# On designing context-aware applications

## Past the phenomenological perspective

Author: Jaak Vlasveld
Date: October 30, 2009
Key words: context awareness, phenomenology, design, ICT

# On designing context-aware applications

## Past the phenomenological perspective

A thesis written under the supervision of

Johnny Søraker, MA
Prof. dr. Philip Brey
Dr. ir. Bert-Jan van Beijnum

in partial fulfillment of the requirements for

the degree of *Doctorandus*

*in de*

*Wijsbegeerte van Wetenschap, Technologie en Samenleving*

equivalent to a Master's degree

in

Philosophy of Science, Technology and Society

Twente University

Enschede, the Netherlands

2009

# *Abstract*

Context awareness is a design approach in computer science that creates computer applications that take the situation of its users into account. These applications rely on their additional means to interact with the world, such as sensors, actuators, and networking facilities, which are used to determine the contexts. This contextual information is used by context-aware applications to pursue their users' goals more effectively and efficiently than their non-context-aware counterparts do. Recently, initiatives surfaced to address the ad-hoc nature of context awareness as a research field, the most prominent one being an approach that is based on phenomenological principles. This thesis contrasts the phenomenological approach with the classical approach to context awareness, evaluating to what extent their methods are suitable to achieve their aims, and outlining the underlying assumptions of both approaches. After this evaluation, new assumptions for a phenomenological approach to context awareness are proposed and philosophical theories are introduced to support such an approach.

The evaluation shows that the phenomenological approach is unlikely to fulfill the needs of designers of context-aware applications and that the role of phenomenology in the design of context-aware applications needs to be reassessed. Context awareness is an engineering approach, which fits well to the main method of the classical approach: to cooperate with end-user experts in designing applications to function in specific contexts. The phenomenological approach, however, invites designers to focus on the theoretical side of how people process contexts, which would fit better to artificial intelligence than to the engineering paradigm of context awareness. Using theories from the phenomenologist Dreyfus, this thesis argues that context awareness can benefit from phenomenology, but only if its theories are used to support the designers of context-aware applications in their cooperation with end-user experts. Dreyfus argues that in some activities, people have more expertise than computers can have, and a phenomenological approach can be used to benefit optimally from the expertise of the designers, the users, and the end-user experts. Theories from Ihde are introduced to argue that a context-aware application is designed to perform sensomotoric actions for its users, and that a designer should decide if the application is closely related to its user or to the world itself, and if the user will focus his attention on the application or on the world. Searle's theories are used to explain why it is difficult for an application to know its user's goals and to identify the moments where an application should ask for additional user input. Finally, it is argued that these theories are indeed able to support designers of context-aware applications because they allow them to combine theoretical and practical expertise in novel ways.

# *Preface*

The possibilities and limits of technology have always fascinated me, as long as I can remember. My first steps in science were the exploration of the sky as a twelve-year-old boy: during the day using a self-made pinhole sunspot projector, in the evening with binoculars and a borrowed constellation map. At that time, I had already taken my first steps into computer science; I cannot even remember the age at which I programmed my first animation on a second hand Commodore, but it could not have been long after I got my first bike. However, my best early scientific memories are the ones from the many Sunday morning discussions I had with my father. There is no doubt in my mind that these open minded discussions on everything that is important, from the Greeks to infinity, facilitated by a patient, caring mother with good thee, shaped how I up to today think that good philosophy should be performed: with passion and warmth, looking forward by taking a step back. Preferably around a solid, oaken table.

When I had to choose where to go for my studies, I considered Cognitive Artificial Intelligence in Utrecht, but in the spirit of first taking a step back, I decided it would be nice to separate the philosophical foundation from the technical knowledge. This is why I chose for Twente University, where the study on Philosophy of Science, Technology and Science was offered. At first, I even wanted to take a step even further back; doubting that computer science was on the right track, I had also started on a study in Physics. Realizing this was the mistake of a dreamer, I quickly switched to Telematics. This would be quicker, and Telematics was strategically right, because it shared the principle of building networks with the human brain. The first time this apparent overly ambitious plan resulted in anything concrete was when I performed my Bachelor's thesis at the Human-Computer Interaction-department. Under the supervision of the always inspirational Anton Neijholt and Rieks op den Akker, I was involved in determining the interactional ins and outs of a virtual, embodied presenter. The fascinating link between behavioral science and the design of computer applications that show signs of intelligence became apparent when Rieks presented me, shortly before my final deadline, an essay by the sociologist Goffman that perfectly summarized all I was able to deduce and more. Sociology: 1 – software modeling: 0. Within the same cluster of research projects, and in the same building, I found two other scientists willing to supervise a reflecting computer science student for a Master's thesis. Bart Nieuwenhuis, for whose company I currently work as a consultant, and Bert-Jan van Beijnum were involved in several projects that involved context awareness, and they were open to a reflective approach to determine more exactly what they were doing and what their challenge actually was. We were convinced that, in cooperation with Rieks, we would get some interesting results. I think that we did, and the path we followed in the creation of that thesis helped shape the challenge that was formulated for this thesis.

It was also during the writing of that thesis that I started involving my great friends in my academic endeavors, and I developed the gift or curse of turning my problems into everybody's problems. I was often warned that graduating is a lonely process, but up to this point, I see academic writing as a similar activity as our Sunday morning discussions: a cooperative effort that should be fun and challenging. If I have been too focused the last months to do justice to this principle I apologize: this would be completely my mistake, and please accept these acknowledgements as a first step of making up for that.

Bram Hendriks has always been very welcoming, and an amazing sounding board; nothing beats a conversation on your couch with therapeutic coffee. I feel this thesis is almost as much the achievement of Maarten Zeinstra as it is my own; from him, I gained so many interesting views on phenomenology and applying it, and on critical examination of ideas and technologies, that I often forgot to write it down. I hope you are satisfied with the result. Andrea Komornikova has been like a muse, but then one with a stick and a medical encyclopedia. My sister Mijke and my mother Annette helped me define the structure of and distinctions in chapter 2 and 3. Els van der Kar inspired me with interesting views on technology development in Living Labs. Peter-Paul Verbeek took the time to talk to me several times, and all of them were equally productive and interesting. Some very bright people, whom I know appreciate the charms of a solid, oaken table as much as I do, proofread, criticized, and supplemented my theories many times, including Astrid Molenveld, Lucas ten Napel and Richard Heermsink. Even more good friends were my emotional and rational outlet: sometimes things just need to be said before they can be true, and they need to be done before they can be fun. Sanne, Frans, Liesbeth, Wouter, Lisanne, Janneke: I would like to thank you as much as the people I mentioned above. And I should not forget Ideefiks, a location that I, when I was part of the board in 2003, learned to be the perfect location for any kind of good discussion.

