussed thus far, as it is not a model in itself, but rather guides the dynamic generation of new model instances, based on the meta-model, defined at design time. The outline of the meta-model resulted from the context information and context management objectives that are deemed necessary for the interactions in the application scenario that is to be modeled. The features that are included in this outline are "the representation of granularity levels, different context types, properties and constraints of context entities, relationships among context entities, and spatial and scope representation" [74]. The modeling approach can thus be adapted in accordance with what is needed in a certain scenario.

The context meta-model features are described in fig. 3. Since *model* is another word for a representation, the figure depicts the important attributes, i.e. features, for the representations of entities and situations in a certain interaction (context). Useful models must support the detection of states to switch between situations and contexts and trigger actions [74].

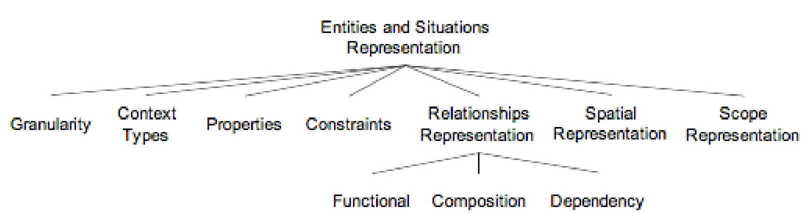


Figure 3: The features

The first feature from fig. 3 is *granularity*. This feature relates to the different levels of detail that contextual data can have. These possible granularity levels affect the abstraction and inferal of context information. Sometimes a certain granularity level is required to be able to obtain a certain goal, e.g. "services for providing functionality based on nearby facilities require finest levels of granularity" [74]

The context types and property features entail the ability of models for representation of raw context data after it is transformed (preprocessed). A model can represent data in different ways, and the way this data is expressed depends on the data type and properties of the data. Moreover, the context model should be able to handle different types of contextual information and manage the information depending on its type [74].

The constraints feature is very important in modeling and understanding context entities and relationships among them. It is important that context modeling supports verification of the model and context reasoning techniques. Also there is a need for dealing with inconsistent, uncertain and ambiguous data; Thus, constraints are necessary to verify consistency [74].

The representation of relationships among context entities supports the development of new context facts from existing contextual information [74].

And finally, spatial and scope representation help in quality-based reasoning, management, and provisioning. The reasoning space should have boundaries to be able to make use of it (it should not be too extensive). And scope representation is important to enforce privacy of context information [74].

2.6 Model selection

To select the context model of choice, we propose to execute three phases: i) model scoring, ii) model ranking, and iii) quality control. In Section 2.4, we defined the requirements for a context model. In the next section, we use these requirements to score each model type. Subsequently, using the model scores, we rank the model types. Last, we discuss how to assess the models' quality.

2.6.1 Scoring the Modeling Approaches

An analysis of how the context models from sec:viewstech score in meeting certain requirements, is done by [42]. Such an analysis is crucial in getting a better understanding on which modeling approach is good at what. The overview of [42] is shown in tab:scoredmodels.

Modelling Approaches		Requirements					
		dc	pc	qua	inc	for	app
1	Key-Value Models	$-$	$-$	$-$	$-$	$-$	$+$
2	Markup Scheme Models	$+$	$+$	$-$	$-$	$+$	$++$
3	Graphical Models	$--$	$-$	$+$	$-$	$+$	$+$
4	Object-Oriented Models	$++$	$+$	$+$	$+$	$+$	$+$
5	Logic Based Models	$++$	$-$	$-$	$-$	$++$	$-$
6	Ontology Based Models	$++$	$++$	$+$	$+$	$++$	$+$
7	Machine Learning	$+$	$+$	$-$	$++$	$++$	$--$

Table 6: The parametric evaluation matrix as proposed by [42].

To understand tab:scoredmodels an explanation of the abbreviated requirements is needful:

- dc = Distributed Composition;
- pv = Partial Validation;
- qua = Richness and Quality of Performance;
- mc = Incompleteness and Ambiguity;
- for = Level of Formality;
- app = Applicability to Existing Environments (adaptability);

So what [42] did was provide a list of requirements for context models, then look at the existing modeling approaches, and score each context modeling approach according to the set out requirements. In their evaluation matrix (tab:scoredmodels) the scores consist of a set of four values: $\{--, -, +, ++\}$.

2.6.2 Extending the Requirements

A requirements list which tries to encompass all requirements from AI literature is given in Section 2.4 and more specifically in Table 4. In short, it consists of the following requirements:

- Scalability; Extendability
- Security & Privacy
- Heterogeneity
- Timeliness
- Adaptive
- Reasoning; Decision Support; Inference
- Quality; Imperfection
- Incompleteness; Fault-Tolerance; Ambiguity
- Generic
- Learning
- Efficiency; Performance

- Aggregation
- Relations
- Formality
- Context Management
- Distributed; Flexible
- Technologies; Lightweight
- Usability; Re-usability; Natural-Projection
- Reliability; Robust systems
- Validation
- Constraints; Preference Compliance
- Visibility
- Delivery

First of all, there are some requirements that overlap or are the same between [42] and Table 4, these are represented in Table 7.

Table 7: Relation between the requirements listed in Table 4 and those of Moore et al. [42].

Moore et al. [42]	**Table 4**
Distributed Composition (dc): "pervasive context-aware computing is (generally) implemented in dynamic distributed systems, often in ad-hoc networks. It must accommodate these characteristics."	Distributed; Flexible
Partial Validation (PV): "given the potential for errors in defining contextual relationships between entities a desirable characteristic of any context modeling approach is the ability to validate and partially validate contextual knowledge on a structural level as well as on instance level against a context model as a result of distributed composition."	Validation
Richness and Quality of Performance (QuA): "sensor derived data is variable (often continuously variable) as is the quality captured from a diverse range of sensor types, context modeling approaches must therefore inherently support quality and richness indication."	Quality; Imperfection
Incompleteness and Ambiguity (Inc): "contextual information may suffer from incompleteness and ambiguity. Context models must incorporate the capability to handle this issue by interpolation of incomplete data on an instance level."	Tolerance; Ambiguity
Level of Formality (for): "the description of (contextual) facts and interrelationships in a precise and traceable manner represents a significant challenge. It is desirable therefore that (in an interactive scenario) each party shares a common understanding and interpretation of the contextual data exchanged."	Formality Relations
Applicability to Existing Environments (adaptability) (App): "it is important that a context model is adaptable to enable use in existing domains, systems and infrastructure (such as ad-hoc networks and Web Services) that utilize contextual information to personalism service provision and match users in co-operative computing."	Scalability; Extendability Adaptive

Second, a few of the requirements from Table 4 are very general and will be met - or can be met - by any context model discussed. These general requirements are:

- Security & Privacy: this is a requirement that should be complied whichever context model is incorporated.

- Context Management: for every system a good management is necessary. This is however a meta-requirement and thus we do not incorporate it here.
- Reliability; Robust systems: the system and model should be trustworthy and not prone to errors. This is true for all context models.
- Visibility: this requirements is also true for all context models. The application can be made to run on the background or actively interact with the user. For most context-aware systems the former will be preferred. Regardless of the context model, the system can be made invisible to the user.
- Generic: this requirement states that models should not be domain specific, but open to different types of applications. Since the context models provided are all generic in principle, but can be adjusted to a specific domain, this requirement is termed to be too confusing to score and is left out.
- Delivery: delivery is a requirement that comes after the choosing of the model. When a model is chosen, then it becomes possible to look at how the information gained from the model - the context - can be delivered to the user or to another system. So in choosing a context modeling approach it is not yet relevant.

2.6.3 Model Scoring

The requirements from Table 4 that can still be scored by how well they meet the appliance in every context model (from Section 2.5 are the following:

- Heterogeneity (he)
- Timeliness (ti)
- Reasoning; Decision Support; Inference (re)
- Learning (lear)
- Efficiency; Performance (eff)
- Aggregation (agg)
- Technologies; Lightweight (light)
- Usability; Re-usability; Natural-Projection (usab)
- Constraints; Preference Compliance (constr)

These eleven requirements are scored for each context model. The results are shown in Table 8.

Table 8: Context models scored to the requirements from the literature, from which an overview can be seen in Table 4.

Modeling Approach	**Requirements**								
	he	ti	re	lear	eff	agg	light	usab	constr
1 Key-Value Models	$-$	$--$	$--$	$--$	$+/-$	$--$	$++$	$-$	$--$
2 Markup Scheme Models	$+$	$--$	$--$	$--$	$-$	$--$	$+$	$-$	$--$
3 Graphical Models	$++$	$--$	$--$	$+$	$++$	$+/-$	$-$	$+$	$-$
4 Object-Oriented Models	$++$	$++$	$--$	$+$	$+/-$	$++$	$+/-$	$++$	$++$
5 Logic-Based Models	$+$	$+$	$+/-$	$-$	$+$	$+$	$-$	$+/-$	$++$
6 Ontology-Based Models	$++$	$+/-$	$++$	$+$	$+/-$	$+$	$--$	$+$	$++$
7 Machine Learning	$++$	$+/-$	$++$	$++$	$+/-$	$++$	$--$	$++$	$++$

What can be concluded from Table 8 is that, depending on the application one wishes to build, the requirements might differ. For example if, in a certain application learning, i.e. the evolution of context, is not so important, then a Key-Value Model might be applied, under the condition that other requirements are also fulfilled in this model. In this approach the right context model given the requirements rolls out.

2.6.4 Model ranking

Here we will give grades to the symbolic scores ranging from 1 to 5 where $--$ is equal to a score of 1, $- = 2$, $+/- = 3$, $+ = 4$ and $++$ is is equal to a score of 5. The scores from Table 6 are given a different grade, since the set of symbols there is: $S = \{--, -, +, ++\}$, which has a size of four symbols instead of five. The set S is normalized using the formula: $X^{'} = a + \frac{(X - X_{min})(b-a)}{(X_{max} - X_{min})}$, where a and b are the minimum and maximum range we want to scale to, in our case to [1,5] ($a = 1$ and $b = 5$), X_{min} and X_{max} are the minimum and maximum of the range we have and wish to scale (to a and b), and lastly, X are the different values in the range we want to map. We say that in scores from Table 6 are: $--=1$, $-=2$, $+=3$, and $++=4$. So $X_{min}=1$ and $X_{max}=4$. The range of the normalized scores from Table 6 now becomes: $\{1, \frac{7}{3}, \frac{11}{3}, 5\}$. These scores are added to the scores from Table 8.

The overall rounded off scores resulting from this approach are shown in Table 9 in ascending order. From this it follows that the Machine Learning, Logic-Based, and Object-Oriented modeling approaches are the three most highly preferred approaches in terms of the requirements set in Section 2.4 from the literature.

Table 9: Modeling Approaches and their total scores in meeting context-awareness requirements (in ascending order).

Modeling Approach	**Total Score**
Key-Value Models	32
Markup Scheme Models	38
Graphical Models	44
Logic-Based Models	50
Machine Learning	58
Object-Oriented Models	59
Ontology-Based Models	60

2.6.5 Model Quality: Some considerations

Context information is inherently related to uncertainties. These uncertainties need to be taken into account when raw context data is processed [63]. [63] name several aspects related to the Quality of Context (QoC): "the uncertainty of sensed data, transmission and update protocols, consistency between data from several providers, *and* the trust placed into the information from individual providers" (the italic part is changed by the author).

The processing of context information needs to be done in a quality-aware way [63]. However, it must be noted that, as proposed by [13]: "QoC is not requiring perfect context data, such as all data with the highest possible precision and up-to-dateness, but having and maintaining a correct estimation of the data quality. In fact, if the context data distribution is not aware of data quality, possible service reconfigurations could be completely misled by low quality data". The QoC parameters that [13] then set out are the following:

1. Context data validity: specifies validity context information of a given type must comply with (e.g. a month time context data must conform to the Gregorian calendar format);

2. Context data precision: evaluates the degree of adherence between real, sensed, and distributed value of a certain context information type; for instance, the delivering of a location can be using ultra-wide-band-based location data or more standard GPS-based information;
3. Context data up-to-dateness: expresses the usefulness of particular data and how that changes over time; for instance, the up-to-dateness of location information of a fixed resource (e.g., a GPRS antenna) is higher than the one of a mobile entity (e.g., a user). This knowledge can be helpful for context data distribution as it prevents the usage of large bulks of information (e.g. sensing of fixed resources can be done less often).

These QoC parameters can be extended by the following parameters posed by [74]: (1) default user profile (e.g. user preferences) information does not comply with a certain situation, and (2) temporal effects can limit the accuracy (i.e. precision) of sensors.

A categorization of quality parameters in accordance with when in the framework they are used is posed in [30]. They state that we can identify: data acquisition, data representation and data usage. Data acquisition is related to the quality parameters important in the sensing and capturing of context information. Data representation is related to the quality parameters used to specify good and understandable representations of context information, that are machine-readable and human-understandable. Data usage quality parameters are related to the context of the data acquisition itself, to determine if the data has significance. The quality parameters related to this classification are shown in table 10.

Table 10: Classification of quality parameters.

Aspect	**Quality Parameter**
Data acquisition	resolution, precision, accuracy, range, freshness, location, coverage
Data representation	units, format, understandability, aliases
Data usage	trustworthiness, completeness, relevance, comparability, availability

To be able to say something of the value of one quality parameter it might be necessary to have some knowledge on its dependency on another quality parameter, e.g. we can possibly only say something about the quality of the *understandability* of the data representation if two measures have the same unit of measurement, so that we can compare them. An overview of the dependencies of the quality parameters from [30] is shown in table 11.

Table 11: Quality parameters dependencies from [30]

Quality Parameter	**Dependencies**
Trustworthiness	precision, accuracy, freshness, range, location, coverage, comparability
Comparability	units, format, understandability, aliases, coverage, freshness, accuracy, range, precision
Completeness	all required parameters (determined by the application)
Relevance	determined by the application or the user (e.g. freshness, coverage, location, accuracy, etc.)
Understandability	units, format, aliases

2.7 Conclusion

The aim of this chapter was to give a thorough overview the literature on *context* and *context-awareness* in the Computer Science domain. The perspective taken is the engineering perspective, as discussed in the introduction of this chapter. The focus herein lies on computer calculations and the technological implementation of context-aware systems. In order to do so requirements for context models are needed as well as reasoning techniques. The focus of this chapter lies on the literature that discusses the techniques needed to build a practical system.

The framework of such a system is that it can gather data, transform it, reason on it (i.e. score it according to relevance), and use it in a middleware layer. The middleware layer is a layer that decides on a possible adaptation to the behavior of the system (or not). This framework, i.e. model, is set out to follow a closed-loop: from data acquisition (from sensors) to situation adaptation which means that the state of the *context* should be measured again (is context-aware), in other words: data acquisition from sensors should be performed again.

This chapter started with definitions in the literature of *context* and *context-awareness*, including a stipulative definition. The stipulative definition of *context* in this research is:

> "Context is any information useful to characterize the state and situation of individual entities and the relationships among them. An entity is any subject (virtual, physical, etc.) that is considered relevant to the interaction between the system and the user, including the user and system themselves."

And for *context-awareness* it is:

> "A system is context-aware if it uses context to provide, represent and deliver relevant information and/or services to the user, where relevancy depends on the user's task or situation. This relevant context information must be modeled in such a way that it can be pre-processed after its acquisition from the environment, classified according to the corresponding domain, handled and reasoned about to be provisioned based on the system's requirements, and maintained to support its dynamic evolution. This is done with the greatest possible accuracy."

After the definition of context and context-awareness the possible types of information that can be sensed (or acquired), i.e. data types, and the classification of this information is elaborated on. A categorization is useful to be able to characterize context information types and management categories. There are different ways of looking at a classification: on a meta-level or more specifically, from the viewpoint of a human user or from the system (wherein the user is part of its context, and many more examples). There are different types of sensors that can take measurements on the current situation of entities. Then, this raw measurement data can be processed, e.g. interpreted, to be able to use it as context information. This context information can be formulated along several axes. From static to dynamic, internal to external, nonvolatile to volatile, non-transient to transient, and many more. The dimensions for classification vary from domain to domain, and application to application. In the Computer Science domain the most used categories are: user (human), space (location), time, virtual (computational), type of activity, and devices (hardware). The user component then consists of several elements, being: a user's preferences, beliefs, desires, relations, characteristics (e.g. age, gender), emotions, and more. It depends on the task and application at hand which elements are used to define the user context information. Location (or space) is the context factor that was thought of as the most important, or at least firstly researched, context factor. It is the physical location of a user (or system), including its surroundings (e.g. GPS location). The time factor has a historical component which is very relevant, for a lot of systems knowing the past behaviour of user and system helps in determining a current situation or state and in general: context. The virtual component mostly refers to the virtual location and environment of a user or system. The type of activity is a context factor that is determined by reasoning on the other context factors, combined the system can infer on what is currently happening. The devices (hardware) component refers to the sensors used, the tools used by user and system and the way the system adapts and responds to the user, and vice versa. For example, a mouse can be thought of as a device which aids the user in interacting with a computer, or the touch-screens used in a self-service cash register and the way the interface of such a system looks can influence the interaction between system and user.

To be able to model context requirements are needed. The challenges that context-aware systems deal with is that interactions are adaptive and constantly changing, and thus, so does context. Furthermore, it is a challenge to determine which information should be sensed, the design and the development still require a lot of engineering work, and the integration of existing component technologies (and the large amount of heterogeneous sources) remain challenges. From the viewpoint of the user it is challenging to exactly know what the user wants, to know its emotions, as some people do not even know their own at times. Moreover, some contextual factors cannot be satisfactorily be represented in a numeric value, e.g. tiredness. The list of requirements for context-awareness systems is long, as it is a combination of the requirements from many literature articles. The requirements are based on different levels of the system (meta-level to specific context elements). For example, on a meta-level a good management of context is an important requirement (the management of context acquisition, classification, modeling, handling, exploitation, maintenance and evolution), but more specifically related to context acquisition is the requirement that there might be incomplete, contradictory or uncertain data and that the system should support the measuring of the quality of the data.

After the outline of the requirements the context modeling approaches are set out. In the literature the main approaches are: Key-Value Models, Markup Scheme Models, Graphical Models, Object-Oriented Models, Logic-Based Models, and Ontology-Based Models. This list is extended with Machine Learning Models and Meta-model-based Models, as they were deemed relevant new approaches to add. The modeling approaches differ in how they handle the contextual information, and thus can be termed *data structures* as well. They differ in how well they handle complex information, how easy they are to implement and use, how well they scale, and on many other fronts. In summary, they differ in the way they handle the set out requirements, some models are better at certain requirements where others are bad at and vice versa. Choosing the modeling type always depends mostly on the application that it is used for. However, in general the top three of models that score best on all requirements are: Machine Learning, Object-Oriented Models, and Ontology-Based Models. Not all requirements are included as some would give the same score for each of the modeling approaches.

Context information is inherently related to uncertainties. The quality of context is therefore deemed very relevant in the literature, and the three aspects that can (and should) be taken into account are: context data validity, context data precision, and context data up-to-dateness. There are several parameters that can be measured and that make up the three aspects.

The aim of this chapter to provide an extensive literature survey is met, and focuses mainly on the definition of *context* and *context-awareness*, the different context data types and the possible categorization of this contextual information, the requirements and challenges of a *context-aware* system, and the possible context models, i.e. data structures, that can be used to build a *context-aware* system. Lastly, the models are scored to how well they meet the requirements from the literature, and several quality of context parameters are provided. All in all this gives a good understanding of the state of the art in the literature on *context* and *context-awareness* in the Computer Science domain, and serves as a building block in building our own context model and context-aware system.

The literature survey could be extended with a summary of the outcomes of the practical implementations of several context-aware systems. This would help in pinpointing the strengths and weaknesses of the context modeling approaches given certain situations. However, it takes a long time to provide this overview given the large amount of literature articles on the topic (in the Computer Science domain alone). It would be interesting in this regard to apply all the outlined modeling approaches to the same dataset, to really understand which requirements are met and which are not. Now, some context models were said by some papers to meet a requirement, while in another paper, with a different application, it scores different on a certain requirement. It is interesting to perform more research on this.

What is interesting from the requirements in the literature is that there a big part of subjects that come forward in a lot of literature studies, but the names they give these requirements and the levels at which they set the requirements differ greatly. The study on context models could benefit greatly from well-researched guidelines on requirements and which are relevant in which scenario. This literature survey aimed at doing so, but the research can be extended further in the future. The same goes for the context information categorization, which also differs quite some in the literature (in how the categories are termed and interpreted), but still most categorization schemes in the literature cover broadly the same topics. Set out guidelines and protocols for what to do when forms a solid basis for researchers and can help future research in that developers (researchers that implement the system) and (theoretical) researchers speak the same language and the knowledge is more easily transferable across research and applications.