But, of course, the ones I owe the most are my supervisors. Bert-Jan van Beijnum, thank you for still being the inspirational, passionate and open minded researcher that I remember from my previous thesis. Philip Brey, who influenced my thought process from as early as 2002, who impacted the structure, quality, and contents of my thesis in a big way, and without whom the contents of this thesis would be all over the map. However, most of all I would like to thank Johnny Søraker, who not only personifies all the qualities of a philosopher that I hold dear, but also proofed to be a good coach, guide, and friend. I appreciate the spirit in which you were able to keep me passionate about linking the practical promise of formal software design as the provider of building blocks to the dream of one day creating applications that suitably and comfortably complement our intellectual lives.

# *Index*

# 1 Introduction

The purpose of this thesis is to research the design of context-aware applications[1]. Context awareness is a young research field that is currently maturing, which involves an evaluation of the approach by which context-aware applications are being designed. The classical approach to context awareness is being contested by a phenomenology-based approach, and although both approaches have the same goals, they have different methods for achieving these goals. I will compare the methods used in the two approaches, deduce their underlying assumptions, and present an improved set of assumptions in the form of philosophical theories that are linked to the goals of context awareness.

The next section will introduce context awareness. After that, the focus of this thesis will be introduced: the adoption of a methodology that brings about better integration of context awareness research in other relevant fields of research. Finally, this chapter will present how the remainder of this thesis will help improve this integration.

## 1.1 An introduction to context awareness

Imagine an application that recognizes when it is too dark for you to read its display properly, after which it adjusts its light settings accordingly. Or, imagine that you tell your mobile phone to print the picture that you are viewing and your mobile phone automatically prints it at the nearest available printer, after which the printer just as automatically selects the right means to bill you for the printing. Or, imagine a meeting recording system that pauses the recording during meeting breaks. Such systems do exactly what we want them to do and as such they are the ideal systems as far as both users and creators of computerized systems are concerned. But today, computer users still experience the applications they use as rigid and inflexible, and the information that the applications present is often irrelevant or downright wrong. Although, it could be that technological developments might finally have improved to a point where steps towards achieving these ideals can be taken. Computer applications are nowadays no longer limited to software programs that run on a computer and display results on a screen;

[1] Earlier, I wrote a MSc thesis on the role and meaning of context and contextual information in designing context-aware applications. That specific thesis focused on which concepts were used in the design process and what methodologies context awareness used in order to tie together all the elements of designing the applications. While that thesis focused on the challenges and difficulties that designers are faced, the current thesis analyzes and criticizes a particular development in the design praxis: the search for a common conceptual framework and the people and the values that are involved in using such a framework. So, while my previous thesis focused on the “what”, this thesis focuses on the “how”. For my previous thesis, see “Vlasveld, J. (2007). On context in context-aware applications.”

applications can take all sorts of forms and shapes, ranging from mobile phones to vacuum cleaners with computer chips inside. They are also more and more present in all parts of our lives. This creates new opportunities for building smarter devices by designer a better type of applications. Many computer scientists have perceived this as a new reason to tackle the issue of inflexibility of applications that misunderstand its users. Context-aware applications are expected to make a valuable contribution to this endeavor. The intersection of networking technologies, sensor technologies and computing technologies as an engineering effort has received little attention from philosophers. An exploration of context awareness might prove an interesting greenfield for similar design fields.

Context awareness builds on recent developments, such as those described above, with as a stimulating force the advancements in interaction technologies. It is difficult to explain what context awareness is by giving a definition: researchers of context awareness do not agree upon such a definition, the literature does not define what it means for an application to be “aware”, and the question as to how to define “context” is not answered clearly. However, Anand Dey presented a working definition that is widely adopted. His view is that applications are context aware when they use context to interact more usefully than when they would not use context. Context is defined by him as follows[2]:

*“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.”*

Context awareness is a research field that aims to help designers of software applications to make their applications more robust, effective and efficient. Context awareness can help design such applications by the systematic use of means to perceive what is happening in the world, such as interpreting thermometers, camera images, flight time tables or other sources of information. It should also systematically use means to impact on the world, such as displaying or altering information, ordering products or operating mechanical tools. Context awareness uses sensors and actuators to achieve the goals of its user in a better way. The belief is shared that applications become better when they display behavior that seems right considering the context.

An example of a context-aware application is a projector that is sensitive to the level of illumination of the room in which it is being used. Imagine a projector that is operating in a room that is illuminated relatively brightly. It would have to project an image that is more bright than average. If this projector can sense the level of

[2] Dey, A. K. (2001, February). Understanding and using context.

illumination and can draw the conclusion that it should adjust its projection brightness, the projector could be called context-aware (with the room's illumination being a contextual factor). The assumption is that the user of the projector does not merely want the projector to project an image on a screen, but, rather, that the user actually wants to be able to see the image, and the projector can achieve that goal better if it acts on the level of illumination of the room. This is a typical challenge for context-aware applications and their designers.

## *1.2 A challenge to designing context-aware applications*

The previous section introduced the view on context awareness that is widely shared between all the researchers in this area. And, for a while, research on context awareness was indeed based on shared methods. Recently, a new methodology was proposed: an approach that objects to the formal engineering approach and proposes to replace it with an approach that has aims that are more scientific. This section will introduce this challenge, and it will argue that this challenge threatens to introduce a conflict in context awareness' research programme: by introducing the frame problem, it can introduce a problem that cannot be solved by the engineers working on context awareness. This thesis will set out to avoid this problem while salvaging the benefits of this new approach.

Context awareness has its roots in computer science. More specifically, it was first explored by specialists in computer-computer interaction and specialists in human-computer interaction. The term context awareness was coined by Schilit in 1994, when he was working on a technical solution to enabling interaction between mobile devices. Dey refers to this widely respected paper by Schilit in his own, equally famous 1999 publication[3] on the definition of context and context awareness. That Schilit's paper was published in "the number one most-cited journal in telecommunications"[4] illustrates both the technical nature of the research and the fact that context awareness is a challenge that was first taken up by the telecommunication research field. This field is known for its strict engineering methodology, with researchers who create computers, networks, and software according to well-defined designs in a way that their correct functioning should be verifiable. Furthermore, their products should in the first place be useful and reliable.

Sometime later, this classical approach to context awareness started to receive criticism from the research field of human-computer interaction. Their most popular publication platform, the *Human-Computer Interaction* journal, published, for example, a

[3] Dey, A. K. and G. D. Abowd (1999). Towards a better understanding of context and context-awareness. p. 3

[4] http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=65

paper by the eminent computer scientist Dourish on the assumptions that underlie the classical approach to context awareness. His statement was that context awareness would benefit from adopting a foundation based on phenomenological principles. He argued that this was necessary because context awareness should be seen as something that goes beyond traditional computer science alone: it should also be related to other fields of science. Being both a computer scientist and an anthropologist, Dourish was in a good position to evaluate context awareness in this way: he has an excellent grasp of the needs of the context awareness community and he is intimately familiar with phenomenology. Dourish believes that non-phenomenological approaches often miss relevant information about the world. According to phenomenology, all classical sciences focus too much on the knowledge that is accumulated by these sciences, such as the laws of physics, behavioral rules, and other scientific facts. This point of view takes seriously the position that scientific facts can be wrong or incomplete, or that at one point in time one can reach different conclusions. Furthermore, phenomenologists believe that people will discover a lot of valuable information by discussing what they see from their own experiences. By careful inspection of their own experiences, they claim to be able to present information about these phenomena that other sciences cannot. It is the aim of the phenomenological approach to improve context awareness by incorporating this phenomenological knowledge. More specifically, the phenomenological approach to context awareness holds that a phenomenology friendly description of what happens when people use context-aware applications will enable designers to cooperate more easily with fields of science that have knowledge that is relevant to the design of context-aware applications.

Dourish, like many other researchers of human-computer interaction, has close ties with artificial intelligence. And, it is artificial intelligence that has received extremely harsh criticism from phenomenology. In as early as 1972, the phenomenologist and philosopher of mind Dreyfus presented a critique on artificial intelligence that even now describes a problem that artificial intelligence cannot solve: the frame problem. If context awareness wants to benefit from knowledge from human-computer interaction, it should be careful not to become susceptible to the same critiques as artificial intelligence was in its early days. Thus, the challenge that I shall set in this thesis is to combine the best parts of both the classical and the phenomenological approach to context awareness in a way that does justice to the warning that Dreyfus gave us. I will argue that what they will gain is a means to involve theories from other fields of science than computer science more easily, without running the risk of abandoning their engineering practices in favor of more scientific ideals. I will address this challenge by analyzing if there are parts of the phenomenological approach that are vulnerable to Dreyfus' critique. I will also outline the

assumptions that underlie the methods of both approaches, evaluate which parts should be kept and combined, and introduce a solution for the missing parts.

## *1.3 Research questions*

This thesis aims to answer the following questions:

- What are the assumptions that should underlie an approach to context awareness from a phenomenological perspective?
    1. What are the goals, methods, and assumptions of both the classical and the phenomenological approaches to context awareness, and who are involved in these approaches?
    2. To what extent can a phenomenological approach in general and this phenomenological approach in particular effectively address the weak points of the classical approach?
    3. Which new assumptions does context awareness need if it wants to benefit from a phenomenological approach?
    4. What are the implications of this new approach for context awareness and for phenomenology itself?

The previous section outlined the historical roots of the phenomenological approach's challenge to the classical approach, introducing the question how the design process is impacted by this challenge. The classical approach to context awareness is a viable approach, and the phenomenological approach claims that it can improve on the classical approach. The goal of this thesis is to investigate this claim to address the possibility that the challenge might result in an unwanted complication, and subsequently propose an approach that avoids that complication. To this goal, the research question is divided in four sub questions, each one of which will be discussed in a separate chapter.

The first question serves to illustrate the complexity of context awareness and provide the analysis that precedes the coming evaluation. The phenomenological approach proposes to do something different than context awareness was used to; this question serves to explore what they want to do differently, how they want to do it differently, and how this came to be. It will be argued that the classical approach relies on knowledge from end-user experts and that the phenomenological approach wants to rely on theories on how people process context.

The previous section stated that there is a reason to be careful with this new approach because this approach might be susceptible to its own difficulties. It was also argued that the approach might be better in some ways than the classical approach. The second sub question serves to identify what the dangers are of this phenomenological

approach to context awareness. It will be argued that phenomenology itself dictates that phenomenology should only be applied in context awareness with a limited scope.

As the previous section argued, both the classical approach and the phenomenological approach have strong and weak points, and they are to some extent incompatible. The third research question assumes that it is possible to create a new approach that combines parts of the two approaches and has more strong points and/or less weak points than either of the two approaches separately. The second research question resulted in a first assumption for such an approach. This third research question is about proposing a set of assumptions that are useful for fulfilling the initial goals for introducing a phenomenological approach.

The creation, introduction, and adoption of a completely new design methodology for context awareness is outside the scope of this thesis. The fourth and last research question, however, serves to illustrate that the newly introduced assumptions are compatible with current design practices in context awareness, and offer designers of context awareness the means and vocabulary to do more than they could do before the instruction of these assumptions, which are inspired by phenomenological theories and by logical analysis.