It is clear from the literature that context is inherently uncertain and that context modeling is very application specific. It is hard to overcome these challenges, but the ideas for future research posed above, and the sharing of databases, guidelines, protocols, and simply the performance of many more research (in lab and real life settings) should, and will, aid to the development of context models and the building of solid practical context-aware systems.

3 Practical Implementation Vanderlande

3.1 Introduction

Context-awareness and the use of *context* factors are relevant and interesting for *real world* systems, albeit systems used in a company, online, by an individual, and many other type of systems. Where the previous chapter dealt with a literature approach regarding context-awareness and context factors, the content of this chapter is concerned with the practical implementation of a context-aware system. The system that will be proposed in this chapter is constructed for the company Vanderlande and build using data from the factory Onninen in Hyvinkää, Finland plus external real world data retrieved from several online sources.

As underpinned in [21], software systems can optimize their processes by exploiting context factors; Such factors can stem from the system itself, their environment and/or interactions between the system and its environment. Furthermore, a worker's environment can (sub)conciously affect worker performance [53]. Therefore, it is interesting to look at which contextual factors we can use to improve process optimization and to increase worker productivity.

This chapter will discuss how Vanderlande can use data from their workstations (i.e. picking stations) to model *context* and make their systems more *context-aware* (which - then - contributes to the UX and process optimization). To be able to do so, an initial brainstorm-session was held at Vanderlande's office building in Veghel. The results of this brainstorm-session can be found in Appendix A. The brainstorm-session, together with the picking station data delivered by Vanderlande, and external sources are used as input data to this practical implementation chapter.

In the previous chapter (Section 2) the context information categorization, requirements and models were discussed, ending with a scoring of the models. As the dataset from Vanderlande is quite limited there are not a lot of categories distinguishable. The dataset encompasses information on dates, number of operators at work, events handled, the identity number of the order etc. In other words, the data is quite limited to what can be measured by the working station that the operators work at, but it does not encompass e.g. physical information on the working environment or user profile information. The sensors used are only virtual sensors. The requirements for the system are therefore not all met. The main objective is to incorporate external data and use heterogeneous data and combine this (as outlined in the introduction). The aim here is to see if, from a limited dataset, it is possible to see if certain factors interact with (are related to) each other. And to investigate if external factors have an effect on the internal parameters. This practical implementation is therefor limited in scope, as it does not provide a concise and full context model, but rather it is exploratory.

A global overview of a possible computational architecture (i.e. framework) - of a system that makes use of context information - has three layers (it is a simplification of the framework proposed in the previous chapter); The lowest layer specifies a format for acquiring and storing data in a way that facilitates retrieving all context factors related to a query (this layer contains the primary context: everything we can measure). The second layer extends this data with all contextual factors we can calculate and model based on the measured information (this is the secondary context, it contains everything we can compute about the data). On top of the information architecture, a decision module is introduced as the third layer, which retrieves the relevant contextual factors and uses these for a task. For the case study in this chapter this would be to find the relation between contextual factors and throughput. As throughput is the amount of work done per operator in a specific time frame, it is interesting to relate this to several other context factors and infer on their relations. The outcome of this inference can be a business strategy for the company (i.e. a way to optimize their processes), an improvement of the UX, or simply provide an insight into their people, products and functioning.

A visualization of the layer architecture is shown in Figure 4. The practical implementation of this chapter will only concern itself with the first two layers (the information architecture), as a decision module would require a lot more data and time to implement properly. The third layer is especially interesting if there is a lot of context information at hand. For example, models that are learned for a specific task can be shared between datasets (and companies), in that way multiple datasets can extend and enrich each other, as long as the dependent variable is the same. However, this chapter will form an introduction into the use of context factors and a way of applying calculations on the dataset at hand without having access to an extended dataset (which will be the case in a lot of real world applications).

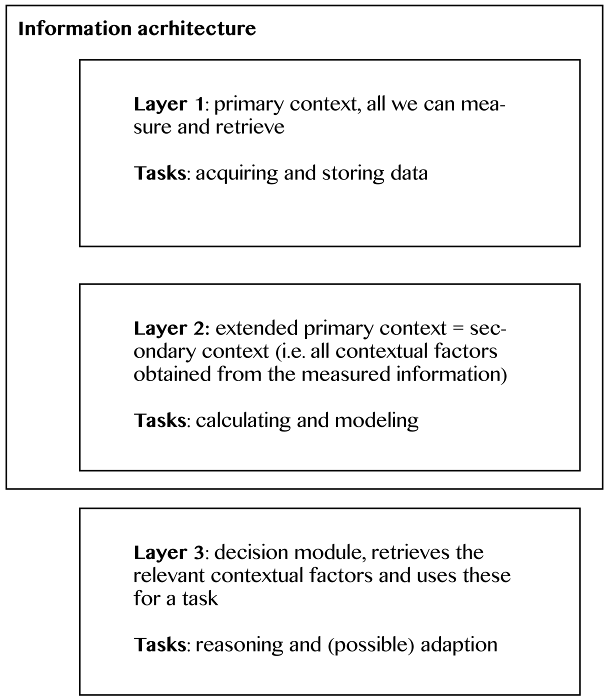


Figure 4: The 3-layer architecture that will be used in this implementation chapter. The first layer being the acquisition and storing layer, the second layer the computing and modeling layer, and the last layer the decision module, where the system decides whether to take action or not. The first two layers make up the information architecture of the system.

Given the dataset from Vanderlande of the Onninen factory, there are three factors that make it substantially more difficult to use conventional statistical tests without pre-processing and summarizing the data beforehand, those are: (i) there is a lack of data in general, (ii) we are dealing with a sparse dataset, and (iii) (most) variables in the dataset follow a non-normal distribution. These three factors have an influence on how the data is analyzed in this chapter, this will be discussed further in Section 3.3.

Business processes benefit from contextual knowledge as makes clear which actions should be taken at what time. The context-aware system in this case reasons on the data of a certain situation and knows which actions are most efficient and relevant given the situation. This makes the business process more adequate, self-managing, automatic, and demanding minimal administrator's guidance [57]. In other words, the business process becomes more efficient.

The outline of this practical implementation chapter is as follows: Section 3.2 will discuss the delivered dataset received from Vanderlande. Section 3.3 explicates how raw data is transformed to be able to analyze it, proposes questions that are deemed relevant regarding the dataset and in modeling *context*, and describes the techniques used to answer these questions. Section 3.4 describes which technical resources are needed and used to process the data. Section 3.5 shows the results from the interpreted data. And lastly, Section 3.6 is a discussion and conclusion, which encompass an interpretation of the results and ideas, and possibilities for the future (for research and implementation).

3.2 The Dataset

As briefly touched upon in the previous section, Section 3.1, Vanderlande delivers logistic systems to airports and companies that work with distribution centers. The sorting systems used in distribution centers are subject of this report. In these warehouses, so called, *operators* put the products of an order from different crates (or other container types) into one crate. This order is then send out to the customer of the order. The work environment of these operators is delivered by Vanderlande and consists of a computer-controlled system that manages the production line (conveyor belt and other distribution machines in a warehouse), on which the crates come and go, and a computer on which the operators can see what the order is, and thus which action(s) they should perform (a picture of such a system can be found in appendix A). Vanderlande delivers the systems to a diverse customer base, consisting of supermarket, fashion, and general merchandise (which can be any sort of goods) warehouses.

The factory where the dataset that is used in this report comes from is Onninen in Hyvinkää, Finland, which is the distribution center of the company, its head office is located in Vantaa. They have 50 express stores or selling points in Finland (and are also located in Sweden, Norway, Poland, and the Baltics). Onninen customers include "electrical, heating, plumbing, ventilation, air conditioning and refrigeration contractors, industrial companies, power plants, public organisations, and retailers", [67] as stated on their website. The products they offer are manifold and are usually from business to business or to consumer (in their express stores). At the end of 2016 they employed 1,100 people in Finland, which went up to 1,200 people at the end of 2017 [67].

The dataset is delivered in *csv*-format (comma separated values) and consists of $816,362$ observations (also termed *tasks* or *events*) of 20 variables. An overview of the names of the variables, a short comment on what they mean, and the number of distinct values that the variables have, is shown in Table 12.

Table 12: Overview of the outcome of context dimensions and context parameters at Vanderlande

#	Variable Name	Meaning of Variable	No. Distinct Values
1	IDEVENT	unique ID of the event	816, 362
2	PERIODID	ID of the period in which the event occurred	73, 777
3	INSERTTS	timestamp when the event was inserted into our database	97
4	EVENTTS	timestamp when the event occurred	98
5	PRIMARYLOCAL-CODE	customer specific non-unique stock keeping unit (SKU) number	19, 669
6	UNITOFMEASURE	The UoM is the number of items of a SKU managed in the warehouse, e.g. pieces, litres, kilograms	7
7	DELIVERYORDER-ID	ID of the delivery order	155, 668
8	DELIVERYORDER-LABEL	label of the delivery order	155, 668
9	DELIVERYORDER-LINEID	ID of the delivery orderline	760, 457
10	DELIVERYORDER-LINELABEL	label of the delivery order-line	25
11	OPERATOR	ID of the operator executing the pick	33
12	STSUID	Single-SKU TSU id (product tote)	61, 833
13	STSULABEL	Single-SKU TSU label (barcode of product tote)	61, 720
14	DTSUID	delivery TSU id	212, 130

Continued on next page

Table 12 – *Continued from previous page*

#	Variable Name	Meaning of Variable	No. Distinct Values
15	DTUSLABEL	delivery TSU label (bar-code)	197
16	TASKID	unique ID of the pick task	816, 362
17	SKUID	ID of the SKU internal	19, 667
18	SKUMATCHCODE	SKU identifier for the customer	19, 667
19	SPECQUANTITY	specified quantity to be picked	476
20	PROCQUANTITY	processed quantity actually picked	466

3.2.1 Describing the Dataset

All 816, 362 observations have 20 characteristics, each of these characteristics has one value for that observation, which is either a number (or code) or it is one of a set of predefined values (i.e. a factor variable). Factor variables can have one of several levels i.e. values. For example, all the observations in the dataset have a UoM (characteristic) which can take one of seven values. In fact, there are only two variables that are not a factor, they are unique: IDEVENT and TASKID. The rest of the variables can take one of a few values (e.g. one of a few dates, one of a few packaging materials et cetera). Most variables are identification numbers (IDs) or labels, which are mostly bar-codes or numbers to identify crates, goods, orders et cetera, to be able to track the goods, crates, orders and clients.

The first variable is the IDEVENT, which is a number (code) that indicates an event (also termed task or pick). The same holds for the sixteenth variable TASKID. This is also a unique identification number (ID). In the rest of this report, the terms *event*, *pick* and *task* will be used interchangeably, as they all refer to one unique action.

The second variable PERIODID is the ID of the period in which the event occurred. The differences between the values for this variable are always 60 or a multiplication of 60. This could indicate that the handled events are registered per hour, however it is difficult to trace back which hours of the day are meant. An event with a certain PERIODID happens in a certain hour, but we do not know which hour is the first and sometimes the difference in PERIODID value on one day is so high (a large multitude of 60) it could mean that an event occurs in the morning and the other in the afternoon, but as we do not have information on shifts it is impossible to draw definitive conclusions on this information.

The third and fourth variables INSERTTS and EVENTTS are both timestamps. The first is the timestamp of when the event was inserted into the database and the second of when it is executed. The timestamps do not include a specific time (this is, as said, probably included in the PERIODID), but the values for the variables are a date for each event. The period that the dataset covers ranges from August 18 to December 21 of 2017. August 18 is a Friday and the next date is August 21, which could imply that the factory is closed during the weekends. However, in the weekend following the weekend of August 19 and 20 there were events/orders coming in on Saturday September 2, and there were also events executed on this day, which implies that on Saturday September 2 work was done in the factory. There are more weekenddays like this. The total number of days between August 18 and December 21 is 125. From this there are 98 days for EVENTTS and 97 days for INSERTTS. The date "17-12-17" is in EVENTTS but not in INSERTTS, this means that there is one more day on which tasks are processed than on which they are inserted into the database. The dates on which the tasks are being processed in the factory (EVENTTS) are shown in Figure 5.

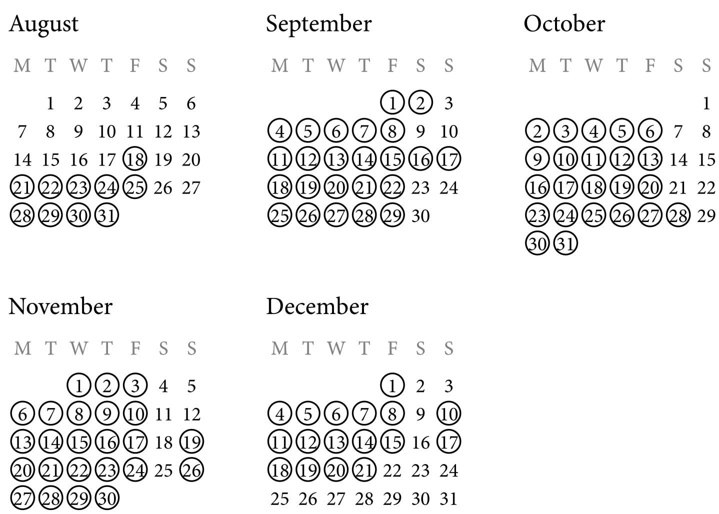


Figure 5: The dates in the dataset on which the events occurred (EVENTTS, see Table 12).

The fifth variable is PRIMARYLOCALCODE. As shown in Table 12 this is a customer specific non-unique Stock Keeping Unit (SKU) number. To clarify, a SKU is an identifying number for the inventory or stock of an item; Each item has certain characteristics and that is why it belongs to a certain inventory, the SKU is the identifier for that particular item in that inventory. Furthermore, it is customer specific. What this means is not clear. It could be that certain stock is specifically used for certain customers, and that different products belong to the same stock (i.e. have the same SKU) because they are all for a specific customer. But it could also be that the stock of a specific product is divided for several customers, but that different products can not have the same SKU. Of course, there are also other ways to interpret this variable. Given the dataset we do not know how to interpret this variable.

The sixth variable, the Unit of Measure (UoM), can have one of seven values, which are the possible packaging materials. For example an event can have UoM "BAG", which means that the packaging of the ordered product is a bag, or it can have "BLI", which means the packaging is a blister pack. Actually, there are six packaging materials but the value can also be empty (the seventh option). What this seventh option entails is not clear. It could be that there is no packaging, or that these are missing values.

The seventh to tenth variables are all concerned with the delivery order. The seventh and eigth variable DELIVERYORDERID and DELIVERYORDERLABEL concern themselve with the delivery orderline and each can take one of 155,668 values. one being the ID and the other the label. Some of the delivery order IDs encompass one event while other encompass 59, and everything in between (with some outliers). The same goes for the labels. Then, the DELIVERYORDERLINEID can take one of 760,457 values, and each of these values occur in between 1 and 25 picks (both mostly once). The DELIVERYORDERLINELABEL then can take one of 25 values, which occur between 10,248 (the 25) and 170,748 (the 1) events. This could imply that there are 25 conveyor belts (i.e. delivery order lines), as the the label 1 is used most often and then in ascending order it goes to label 25. However, we do not know if this is the right interpretation.

The eleventh variable is OPERATOR. It is the ID for the operator who handled the event. In total there are 33 pickers (i.e. operators).

The twelfth to fifteenth variable, and also the seventeenth and eighteenth variable, are all related to the SKU and TSU. What a SKU is, is already explained. A Transport and Storage Unit (TSU) is the code for the type of unit where the goods are kept in, e.g. a pallet, or a plastic crate. A TSU thus can keep several SKUs. Where STSUID and STSULABEL are concerned with the ID and label of the single transport and storage units, and DTSUID and DTSULABEL are concerned with the delivery ID and label for the total delivery transport and storage units. The SKUID and SKUMATCHCODE are then the internal ID for the SKU and the external identifier for the customer. Both have the same amount of values they can take (19,667). And the amount of values these can take comes very close to PRIMARYLOCALCODE (19,669), which is also concerned with the SKU. However there is a difference of two, which is unexplained.

Lastly, the nineteenth and twentieth variable consider the specified quantity to be picked, and the processed quantity actually picked. These variables consider the amounts of goods handled in one event (the specified amount and the amount actually processed). This means that one event or order can comprise of several goods. Whether these are big or small is unknown. For example, if the processed quantity is twenty and is shipped in a blister packaging, then it might be bigger or smaller than one product in a bag, we do not know this.

Furthermore, there are eight variables that have missing values; DTSUID and DTUSLABEL both have 86 missing values, and DELIVERYORDERID, DELIVERYORDERLABEL, DELIVERYORDERLINEID, DELIVERYORDER-LINELABEL, and PRIMARYLOCALCODE all have 4, 379 missing values. Which can either mean that this empty value is an actual value, i.e. an empty field also means something, or the value is really missing (e.g. not processed). The rest of the variables have zero missing values.

3.2.2 Open- and Closed World Data

The Literature. In the literature about information databases (wherein different *context* features, parameters or factors are saved) often the terms *internal* and *external* data are used [5][12]. What is meant by these terms in general, is that *internal data* is local data (i.e. context parameters) that concerns attributes stored in the database, and *external data* involves attributes outside the database [12]. In other words, external data comes from the external, wider, world. While internal data is the data at hand in a company, lab or other research department. In [5] the distinction between internal and external context is explained as: the distinction between specific domain context, i.e. closed world or internal data, and general and independent domain context, i.e. open world or external data.

The difference between the two data types is, according to [75], that internal databases need to access operational data, and external databases need to access external, online databases. One can also combine the two data types, this is done by an integration of internal and external data, termed *complex* data by [32]. [32] further explains the distinction between *internal* and *external* data as internal context parameters being the factors used to model users given a certain situation i.e. the contextual factors described from a users (human) perspective, and external context parameters as being the factors that describe environment and device properties that influence the situation. In summary, what to designate as internal or external depends on your viewpoint. Here, the data from the factory of Onninen Finland will be regarded as internal data. Where open world, online sources, such as e.g. weather forecasts, are external data sources.

In the business intelligence and innovation research domain(s), the terms are also used. External data serves as an extension to internal data. *External data* is said by [73] to "increase efficiency, understand customers or gain new insights". Furthermore, external data needs to be processed (i.e. transformed) and integrated with the internal data. It is important that external data is deemed reliable which can be measured according to its accuracy and appropriateness [73].

Open World Data in our System. Next to the Onninen data delivered by Vanderlande, several external (i.e. open world) resources are used to extend our dataset. From the brainstorm-session (see Appendix A) it followed that, for example, *world events*, *economical data* and the *weather* could be useful in determining if external factors influenced the operators in the workstations. These external, or rather open world, variables were added to the dataset.

Table 13 shows the world events that will be used in this research report. As you can see, world events can belong to one of several categories. The categories used are Crime, Holiday, Misc, Sports and Holidays, but of course one can think of other ways to categorize these world events. As we want to calculate if these world events have an impact on the internal factory data we will have to transform the data to be able to use it in our system, this will be discussed in more detail in section 3.3.

Table 13: Open World Events, manually retrieved from Wikipedia, in the period that covers the dataset (from August 18 to December the 21 in 2017), with the corresponding category of each event (from the set: {Crime Sports Holiday Misc}).