# 2 Context awareness: an overview

The introduction chapter gave a bird's eye view on the challenge of context awareness that this thesis will address. The goal of context awareness is to create applications that make life easier for people. Context awareness has a shared methodology to achieve this: to design applications in such a way that they gather information and use that information to assist its users to achieve their goals. During the initial years of context awareness research, the applications were created using typical and classical computer science approaches. However, recently a new approach was developed: an approach that is based on phenomenological principles. The task of the current chapter is to analyze both approaches, so they can be evaluated in the next chapter. This analysis will consist of three different steps. Firstly, the reader should get to understand the goals of context awareness better. The goals are shared by both approaches to context awareness and the methodology of both approaches should be fit to achieve these goals. The goals are best explained by describing the types of applications that are created by context awareness and presenting a scenario that describes the usage of such applications. The second step of the analysis consists of analyzing how the designers of context-aware applications acquire the specific knowledge they need to build context-aware applications. Using context is a complex process and if a designer wants to create an application that uses context, he needs to find a way to get the applications to use real information in actual situations. The designer can prepare the application for a list of situations, or he can create a generic algorithm for the application to deal with any situation. This is the main difference in the methods used by respectively the classical and the phenomenological approach. The third step of the analysis explains why the phenomenological approach introduced a shift in emphasis from cooperation with end-user experts towards relying on theories about how context can be processed. We will see that this new approach is the result of a different set of assumptions.

## 2.1 What are these context-aware applications?

The first step of the three-step analysis of context awareness is to determine which goals, methods and assumptions are shared by all researchers of context awareness. Dey presented us with the standard definition of context awareness. In chapter 1, it was stated that Dey defines a context-aware application as an application that uses context to interact more successfully, and he states that the context consists of the information that is relevant for the interaction between the application, its user, and its environment. While this is the only widely accepted definition, it does not give us much information on context awareness. It does raise a number of questions, such as "how should the application use context", "when is the interaction successful", and "how to determine

which information is relevant?" These questions are indeed hard to answer, and, as Dey also recognizes, these questions are best answered by giving examples of what these applications achieve for their users.

These context-aware applications address a specific need. Often, users of computer applications experience these applications as rigid and inflexible, and they feel that they do not know what their computers want them to do in order to get where they want to be. These computer applications are too demanding, and if they fail to do what their users want them to do, a situation arises where the computer and the users seem to be unable to understand each other. There are attempts to design applications in such a way that users can understand them more easily, and usually, these attempts are related to the research fields of user interface design and human-computer interaction. In context awareness, the aim is that the user does not necessarily have to understand computers and applications: the user should state his goals and the application should pursue them while requiring as little attention of the user as possible. This emphasis on goals and actions implies that context awareness is about more than good interface design or good interaction design: it is about good design in general.

The application does not plot its course by reading the mind of its users, but by determining the context in which the communication with its users takes place. It has a range of possible actions it can perform, and it has to perform those actions that are effective at efficiently reaching its users goals. Effective means that the goal must indeed be reached. Efficient means that it must not require unnecessary effort from its users. In one context, different actions might be more convenient to achieve a specific goal than in another context. The application should be designed to determine the appropriate action given a specific goal and the current context.

There is already some agreement on the methods that should be used to create such applications and their underlying assumptions. The main assumption is that computers are currently all around us and that they are more and more equipped with networking technology and other means to interact with the world. The interaction can consist of "reading" the world with, for example, cameras, temperature sensors, height detectors, microphones, et cetera. It can also consist of performing actions with, for example, printers, telephones, projectors, and even with actuators connected to locks and coffee makers. A method that is shared by all researchers of context awareness is using these possibilities and reading the world in order to act in the world in a way that helps its users. A context-aware coffee maker can, for example, decide to make espresso when the outside temperature is above 30 degrees while in other circumstances it would make a large cup of American coffee.

### 2.1.1 Three prototypical applications

While this helps us a little bit to understand what context-aware applications do for us, an overview of typical forms of context-aware applications will help to recognize them more easily. This paragraph will discuss three categories of context-aware applications that we might be able to encounter now and in the near future. These categories are artificially intelligent agents, mobile wearable devices, and ambient intelligence systems. While an application that fits one of these three categories might not always be context aware, they can typically be made context aware, and they can always benefit from some of the techniques that are employed by context awareness.

Artificially intelligent agents are systems that are seen as relatively autonomous: within certain bounds, they can act independently. Moreover, we perceive them as acting independently. They are agents that emulate behavior that is recognizable to humans, and as such are they susceptible to anthropomorphism, i.e., the users of the agents attribute typical human characteristics to them, such as having emotions, needs, and opinions. Sony's AIBO is an example of an artificially intelligent agent: a robot dog that in appearance and behavior partially resembles a real dog. When comparing the AIBO to the other two prototypical applications it is notable that the AIBO is an agent amongst many other comparable agents such as its user: the person with whom the AIBO communicates.

The second type of system is the mobile wearable device. A typical example of a mobile wearable device is the mobile phone, but more interesting examples are smart phones, such as the iPhone, and devices that are designed for specific applications, such as health-monitoring devices, since they have more extensive sensing and actuating capabilities. Central to the concept of mobile wearable devices is that it is worn on or near the body of a person during many of his activities. To its user and to other parties, it seems that the device and its user are usually in the same situation. An example is a bus application that is described by Korpipää and Mäntyjärvi[5]. A user has an application on his mobile device that displays the bus schedule. The mobile device's sensors measure it is going up and down very fast and that the mobile device is held at hand. From the clustered information, it is induced that the user is running, probably towards the bus. Subsequently, the display's colors and font size, and the amount of information displayed are adjusted to keep the screen readable for a running person.

The third prototypical application covers ambient intelligence. Such applications address the principle that computing devices and networks are omnipresent: small

[5] Korpipää, P. and J. Mäntyjärvi (2003). An ontology for mobile device sensor-based context awareness.

applications are present everywhere in our living areas and they can all be interconnected[6]. The aim is to support users in their activities in a way that requires as little interaction as possible. Context awareness is one of the means to implement this functionality. The resources that are available for such applications also enable the system to create functionality that is more complex. The two previously mentioned applications are typically more limited in resources; for practical purposes, a mobile wearable device, for example, has fewer sensors, and even an agent is more limited in the number of possible actions he can perform or initiate. Ambient intelligence can be a system that controls or uses artificial intelligence agents and mobile wearable devices, creating a trusted environment with these applications. The relation to its users is likewise an overarching one: it can be assumed that all user activity is potentially processed by the ambient intelligence system. The use of ambient intelligence is illustrated by the development of a virtual meeting room. Twente University's HMI group showcases such an application in the light of the FP6 AMI-project. The virtual meeting room can assist pro-actively during meetings by interpreting data from many different types of sensors[7]. Examples of its activity are to turn down the lighting in the room at the start of a presentation, to take minutes and to play back a representation of the meeting from various angles. The use of contextual information helps to process the sensor data and to reason about it. Artificially intelligent agents, mobile wearable devices and ambient intelligence systems have this in common: the aim and possibility to use the device's physical presence to assist users in achieving their goals.

### 2.1.2 A scenario

The previous sections shed some light, in both an abstract and more concrete manner, on what context-aware applications are supposed to do. A scenario where context-aware applications are introduced will serve to make this even more tangible: it will illustrate how context-aware applications can enrich our lives. Context awareness aims to produce technology that is accessible and robust, and thus it wants to make high-tech products suitable for introduction in mundane, everyday activities while assuming that traditional complex high-tech products often are not suitable for such activities. To give a proper illustration, this paragraph will introduce a scenario where a context-aware application is embedded in a mundane, everyday activity. A small part of the scenario will be the sound of the future, but the bigger part is technologically feasible today. In its entirety, the scenario will demonstrate that a context-aware application indeed relies on its sensors

[6] Brey, P. (2005). Freedom and privacy in ambient intelligence, pp. 167-158
[7] Rienks, R., A. Nijholt, and D. Reidsma (2006). Meetings and meeting support in ambient intelligence, pp. 2-3

and actuators to perform actions that fit to the situation and get its users closer to their goals in a way that requires less effort of its users than in the traditional scenario.

Let us choose for cooking as the central activity of our choice. Cooking is an everyday activity, and a convection microwave is a common kitchen appliance. I will consider a convection microwave with grill in a kitchen that can be equipped with additional high-tech interfaces, such as other sensory equipped kitchen appliances. We will see two scenarios. The first scenario is based on using the traditional convection microwave and the second scenario will involve the context-aware convection microwave. Cooking is a complicated process and even the simplest of activities requires the chef du jour to have practice, expertise, and a firm grasp of what he is and should be doing. The context-aware solution allows him to focus his attention both on the less technical parts and the more interesting parts of the dinner preparation process. From a designer's point of view, it is striking how the person using the context-aware application needs to be less involved with the technical details of the trajectory. It is similarly striking that typical human concepts are implicit throughout the entire process of cooperation between the user and the high-tech application, while advanced reasoning mechanisms are employed to utilize the information gathered by the context-aware application.

In short, we will see an application in action that is typical for context awareness in the way it processes information to act in order to make life less complicated for its user. At first, the two scenarios will appear to be strikingly similar, but the closer reading that follows will demonstrate how the traditional scenario is enriched by the typical context awareness features. This enrichment stems from how the user attention is directed away from the devices to the activities, which is made possible by the well thought out way the sensed information is combined to work towards the user's goals.

*Traditional scenario: John is preparing dinner. He will be serving grilled chicken with salsa verde to his three guests, who will arrive for dinner at 19.00. John defrosts the frosted chicken fillet in the microwave, and prepares the beans and eggs for the salsa verde by boiling them. When the chicken is defrosted, he estimates the time and intensity needed for the chicken to be ready and postpones starting the grilling of the chicken until the moment he judges to be reasonable. While preparing the other parts of the salsa verde, he monitors the boiling eggs, the simmering beans and the grilling of the chicken. When they are all as good as ready, he lays the tables for his guests, who are arriving. Needless to say, John spends most of his time watching the clock and the progress of the different parts of the dish and it is the result of his expertise that he managed all of his tasks without problem.*

*Context-aware scenario: John is preparing dinner. He will be serving grilled chicken with*

*salsa verde to his three guests, who, according to his digital agenda, will arrive for dinner at 19.00. John takes frosted chicken from his smart freezer, which notifies the other kitchen appliances about this action. John puts the chicken in his context-aware convection microwave and presses the prepare-button. The microwave has aggregated information on John's plans and takes decisions based on this information. The agenda informed it about when the meal should be ready, the freezer informed it that the cold item that weighs 600 gram is chicken, and its own sensors informed him that the chicken is currently minus 18 degrees Celsius. This enables the microwave to decide on the best defrosting and grilling intensities and timing. John prepares the beans and eggs for the salsa verde by boiling them. While preparing the other parts of the salsa verde, he monitors the boiling eggs and the simmering beans. When they are all as good as ready, he lays the tables for his guests, who are arriving. When they do arrive, the chicken has finished grilling. While John still needed to manage his cooking properly, he did not need to worry about the timing and intensities of the defrosting and grilling, allowing him to focus on other tasks.*

The first issue I would like to direct the attention at is that the actions that are performed by the application (the context-aware convection microwave) require less of the explicit attention of the user (John) than the actions that are performed in the traditional scenario. John is interested in the end-result of the microwave's preparation process, the actual grilled chicken; he is less interested in whether or not the chicken should be defrosted and at what speed, how long it must be grilled, when the grilling should start, at what temperature, et cetera. Likewise, the application does not need these explicit action parameters: all the microwave needs to know is that John at this point wants the microwave to prepare the chicken properly.

This specific microwave is enriched with functionality to follow proper lines of reasoning or perform the proper inferences, namely that the menu and the appointments made require the chicken to be grilled at 19.00 using a specific temperature. This information has been communicated before, although not necessarily to the microwave itself. However, this information is shared by the ICT infrastructure, thus making efficient use of the time and energy of the people involved. This information is gathered by the application in a way that is similar to how it gathers information with traditional sensors such as heath sensors and cameras, and we will consider any information gathered through its sensing and input infrastructure to be sensed information. The microwave uses its sensors to determine what is happening; for example, it is determined that what is placed in the microwave is chicken, the temperature and the weight of the chicken is measured, and there is communication about the dinner appointments with the agenda. By interpreting this information and relating it to the goal of having well prepared

chicken, the microwave determines the proper grilling procedures. As can easily be imagined, this requires a lot from the designer of this application and this is probably too complex for software engineers to handle without help from people with other expertise. To conclude, some steps might benefit from technologies that are not yet readily available or production-ready, such as sensors to determine the type of food more accurately.