Date	**World Event**	**Category**
August 18, 2017	Stabbing attack in Turku, Finland. Two people died	Crime
August 20, 2017	Minute of silence to honour victims of Turku stabbing	Crime
August 26, 2017	Finnish Nature Day (the flags will be raised, bonfires lit, dinners enjoyed and Finnish nature gets all the credit it deserves)	Holiday
August 31 till September 3, 2017	The Red Carpet Film Festival of Hyvinkää	Misc
September 2, 2017	2018 FIFA World Cup qualification (pool I of Finland)	Sports
September 5, 2017	2018 FIFA World Cup qualification (pool I of Finland)	Sports
September 9 and 10, 2017	US Open Finals (tennis)	Sports
September 12, 2017	Elections in Norway	Misc
September 13, 2017	Baseball major league finals	Sports

Continued on next page

Table 13 – *Continued from previous page*

Date	World Event	Category
September 17, 2017	EuroBasket 2017 (basketball championship (EU))	Sports
September 22, 2017	2 refugees stabbed themselves in front of Parliament	Crime
October 6, 2017	2018 FIFA World Cup qualification (pool I of Finland)	Sports
October 9, 2017	2018 FIFA World Cup qualification (pool I of Finland)	Sports
October 24, 2017	Ransomware (Security researchers report on the outbreak of the ransomware nicknamed Bad Rabbit, which has affected computer networks throughout the world)	Crime
October 26, 2017	Rail accidents in 2017 (four people are killed and four conscripts are injured after a passenger train collides with an off-road military truck in Raseborg, Finland)	Misc
October 29, 2017	FIA Formula One World Championship, 2017 World Rally Championship, WTA Finals (Tennis)	Sports
October 29, 2017	Clock was set to winter time, so turned backward one hour	Misc
November 1, 2017	Game 7 was played (US baseball)	Sports
November 4, 2017	All Saints Day in Finland	Holiday
November 12, 2017	Fathers day in Finland	Holiday
November 13, 2017	2017 World Rally Championship (Sweden-Italy)	Sports
November 19, 2017	NASCAR Cup Series (stock car racing)	Sports

Continued on next page

Table 13 – *Continued from previous page*

Date	World Event	Category
November 27, 2017	Prince Harry announces wedding to Meghan Markle	Misc
December 6, 2017	Finland celebrates its 100 years of independence	Holiday
December 10, 2017	2017–18 Premier League Finals	Sports
December 16, 2017	2017 FIFA Club World Cup	Sports

Another open world factor that is incorporated in this report is economical data, more specifically the Consumer Price Index (CPI), which is calculated per month. Weather data, which consists of several variables (e.g. precipitation, snow etc.), is calculated per day.

In the next section (Section 3.3) the transformations needed to be able to use the open world data are set out, as well as the questions to be asked to get relevant results from the total dataset (from Onninen and open world), and the techniques used to deliver these results.

3.3 Methods

As shown in Section 3.2 the data from Onninen consists mostly of ID and label values. However, the most interesting aspects are the 33 operators (i.e. pickers at the picking stations), the tasks (i.e. events) they perform, the dates on which they work, the amount of goods that each task consists of, and the type of packaging that a task has. Even though some of the IDs and labels (e.g. for the delivery order and SKU) might also be interesting, it is unclear how to interpret them correctly, and therefor information available on these variables is too limited to use it effectively. the closed world data (from Onninen) is extended with open world (from external sources) data, to increase the dataset and the possible (context) factors that influence the internal (closed world) data.

This section will discuss how the open world data from the previous section is transformed to be able to make use of it in our system, which questions we want to answer regarding the total dataset (of internal and external sources) and which techniques we will use to do so.

3.3.1 Data Transformations

The external, open (i.e. real) world data that will be used in this report, as already mentioned in the previous section, is the Consumer Price Index (CPI), the weather, and the events as shown in Table 13.

To be able to use the events listed in Table 13, we need to look at what is relevant. Even though all world events in the table happen in the time period that the dataset covers, some world events happen when the factory is not opened. These events might have an influence on other factors in the dataset on the day beforehand (e.g. in the case of holiday events) or the day after (e.g. in the case of crime events, but possibly for holidays as well). Thus, if there is a world event on a certain date when the factory is not open this world event might be used as a context factor on the day before or after, when the factory is open. The world events can be transformed to be an array of 0s, 1s and 2s: a 0 when an world event is not happening, a 1 on the day before or after (not for crime events, as they are not known beforehand), and a 2 on the day a world event is happening. The array should have a length of 98: the 98 days on which events occur in the factory.

Instead of 0s, 1s and 2s it is also possible to look at the world event categories as either positive or negative. For example, crime world events are negative, while holidays are positive. There is also a world event array saved like this (as extra array), where Sports and Holiday are 2, and Crime is -2 (and the before and after days are 1 and -1). And lastly, Misc is divided between 2 and -2, for example the rail accidents are -2 while the announced marriage of Prince Harry and Meghan Markle is 2. In the first array all events are 2, the days before and after are 1, and the days on which no events occur are 0, in the second array still all days on which no events occur are 0, but some events are 2 and some are -2, and the days before and after are 1 and -1.

The CPI is a number, thus it does not need transformation. However it is calculated per month, and thus the other variables in the internal dataset that are relevant to measure related to the CPI should be transformed to months as well. The CPI data is shown in Table 14 (the year is left out, since all months are from 2017). Next to the CPI, Table 14 also shows the annual changes of the CPI in percentages.

Table 14: The CPI and annual change in % in Finland from August to December of 2017 [68].

Month	**Overall CPI 2015=100 Point figure**	**Annual Change in %**
August	101.1	0.7
September	101.3	0.8
October	101.3	0.5
November	101.6	0.8
December	101.5	0.5

The weather contains several aspects. The weather variables are downloaded from [69] and contain the daily observations of all days between August 18 and December 21 2017. The weather variables are (with extra information from [69]):

- Precipitation amount, the daily precipitation is measured between 8 a.m. (9 a.m. in summertime) and 8 a.m. (9 a.m. in summertime) local time the following day and is given in millimetres (snowfall=water equivalent);
- Snow depth, is measured at 8 a.m. local time (9 a.m. during summertime), accuracy +- 2 cm;
- Air temperature, this is in fact the mean, i.e. average, temperature measured on 4 or 8 observations per day;
- Maximum temperature, this is the highest temperature during two 12 hours period, i.e. between 8 p.m. previous evening and 8 p.m. this evening (9 p.m. - 9 p.m during summertime);
- Minimum temperature, this is the lowest temperature during two 12 hours period, i.e. between 8 p.m. previous evening and 8 p.m. this evening (9 p.m. - 9 p.m during summertime).

Since these variables are also numbers, they do not need transformation.

3.3.2 Questions

From the total dataset certain factors and an interplay of these factors seem relevant. When we look at the context classification diagram proposed in [insert reference to literature section of classification diagram] and we lay this diagram next to the dataset at hand, it becomes clear that our dataset does not cover all the categories in the classification diagram. For example, the user context is unknown to us in the dataset we have here. However using the classification diagram does provide us with a starting point to delineate the context factors that we wish to extract from our dataset.

This extraction is done by forming questions regarding the data, where the answers will require the interpretation and reasoning on the data, which will lead to contextual results, and eventually to conclusions about which actions to take given the context factors.

The questions we wish to answer regarding the dataset are limited in scope to the variables from Section 3.2 that were deemed interpretable (the variables that were inconclusive and open for interpretation are left out) and relevant. Some of the questions are included to get a better understanding of the dataset, others are examined because they can lead to conclusions on UX and process optimization. The questions that we seek to answer in this chapter are:

1. How many of each of the Units of Measure (UoMs) are processed in total? Are there certain UoMs that occur more often, and what might be the reason for that? Do operators handle all sorts of UoMs, or do they focus on one type?

2. On which days and for which number of events does the specified quantity to be picked differ from the processed quantity actually picked? Does this often happen for certain UoMs? Or for certain operators? And does it occur more often on certain days of the week?
3. What is the throughput (i.e. the amount of work (i.e. tasks or events) done per operator in a specific time frame) in total, per week, per day, and for each day of the week? Is there a difference in throughput between days of the week (e.g. between weekdays and weekenddays)? When we include the processed quantities and answer the same questions, does this give different results?
4. Does the amount of operators at work on the same day influence their productivity (perceived workload)? Does the productivity (number of tasks executed per day by each operator) between operators differ, or is it somewhat the same?
5. What influence does the weather have on throughput?
6. What influence do World Events (e.g. football matches, national (holi)days etc.) have on throughput?
7. What influence does the economic state (e.g. Consumer Price Index (CPI), inflation rate etc.) have on on throughput?

3.3.3 Techniques

To be able to interpret the dataset several techniques are needed. As already mentioned in Section 3.1, due to a lack of data (e.g. we do not know the duration of shifts, the physical environment, nor do we have information on the variables in the dataset over longer periods of time (i.e. more than 98 days)), the fact that we are dealing with a sparse dataset and that (most) variables in the dataset follow a non-normal distribution, a (multivariate) analysis of variance ((M)ANOVA) is not feasible, therefor the statistical methods used to analyse the variance are the two-sided and one-sided t-test and the Wilcoxon rank sum test with continuity correction. These techniques and other measures that will be used are:

- Correlation coefficient. Looking at data relation is a sensible step to understand how your different variable interact together. Correlation look at trends shared between two variables. The Spearman method is used here to provide a nonparametric measure of the correlation, as we are not always dealing with variables in our dataset that assume a normal distribution. While Pearson's correlation (the default in R) assesses linear relationships, Spearman's correlation assesses monotonic relationships (whether linear or not), by using the rank correlation. The rho (outcome of Spearman's rank correlation) is a value between -1 and 1 that measures how two variables change relative to each other. A value of 1 means 'if x goes up, y always goes up', and a value of -1 means 'if x goes up, y always goes down'. So it measures how much x and y vary. There is also a value that shows how confident we are about this correlation, it is the probability value.

- The probability value (or p-value) is used to look at the correlation (or asymptotic significance). It shows the probability of observing a certain result. It is usually used together with a level of significance alpha, here 5%, which is the threshold value (i.e. cut-off point). In case of hypothesis testing we say that if the p-value is less than the chosen significance level (alpha), this suggests that the observed data is sufficiently inconsistent with the null hypothesis and so the null hypothesis may be rejected. When looking at the p-value for the normality test the interpretation is somewhat different.
- Normality test, more precisely the Shapiro Wilk test in R. The p-value is also used to see if a dataset approaches a normal distribution. This p-value reports what the chances are that the sample comes from a normal distribution. If this p-value is low, the smaller the chance. Usually a cutoff of a value of 0.05 is used. If the p-value is lower than 0.05 then the sample deviates from normality.
- Mean, is the sum of the values, i.e. datapoints, in the dataset divided by the number of values. We also call this the average.
- Standard Deviation, gives an indication on the variation or the distribution of datapoints in the dataset. If the standard deviation is low than the datapoints in the dataset are close to the mean, if it is high then there is a large spread of the datapoints, i.e. the datapoints are spread out over a larger range.
- Confidence level, it says that when a certain sample dataset is retrieved again and again, then the results would match the results from the actual population 95 percent of the time. So it is the frequency of possible confidence intervals that contain the true value of their corresponding parameter.
- Confidence interval, consist of a range of values (interval) that act as good estimates of the unknown population parameter. There is a 95% chance that the sample mean will fall into the confidence interval, but there is a 5% chance that is does not (we choose 95% and 5% here, it could also be 99% and 1% for example).
- Kurtosis, is a measure of *flatness* of a probability distribution. It says something about the tails of the distribution. A high kurtosis means that the distribution is very spiked, and mostly indicates outliers and large deviation, and a low kurtosis indicates a flat distribution with modest sized deviations. The kurtosis of any univariate normal distribution is 3 for the Pearson formula, and has a correction to be 0 in the Fisher formula. Here we use the Fisher kurtosis formula, thus with a correction, so it is 0 when we speak of a univariate normal distribution.
- Skewness, related to kurtosis is the *skewness*, i.e. the measure of asymmetry of the probability distribution around its mean. Negative skewness indicates that the tail on the left side of the probability density distribution is longer or fatter than the right side, and positive skew means the same but then for the right side.

- T-test, the two-sample t-test tests the null hypothesis that the population means of two groups are equal, based on samples from each of the two groups. It assumes normality for the distributions in each of the two groups. If there is no normality, then the larger the sample the more likely that the test will still be accurate. The one-sided t-test does in essence the same but looks at an alternative hypothesis not of not being equal but of the second group having a significantly smaller or bigger mean than the first group.
- Wilcoxon rank sum test with continuity correction in R, it does measures the same thing as the t-test, but is used if we can not assume normality. It measures if population distributions are identical without assuming them to follow the normal distribution.

3.4 System Description

The system is build using the R programming language. The libraries that are used in R are:

- Psych package for descriptive statistics, including mean, standard deviation, skewness and kurtosis [55].
- The R Stats Package for the correlation coefficient, t-test, and Shapiro Wilk test [65].
- The R Graphics Package for plots, boxplots and histograms [64].

The *csv*-file is loaded into the R script and used in the form of a data frame with the columns being the individual arrays of each variable, and the rows being all variable values for one specific task (or event).

Open world data (such as world events) is acquired from Wikipedia; On Wikipedia world events were looked at for each month and manually filtered on (possibly) relevant information. Furthermore, weather data is downloaded from `http://en.ilmatieteenlaitos.fi/download-observations#!/`, downloaded in *csv*-format as well. It is downloaded for each day between August 18 and December 21 of 2017 and then the 98 days that the Onning factory is open in this period are filtered so that the variables have the same length as the number of days in the Onninen dataset. The CPI is calculated per month and ranges from August to December of 2017. Next to the CPI the annual change in % is also downloaded (all economical data is downloaded from `http://pxnet2.stat.fi/PXWeb/pxweb/en/StatFin/StatFin__hin__khi/`).

A visualization of the system architecture is shown in Figure 6. As you can see the R script uses the abovementioned packages, loads the Onninen data *csv*-file into arrays and manually includes the Open World data from Wikipedia and other sources on the internet. It then performs calculations on the data using the libraries; The results of the performed calculations are shown in the next section (Section 3.5).

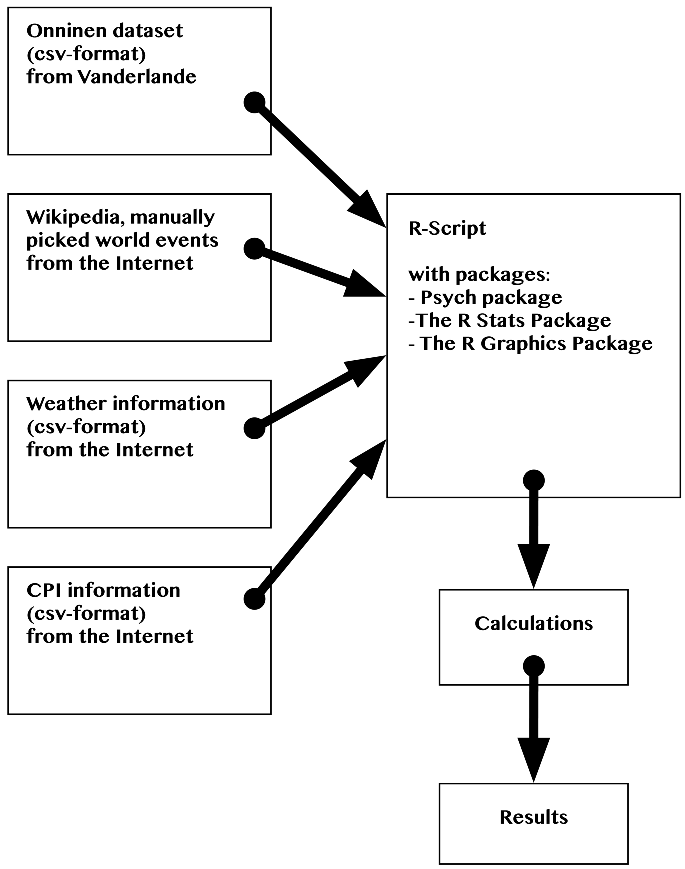


Figure 6: A visualization of the system architecture; the dataset, retrieved Open World data and Onninen data, the R script with the packages used, and the calculations and results.

3.5 Results

3.5.1 Units of Measure

The seven possible units of measure are: {“ ”, ‘BAG’, ‘BLI’, ‘EA’, ‘KG’, ‘PAA’, ‘PAK’}. An overview of how many events each of the UoMs encompassed, how many operators handled each UoM, and the average number of events handled by each operator for a certain UoM type, can be found in Table 15.

Table 15: The amount of events handled for each UoM, the number of operators that process these, and the average number of events of each UoM processed by each operator handling that UoM (from the Onninen dataset). The averages are rounded of to the nearest integer.

UoM	**No. events**	**No. operators**	**Average no. events per operator**
“ ”	4,379	31	141
BAG	20,319	32	635
BLI	225	25	9
EA	732,300	33	22,191
KG	2	2	1
PAA	8,728	32	273
PAK	50,409	33	1,528

The initial things noteworthy in Table 15 are that a number of $4,379$ events have an unknown UoM, the number of operators handling a specific UoM ranges from 25 to 33, with an anomaly of two workers for ’KG’ (possibly due to the fact that there are only two products to be processed for ‘KG’), and the UoM that is most prevalent is ‘EA’.

The ‘EA’ products are handled by 33 operators, which is, with a confidence level of 95%, significantly more than for the other UoMs with a p-value of 0.02895 (< 0.05), using the Wilcoxon signed rank test in R. However, the value of ‘KG’ is very low, and this might bias the result for ‘EA’. If the 2 for ‘KG’ is left out, the Wilcoxon signed rank test in R shows, with a confidence level of 95%, a p-value of 0.04876, which is still < 0.05. If however we do assume normality the p-value for a one-sided t-test in r is, with a confidence level of 95%, 0.08491, which is higher than 0.05. As the dataset consists of only 7 datapoints, and one is left out (the ‘KG’) outlier, these calculations are approximations at best.

When we use the Wilcoxon signed rank test and t-test in R to see if the average handled events for the ‘EA’ UoM is greater than those for the other UoMs the p-values are, with a confidence level of 95%, 1.523e-09 and 0.01563 respectively. Both p-values are smaller than 0.05, and thus the null hypothesis that the average for ‘EA’ is within the range of the other averages, is rejected in favour of the alternative: that the average value for ‘EA’ is significantly greater.

A possible interpretation for the high amount of ‘EA’ events handled, is that ‘EA’ packaging is not a lot of work, or that they are small. A similar observation, but then the other way around, holds for the ‘BLI’ UoM, which stands for blister packaging. Each operator handling an event with a blister packaging handles 9 on average, which would then imply that the packaging is larger or more work (or both) (otherwise fewer operators would be needed to handle only 225 events with blister packaging).

We can immediately conclude from Table 15 that none of the variables (no. events, no. operators and average no. events per operator) follows (or approaches) a normal distribution, and when we use the Shapiro Wilk normality test in R we see that all values are indeed lower than the threshold value alpha < 0.05 (0.0001042336, 0.0005236139, and 1.949374e-05 respectively). So when we test the correlation in R, we do so using the Spearman method as the data does not assume a normal distribution and the amount of datapoints in lower than 10. When we look at the correlation coefficient of the number of operators and the average number of products they handle, this is 0.9819805 (with a p-value of 8.291e-05). And between the number of events and the number of operators it is 0.9819805 (with a p-value of 8.291e-05). And lastly, between the number of products and the average number of products they handled it is 1 (with a p-value of 0.0003968), which is logical since the number of products is divided by the number of operators (always around 31 to 33, with some exception) and thus it correlates fully with the average. The p-values are all less than the significance level alpha = 0.05, and therefor we can conclude that the variables are significantly correlated with a confidence level of 95% and with correlation coefficients of 0.98, 0.98 (rounded of to 2 decimal points) and 1.

If the 'KG' UoM is left out, the correlation coefficient between number of events and number of products is 0.97 (rounded of to 2 decimal points), which is the same as for the number of events and average handled per operator (the correlation coefficient is still 1 for the number of products and average number of events handled per operator). The p-values are 0.0013 (rounded of to 4 decimal points) for both. So, leaving 'KG' out has a negative effect on the Spearman correlation coefficient: it is 0.01 less.

When we look at the distribution of the seven UoM types over the 33 operators and see if this distribution approaches a normal distribution this provides us with the outcomes in Table 16. For each of the UoMs the p-values for the distribution over the 33 operators is calculated and the Shapiro Wilk normality test in R. The lower the p-value of this test, the less likely it is that the distribution of datapoints assumes (or approaches) a normal distribution.

Table 16: The p-values (rounded of to 4 decimal points, except for 'KG', which approaches 0) for the Shapiro Wilk normality test in R, to see if the distribution of the UoMs over the 33 operators approaches a normal distribution.

UoM	**p-value**
" "	0.0042
BAG	0.0102
BLI	0.0141
EA	0.0057
KG	9.191e-12
PAA	0.0135
PAK	0.0475

From Table 16 it follows that none of the p-values is higher than significance level alpha = 0.05, and thus none of the distributions of the UoMs over the operators follow (or approach) a normal distribution. When we calculate how the UoMs are distributed over the 98 days in the dataset (instead of over the 33 operators), the p-values are even lower (with the highest p-value for " " being 0.00003238). So, the UoMs can not be said to be normally distributed over the 98 days nor over the 33 operators.

As we can not use calculations that would be used given a normal distribution, it is interesting to see how the UoM data is spread out. If we go back to the distribution over the operators, we can see the spread of each of the UoMs over the operators in the boxplots in Figure 7. It must be noted here that the mean is calculated differently than in Table 15; In Table 15 the mean was calculated by dividing the handled products for each UoM by the amount of operators that handled them. However in the boxplots for each UoM the number of handled products is divided by the total number of operators. This is done to be able to say something about the behaviour of the operators handling goods and if this differs depending on the UoM, instead of looking at the exact throughput for each UoM as in Table 15. Furthermore, all of the boxplots are normalized to be able to compare them in the way they behave, as they then have the same range: between 0 and 1, this makes it easier to reason about them and visualize them in one image. The normalization formula [23] used for each UoM and each value x in that UoM, i.e. the value of each operator for that UoM, is: $\frac{x-min(x)}{max(x)-min(x)}$. This normalization is done to be able to compare different features to each other [23], in this case: the different UoMs and how each of them behaves in itself and compared to each other. This formula for the normalization is the most basic one, and it can be used to extend the feature to any range by multiplying the formula by a scalar or adding an offset [23].

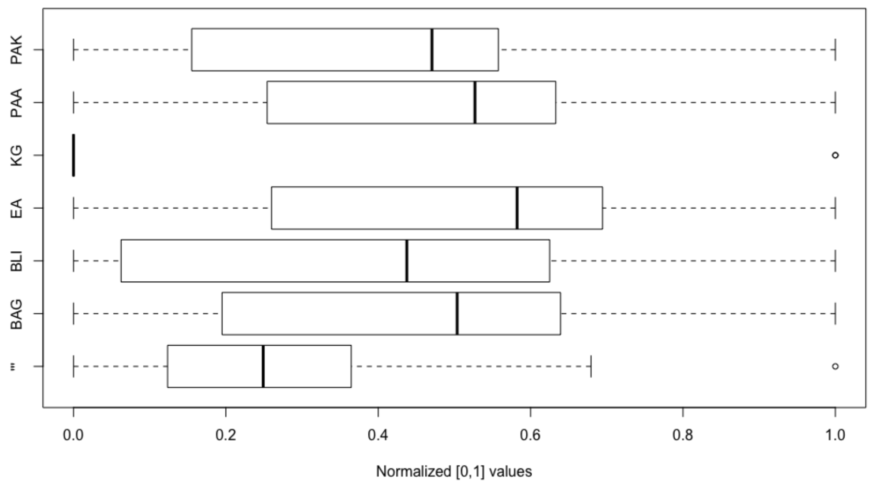


Figure 7: Boxplots, with normalized values, for the observations per operator of each of the seven UoMs that occurred in the dataset.

The black bars in Figure 7 show the median (the middle value) of each UoM, i.e. to the right and left of this line lie 50% of the values. Furthermore, the dots (only shown for KG and “$''$”) are outliers (a value that is higher than $\frac{3}{2}$ times the upper quartile), the end of the whiskers are the greatest and smallest value (it is logical that these have the values 0 and 1 for most UoMs, as they are normalized and their biggest and smallest value are set to 0 and 1). The beginning and ending of the box are also termed the first and third quartile and are the points where 25% (to the left) and 75% (to the left) of the data is at for that variable.

What you can see in Figure 7 is that for most UoMs, except “KG” and “ ”, the median lies at about the 0.5 point, which means that the data is spread quite evenly. The median however is on the right of the box for these UoMs. This indicates that the data is denser when more products are handled. However, as most of the whiskers to the right are longer than the left whisker, the spread for those operators that handle the top 25% of these UoMs is wider. The “KG” UoM is best left out as it does not have any spread: there are only 2 events with this UoM. Lastly, the “ ” UoM behaves quite like the others, except that it has an outlier. If this outlier would be left out, the spread for the 1^{st}, 2^{nd}, and 3^{d} quartile is quite even, and it’s median would (probably) be around the 0.5 point as well. However, the last quartile, i.e. the operators that handle a lot, of events is sparser than the other three.

When we look at how the UoM events handled is spread out over the 98 days that the factory is opened, this presents us the boxplots as shown in Figure 8.

The values in this distribution are also normalized following the same formula as before.

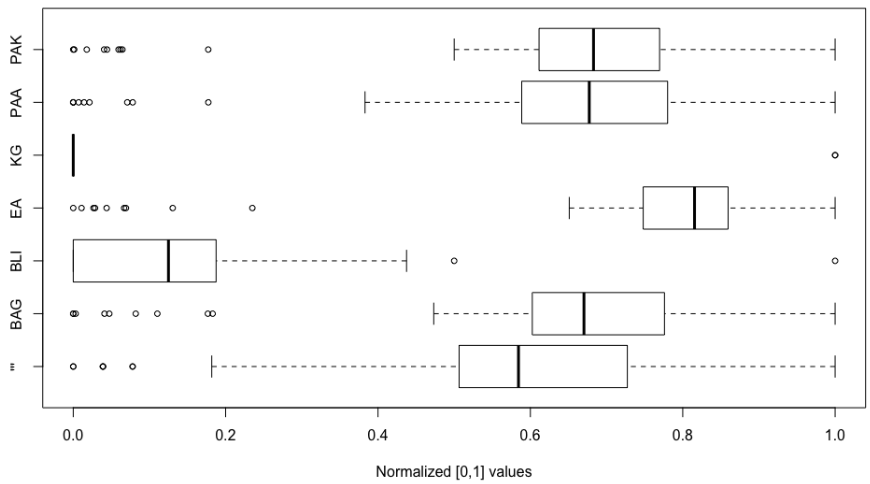


Figure 8: Boxplots, with normalized values, for the observations per day (98 in total) of each of the seven UoMs that occurred in the dataset.

From Figure 8 you can deduce that there are a lot of outliers, and that the boxplots for 'KG' and 'BLI' deviate from the others. For the UoMs without 'KG' and 'BLI', the outliers are mostly to the left, which indicates that it is common that there are days on which a low, or very low, amount of events for a particular UoM are handled. For example, for 'BAG' there are quite a few days when the number of 'BAG' events is zero. Furthermore, for these UoMs the whiskers are quite long, indicating that the days when there is a very low or very high number of events with that UoM are quite spread out, while the middle 50% of the days is quite dense, and steady.

Then, the 'KG' and 'BLI' UoM each have a small number of events that they cover (just 2 for 'KG', and 225 for 'BLI', which is 4,154 away from the number of events of the closest other UoM). This explains why it is - relatively - more common that there is a low or very low amount of events handled for these UoMs than that there is a high amount handled. The high amounts in these cases are outliers.

3.5.2 Specified and Processed Quantities

Before we go into depth in the specified and processed quantities, it is useful to know in greater detail what these variables entail. Each event (of the 816,362) in the dataset has a specified and processed quantity, this implicates that one event (i.e. pick, order, task) consists of differing quantities. This observation is relevant for later results as well, as the throughput can be thought of as not only the amount of events handled but can also be calculated as the amount of quantities that make up an event.

The specified quantity to be picked is (probably) what is put into the system beforehand, i.e. what is determined by the managing department of the factory as what is to be processed (depending on the incoming orders). This specified amount can differ from the quantity actually picked, i.e. what is really processed on a workday. This can be less (it is never more) due to different circumstances (e.g. a workstation is out of order, the amount of workers that is scheduled is too low or too slow etc.). We do not know what the reason for the difference is.

When we go back to the question at hand, we observe in the dataset that for 10,625 of the 816,362 events the specified quantity is more than the processed quantity. Hence, this is the case in $\frac{10{,}625}{816{,}362} = 1.30\%$ (rounded of to 2 decimal points) of the total events. Furthermore, the total number of specified quantities is $12,150,374$, and the total number of processed quantities is $11,900,716$. This means that there is a difference of $249,658$ between the two. Consequently, this implicates that of the $12,150,374$ specified quantities a number of $249,658$ quantities does not get processed which is $\frac{249{,}658}{12{,}150{,}374} = 2.05\%$ (rounded of to 2 decimal points) of the total specified quantities. These $249,658$ "missed" quantities cover a range of, as said, $10,625$ events. Which means that on average per "missed" event $\frac{249{,}658}{10{,}625} = 23.50$ (rounded of to 2 decimal points) quantities are "missed". (Note: from now on we will use the term *missed event* to describe the events where the specified quantity differed from the processed quantity)

Table 17: For each UoM the number of events where the specified and processed quantities differed, and the percentage of this from the total number of events for each UoM (rounded of to 2 decimal points).

UoM	**# missed events**	**% from total**
" "	4,379	$\frac{4{,}379}{4{,}379} * 100 = 100.00$
BAG	117	$\frac{117}{20{,}319} * 100 = 0.58$
BLI	1	$\frac{1}{225} * 100 = 0.44$
EA	5,821	$\frac{5{,}821}{732{,}300} * 100 = 0.79$
KG	0	$\frac{0}{2} * 100 = 0.00$
PAA	65	$\frac{65}{8{,}728} * 100 = 0.74$
PAK	242	$\frac{242}{50{,}409} * 100 = 0.48$

The most striking result from Table 17 is that all the " " (i.e. empty) UoM events are, in fact, missed events (where the specified quantity differed from the processed quantity). This result implies either that the empty " " UoM is in fact empty, because the quantity that was specified to be handled was not handled at all or that these cases have no packaging, which makes it very unlikely that all of the events with this packaging are missed. If we assume the first reasoning to be correct, this means that the empty UoM should be left out when we consider throughput in the next section, as it aliases how much work is actually done.

Next to this conclusion, Table 17 does not give much information other than that for the rest of the UoMs the percentage of events where there is a difference between specified and processed quantities is very low (never more than 0.79%). This is logical, as the empty UoM events are already $\frac{4{,}379}{10{,}625} = 41.21\%$ (rounded of to 2 decimal points) of the total number of missed events.

When we look if there are more missed events for certain operators, we encounter that there are 31 of the 33 operators that have performed missing events. Or, in other words, 2 operators did not miss any events. The range of how many events the 31 operators have missed lies between 6 and 843, which is quite a large range.

Using the Shapiro Wilk test in R to test for normality, i.e. if the distribution of missed events is equal over the 31 operators that have missed events, a p-value of 0.1641 is calculated, which is higher that 0.05 and therefor we can say that the distribution approaches a normally distributed. You can see the histogram of the density of the 31 operators that have missed events in Figure 9. As you can see the curve goes somewhat up, and aims at a bell curve, but is quite flat and wide.

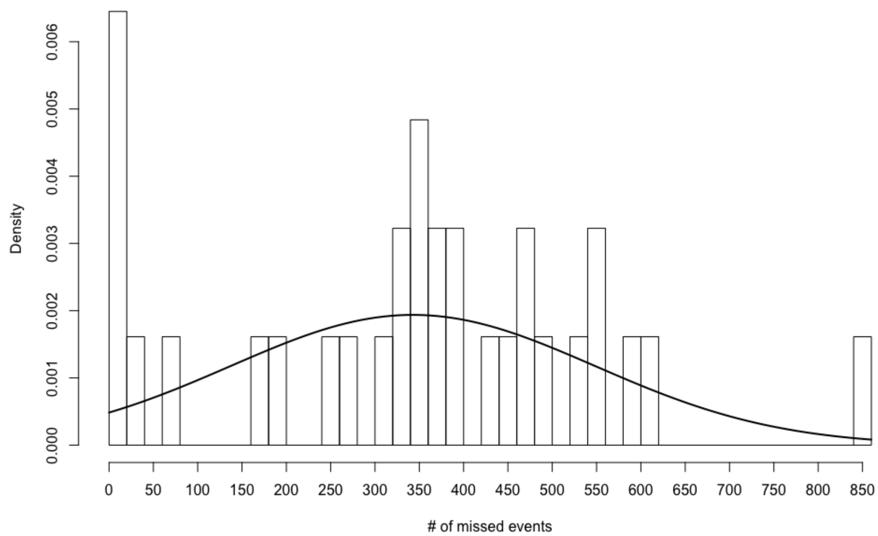


Figure 9: Histogram of the density of the 31 operators that have missed events.

In Table 18 you can see if the missing events occur more on certain days of the week. At first glance, the frequency for the Tuesday seems to be quite higher than for other days of the week. But when we calculate the relative frequency, there are in fact the most events missed on the weekenddays. Furthermore, relatively the fewest events are missed on Mondays. And, also relatively, there is still a peak for Tuesdays, if we only look at weekdays.

Table 18: The frequency of missed events for each day of the week, the number of times that each day occurs in the dataset, and the relative frequency (which is the mean for that specific day of the week) of missed events, by dividing the overall frequency of missed events per day by the number of times that the weekday occurs in the dataset, and the standard deviation (these last two parameters are rounded of to 2 decimal points).

Day of the Week	**Frequency**	**# Days**	**Relative Frequency** $(\frac{Frequency}{\#Days})$	**SD**
Monday	1,775	18	98.61	41.30
Tuesday	2,052	18	114.00	34.22
Wednesday	1,910	18	106.11	35.40
Thursday	1,960	18	108.89	33.45
Friday	1,959	18	108.83	41.40
Saturday	365	3	121.67	4.51
Sunday	604	5	120.80	25.59
Total	10,625	98	778.91	215.87

It is interesting to see if there is also a big difference in frequency between Mondays, Tuesdays etc. Because, it might be that there are a lot of missed events on Tuesdays, but that this was just because of a few incidents. Therefor, we look at the mean (i.e. relative frequency) and standard deviation for each of the days. This is also shown in Table 18. As you can see, the standard deviations for Monday and Friday are the highest, then in the middle there are Tuesday, Wednesday and Thursday, while the low and lowest standard deviations are on Sunday and Saturday respectively. The low standard deviations for the weekenddays are explainable as there are only 3 Saturdays and 5 Sundays in the dataset, and the amount of missing values for these days is high, but is around the same amount (its mean) on its days (the datapoints are dense, i.e. close to each other). The range for missed events on Saturdays is 9, and on Sundays 64, while it is 149, 167, 156, 160, and 157 for Monday to Friday respectively.

The relative frequency is normally distributed over the weekdays, with a confidence level of 95%. As the p-value of the Shapiro Wilk normality test is 0.6732 which is definitely higher than alpha = 0.05. That the standard deviation for Tuesday is not around the highest but also not the lowest, indicates that its high frequency of missed events is not just by chance, but might actually have a identifiable cause. However, if we use the t-test (as the variable is normally distributed) in R to see if the relative frequency value for Tuesday is higher than the mean relative frequency of the rest of the days, then this gives a p-value of

0.2107, which is higher than significance level alpha = 0.05, and thus the relative frequency of missed events on Tuesdays is the highest, but not significantly higher than for the other days. Furthermore, the relative frequency on Mondays is, with a confidence level of 95%, lower than on the other weekdays calculated using a one-sided t-test in R, which gives a significance level of 0.001377 this is smaller than alpha = 0.05.

The high amount of missed events on weekenddays is on average $\frac{121.67+120.80}{2} = 121.235$. Using the one-sided t-test in R to see if this value is significantly higher than for the other weekdays (Monday to Friday) gives, with a confidence level of 95%, a p-value of 0.002593. This p-value is lower than significance level alpha = 0.05, and we can therefor say that the relative frequency of missed events is, with a confidence level of 95%, higher on any of the weekends than on any of the other days.

3.5.3 Throughput

The throughput is the amount of work done per operator in a given time frame (i.e. window). We will look at the time frames of the whole period (all 98 days), for each week, each day, each weekday (all Mondays, Tuesdays, etc.), and for weekdays versus weekenddays. Also, the throughput is calculated with and without taking the processed quantities into account, to see if this has an effect (events that have larger quantities possibly take longer to process and influence throughput). Furthermore, the throughput calculations will be done with and without the events with an empty UoM, as seen in the previous section. For the same reason as before, to see if there is a difference, as we do not know if these empty UoMs mean that these events are not handled.

The total throughput can be looked at as either being the total of $816,362$ events, the total of events when quantities were processed, so minus the $5,122$ empty events (that were not handled i.e. the missing events), which is $816,362 - 5,122 = 811,240$ of the events. Or the events where quantities were processed multiplied by the quantities processed, which is: $11,900,716$. Dividing all of these by 33 (the number of operators at work during the 98 days that the dataset covers), gives a throughput of:

- $\frac{816,362}{33} = 24,738.24$, this is the total throughput over the 98 days per operator, including the events which had 0 processed quantities.
- $\frac{811,240}{33} = 24,583.03$, which is the total throughput over the 98 days per operator, without the events which had 0 processed quantities.
- $\frac{11,900,716}{33} = 360,627.8$, which is the the average number of quantities processed per operator.

For ease of reading, the three ways of calculating throughput will be termed: absolute, relative and processed throughput. Absolute throughput is the throughput calculated over all the events, including the events where there were no quantities processed. Relative throughput is the net throughput: all events minus the events where no quantities were processed. And lastly, processed throughput is the relative throughput multiplied by the quantities processed in each of the events.

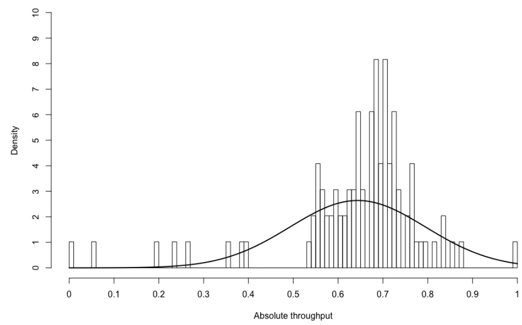
In Table 19 you can find the table for the p-value (from the Shapiro Wilk normality test in R), mean, standard deviation, confidence interval, skewness and kurtosis of the three throughput types calculated over the 98 days. The throughput is calculated for each day and then divided by the amount of operators to get the average throughput of that day per operator.

Table 19: For the absolute, relative and processed throughput of the total 98 days in the dataset, the p-value (from the Shapiro Wilk normality test in R), mean, standard deviation, confidence interval, skewness and kurtosis.

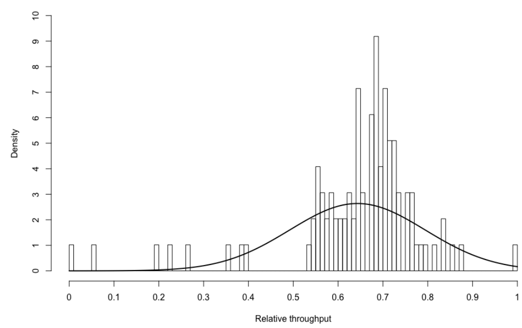
Throughput	**p-value**	**Mean**	**SD**	**Conf. Int.**	**Skew.**	**Kurt.**
Absolute	1.112e-09	417.54	97.80	1.00 - 648.00	-1.88	5.31
Relative	1.093e-09	414.90	97.29	1.00 - 644.50	-1.88	5.30
Processed	1.215e-09	6,021.16	1451.75	20.00 - 8,526.75	-1.93	4.84

What Table 19 shows is that none of the throughputs follows a normal distribution over the 98 days in the dataset. Furthermore, the means for the absolute and relative throughput are quite close to each other, as well as their standard deviations. For the processed throughput mean and standard deviation are bigger, which is logical as the relative throughput is multiplied with the processed quantities, which means that the values for this variable are larger.

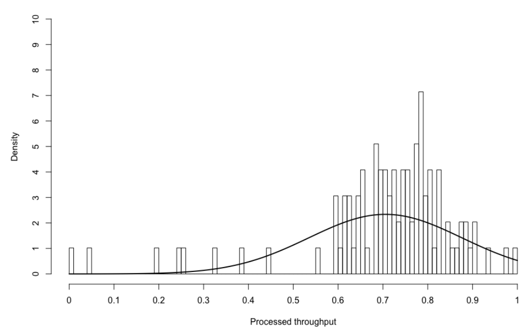
To interpret the skewness and kurtosis it is useful to look at the histograms for the three throughputs, which are shown in Figure 10.



(a) Absolute Throughput



(b) Relative Throughput



(c) Processed Throughput

Figure 10: The histograms and their curve for the absolute, relative and processed throughput when looked at as distributed over the 98 days in the dataset. The throughput is normalized and measured against density. The y-axis shows the density ranging from 0 to 10, and the x-axis are the normalized throughputs (from the formula used before for normalization) ranging from 0 to 1 for all three throughput types.

A high kurtosis means that the distribution is very spiked, and mostly indicates outliers and large deviation. From Table 19 you can see that the kurtoses are quite high (normal is around 0). When we then look at Figure 10, we see quite high spikes for each of the three throughputs, in particular there are - for all throughputs - spikes to the left of the highest spikes and the height of these spikes seems quite steady over the total range (0 to 1 in the figures). Also their deviations are quite high which also indicates large deviation (and possibly outliers). The processed throughput shows a somewhat lower kurtsosis, which we can see in Figure 10 as the curve being somewhat flatter (not much) than the other two curves.