There is more that can be said about the functioning of the microwave than that it has to communicate and use its sensors in order to create a meaningful impression of the situation. Part of the process consists of applying rules: recipes can be seen as a collection of rules and the schedules found in a digital agenda can be processed in a rule-based manner. If the microwave has functionality to determine what John placed in it, this functionality is likely to work with pattern recognition. Furthermore, the microwave is supposed to be able to perform basic cooking operations, it is expected to function as well as any chef who has a decent grasp of preparing food in convection microwaves. Finally, whereas the functioning of traditional microwaves is not described in concepts that are typically used by humans, the concepts of the context-aware microwave are part of our common language. This last point might require clarification. When we discuss the architecture of a microwave, we discuss how it works by sending electromagnetic waves to water, sugar, and fat cells for a period of time and at an intensity that is large enough for the cells to absorb enough energy to get “excited”. The concepts that are used to discuss the architecture of a context-aware microwave are more recognizable: the type of food is described in terms that match cooking instructions, dinner times are matched to agenda information, et cetera, resulting in a design where the emphasis is no longer on concepts that are not human centered.

The final interesting aspect that I will mention here is to what prototypical application the microwave will conform and thus what perspective we will have towards it, and what perspective the designers will have to assume it has towards us. The three options are agency (as with artificially intelligent agents), mediation (by a device in a way mobile wearable devices mediate) and overview (as by ambient intelligence systems). We should note that this is a moment where the designer of the application not only can but even needs to make an important decision. It is in theory possible to choose any of the three options, but as soon as he chooses one, he should adhere to this choice throughout his design of that application. Whichever choice the designer makes, there will be pros and cons that depend on the specific application with its specific usage situations. In the current application, the application is intuitively perceived as an agent, which suggests that this is the easiest option when communicating with people who are not (yet) closely involved in the design of the application. However, there are arguments

to consider the other perspectives. It is true that the context-aware convection microwave must try to deduce what the chef is thinking when he places the food in the convection microwave, shortly bringing the both of them closely together. This would indicate that the application could fit to a (stretched) definition of mobile wearable devices. However, this presence in the world of the microwave and the chef as a unity lasts only moments and then they will each go their separate ways again. It is more likely he sees his context-aware convection microwave on the same level as an assistant: “here is the food, now, would you please prepare it?” The God’s eye view option is also very viable. However, in that case the context-aware convection microwave must always be used in combination with other systems, such as equally context-aware refrigerators and even more intelligent overarching ambient intelligence system. This requirement is needlessly complicating the scenario for now. It is more likely that the functionality needed for such a complex system is developed step-by-step. In a later stadium, this agent might be incorporated in an ambient intelligence kitchen system.

### 2.1.3 Summary

The three main categories of context-aware applications can be summarized as follows. There are applications that have agency, applications that mediate for its user, and applications that rely on their overview on the world. The first appears to be an actor on a comparable level as the human user, the second appears to extend the body and mind of the human user of the application, and the third appears to stand above the human user. All three attempt to achieve the main goal of context awareness:

G1. To improve the usability of the applications
G2. To create applications that pursue the goals of its users
G3. To create applications that require less attention of its users

The applications are also all created by using similar methods:

M1. The applications interpret what it reads about its environment
M2. The applications use contextual information to perform an action

The designers of these applications use these methods to adept to the following assumptions:

A1. Computer applications have more and more means to interact in the world
A2. Computer applications are more and more connected to networks
A3. People use more and more computer devices

## *2.2 Three groups of specialists*

The goals, methods and the corresponding assumptions that are mentioned in the previous section are shared by all researchers of context awareness. If we want to know more about the methods and assumptions of context awareness, we can look at the methods and assumptions that are not shared. This section will discuss two different methodologies that are used when creating context aware applications. The first is cooperation between application designers and end-user experts: people who have knowledge of the domains in which the applications are going to be used. The second is cooperation between application designers and context-processing theorists: people who have knowledge of the way context can be processed, particularly by people. The next section will discuss how these two methodologies set apart the classical approach from the phenomenological approach, while the next chapter will discuss the implications that follow from the traditions of these groups.

As was noted in section 2.1.2, creating context-aware applications is a complex process, and often the knowledge of computer application designers does not cover knowledge on the actual situations the context-aware applications will encounter. This issue can be addressed in two ways. The first is to acquire specific knowledge on the situations that the application will encounter from people who have ample experience with these situations because that is part of their job: end-user experts. The second is to acquire generic knowledge on how people process contextual information, because people seem to be able to make decent choices even when they are no experts on the issue at hand: context processing theorists. While the distinction might not be completely black-and-white, the classical approach clearly prefers reliance on cooperation with end-user experts, while the phenomenological approach prefers reliance on using good theories of context processing. This shift has a large impact on the design of context-aware applications and this section will set out to assess this impact by outlining how the two different sources of knowledge (end-user experts and context processing theorists) relate to the group using this knowledge (the computer application designers).

### 2.2.1 Computer application designers

The most obvious group of people that are involved in the creation of context-aware applications are the computer application designers. They are the ones who are in fact creating the applications. Computer science has a relatively short but turbulent history when compared to other fields of science. During this time best practices were developed: norms, written and unwritten rules that state how software applications can be developed best. Context awareness is technology at an experimental stage and as

such, it has not matured enough to be seen in all parts of life. However, notwithstanding its complex and novel nature, it is a very promising concept. Some of the applications are currently moving to usage domains; although they are kept simple and they typically only use a small part of the available sensor data, they are already powerful applications. However, most of the context-aware applications are still under development and can be seen working in laboratory settings: there is strict supervision on their operations and everybody knows that the applications cannot fully be trusted to be without errors because they are still in an experimental phase. Still, the number of people and the number of research groups involved in developing context-aware applications is large. Research is typically being performed at universities and at companies that develop high-tech domestic and user appliances, such as Philips, Nokia and many others. Since they are creating typical domestic appliances applications, these research groups and these companies want to develop robust applications that they can sell for a profit. Considering the wide variety of universities and companies that are involved in these efforts, it can be expected that there is a large variety of best practices. However, the computer scientists are not developing context-aware applications all by themselves; the research field cooperates both with people who have practical knowledge of the usage domains of the applications, and with people who have knowledge of what context itself is and how people use context.