Skewness is the meassure of asymmetry. Negative skewness indicates that the tail on the left side of the probability density distribution is longer or fatter than the right side. You can easily see that this is the case for all three throughput types distributed over the 98 days.

When we look at the absolute, relative and processed throughput per week, excluding the first day (August 18) as this day falls in another week, this gives the results in Table 20. Again, all the throughputs are calculated per operator.

Table 20: The absolute, relative and processed throughput per week (18 weeks in total), the throughputs are per operator (calculated by: total week throughput divided by the total number of operators (33), rounded of to 2 decimal points). The number of operators and days is also shown. There is no correction for the number of days, but as there will likely be more operators at work when there are more days in a week, this should not bias the results.

Week	**# Op.**	**# Days**	**Absolute Throughput**	**Relative Throughput**	**Processed Throughput**
1	108	5	386.49	384.33	5,690.15
2	121	6	367.33	364.82	5,316.04
3	101	5	428.30	426.08	6,105.40
4	107	7	501.03	497.55	7,403.88
5	96	5	470.05	467.30	6,796.12
6	105	5	421.43	419.37	6,254.14
7	97	5	473.07	470.49	6,675.92
8	103	5	458.41	455.85	6,741.08
9	110	5	412.95	410.45	6,300.81
10	112	6	432.15	429.63	6,026.46

Continued on next page

Table 20 – *Continued from previous page*

Week	**# Op.**	**# Days**	**Absolute Throughput**	**Relative Throughput**	**Processed Throughput**
11	107	5	423.42	420.84	6,454.33
12	108	5	429.65	427.12	6,257.56
13	102	6	456.91	453.79	6,567.97
14	110	6	452.66	449.04	6,483.27
15	102	5	460.06	456.84	6,607.59
16	91	6	417.84	415.03	6,014.86
17	103	6	430.19	427.17	6,166.19
18	80	4	400.10	397.19	5,437.49

When we take the data from Table 20 and see if the three throughputs are normally distributed over the 18 weeks, this gives the results in Table 21, using the Shapiro Wilk test in R.

Table 21: The p-values for the distributions of the three throughputs using the Shapiro Wilk test in R.

Throughput Variable	**p-value**
Absolute	0.9635
Relative	0.9684
Processed	0.8173

As the p-values in Table 21 are all definitely above 0.05, the throughputs are accepted as being normally distributed over the 18 weeks. Which is interesting, as they were not over the 98 days. So where there are deviations and spikes for the throughput distributions over the 98 days, they are less so for the throughput distributions over the 18 weeks. To verify this, the calculations of Table 22 are used. And as expected, the kurtosis and skewness of the throughput distributions over the weeks are lower than over the 98 days. The skewness values all approach 0, and also the kurtosis values are just a bit spiked (approaching 0 for the processed throughput). Furthermore, the standard deviations are lower, while the means do not differ that much from the means over the 98 days. What can be concluded is that when the dataset is divided into weeks, instead of looked at per day, then the throughput over the week levels out, and is somewhat the same, or at least normally distributed, over the weeks, while there are larger differences, deviations and outliers if we look at the throughput per day, meaning that days differ but weeks less so. On some days things might happen that level out in weeks.

Table 22: The mean, standard deviation, skewness, and kurtosis for the variables absolute, relative and processed throughput over the 18 weeks in the dataset.

Throughput Variable	**Mean**	**SD**	**Skew.**	**Kurt.**
Absolute	434.56	32.86	-0.05	-0.53
Relative	431.83	32.63	-0.06	-0.52
Processed	6,294.4	504.31	-0.06	-0.02

Having looked at the throughput per operator calculated per week and per day, we can conclude that the difference between the absolute and relative throughput is not distinct (their mean is different) over the 98 days and over the 18 weeks. With a confidence level of 95% the first is calculated using the Wilcoxon signed rank test with continuity correction in R (as the throughputs over the 98 days do not approach a normal distribution). This gives a p-value of 0.6676. And using the t-test to see if the dataset differ distinctly over the 18 weeks, also with a confidence level of 95%, gives a p-value of 0.804. Both p-values are greater than confidence level alpha = 0.05 and thus the absolute and relative throughput sets do not differ, with a confidence level of 95%, over the 98 days nor over the 18 weeks. Therefor, from now on we will only use the the relative and processed throughput. The relative throughput is chosen above the absolute throughput since we do not know the reason why some specified events are not processed, and therefor taking them into account when calculating the throughput does not seem proper. Furthermore, as the difference between absolute and relative is not big, it does not matter a lot which of the two we choose, and only looking at the relative throughput gives us enough information on the throughput of the events.

When considering the throughput on each day of the week, Monday to Sunday, separately, to see if there is a difference in throughput depending on the day of the week, the results are shown in Table 23. There are in total 18 Mondays, 18 Tuesdays, 18 Wednesdays, 18 Thursdays, 18 Fridays, 3 Saturdays and 5 Sundays in the dataset.

Table 23: The p-values for the distributions of the relative and processed throughputs using the Shapiro Wilk test in R, the mean, standard deviation, skewness, and kurtosis for each of the weekdays (Monday to Sunday) in the dataset. All statistics are calculated with and without the outlier of December 17, which is on a Sunday, when only one event was handled by one operator, with a quantity of 20. The Sunday results without the outlier are in italic.

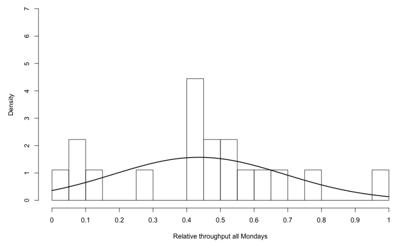
Panel A: Throughput for each day of the week for the relative throughput.

Day	**p-value**	**Mean**	**SD**	**Conf. Int.**	**Skew.**	**Kurt.**
Monday	0.5902	441.24	49.66	355.87 - 551.50	0.13	-0.42
Tuesday	0.1998	440.77	40.98	358.29 - 495.90	-0.61	-0.61
Wednesday	0.0007	404.91	83.11	125.67 - 494.45	-1.95	4.29
Thursday	0.2389	466.16	48.73	361.80 - 563.06	0.20	-0.31
Friday	0.1131	409.46	42.27	349.52 - 525.81	1.01	0.88
Saturday	0.1279	376.22	232.86	226.50 - 644.50	0.38	-2.33
Sunday	0.7083	121.07	100.99	1.00 - 247.67	-0.03	-2.00
	0.8482	*151.08*	*87.13*	*37.00 - 247.67*	*-0.22*	*-1.89*

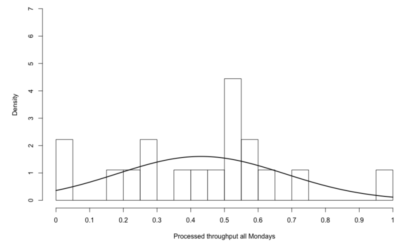
Panel B: Throughput for each day of the week for the processed throughput.

Day	**p-value**	**Mean**	**SD**	**Conf. Int.**	**Skew.**	**Kurt.**
Monday	0.5173	6,456.27	818.64	5,040.26 - 8,330.78	0.14	-0.26
Tuesday	0.3231	6,358.20	554.00	5,201.42 - 7,376.86	-0.51	-0.32
Wednesday	0.0024	6,039.77	1,217.63	2,067.67 - 7,541.55	-1.72	3.58
Thursday	0.9580	6,845.52	878.42	4,741.40 - 8,526.75	-0.28	-0.01
Friday	0.0140	5,869.95	596.26	5,046.71 - 7,678.90	1.41	2.14
Saturday	0.1733	4,544.92	2,604.76	2,811.17 - 7,540.25	0.37	-2.33
Sunday	0.7538	1,636.90	1,511.56	20.00 - 3,825.33	0.27	-1.78
	0.9287	*2,041.12*	*1,398.96*	*441.67 - 3,825.33*	*0.14*	*-1.91*

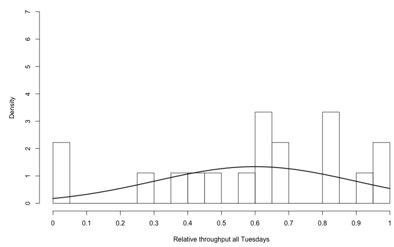
In Figure 11 you can see the distributions over the weekdays in histograms with their curve.



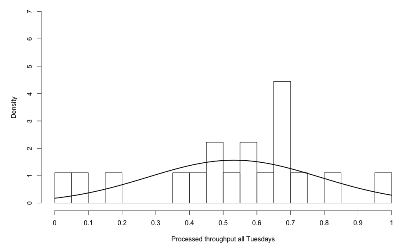
(a) Monday, relative



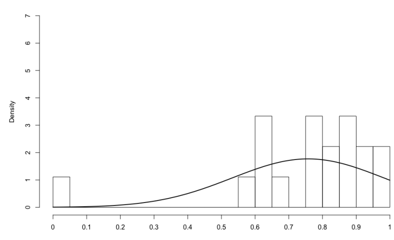
(b) Monday, processed



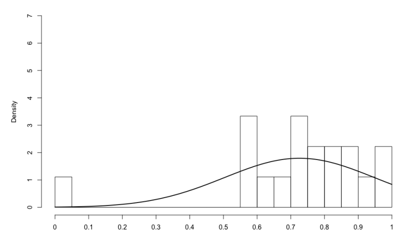
(c) Tuesday, relative



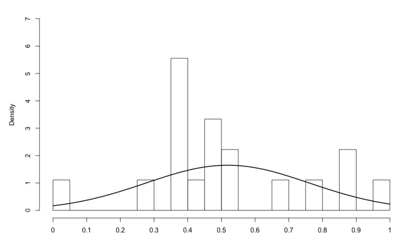
(d) Tuesday, processed



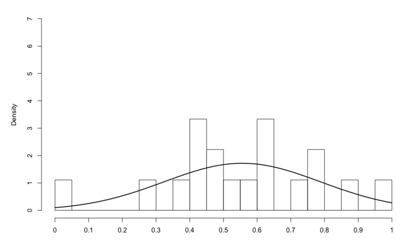
(e) Wednesday, relative



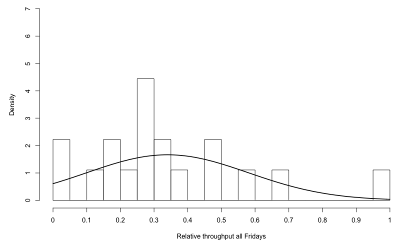
(f) Wednesday, processed



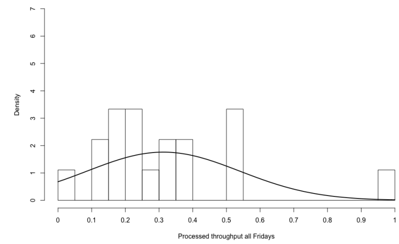
(g) Thursday, relative



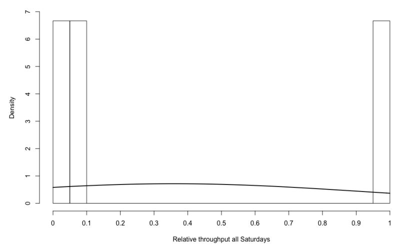
(h) Thursday, processed



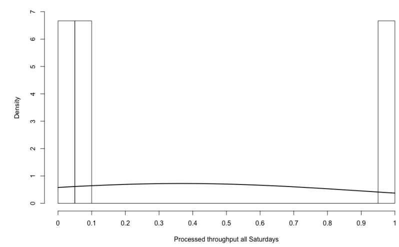
(i) Friday, relative



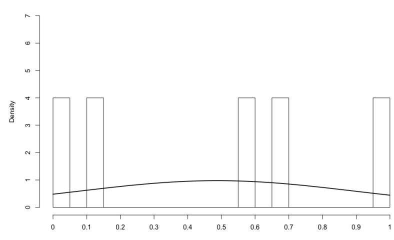
(j) Friday, processed



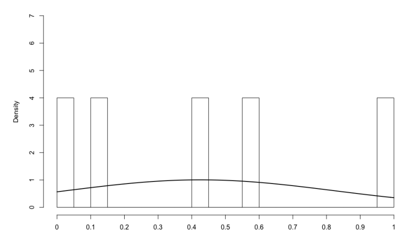
(k) Saturday, relative



(l) Saturday, processed



(m) Sunday, relative



(n) Sunday, processed

Figure 11: Images of the distributions of the relative throughput (the total number of events on a day divided by the total number of operators at work on that day) and the processed throughput (the relative throughput times the number of quantities processed for each event) for each day of the week combined. On the x-axis the label is: "Relative/Processed throughput all [Day of week] ", and the values are normalized so they all range from 0 to 1. On the y-axis the density is plotted ranging from 0 to 7 in all plots.

What is interesting from Figure 11 is that all weekdays (Monday to Friday) seem to somewhat approach a bell-curve, while the weekenddays (Saturday and Sunday) definitely do not. When we use the values of Table 23 and the plots in Figure 11, we can deduce what happens; For the relative throughput, all days approach a normal distribution except Wednesdays. For the processed throughput the Wednesdays but also the Fridays do not approach a normal distribution. This is remarkable, since the plots for the Saturdays and Sundays do not look like a normal distribution at all. The normality test is not trustworthy for these days, as it can only consist of outliers and still approach a normal distribution because there are simply not enough observations in the set. The large standard deviations for the Saturdays and Sundays unperpin this. The large standard deviation indicates that the datapoints in the set are prone to being outliers. Furthermore, the negative kurtosis of about -2 for the Saturdays and Sundays (for relative and processed throughput) entails that the shape of the curve is somewhat flatter: the peak is wider and the distribution has broader shoulders. When we take a look at the Saturday and Sunday plots in Figure 11, we see that the curve is indeed quite broad and the peak is wide. This is logical as there are only three Saturdays and five Sundays. Therefore, a difference - even small - has a large effect on the distribution. The effect of removing the outlier for the Sunday seems to have an effect on the p-value (even more normality! This is very likely biased) mean, standard deviation and confidence interval, but does not really seem to influence the kurtosis. It is striking that the kurtosis and skewness get bigger when removing the outlier in the relative throughput Sunday variable, but they get larger for the processed throughput Sunday variable.

For both throughputs the highest averaged throughput is on Thursdays. And Thursdays very much approach normality for the processed throughputs, but less so for the relative throughputs. When we look at the Thursdays in Figure 11 and compare these plots to the other plots it is clear that the kurtosis (peakedness) is quite normal as well as the skewness (the center of the curve is more or less in the middle). When we calculate if the mean value of the Thursdays is, with a confidence level of 95%, greater than the mean for the rest of the days, using the Wilcoxon signed rank test in R (as the distribution over the weekdays does not assume a normal distribution), the p-value is 0.01563 for the relative throughput and also 0.01563 for the processed throughput. As both p-values are < 0.05 we can say that the value for the Thursdays is, with a confidence level of 95%, greater than the mean for the rest of the days.

Furthermore, the plot in Figure 11 for Wednesday (relative and processed) clearly shows the skewness of the curve to the right, which corresponds to the observation that the Wednesdays have a high negative skewness, for both relative and processed throughput. Also, the mean relative throughput is lowest on Wednesdays with quite a large standard deviation (if we do not include Saturday and Sunday). However, using the Wilcoxon signed rank test in R, the mean relative throughput for Wednesdays is not significantly ($0.3438 > 0.05$) lower than the mean throughput for the other days. The skewness is, with a confidence level of 95%, less than for the other weekdays ($0.01563 < 0.05$ for both relative and processed throughput). And the same goes for the kurtosis ($0.01563 < 0.05$ for both relative and processed throughput).

When we look at Fridays for the processed throughput, there is a big difference with Fridays for relative throughput. For processed throughput the Fridays do not approach normality, have the lowest mean of the week (excluding Saturdays and Sundays) and have a high skewness and kurtosis. However, the mean is, with a confidence level of 95%, not lower than for the rest of the days ($0.5781 < 0.05$), using the Wilcoxon signed rank test in R. However the skewness is, with a confidence level of 95%, greater on Fridays than for the other weekdays for the relative ($0.01563 < 0.05$) and processed ($0.01563 < 0.05$) throughput. While the kurtosis is, with a confidence level of 95%, greater than for the other weekdays ($0.03125 < 0.05$) for the processed throughput, but not for the relative throughput ($0.2188 > 0.05$).

Furthermore, for the relative and processed throughputs both Fridays are more skewed to the left (see Figure 11), given the high positive skewness (see Table 23). Furthermore, what is striking on Fridays is that the kurtosis is a lot bigger for the processed throughput than for the relative throughput, which is also visible in Figure 11; When looking at the bins of the histogram the one for "Friday, processed" shows more higher peaks over a wider range.

When we dive deeper into the days of the week it is also possible to group the week- and the weekenddays. The results from the statistical calculations of this are shown in Table 24.

Table 24: The p-values for the distributions of the relative and processed throughputs using the Shapiro Wilk test in R, the mean, standard deviation, skewness, and kurtosis for the weekdays (Monday to Friday) and the weekenddays (Saturday and Sunday) in the dataset combined. All statistics are calculated with and without the outlier of December 17, which is on a Sunday. The weekend results without the outlier are in italic

Panel A: Throughput for the week- and weekenddays combined for the relative throughput.

Period	**p-value**	**Mean**	**SD**	**Conf. Int.**	**Skew.**	**Kurt.**
Week	4.289e-06	432.51	58.53	125.67 - 563.06	-1.37	6.73
Weekend	0.0913	216.75	196.87	1.00 - 644.50	1.02	0.06
	0.0620	*247.57*	*190.66*	*37.00 - 644.50*	*1.07*	*-0.09*

Panel B: Throughput for the week- and weekenddays combined for the processed throughput.

Period	**p-value**	**Mean**	**SD**	**Conf. Int.**	**Skew.**	**Kurt.**
Week	0.0003	6,313.94	895.46	2,067.67 - 8,526.75	-0.86	4.26
Weekend	0.3695	2,727.41	2,347.18	20.00 - 7,540.25	0.77	-0.45
	0.3441	*3,114.18*	*2,243.05*	*441.67 - 7,540.25*	*0.82*	*-0.52*

What is striking from Table 24 is that even though the weekdays (Monday to Friday) seemed to follow a normal distribution in Figure 11 (except for the Wednesdays), the p-value for the Shapiro Wilk test when they are combined is very low. The p-values for the weekend are however > 0.05 (= alpha) and thus for the weekend we can assume a normal distribution. However, this is of course arbitrary as there are only 8 values for the Saturdays and Sundays. If we instead of a t-test, use the Wilcoxon rank sum test with continuity correction in R, as it does not assume normality, the p-value for the relative throughputs, of week- versus weekenddays is 0.0005994 (< 0.05), indicating that the mean for the relative throughput is with a confidence level of 95% significantly different for the weekdays versus the weekenddays combined. For the processed throughput test between week- and weekenddays the p-value is 0.0004067 (< 0.05 (= alpha)), so the mean between week- and weekenddays is also significantly different with a confidence level of 95%.

If now, we remove one outlier from the Wednesday and calculate the Shapiro Wilk normality test again for the weekdays a p-value of 0.1643 comes out for the relative throughput, and of 0.4902 for the processed throughput. Both p-values are higher than 0.05 and therefor we can assume normality, from only removing one outlier! This shows that we need much more data to know for sure if certain effects and conclusions hold.

3.5.4 Perceived Workload & Productivity

A subject related to the throughput is the difference between events (or tasks) processed by each operator, i.e. the individual differences between operators. There are 33 different operators at work during the 98 days that cover the dataset. When we take a look at the productivity per operator, the number of days each of the operators works, the number of tasks he or she performs in total and their productivity (i.e. $\frac{No.Tasks}{No.Dates}$) the results are shown in Table 25.