### 2.2.2 End-user experts

A context-aware application is being developed, as all computer applications, for a specific purpose. This is so self-evident that it almost sounds redundant. Its non-triviality in this case stems from the weight that needs to be given to the specifics of this purpose. In context awareness, the designer explicitly does not design an application that performs a function, but an application that is to operate in some contexts. The context is not merely an issue to take into account as it is for other applications: processing it is a crucial mechanism for the application's operation. The domains in which the application is going to operate play a particularly big role in the design process.

When designing an application, or when taking the first steps towards programming any software, a good computer scientist knows that he should make an analysis of the application domain. This means that he should capture the requirements of the future users of the application, create scenarios that display how the application will be used and capture what would happen if the application would not exist. The application that is being developed should then conform to the criteria that follow from this analysis, enabling the future user to operate the application according to his expectations and without seeing any big surprises.

Applications that have no elements of context awareness operate best in an environment that is static compared to the environments in which context-aware applications can operate. In fact, classical, non context-aware applications are designed for operation in one or a few specific contexts. Such an application *can* operate in different environments, but it will not adapt to these environments. Even more, it will function optimally when the environment corresponds optimally to the environment the designers had in mind when creating the application.

Context-aware applications typically are able to adapt to a range of relevant and predictable changes in the environment. This word “predictable” will prove cause for considerable conceptual differences within the design of context-aware applications. It is an assumption of the classical approach that their applications only need to be sensitive to a predictable range of contexts, while the phenomenological approach sometimes assumes that the applications in the end should be sensitive to all possible contexts.

The context-aware applications that are now being developed often use a practical approach with a bottom-up character, with the use of expert knowledge about the application domains to make sure the application is prepared for a sufficiently large variation in contexts. How large this variation should be, is a judgment call that is made by the designer. This expert knowledge is information that on request is provided by persons from companies or institutions who work in cooperation with the designers, and these persons are called end-user experts. Many of the applications that are now being developed are to be used in houses (domestic appliances), offices, medical situations, mobility or everyday situations (such as with applications for mobile phones) and thus people who have practical experience with domotics, who are office workers, medical practitioners, truck drivers, or even people from everyday households.

An example of knowledge from an end-user expert can be seen in the following situation. Consider a context-aware application that assists a user in finding an appropriate restaurant and imagine that a person in Barcelona uses this application at 1 p.m. local time. It is hard for a computer scientist to guess the kind of restaurant his application should use for his response. However, most hospitality managers, for example, might be able to provide the proper social rules for making restaurant reservations in Spain.

### 2.2.3 Context processing theorists

Contrasting with the practical knowledge of the end-user experts, many sciences contribute to knowledge on context processing in the human mind. This is particularly relevant information, because people are considered able to process context almost

effortlessly. Researchers of artificial intelligence for instance already have experience in cooperating with researchers of the mind, brain, and intelligence. These researchers are active in fields such as cognitive science, psychology, neuroscience, anthropology, philosophy, physiology, and biology: all fields that are concerned with how people think and act. Researchers of context awareness, and especially the researchers from the phenomenological approach, have a specific interest in the knowledge that can be provided by the sciences of the mind. They want to develop generic procedures for applications to deal with contextual information, and they expect the sciences of the mind to be able to provide them with a means to do so.

We will see what these generic procedures should be and how this differs from artificial intelligence's quest. The word "context" used to refer to how people were able to interpret words in a text correctly: by interpreting not only a specific word, but also the words that are in front and after that word. The word "context" has a broader meaning now, namely all information, not only textual information, which is used to interpret any other information correctly. This illustrates how people process context almost effortlessly, as opposed to computers, which require large efforts of application designers to reach a decent but often less good result. The brain is seen as a system that is context aware, and if researchers are able to figure out how human interpretation works, they might know more about how computer systems might interpret context. Researchers of context awareness want to know how a system can be designed that can interpret as many contexts as possible with minimal effort. If a researcher knows more about context in general, he might be able to design context-aware applications that can interpret all contexts with less effort than during earlier stages. He might also be able to reuse his application with less effort in another domain than the domain for which he originally designed his application[8].

### 2.2.4 Conclusion

This section's conclusion is that computer application designers, and especially designers of context-aware applications, can cooperate with end-user experts to gain additional insight into the contexts in which the application will operate, and that they can also cooperate with context processing theorists to create generic algorithms for applications that are to function in several contexts. A good computer designer, in any design branch,

[8] This fits perfectly to the development of middleware. Middleware is a generic set of application parts that can be reused and customized for specific applications. This is part of a methodology that is seen often in application engineering, especially when there is convergence in the goals and methods that are applied to achieve these goals. There are currently several attempts to create broadly accepted middleware solutions for context awareness, including approaches that claim to be phenomenology-related. A thorough overview of context-aware middleware approaches, their purposes, and their relation to phenomenological views is given in Johnsson, M. (2007). Sensing and making sense, designing middleware for context aware computing.

not only in context awareness, is supposed to study the domain in which his applications are to be used. In context awareness, however, this activity is more important in context awareness than in other design branches. How the computer designers should capture the knowledge of end-user experts in these domains, and how his applications should process this knowledge, is a difficult question that is part of any computer application engineering effort. However, context awareness can also benefit from the knowledge of researchers on how people process context. Cognitive scientists, for example, have knowledge that can be used to compare how people process context to how applications should process context. Social scientists can help with giving a better definition of "context". As context awareness matured, designers of context-aware applications required more and more input from end-user experts and researchers of human context processing. The next section will show how the classical approach came to rely more on knowledge from end-user experts and the phenomenological approach came to rely more on knowledge on context processing theory.