Table 25: The number of dates and number of tasks that each of the individual operators has, and the productivity per operator (calculated by dividing the number of tasks per operator by the number of dates he or she works)

Operator	No. Dates	No. Tasks	Productivity
1	5	846	169.20
2	6	33	5.50
3	54	19,632	363.56
4	83	51,413	619.43
5	41	20,321	495.63

Continued on next page

Table 25 – *Continued from previous page*

Operator	**No. Dates**	**No. Tasks**	**Productivity**
6	59	26,724	452.95
7	1	728	728.00
8	52	21,395	411.44
9	1	441	441.00
10	65	32,806	504.71
11	48	20,608	429.33
12	75	29,430	392.40
13	9	4,163	462.56
14	81	38,718	478.00
15	65	25,627	394.26
16	89	37,224	418.25
17	1	591	591.00
18	70	42,783	611.19
19	86	38,473	447.36
20	80	34,151	426.89
21	84	46,442	552.88
22	75	27,010	360.13
23	87	34,291	394.15
24	81	34,165	421.79
25	82	33,527	408.87
26	80	33,913	423.91
27	89	37,289	418.98
28	89	39,692	445.98
29	81	34,871	430.51
30	5	1,858	371.60
31	61	13,075	214.34
32	10	3,463	346.30
33	87	30,659	352.40

In Table 25 we can see that there are quite large difference in the number of dates that operators work, ranging from 1 to 89 (of the 98 days in the dataset), and none of the variables in Table 25 follows a normal distribution (with p-values of 4.028e-05 for the number of dates, 0.005883 for the number of tasks, and 0.004459 for the productivity, all < 0.05 (= alpha), using the Shapiro-Wilk normality test in R).

It is interesting to see if the productivity of operators that work fewer days is lower. To calculate this the cutoff value is set at 11 days. There are 8 operators that work fewer than 11 days, which means that there are 25 operators that work more than 11 days, in fact these 25 work between 41 and 89 days. The mean productivity of the 8 operators that work few days is 389.39 with a standard deviation of 227.20. For the 25 operators that work more than 40 days the mean productivity is 434.77 with a standard deviation of 82.53. The standard deviation is lower for the operators that work more days, but this can be due to the fact that there is an outlier: operator number 2, which performs only 33 tasks in 6 days.

When we take a look at the mean productivity for the operators that work few days and those that work more days, and assume that the operators that work fewer days have a lower productivity, we can say with a confidence level of 95% that we can not say that this is the case, as the p-value is $0.3178 > 0.05$ (using the one-sided Wilcoxon rank sum test). In conclusion, if operators work fewer days this not mean that they have a lower productivity than operators that work more days. And the same holds for the other way around: if operators work more days this does not mean that they have a higher productivity than the operators that work fewer days ($0.6968 > 0.05$).

Furthermore, the perceived workload addresses if there are more or fewer tasks processed depending on the amount of operators at work that day. When we look at the amount of operators at work on each of the 98 days in the dataset, this amount ranges from 1 to 25 (and does approach a normal distribution with a p-value of 5.013e-15 being < 0.05 (= alpha) using the Shapiro Wilk normality test in R). There are 9 days on which 6 or fewer operators are at work, and the other 89 days there are eighteen operators or more at work. If we divide these days into two groups (one of 9 and one of 89) and look at their throughputs, then the mean relative throughput on days where there are fewer (than seven) operators at work is significantly lower, with a confidence level op 95%, than for days when there are more (than seventeen) operators at work, with a p-value of 6.519e-05 (< 0.05 (= alpha)). The same holds for the processed throughput, with a p-value of 4.579e-05 (< 0.05 (= alpha)) (both using the Wilcoxon rank sum test with continuity correction in R, as the variables do not approach a normal distribution). In conclusion, if there are more operators at work on a certain day the average throughput on that day is significantly (with a confidence level of 95%) higher, and thus it seems that perceived workload has an effect on throughput.

Lastly, we look at the productivity distribution per operator, this is done by using a boxplot for each of them, since this is the easiest way to describe their distribution and compare them in one image. The image is shown in Figure 12.

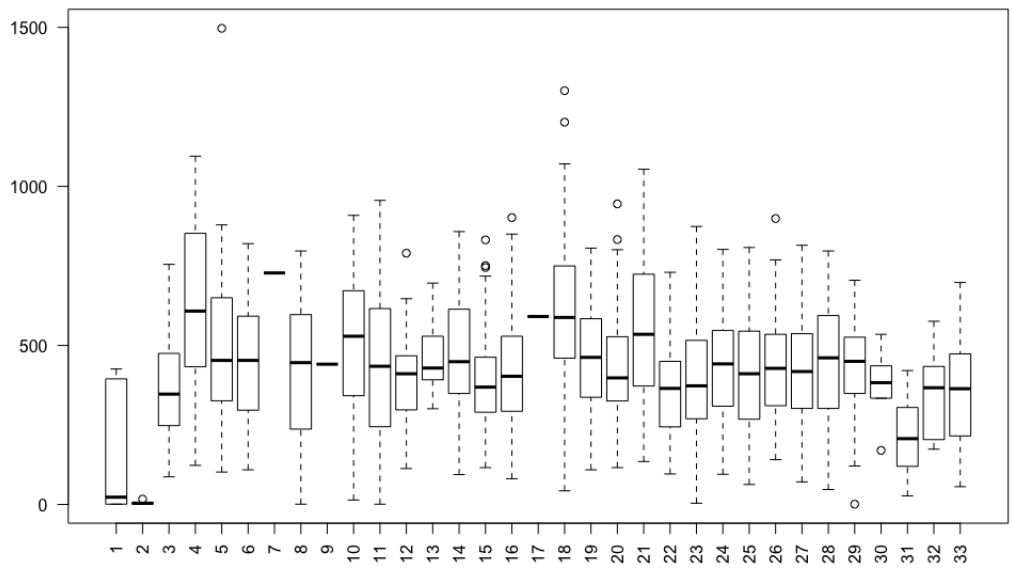


Figure 12: Boxplots for the distribution of the amount of work that each operator does spread out on each of the days that he or she works (ranging from 1 to 89 days, depending on the operator).

From Figure 12 you can immediate see that some operators work few days. These are the ones where you cannot really see the box, but rather just a line and maybe a dot. The operators that work less than 11 days are 1, 2, 7, 9, 13, 17, 30 and 32. And as you can see operator 2, 7, 9 and 17 only have a line or a line and a dot. Operator 1 is a special case in that he or she works 5 days in total, but on two of the days only handles 1 event, therefor the bottom whisker is almost not visible.

For the other operators (that work more than 40 days) the medians seem to lie quite closely together. indicating that the differences among operators are not that different if they work more than 40 days. However, when we look more closely there is still a big deviation. It would be interesting to see if it is possible to bring the spread for the operators more in one line over the operators. Outliers (the dots) are quite common, but lie mostly above the top whisker, indicating that this is an outstanding high throughput for that operator, which is not a bad thing. Furthermore, for most operators the boxes are somewhat more dense than their whiskers indicating that in 50% of their workdays they perform in a similar way, but in the other 50% they have a higher or lower throughput which values are more spread (so can be much higher or lower). In general the top whiskers seem longer for most operators, indicating that the spread of the workdays in which they have a high throughput is larger.

3.5.5 Weather

The five weather variables that are taken into account in our system are: (1) Precipitation amount, (2) Snow depth, (3) Air temperature, (4) Maximum temperature, and (5) Minimum temperature. The correlation coefficient of these five variables with relative and processed throughput is shown in Table 26.

Table 26: The correlation coefficients (using the Spearman's rank correlation rho in R) for each of the weather variables with the relative and processed throughputs (rounded of to 4 decimal points). The throughputs and weather variables are calculated per day (98 days in total).

Variable	**Cor. Relative Throughput**	**Cor. Processed Throughput**	**p-value Relative Throughput**	**p-value Processed Throughput**
Prec. amount	0.2869	0.2431	0.0042	0.0159
Snow depth	-0.0324	-0.0861	0.7518	0.3990
Air temp.	-0.0325	-0.0018	0.7506	0.9863
Max. temp.	-0.0675	-0.0175	0.5093	0.8640
Min. temp.	0.0319	0.0463	0.7553	0.6507

From Table 26 it follows that there is not a high correlation coefficient for any of the of the weather factors and the throughput, which is measured per day. Only the precipitation amount shows a little correlation (more than 25%), and this is the only value with a p-value of < 0.05 (= alpha). The other all have higher p-values and therefor the other variables seem to be uncorrelated.

3.5.6 World Events

As the world events from Table 13 are the world events selected from the period between August 18 and December 21 of 2017, but do not fall exactly on days that the factory was opened, two transformations - as proposed in section 3.3 - are applied. The results from these transformations is shown in Table 27. The dates are changed to the format $Year - Month - Day$, to be able to use it in R.

Table 27: The world event arrays as they are used in R. Array 1 gives each world event a value of 2 and each day before or after the world event a 1 (except for events that could not be known beforehand, such as crime events which do not get a 1 on the day before). Array 2 does the same except that it gives negative events a -2 and the day before or after a -1. The total of the days between August 18 and December 21 are shown, to make clearer how the arrays are made up.

All Days	**Date Factory Open**	**Date World Event**	**Array 1**	**Array 2**
2017-08-18	X	Crime	2	-2
2017-08-19				
2017-08-20		Crime		
2017-08-21	X		1	-1
2017-08-22	X		0	0
2017-08-23	X		0	0
2017-08-24	X		0	0
2017-08-25	X		1	1
2017-08-26		Holiday		
2017-08-27				
2017-08-28	X		0	0
2017-08-29	X		0	0
2017-08-30	X		1	1
2017-08-31	X	Misc	2	2
2017-09-01	X	Misc	2	2
2017-09-02	X	Misc, Sports	2	2
2017-09-03		Misc		
2017-09-04	X		1	1
2017-09-05	X	Sports	2	2
2017-09-06	X		1	1
2017-09-07	X		0	0

Continued on next page

Table 27 – *Continued from previous page*

All 98 Days	Date Factory Open	World Event	Array 1	Array 2
2017-09-08	X		1	1
2017-09-09		Sports		
2017-09-10		Sports		
2017-09-11	X		1	1
2017-09-12	X	Misc	2	2
2017-09-13	X	Sports	2	2
2017-09-14	X		1	1
2017-09-15	X		0	0
2017-09-16	X		1	1
2017-09-17	X	Sports	2	2
2017-09-18	X		1	1
2017-09-19	X		0	0
2017-09-20	X		0	0
2017-09-21	X		0	0
2017-09-22	X	Crime	2	-2
2017-09-23				
2017-09-24				
2017-09-25	X		0	0
2017-09-26	X		0	0
2017-09-27	X		0	0
2017-09-28	X		0	0
2017-09-29	X		0	0
2017-09-30				
2017-10-01				
2017-10-02	X		0	0
2017-10-03	X		0	0
2017-10-04	X		0	0

Continued on next page

Table 27 – *Continued from previous page*

All 98 Days	**Date Factory Open**	**World Event**	**Array 1**	**Array 2**
2017-10-05	X		1	1
2017-10-06	X	Sports	2	2
2017-10-07				
2017-10-08				
2017-10-09	X	Sports	2	2
2017-10-10	X		1	1
2017-10-11	X		0	0
2017-10-12	X		0	0
2017-10-13	X		0	0
2017-10-14				
2017-10-15				
2017-10-16	X		0	0
2017-10-17	X		0	0
2017-10-18	X		0	0
2017-10-19	X		0	0
2017-10-20	X		0	0
2017-10-21				
2017-10-22				
2017-10-23	X		0	0
2017-10-24	X	Crime	2	-2
2017-10-25	X		1	-1
2017-10-26	X	Misc	2	-2
2017-10-27	X		1	-1
2017-10-28	X		1	1
2017-10-29		Misc, Sports		
2017-10-30	X		1	1
2017-10-31	X		1	1

Continued on next page

Table 27 – *Continued from previous page*

All 98 Days	Date Factory Open	World Event	Array 1	Array 2
2017-11-01	X	Sports	2	2
2017-11-02	X		1	1
2017-11-03	X		1	1
2017-11-04		Holiday		
2017-11-05				
2017-11-06	X		0	0
2017-11-07	X		0	0
2017-11-08	X		0	0
2017-11-09	X		0	0
2017-11-10	X		0	0
2017-11-11				
2017-11-12		Holiday		
2017-11-13	X	Sports	2	2
2017-11-14	X		1	1
2017-11-15	X		0	0
2017-11-16	X		0	0
2017-11-17	X		0	0
2017-11-18				
2017-11-19	X	Sports	2	2
2017-11-20	X		1	1
2017-11-21	X		0	0
2017-11-22	X		0	0
2017-11-23	X		0	0
2017-11-24	X		0	0
2017-11-25				
2017-11-26	X		0	0
2017-11-27	X	Misc	2	2

Continued on next page

Table 27 – *Continued from previous page*

All 98 Days	**Date Factory Open**	**World Event**	**Array 1**	**Array 2**
2017-11-28	X		1	1
2017-11-29	X		0	0
2017-11-30	X		0	0
2017-12-01	X		0	0
2017-12-02				
2017-12-03				
2017-12-04	X		0	0
2017-12-05	X		1	1
2017-12-06	X	Holiday	2	2
2017-12-07	X		1	1
2017-12-08	X		0	0
2017-12-09				
2017-12-10	X	Sports	2	2
2017-12-11	X		1	1
2017-12-12	X		0	0
2017-12-13	X		0	0
2017-12-14	X		0	0
2017-12-15	X		1	1
2017-12-16		Sports		
2017-12-17	X		1	1
2017-12-18	X		0	0
2017-12-19	X		0	0
2017-12-20	X		0	0
2017-21-21	X		0	0

In Table 27 you can see how the arrays are build up. In the first array (Array 1), for each world event the day itself gets a 2 in the array and the day before and after the world event get a 1. However, since not all dates are represented in the Onninen dataset (i.e. the factory is not open everyday in the time period) sometimes the index of the array jumps from 0 to 2 or vice versa, or the 2 is not represented in the array because on the date of the world event the factory was closed.

In the second array (Array 2) the same approach as for Array 1 is used, with the difference that negative events, such as crime or accidents get a -2 on the date that it happens and a -1 on the day after. Note that crime events and accidents can not be known in advance and thus do not have a 1 (for Array 1) or -1 (for Array 2) on the day before it happened. Table 28 shows the correlation coefficients between the relative and processed throughput and Array 1 and Array 2 respectively.

Table 28: Correlation coefficients (rounded of to 4 decimal points) calculated in R of the relative and processed throughput with Array 1 and Array 2, i.e. the arrays that represent the presence (and absence) of world events.

	Relative Throughput	**Processed Throughput**
Array 1	-0.2550	-0.2917
Array 2	-0.1973	-0.2101

As you can see in Table 28, all correlation coefficients are negative. This means that if one of the correlation variables goes up, the other goes down or vice versa. Related to the world events it means that a world event negatively influences throughput. The correlation coefficient for Array 1 is somewhat higher, indication that there is more relation when the world events are not divided into positive and negative events. However, all four correlation coefficients are not high (at most about 30%), and so we can not speak of a correlation.

If we now make an array of just the world events (so only the events where Array 1 is 2), we get an array of 0s and 1s (0 for no world event, and 1 when a world event happens). There are 19 days with world events in the dataset. If we look at the difference between the throughputs on those 19 days compared to the throughputs on the remaining (98 - 19 =) 79 days and perform a one-sided Wilcoxon rank sum test with continuity correction in R, to see if their means differ significantly, the results are the following:

- With a confidence level of 95% we can state that the relative throughput on days when a world event happened was significantly less than on days when no world event happened ($0.03406 < 0.05$ (= alpha)).
- With a confidence level of 95% we can state that the processed throughput on days when a world event happened was significantly less than on days when no world event happened ($0.01937 < 0.05$).

When we now take all the 1s and 2s as belonging to a group (all 1s and 2s are being transformed to 1s), we divide the dataset into 46 days with world events and 52 days with no world events (as we can reason that days before and after world events also impact the throughput). If we now compare the means and say that the days with world events have a lower mean, the p-values are: 0.08754 for relative throughput and 0.04151 for processed throughput. So only the p-value of the processed throughput is < 0.05 (= alpha). In conclusion, we can state with a confidence level of 95% that the processed throughput on days that have world events and the day before and/or after is significantly less than on days that have no world events. It is interesting that the p-value for the processed throughput was not lower than our significance level alpha (= 0.05), and the p-value for processed throughput is. It could be the case that on days surrounding world events there are still orders, but that the quantity of those orders is different from days when there are no world events. It is interesting to research this further.

If we take a more thorough look at the behaviour of the divided datasets: those with (surrounding) and without world events (so the set of 46 versus 52), we get the results as shown in Table 29.

Table 29: The p-value, mean, standard deviation, confidence interval, skewness and kurtosis for the relative and processed world events versus the other events in the dataset.

Throughput	**p-value**	**Mean**	**SD**	**Conf. Int.**	**Skew.**	**Kurt.**
Relative, World Events	0.0001792	395.88	125.80	1.00 - 644.50	-1.27	1.91
Relative, Other Events	5.744e-06	431.72	58.52	148.00 - 537.72	-2.02	8.37
Processed, World Events	0.0001538	5,682.02	1,832.94	20.00 - 8,526.75	-1.32	1.64
Processed, Other Events	2.135e-06	6,321.18	921.2	1,711.50 - 8,003.94	-2.21	9.72

The most striking result in Table 29 is that the kurtosis values for both relative and processed throughput are very high for the other events (non-world events), even though their standard deviations and ranges are lower. Furthermore, they are more skewed. From that observation we can conclude that the datapoints are more spread over a smaller range and therefor the peak is more wide and the shoulders broader, but the standard deviation remains small. Furthermore, we can conclude from Table 29 that the mean, standard deviation, skewness and kurtosis are smaller for the days surrounding world events, than for days without world events, but that on the days without world events it is more difficult to draw conclusions and interpret what is happening: these days are more prone to outliers, and do not approach a normal distribution.

3.5.7 Economic Data

In Table 14 the data used as economical data is already displayed. The relative and processed throughput is also converted to cover the five months. This is done by accumulating the average throughput for each day in that month divided by the number of days in that month. The total dataset for the economic calculations is shown in Table 30.

Table 30: The CPI and annual change in % in Finland from August to December of 2017 [68], together with the relative and processed throughput per day in these five months.

Month	**Overall CPI 2015=100 Point figure**	**Annual Change in %**	**Relative Throughput**	**Processed Throughput**
August	101.1	0.7	379.88	5,610.67
September	101.3	0.8	428.18	6,232.79
October	101.3	0.5	448.30	6,476.46
November	101.6	0.8	418.19	6,096.30
December	101.5	0.5	366.89	5,241.80

As the dataset only covers five months, it is difficult to apply statistics on this data. However, we can take a look at the correlation coefficient. The correlation between CPI and relative throughput the coefficient is 0.014 (rounded of to 2 decimal points). And the correlation coefficient for CPI and processed throughput is -0.06 (rounded of to 2 decimal points).

Furthermore, the correlation coefficient of annual change and relative throughput is 0.16, and for annual change and processed throughput it is 0.25 (both rounded of to 2 decimal points).

3.6 Discussion

The aim of this chapter was to provide an information architecture using the Onninen factory dataset delivered by Vanderlande and external open world data. The focus lied on the heterogeneity of context data and the usage of possible open world data. This chapter serves as an initialization of a direction in the development of a system that uses context in the advancement of process optimization and to enhance the UX. Already at the introduction of this chapter it became clear that the dataset has a threefold drawbacks that make it difficult to perform straightforward analysis on the dataset: (i) a lack of data, (ii) a sparse dataset, and (iii) a non-normal distribution of the variables in the dataset. The lack of data and sparse dataset indicate that the real data is not normally distributed, and even if it is we cannot conclude this with enough confidence. A lot of tests, such as MANOVA and ANOVA, but also the t-test assume a normal distribution, which means these analyses are very hard to perform without first doing a lot of pre-processing (to minimize sparseness) on the dataset. Therefore, in this result section all the signified context factors are investigated separately, which means that there might be interaction effects between the factors as repeated t-tests and Wilcoxon ranked sum tests are performed, these effects can lead to overestimation and make the results less reliable. However, these separate tests showed such significance that we may consider them as important.

3.6.1 Units of Measure (UoM)

From the observations regarding the UoM we can conclude that the type of packaging are not evenly distributed over the number of events, the number of events handled with an 'EA' packaging is significantly higher than for the other UoMs. Furthermore, almost every operator handles all types of UoM and thus the distribution of work regarding UoM type is quite equal (yet does not approach normality). As we do not know more about what all the packaging sorts entail, and we also do not know anything about their sizes (we can have a small or big blister packaging) we should not make assumptions. We do know that the empty UoM is in fact empty, it contains no quantities (as the later section concerning quantities showed). There are only two operators that do not handle any empty UoMs. Furthermore, there are quite some days on which a low amount of a certain UoM is handled, these are outliers in the boxplot of Figure 7, and there are quite a lot. How to interpret this needs further research.

3.6.2 Specified and Processed Quantities

Only about 1.30% of all events differ in specified and processed quantity, which in effect means that these are events that were put into the system but were not handled for whatever reason. Of these events $4,379$ are events with an empty UoM, which means that they have no packaging (as their packaging is not specified). It is very unlikely that every packaging with an empty UoM gets specified but not processed (100%), which indicates that these events might not be designated to be processed at all. As we do not know what is the case, it was decided to be better to leave out the empty UoMs for the other results, or at least calculate certain events with and without them. Of the 33 operators, there are 31 that have missed events. The number of missed events for each operator lies within a range of 6 and 843, which is quite large. However, the distribution of missed events over the 31 operators that have missed events does assume a normal distribution. And so does the relative frequency of missed events distributed over the days of the week. The relative frequency of missed events is significantly lower on Mondays. This is interesting as one might reason that on Mondays workers are more efficient and faster in performing their work because they have just had a weekend in which they have rested or did leisure activities.

3.6.3 Throughput

The throughput is divided into absolute, relative and processed throughput, which is in effect: the total throughput (number of events handled per operator), the total throughput minus the events where no quantities were processed, and the total throughput minus the events where no quantities were processed multiplied with the number of quantities processed. None of the throughput types follow a normal distribution, and they are all skewed to the right and have a high kurtosis (a lot of high peaks, deviations and outliers). It became clear that the absolute and relative throughput did not differ significantly (with a confidence level of 95%). Therefore, the relative throughput variable was chosen to be used in the rest of the study over the absolute throughput, because the relative throughput leaves out events with no processed quantities (which seem to be events where no goods were handled). While the distribution of throughputs over the 98 days did not follow a normal distribution, it did so over the 18 weeks (leaving out the first date, August 18, as this date falls separately into an other week). Over the 18 weeks the skewness and kurtosis were also lower. What can be concluded is that when the dataset is divided into weeks, instead of looked at per day, then the throughput over the week levels out, and is somewhat the same, or at least normally distributed, over the weeks, while there are larger differences, deviations and outliers if we look at the throughput per day. If we now look at the throughput on each day of the week, Monday to Sunday, combined, then we see that for the relative throughput only the Wednesday does not follow a normal distribution, which is evident in its significantly greater negative skewness and kurtosis (which is also the case for the processed throughput), and also its relative large standard deviation. Furthermore, the Wednesday has a low mean, however this is not significantly lower than the rest of the days.

For the processed throughput, the Wednesday as well as the Friday do not assume a normal distribution. And the means for these two days are lower than for the other days as well (however not significantly). The skewness is significantly greater on Fridays for both relative and processed throughput, while the kurtosis is only significantly greater for the processed throughput. The skewness on Fridays is more to the left indicating more lower values. While the skewness on Wednesdays is more to the right, indicating more higher values. In each case the data is more spread out and does not approach normality. For the Wednesdays this might be due the fact that a lot of public Holidays take place on Wednesdays in Finland.

Thursdays have significantly the highest throughput, which might give reason to believe that the management of the factory Onninen should schedule in more operators on Thursdays, because they seem more productive then. However, we do not know *why* the operators have a higher throughput on Thursdays, so this should be investigated further. The Saturdays and Sundays in both throughput sets behave quite differently, but a thorough analysis is hard, as both sets combined only consider 8 values (3 for Saturdays and 5 for Sundays), therefore the statistic results they present are most likely not so trustworthy.

3.6.4 Perceived workload

The number of days that an operator works differs per operator, and can range between 1 and 89 days. The productivity per operator is calculated by dividing the total number of tasks per operator by the number of dates that the operator works. There are 8 operators that work fewer than 11 days, the others all work more than 40. The mean productivity for operators that work few days versus those that work more days is not significantly lower. However, the *throughput* is significantly lower when there are fewer than 7 operators at work on a certain day, indicating that perceived workload has an effect on throughput, i.e. the more other operators are at work on a certain day, the more throughput each operator produces that day.

3.6.5 External Open World Data

The weather data showed no correlation with the relative and processed throughput. Only the precipitation amount seemed to have a little correlation, even though it is no more than 30%. The precipitation relates to the falling of drizzle, rain, hail, snow etc. However, there seems to be a positive correlation, indicating that if there is precipitation, then the throughput goes somewhat up. This is interesting, but as the correlation coefficient is not so high, it might not really be of meaning.

The world event data showed no correlation with the relative and processed throughput. However, the throughput on days when world events happened is significantly lower, with a confidence level of 95%, than for days when no world event happened, for both relative and processed throughput. And the same holds for the days surrounding world events, but then only for processed throughput. It could be the case that on days surrounding world events there are still orders, but that the quantity of those orders is different from days when there are no world events. It is interesting and needed to research this further.

The economic data showed no correlation with the throughput. More information is needed to use the economical information because now the economical dataset only contains five datapoints which makes it difficult to make assumptions or use calculations on the CPI data and throughput per month. Data over larger periods of time would be interesting to investigate. The specific financial situation of the operators at work in the factory is interesting to gather, to - for example - see if know if the external data source (e.g. CPI) influences them the productivity of the operator.

3.6.6 Issues With The Dataset

There are several issues with the current dataset; First of all, a lot of variables are hard to interpret and therefor hard to use in our analysis. They are hard to interpret from the perspective of this research, since they are related to administration, whereas we seek to also identify the value of context factors that they are not currently tracking. In other words, most of the data consists of labels and IDs, which are relevant for tracking goods inside the factory and for stock keeping, but less so as *context* information. Secondly, we do not have a lot of information on the physical works environment *context* information such as 'Lighting conditions of workstation and around workstation', 'Materials used for the workstation' or 'Distance between workstations'.

Furthermore, the data is provided per workday, and we do not have information on other time units, such as at which minute of the day a task is executed. If we would have such information it would, for example, be easier to look at the productivity of each operator separately and at time frames of the day that were more productive in general. These examples show that the data in the provided dataset as is, does not provide a lot of *context* information that can be used in the striving for a *context-aware* system. What is also missing is personal data of the operators (their user profiles), which can help in deepening the understanding of the factory data. User profile parameters are (e.g.) age, gender, family situation, digital proficiency etc.

Additionally, when linking open world parameters (e.g. the weather, world events etc.) to closed world parameters (data delivered by a company), we still can not know for sure that it was the sunny weather or a football match that influenced the closed world parameters, or maybe it was just chance. Even if we see a correlation, this does not mean that it was truly there.

3.6.7 Future Research

From the results it follows that possible additional data that is deemed interesting are: more data in general (more data makes it possible to provide better analysis, so more data on what was delivered now by Vanderlande already makes it more interesting), more meta information on the variables in the dataset, physical environment parameters from the factory (e.g. workstation look and feel, music, noise, sound information etc.), user information (profile: e.g. age, gender, and preferences and emotions), and more specific time information (e.g. throughput per minute, hour etc., and the shifts of the operators (do they get tired at the end of their shift?)). Adding this information to the system heightens the chance of interesting results, makes the dataset less sparse and more likely to approach normality, and adds to the context-awareness and process optimization of the system.

To overcome the problem of not being able to know if open world data such as the weather really was the cause of the behavior of some other internal variable, a lot of extra research is needed. The company can for example make adaptations on the assumption that there is a correlation, and if these adaptations do not affect the process data then this means that the observed correlation was not - in fact - a correlation. Another approach is to build an initial model that can predict the effects of weather with a confidence measure. If these results are significant enough, a spearhead picking station should have its worker scheduling algorithm modified to account for the weather information in order to validate the results of the previous experiments. Preferably this facility is geographically nearby other facilities, which can serve as the ground truth.

Another interesting research direction that could add to the *context-awareness* and provides richer results is the usage of controlled experiments. An example of a controlled experiment is to play music in a warehouse and not play music in a similar warehouse and see if this affects performance in any way, as music has an influence on work and performance [71][72].

The amount of impact of the context factors and the interesting things touched upon here need future work to understand and make use of fully (i.e. to know how to best act on this information). A follow-up project can focus on the building of the whole context model, also with the decision module for adaptations (automatic or not). Including the decision module approaches a full context-aware system and testing it gives even more information on user experience regarding interaction with the system.

Finally, the prospect of sharing data (especially context information) between organizations without actually sharing the (potentially classified) data could be explored further. Sharing context information can help greatly in improving results; As there is more input - we are dealing with larger amounts of data - we can say more about the context factors that play a role and we can say more about types of situations. As such, we become more efficient in adapting to these situations in the appropriate way; Albeit for process optimization, UX or otherwise.

3.6.8 Conclusion

In conclusion, if there is more information (i.e. data) available in general about the world and individuals, and maybe even more environmental data of the factory (in this case study), then a system can be build that not only sees indicators of where process optimization can take place, but at the same time such a system can incorporate context and reach a (higher) level of context-awareness. It does so by means of a decision module, that can be build when more data is available. If this system is then tested under different circumstances, with for example user feedback, we can optimize it for user experience and process optimization. This report should thus serve as an initialization of what is possible with real world data, and what could be added, rather than as an end-report on context, process optimization and UX.

4 General Discussion

The aim of this thesis was to combine a thorough literature survey on context and context-awareness and use this framework as the building blocks of a practical implementation of such a system. Section 2 discussed the literature extensively. It gave a possible classification of context information, a set of requirements (combined from several papers), an overview of which context modeling approaches scored best on which requirements, and some measurements for the Quality of Context. However, when building the practical implementation with these guidelines as starting point, it quickly became clear that the real world data did not fit onto the blueprint of the theoretical world.

In the Computer Science domain the most used categories are roughly: user (human), space (location), time, virtual (computational), type of activity, and devices (hardware). The set of requirements covered a list of 23 demands, and the modeling approaches consisted of 8 types (ranging from Key-Value pair Models to Machine Learning Models). From the literature it also followed that with the Big Data era, two problems came along: heterogeneity and contextual data. Meaning that data has limited value when it is not paired with its context, and usually internal company data is not connected to other, external, universal data. And, data that makes up the *context* is mostly heterogeneous (in source and data type), and thus there is a need for a good way to process and integrate this data to be able to infer something about the *context* of the user [20] [45]. The practical implementation then mostly focused on these two aspects.

External, open world, data was added to the dataset from Vanderlande and includes: weather data, world events and economical information. The variables in the dataset from Vanderlande mainly mostly relate to the logistics in the factory, and the sensors are mostly virtual, there is no information on the users (in this case operators in the factory), the location is unknown, the date is the only time component interpretable, and the physical environment is also unknown (e.g. is there artificial or sun-light, is music playing in the background etc.). This shows that not all categories from the literature are covered. However, when the amount of operators at work on a certain day is integrated with the number of events handled, the number of quantities processed, and the number of other operators at work on that day, interesting context factors evolve.

Based on the results of the practical implementation, it became clear that context factors are related to worker competency. As such we know that there are 31 operators that have missed events, and these missed events are distributed normally over these 31 operators. The number of missed events is significantly lower on Mondays, which is logical as the operators then just had a weekend to rest or do leisure activities. The weekenddays show the highest amount of missed events, but as there are only eight weekenddays in the dataset, it is difficult to draw conclusions from that. The significantly higher amount of missed events on weekenddays should be researched further. It indicates that operators are less accurate on weekenddays, which are of course usually free days. So they might be tired, or not motivated on these days. However, eight measurements is too low a number to draw conclusions.

The throughput is calculated using the relative throughput (throughput per operator leaving out the missed events) and processed throughput (relative throughput multiplied with the quantity handled for that event). The throughputs over the total of 98 days shows some spikes and deviations in its distribution, it does less so if it is calculated over the 18 weeks in the dataset. This means that even though there are differences over the days, the weeks level this out and the weeks do approach normality. This makes it interesting to look at specific days in the week that perform in a certain way. Doing so shows us that the Wednesdays (for relative throughput) and Wednesdays and Fridays (for processed throughput) deviate from normality. There are a lot of Holidays in Finland on Wednesdays which might indicate the deviating behavior of the throughput on these days. The behaviour on Fridays needs further research, but might be due to the fact that it is almost the weekend: its processed throughput is lowest (not significantly though). Furthermore, the Saturdays and Sundays have a very low representation (three Saturdays and five Sundays), which makes the interpretation of their distribution difficult. The means for the week- versus weekenddays is, with a confidence level of 95%, significantly different (lower). However, much more data is needed to be able to draw trustworthy, justifiable conclusions, as removing one outlier can change a dataset from not approaching normality to assuming a normal distribution.

The productivity is not higher for operators that work more days. However, when there are more operators at work on a certain day then the throughput is higher, indicating that the perceived workload (imposed by the presence of others) has a positive effect on productivity.

The open world data (weather, world events and CPI) at first glance did not seem to correlate to the throughput (relative or processed). The precipitation amount (rain, drizzle etc.) did show some correlation, however this was not more than 25%. The CPI did not show any correlation, nor did the world events. However, when using the world events to divide the days in the dataset into days with world events (including and excluding surrounding days), it did show that the average throughput was lower in the group of world event days (including surrounding days) for relative and processed throughput, and was also lower in the group of (just the) world event days (excluding surrounding days) for processed throughput.

The effects that are measured from the practical implementation show that world events, certain days of the week and the perceived workload play a role in worker competency. The business process could be adapted to these findings. It would be very interesting to verify the findings of this practical implementation, by using (M)ANOVA. In order to do so, much more data is needed.

The requirements that idealistically should be met in a satisfying context-aware system are not all met in this real life system, mainly because the decision module is not implemented. Thus requirements distribution, adaptive, delivery, learning, reliability, and generic, are not met here. The most important requirements that are met are: heterogeneity, inference, incompleteness, lightweight, imperfection, privacy, relations, aggregation, usability, and validation. Furthermore, the requirements that are not relevant here are: preference compliance, visibility, distributed, context management, scalability, timeliness, efficiency, formality, and constraints. These are mostly irrelevant as there is no new information added and the application does not need to perform at runtime, but it rather is (for now) just the building block on which a decision module can be build. As you can see the "formality" requirement is also not met, meaning that as of yet the practical implementation did not implement a context model. However, for this dataset a lightweight context model would suffice at it is very specific to the distribution center. If the dataset is extended to more companies a more complex context model would be more relevant.

Furthermore, it would be very interesting regarding the context classification posed in the literature section (Section 2) to extend the dataset with more time, physical environment, user, and social information. As of now the dataset mainly comprises of variables that are interesting regarding the logistics in the factory. However, a context-aware system can make use of many more types of data to really be able to make a full computational model of this context. From the literature section it would be great to follow a top-down approach and say: "This is all we can do! Model this, model that and make it work". However, in practice we are dealing with a certain dataset and this offers restrictions and limitations on what is actually possible, this is the bottom-up approach. And in this thesis it becomes clear that these two approaches differ quite a lot from each other.

Today, mostly large companies, such as Google and Amazon, have very large datasets from which they can build recommender systems and context-aware systems that make use of virtual and physical context factors (e.g. smart homes, automatic feedback systems and the like). But if there would be a database covering a lot of datasets from different companies, groups, and individuals of all sorts, then data would not be mostly available for big companies, but context-awareness could be applicable in a much wider range of scenarios. The drawback from sharing data is that it touches upon privacy issues, but if this data is encrypted or depersonalized (as in: not traceable to a person or location), then a lot of applications can benefit from the uses of context-awareness.

Even though there is much to gain from the employment of context, it remains to be seen if what context-awareness brings forth is attractive and useful for the human user. As a context-aware system should aid in the tasks a user performs and not interrupt or obstruct the user, much more research into the UX of context-aware systems is needed. As we aim at building systems that behave like humans, it remains to be seen if this is what users want. We can not ignore the fact that humans are (up till today) superior in the performance of certain (semantically charged) tasks, where computers are superior in the performance of a different kind of tasks (processing and make calculations on many types of data and big datasets). Research should point out which hybrid - between humans and computers - is the most favorable to use (in which scenario). This might benefit process optimization and UX much more, than trying to bridge the semantic gap and try to make computers do what humans do. As maybe, making computers do what humans do, is not what humans want? More research into this field is very interesting, as we are still forming (through research and development) what context-aware systems should look like and do.

References

[1] Gregory D Abowd, Anind K Dey, Peter J Brown, Nigel Davies, Mark Smith, and Pete Steggles. Towards a better understanding of context and context-awareness. In *International Symposium on Handheld and Ubiquitous Computing*, pages 304–307. Springer, 1999.

[2] Gregory D Abowd and Elizabeth D Mynatt. Charting past, present, and future research in ubiquitous computing. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 7(1):29–58, 2000.

[3] Vikas Agrawal, Lokendra Shastri, Munindar Singh, and Tim Finin. Activity context representation: Techniques and languages. 2012.

[4] Jose Aguilar, Marxjhony Jerez, and Taniana Rodríguez. Cameonto: Context awareness meta ontology modeling. *Applied Computing and Informatics*, 2017.

[5] Jose Aguilar, Marxjhony Jerez, and Taniana Rodríguez. Cameonto: Context awareness meta ontology modeling. http://www.sciencedirect.com/science/article/pii/S2210832717301643, 2017. [Online; accessed 19-March-2018].

[6] V. Akman. Rethinking context as a social construct. *Journal of Pragmatics*, 32(6):743–759, 2000.

[7] Polzonetti Alberto. User-centric mobile services: context provisioning and user profiling. In *Proceedings of the 11th Annual International Digital Government Research Conference on Public Administration Online: Challenges and Opportunities*, pages 122–130. Digital Government Society of North America, 2010.

[8] U. Alegre, J. C. Augusto, and T. Clark. Engineering context-aware systems and applications: A survey. *The Journal of Systems and Software*, 117:55–83, 2016.

[9] Theodoros Anagnostopoulos, Christos Anagnostopoulos, Stathes Hadjiefthymiades, Miltos Kyriakakos, and Alexandros Kalousis. Predicting the location of mobile users: a machine learning approach. In *Proceedings of the 2009 international conference on Pervasive services*, pages 65–72. ACM, 2009.

[10] Danilo Ardagna and Barbara Pernici. Adaptive service composition in flexible processes. *IEEE Transactions on software engineering*, 33(6), 2007.

[11] J Augusto, Asier Aztiria, Dean Kramer, and Unai Alegre. A survey on the evolution of the notion of context-awareness. *Applied Artificial Intelligence*, pages 1–30, 2018.

[12] Okba Barkat and Ladjel Bellatreche. Linking context to ontologies. In *Semantics, Knowledge and Grids (SKG), 2015 11th International Conference on*, pages 57–64. IEEE, 2015.

[13] Paolo Bellavista, Antonio Corradi, Mario Fanelli, and Luca Foschini. A survey of context data distribution for mobile ubiquitous systems. *ACM Computing Surveys (CSUR)*, 44(4):24, 2012.

[14] C. Bettini, O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan, and D. Riboni. A survey of context modelling and reasoning techniques. *Pervasive and Mobile Computing*, 6(2):161–180, 2010.

[15] Cristiana Bolchini, Carlo A Curino, Elisa Quintarelli, Fabio A Schreiber, and Letizia Tanca. A data-oriented survey of context models. *ACM Sigmod Record*, 36(4):19–26, 2007.

[16] Felix Burgstaller, Dieter Steiner, and Michael Schrefl. Modeling context for business rule management. In *Business Informatics (CBI), 2016 IEEE 18th Conference on*, volume 1, pages 262–271. IEEE, 2016.

[17] Giacomo Cabri, Mauro Leoncini, and Riccardo Martoglia. Ambit: Towards an architecture for the development of context-dependent applications and systems. In *Proceedings of the 3rd International Conference on Context-Aware Systems and Applications*, pages 64–68. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2014.

[18] CL Philip Chen and Chun-Yang Zhang. Data-intensive applications, challenges, techniques and technologies: A survey on big data. *Information Sciences*, 275:314–347, 2014.

[19] Damien Cram, Béatrice Fuchs, Yannick Prié, and Alain Mille. An approach to user-centric context-aware assistance based on interaction traces. *MRC 2008, Modeling and Reasoning in Context*, 2008.

[20] Wei Dai and Jonathan J Liu. Context aware adaptive services framework. In *Service Systems and Service Management (ICSSSM), 2010 7th International Conference on*, pages 1–6. IEEE, 2010.

[21] Rogério De Lemos, Holger Giese, Hausi A Müller, Mary Shaw, Jesper Andersson, Marin Litoiu, Bradley Schmerl, Gabriel Tamura, Norha M Villegas, Thomas Vogel, et al. Software engineering for self-adaptive systems: A second research roadmap. In *Software Engineering for Self-Adaptive Systems II*, pages 1–32. Springer, 2013.

[22] María de Lourdes Martínez-Villaseñor and Miguel González-Mendoza. Sharing and reusing context information in ubiquitous computing environments. In *Artificial Intelligence (MICAI), 2014 13th Mexican International Conference on*, pages 227–230. IEEE, 2014.

[23] Horst Eidenberger. *Handbook of Multimedia Information Retrieval.* atpress, Vienna, 2012.

[24] Mostafa Erfani, Mohammadnaser Zandi, Juergen Rilling, and Iman Keivanloo. Context-awareness in the software domain—a semantic web enabled modeling approach. *Journal of Systems and Software*, 121:345–357, 2016.

[25] Seblewongel Esseynew, Berhanyikun Amanuel, Mekuria Sinke, Abel Damtew, and Saso Koceski. Context-aware technologies and systems. 2016.

[26] Maria Francesca Faraone, Michele Gorgoglione, and Cosimo Palmisano. Contextual segmentation: using context to improve behavior predictive models in e-commerce. In *Data Mining Workshops (ICDMW), 2010 IEEE International Conference on*, pages 1053–1060. IEEE, 2010.

[27] Fan Feng, Yusong Pang, and Gabriel Lodewijks. An intelligent context-aware system for logistics asset supervision service. In *Computer Science and Information Systems (FedCSIS), 2016 Federated Conference on*, pages 1147–1152. IEEE, 2016.

[28] Fabian Hämmerle, Patrick Nickel, Michael Doering, Johannes Merkert, Marcus Müller, Marcos Rates Crippa, and Christian Mannweiler. Evaluation of context management architectures: The case of context framework and context broker. In *Industrial Technology (ICIT), 2015 IEEE International Conference on*, pages 1870–1875. IEEE, 2015.

[29] Zardosht Hodaie, Natalia Mannov, Sergio Zamarripa, Yung Shin Van Der Sype, Markus Burvall, and Walid Maalej. D6. 1-conceptual model of user observation, application monitoring tools, and context actuators, 2013.

[30] José R Hoyos, Jesús García-Molina, Juan A Botía, and Davy Preuveneers. A model-driven approach for quality of context in pervasive systems. *Computers & Electrical Engineering*, 55:39–58, 2016.

[31] Interaction-Design.org. Context aware computing video 1: What is context aware computing. `https://www.youtube.com/watch?v=jYnViOb2K4A`, 2014. [Online; accessed 28-May-2018].

[32] Mladjan Jovanovic, Dusan Starcevic, and Zoran Jovanovic. Bridging user context and design models to build adaptive user interfaces. In Stefan Sauer, Cristian Bogdan, Peter Forbrig, Regina Bernhaupt, and Marco Winckler, editors, *International Conference on Human-Centred Software Engineering*, volume 8742, pages 36–56. Springer, Berlin, Heidelberg, 2014.

[33] Evangelos Karapanos, John Zimmerman, Jodi Forlizzi, and Jean-Bernard Martens. User experience over time: an initial framework. In *Proceedings of the SIGCHI conference on human factors in computing systems*, pages 729–738. ACM, 2009.

[34] Sotiris B Kotsiantis, I Zaharakis, and P Pintelas. Supervised machine learning: A review of classification techniques, 2007.

[35] Qihua Liu. Context-aware mobile recommendation system based on context history. *Indonesian Journal of Electrical Engineering and Computer Science*, 12(4):3158–3167, 2014.

[36] Mohcine Madkour, Abderrahim Hasbi, et al. Context-aware service retrieval in uncertain context. In *Multimedia Computing and Systems (ICMCS), 2012 International Conference on*, pages 611–616. IEEE, 2012.

[37] Vinícius Maran, Iara Augustin, and DB Saccol. A service for clinicspace architecture to provide context data persistence and context-based selection of documents. In *39th Seminar on Hardware and Software*, 2012.

[38] Shikha Mehta and Hema Banati. Context aware filtering using social behavior of frogs. *Swarm and Evolutionary Computation*, 17:25–36, 2014.

[39] Tarek Ben Mena, Narjès Bellamine-Ben Saoud, Mohamed Ben Ahmed, and Bernard Pavard. Towards a context definition for multi-agent systems. In *Personalized Information Retrieval and Access: Concepts, Methods and Practices*, pages 286–307. Igi Global, 2008.

[40] Boris Moltchanov, Massimo Valla, Carlo Alberto Licciardi, et al. Context management framework and context representation for mno. In *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

[41] Philip Moore, Bin Hu, and Jizheng Wan. Smart-context: A context ontology for pervasive mobile computing. *The Computer Journal*, 53(2):191–207, 2008.

[42] Philip Moore, Bin Hu, Xiaomei Zhu, William Campbell, and Martyn Ratcliffe. A survey of context modeling for pervasive cooperative learning. In *Information Technologies and Applications in Education, 2007. ISITAE'07. First IEEE International Symposium on*, pages K5–1. IEEE, 2007.

[43] Philip Moore and Hai V Pham. Intelligent context with decision support under uncertainty. In *Complex, Intelligent and Software Intensive Systems (CISIS), 2012 Sixth International Conference on*, pages 977–982. IEEE, 2012.

[44] Philip Moore and Hai Van Pham. On context and the open world assumption. In *Advanced Information Networking and Applications Workshops (WAINA), 2015 IEEE 29th International Conference on*, pages 387–392. IEEE, 2015.

[45] Yannick Naudet, Valentin Groues, and Muriel Foulonneau. Introduction to fuzzy-ontological context-aware recommendations in mobile environments. In *Proc. of 1st International Workshop on Adaptation, Personalization and REcommendation in the Social-semantic Web (APRESW 2010), 7th Extended Semantic Web Conference (ESWC 2010)*, volume 31, 2010.

[46] Yannick Naudet, Sabrina Mignon, Lionel Lecaque, Cyril Hazotte, and Valentin Groues. Ontology-based matchmaking approach for context-aware recommendations. In *Automated solutions for Cross Media Content and Multi-channel Distribution, 2008. AXMEDIS'08. International Conference on*, pages 218–223. IEEE, 2008.

[47] Asanga Nimalasena and Vladimir Getov. Context-aware framework for performance tuning via multi-action evaluation. In *Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual*, volume 3, pages 318–323. IEEE, 2015.

[48] Katsunori Oyama, Hojun Jaygarl, Jinchun Xia, Carl K Chang, Atsushi Takeuchi, and Hiroshi Fujimoto. A human-machine dimensional inference ontology that weaves human intentions and requirements of context awareness systems. In *Computer Software and Applications, 2008. COMPSAC'08. 32nd Annual IEEE International*, pages 287–294. IEEE, 2008.

[49] Cosimo Palmisano, Alexander Tuzhilin, and Michele Gorgoglione. Using context to improve predictive modeling of customers in personalization applications. *IEEE transactions on knowledge and data engineering*, 20(11):1535–1549, 2008.

[50] Emilian Pascalau. Identifying guidelines for designing and engineering human-centered context-aware systems. 2013.

[51] Direshin Pather, Janet Wesson, and Lester Cowley. *A Model for Context Awareness for Mobile Applications Using Multiple-input Sources*. PhD thesis, Nelson Mandela Metropolitan University, 2015.

[52] Hai Van Pham, Philip Moore, and Khang Dinh Tran. Context matching with reasoning and decision support using hedge algebra with kansei evaluation. In *Proceedings of the Fifth Symposium on Information and Communication Technology*, pages 202–210. ACM, 2014.

[53] Low Sui Pheng and Quek Tai Chuan. Environmental factors and work performance of project managers in the construction industry. *International Journal of Project Management*, 24(1):24–37, 2006.

[54] PV Pushpa. Context information modelling for internet of things. In *Contemporary Computing and Informatics (IC3I), 2016 2nd International Conference on*, pages 393–399. IEEE, 2016.

[55] William Revelle. Package 'psych'. `https://cran.r-project.org/web/packages/psych/psych.pdf`, 2018. [Online; accessed 2-april-2018].

[56] Ioannis N Roussos and Timos Sellis. A model for context aware relational databases. 2008.

[57] Oumaima Saidani and Selmin Nurcan. Towards context aware business process modelling. In *8th Workshop on Business Process Modeling, Development, and Support (BPMDS'07), CAiSE*, volume 7, page 1, 2007.

[58] Kurt Sandkuhl. Context modeling for knowledge management systems. In *WM*, pages 41–54, 2017.

[59] L. Serafini and P. Bouquet. Comparing formal theories of context in AI. *Artificial Intelligence*, 155(1–2):41–67, 2004.

[60] George Sielis, Aimilia Tzanavari, Roger Dols, Gabriel Hopmans, Peter Dolog, Klaus Schmid, Pascal Grube, Argyris Kouloumbis, and Stavroula Papavasiliou. D3. 1–description of context awareness in idspace. 2008.

[61] R Sudha, MR Rajagopalan, M Selvanayaki, and S Thamarai Selvi. Ubiquitous semantic space: A context-aware and coordination middleware for ubiquitous computing. In *Communication Systems Software and Middleware, 2007. COMSWARE 2007. 2nd International Conference on*, pages 1–7. IEEE, 2007.

[62] Mr Anil R Surve and Vijay R Ghorpade. Pervasive context-aware computing survey of context-aware ubiquitious middleware systems. *International Journal of Engineering*, 10(1):2017.

[63] Nexus Team. Reference model for the quality of context information. *Stuttgart Universität (February 2010)*, 2010.

[64] R Core Team. The r graphics package. `https://stat.ethz.ch/R-manual/R-devel/library/graphics/html/00Index.html`, 2018. [Online; accessed 2-april-2018].

[65] R Core Team. The r stats package. `http://stat.ethz.ch/R-manual/R-devel/library/stats/html/00Index.html`, 2018. [Online; accessed 2-april-2018].

[66] Shiu Lun Tsang and Siobhán Clarke. Mining user models for effective adaptation of context-aware applications. In *Intelligent Pervasive Computing, 2007. IPC. The 2007 International Conference on*, pages 178–187. IEEE, 2007.

[67] Unknown. Onninen Finland. `http://www.onninen.com/group/aboutus/Organizationstructure/Finland/Pages/Default.aspx`, 2011. [Online; accessed 7-March-2018, and 30-April-2018].

[68] Unknown. Statistics Finland » Prices and Costs » Consumer Price Index. `http://pxnet2.stat.fi/PXWeb/pxweb/en/StatFin/StatFin__hin__khi/`, 2018. [Online; accessed 10-April-2018].

[69] Unknown. Weather and sea » Download observations » Weather Observations » Daily observations. `http://en.ilmatieteenlaitos.fi/download-observations#!/`, 2018. [Online; accessed 10-April-2018].

[70] E. L. van den Broek. *Affective Signal Processing (ASP): Unraveling the mystery of emotions*. PhD thesis, Human Media Interaction (HMI), Faculty of Electrical Engineering, Mathematics, and Computer Science, University of Twente, Enschede, The Netherlands, 2011.

[71] E. L. van den Broek, J. H. Janssen, and J. H. D. M. Westerink. *Autonomous closed-loop biofeedback: An introduction and a melodious application*, chapter 35 (Section 5: Applications of Affective Computing), pages 472–482. Oxford Library of Psychology. New York, NY, USA: Oxford University Press, Inc., 2015.

[72] M. D. van der Zwaag, J. H. D. M. Westerink, and E. L. van den Broek. Emotional and psychophysiological responses to tempo, mode, and percussiveness. *Musicae Scientiae*, 15(2):250–269, 2011.

[73] Remi Verhoeven. Capitalizing on external data is not only an outside-in concept. `https://www.compact.nl/articles/capitalizing-on-external-data-is-not-only-an-outside-in-concept/#Reliability_of_external_data`, 2017. [Online; accessed 19-March-2018].

[74] Norha M Villegas and Hausi A Müller. Managing dynamic context to optimize smart interactions and services. *The smart internet*, 6400:289–318, 2010.

[75] Michael T Warmouth and David Yen. A detailed analysis of executive information systems. *International Journal of Information Management*, 12(3):192–208, 1992.

[76] Geoffrey I Webb, Michael J Pazzani, and Daniel Billsus. Machine learning for user modeling. *User modeling and user-adapted interaction*, 11(1):19–29, 2001.

[77] Qiang Wei and Zhi Jin. Service discovery for internet of things: a context-awareness perspective. In *Proceedings of the Fourth Asia-Pacific Symposium on Internetware*, page 25. ACM, 2012.

[78] Mark Weiser. The computer for the 21 st century. *Scientific american*, 265(3):94–105, 1991.

[79] Mark Weiser and John Seely Brown. The coming age of calm technology. In *Beyond calculation*, pages 75–85. Springer, 1997.

[80] Lian Yu, Arne Glenstrup, Shuang Su, and Yang Zhang. Building a context world for dynamic service composition. In *Pervasive Computing and Applications (ICPCA), 2010 5th International Conference on*, pages 336–341. IEEE, 2010.

[81] Stefan Zander and Bernhard Schandl. Semantic web-enhanced context-aware computing in mobile systems: Principles and application. *Mobile Computing Techniques in Emerging Markets: Systems, Applications and Services*, pages 47–96, 2012.

A Vanderlande Brainstorm

A.1 Introduction

On the 9^{th} of November, 2017 another brainstorm-session was held at Vanderlande in Veghel. At this meeting the following five people were present:

- dr Egon L. van den Broek, CRUX, Utrecht University
- René Scheffer, CRUX, Stroomt Interactions
- Michael Verheijden, Vanderlande
- Thomas Schoegje, CRUX, Utrecht University
- Marthe Hegeman, Utrecht University

Again, all the attendees work in, or have experience in, the field of User Experience and work or have worked with context. Thus they are experts in the field. The brainstorm was focused on the case of "operators in a picking station". Vanderlande builds automatic packaging systems for distribution centres. These are really big machines that take care of the transport of physical products in a distribution centre. An example of a system that Vanderlande produced is shown in Figure 13.

Figure 13: Picture of an assembly line system at Vanderlande in Veghel, picture taken on November 9, 2017.

The case study focuses on work done by so called *operators*. These are people that operate a workstations, i.e. terminal. Products in the warehouse are in crates in that are reachable by carts that move on automatic rollers and via little elevators. When an order is made by a customer the system knows which product(s) are part of the order, it selects these and moves them in the right order on the assembly line to the operator. It is the task of the picking operators of the workstations to select the product(s) that make up an order, put them in a crate and push a button to send the order. If, for example, there is an order from a web-shop of three products that are in a warehouse, then the automatic systems pick up the crates with the products in them that make up the current order. The three crates that are filled with products A, B and C are send to the workstation of the operator, he or she then sees on a screen how many products are ordered of each type. The operator then grabs the amount of products needed from the order from each of the crates and then puts it in another crate. If this particular order is complete, the operator pushes a button and the order is sent out to the customer.

The system automatically updates the number of products that should still be in the crate and puts it back in the warehouse. If a crate is empty that should be filled (there is a discrepancy in the amount of products) then the operator can push a button and a new crate comes with the same product.

In principle a system that Vanderlande delivers to a company is ready-made. However, the customer (company) can choose if they want the (ergonomic) ready-made workstation that Vanderlande produced, which is ergonomic and dampens vibrations etc., or if they want another type of workstation. The software that Vanderlande delivers is the same for each workstation for a client.

In general, operators have a low education level since it is not the most difficult or challenging job. Their education level can be lower than MBO. It says on the screen of the workstation how many orders should be taken out of a crate and put into another crate. Some operators do not speak Dutch or English, and the language can be adjusted, for example to Polish (a lot of operators speak Polish). Vanderlande always delivers the system in the language of the country where the company is located, however if the client requests so the system can be translated into other languages. And if different operators that operate the same workstation speak different languages, the language is changeable. This, however, is not logged due to privacy reasons. So there is not data on who is operating a workstation, but there is data on which language the system is operating in at time periods.

The amount of hours that a workstation is active per day depends on the client of Vanderlande (some companies are operative 24 hours per day, or they are in busy months). The operators work in shifts and they do not have a fixed workplace, so they are flexible in where they stand (this can differ from day to day or shift to shift).

There are reports that keep track of how fast operators work, this is shown per station. If the system is malfunctioning then the operators do not tend to let this know to their supervisors; It is their task to pick but not to notify that the system is not working.

You can see how long the system is active, if it is not active for a short period of time (break), how much is processed, and its state (if it is malfunctioning or not). On the level of the warehouse you know the discrepancy of what came in and what went out.

An operator works on one order at a time, this order needs to be completed (a button needs to be pushed to change to this state) and then the next order comes. The system works on multiple orders at the same time, but that is *behind the screens* for the operator. The products of the order are automatically put in the right order.

The packaging of the goods is done automatically. So as a picking operator the only task is to put the automatically delivered products at your workstation in another crate and push a (send) button.

Operators mostly work on-call, so when it is needed they are called to ask if they can come to work. Almost never are operators picking eight hours in a row, mostly they work for four hours in the mornings (for example) and then do something different in the afternoon.

There is no data available on why a supervisor is called, only the status information of a workstation is available. And, if a workstation was malfunctioning, who solved it and when.

The workstation-process is continuous, but the work is done in shifts, and if one operator has a break then another operator does his or her work in the mean time.

A.2 The brainstorm-session

The general types of context that were said to be distinguishable at the beginning of the session are:

- Physical environment
- Computational environment
- State (of system and people)
- Context events
- User (e.g. profile, social environment etc.)

The free brainstorm-session delivered a table of context dimensions and parameters as shown in Table 31.

Table 31: Overview of the outcome of context dimensions and context parameters at Vanderlande.

Context Dimensions	Context Parameters
Physical Environment	Noise, music, sounds (e.g. do employees have own music or not, sound of the machines) Temperature

Continued on next page

Table 31 – *Continued from previous page*

Context Dimensions	**Context Parameters**
	Weather Geographical location Travel distance Light conditions of workstation and around workstation Materials used for the workstation (the workstation offered to clients by Vanderlande are made of wood or bamboo: feels nice, works nice) Quality of the workstation (ergonomic, or not) Distance between workstations Type of product(s) Size of product(s) Variety in products Variety in look and feel of product(s) Look of the crates where the product(s) are in Size of workstation Type of environment where the company is stationed (urban, rural) Level of supervision
Computational Environment	Workload; throughput (how many product(s)) Language of the system Performance of tests (to see if operators are awake and alert) Terminal Type of input device Response time Gamification Frequency of errors Time analysis (e.g. peaks, football) Interface Possibility of reporting errors Feedback (customer ratings)
State	Number of workstations in use at a time State of the workstation (e.g. active, malfunction, on break etc.) Frequency and duration of malfunctions Seriousness of system errors

Continued on next page

Table 31 – *Continued from previous page*

Context Dimensions	**Context Parameters**
	Type of branch/warehouse/customer (e.g. supermarket or *bol.com* etc.) Variation of work, people, shifts, workstation, products Public image of the company Time of week Time of day Time of year Personal state: • Familiarity with the system • Familiarity with the terminal • Recognizability of the product(s)
Events	World events (e.g. football, 'Sinterklaas', Christmas, Ramadan etc.) Staff turnover Absence due to illness Type of shift(s) Duration of shift(s) Break(s) in shift(s) If your picking station goes well, but others do not

Continued on next page

Table 31 – *Continued from previous page*

Context Dimensions	**Context Parameters**
User	User profile: • Date of birth/ age • Gender • Family situation (children, relationship etc.) • Beliefs/ religion • Nationality • Cultural background • Level of education • Type of education • Physical limitations • Mental limitations • Health • Smoking • Digital proficiency • Amount of experience the operator has with operator work • Type of contract (permanent or temporary) Social: • Form of social affiliation • Form of appreciation by company/employer (company excursion, bonus etc.) • Social environment (e.g. alone or with others) • Different groups of people or fixed shift teams • Atmosphere Emotional state Tiredness; diminished focus Sense of responsibility Level of responsibility required by employer Appreciation for the company (employer) Relationship and appreciation of the employee with/for the product

A.3 Prioritising the Context Parameters

In Table 32 you can see the prioritisation of the context parameters. The parameters in **bold** are those that are deemed quantifiable and where data is available, and the ones in *italic* are quantifiable but difficult to acquire or measure. The regular ones are not so easily available or quantifiable.

Table 32: Hierarchy of context parameters.

No. post-its	**Context Parameters**
3	**Day of the week** **World events**
2	Appreciation for the company (employer) **Time analysis, peaks** **Quality of the workstation** Age, gender, education
1	**Type of product(s)** **Variety in products** Level of responsibility required by employer Family situation **Type of environment where the company is stationed (urban, rural)** *Variation of work* **Distance between workstations (mainly the social aspect that could be involved here)** *Staff turnover* **Time of day** **Workload; throughput** **State of the workstation (e.g. active, malfunction, break etc.)**