## *2.3 The classical and the phenomenological approach compared*

This section will argue that the end-user experts enable the classical approach to create applications that perform actions and make choices according to a design plan, while the phenomenological approach created generic procedures that focus on how to present choices and information that other actors can use to steer an application's activity. Up to this point, we have not been able to set the phenomenological approach apart from the classical approach. All that we were able to say is that the phenomenological approach uses phenomenology and the classical approach does not. However, the distinction that was introduced in the previous section between cooperating with end-user experts and relying on context processing theories does give us the proper means to explain the differences between both approaches properly. This section will discuss the impact of the classical approach's cooperation with end-user experts and the phenomenological approach's reliance on context processing theory on the goals and methods of these approaches. It will also discuss what is classical about the classical approach, and what is phenomenological about the phenomenological approach. In parallel, we will also see the assumptions that underlie these methods.

The change that was proposed by the phenomenological approach should be seen in the light of the traditions of computer application designers that are involved in creating context-aware applications. These designers were typically true computer scientists with a logic and mathematics related background. Both Schilit and Dey used to work at Intel, the largest producer of computer chips in the world. While the next chapter

will discuss the differences in the background of computer designers involved in context awareness, this section will suffice with stating that context awareness was popularized by researchers that had ample experience with creating computer applications and verifying that they do what they should do. Furthermore, they believe that applications can be improved by making use of the new possibilities for interacting and making use of the ubiquity of computers in our world. What is typical for the classical approach is that their method to do this is to rely on formal processes such as logic, mathematics, formal verification procedures, and careful planning. This contrasts with the phenomenological approach, which holds that the sensor data should not primarily be related to formal processes, but to the meaning that they reveal when they are used for action or interaction. Dourish, an author who wrote many widely accepted papers on the combination of phenomenology and ICT, is a proponent of this shift in methodology.[9] His statement is that researchers that rely on such classical design techniques often think that their knowledge is absolutely valid and correct. Phenomenology criticizes this assumption, and holds that all knowledge is subjective, is only true until better knowledge has been found, and that designers should be aware of these limitations of knowledge. This awareness is achieved by analysis of one's own experience. The phenomenologist's approach, according to Dourish, is to reflect on what the interaction with a context-aware application will mean to its user and use that reflection to improve the design of the application.

### 2.3.1 The classical approach

According to the views of classical context awareness, contexts can be objectively determined. According to them, it is just a matter of good research to determine what information the application needs to detect. This section will explain how this method works by describing research on context awareness that follows this classical approach. We will see that this research is coupled with information from application domains: the areas where the application is going to perform its functions.

A problem that is typically identified by the classical approach is how to interpret the data applications receives from its sensors. The application has to achieve a goal and it uses this interpretation to find the most suitable way to pursue that goal. There are two solutions that the classical approach typically applies, and both are pragmatic, straightforward, and rely on logical inference or deduction. The first is to start with looking at which sensors are available and what they can measure. Then the designers should try to find creative ways to interpret the sensor data and adjust the application's activities accordingly. The second is to brainstorm about which events might impact on

[9] Dourish, P. (2001). Seeking a foundation for context-aware computing, pp. 235-237

the application's performance, and then try to determine which types of sensors are needed to recognize such events. Both solutions rely on what in software engineering is called abstractions and generalizations, which is part of formal software engineering. An abstraction is a generalization of similar things, where irrelevant details are 'abstracted away' and only the relevant characteristics remain. An abstract occurrence of a phenomenon corresponds to several concrete occurrences, which have a number of elements in common. Formal logic can then be used to reason with these decontextualized abstractions and later they can be re-translated to concrete situations[10].

One of the research groups I am familiar with is the Architectural Services and Networked Applications group (ASNA)[11] at Twente University, a group that created context-aware applications. Dockhorn, one of the researchers of that group, documented the classical approach very clearly, thereby illustrating the practice of designing context-aware applications using the classical approach. She used[12] the Merriam-Webster dictionary for a definition of “context” and defined it as a “collection of interrelated conditions in which something exists or occurs”.[13] In her argument it is said that when an application has information on the user’s context, the application knows more about what it should do for the user, and it knows how it should be done[14]. The challenges she discerned correspond to the following two steps: the application should interpret the context (“what is the current situation?”) and the behavior of the application should be influenced (“what should the application do and how should it do this?”) As an example, the document mentions a telemonitoring application. This specific application was a device that is to be carried by epileptic patients and gathers vital signs in order to predict the likelihood of epileptic seizures (“What is the current situation?”). The vital signs are sent to a central processing facility (“What should it do?”) through the most suitable available networking technology. After it has determined if it can send the data through WiFi, GPRS or SMS (“What is the current situation?”), it can determine how much of the gathered information to send (“How should it do this?”). When necessary, volunteers are automatically contacted (“What should it do?”) to provide assistance.

This example illustrates some key points of the classical approach: the attempt to create an application in a way that makes it easier than before to create good applications, the struggle with the definition of context and context awareness, and

[10] Guha, R. and J. Mccarthy (2003). Varieties of contexts, p. 166-167
[11] After a reorganization, this group ceased to exists and its research was migrated to research groups dispersed over both the computer science department and other departments.
[12] Dockhorn Costa, P., Ferreira Pires, L., and van Sinderen, M.L. (2006). Architectural support for mobile context-aware applications
[13] Idem, p. 24
[14] Idem, p. 2, where is being referred to a document by Dockhorn et al. that is not included in the document's bibliography

reliance on logic in the creation of the application. While it is a goal of the phenomenological approach to create applications that are better than the classical context-aware applications, the classical approach itself is rather modest. They acknowledge that this application could also have been created without principles of context awareness, but this approach allows them to create a better application with less effort. The assumption is that context awareness can simply be "added" to an application[15], not completely unlike how color can be added to a surface. The use of a naive definition from the dictionary illustrates that they are not using the knowledge from context processing theories. In fact, they do not really care what specialists on context think at all, they only care how they can use the information that the device receives from its sensors; a strategy that some classical researchers even mention explicitly[16]. However, the application does rely on the knowledge of the end-user experts, the knowledge of practitioners. Without the practitioners at the Revalidation Centre Roessingh, such as physiotherapists, care workers, and medical practitioners, the application designers would not have known what signs to look for. Not only do the practitioners know, for example, what the signs of an epileptic seizure are, they are also intimately familiar with the daily habits of their patients: they have the knowledge that computer application designers need to have in order to make reasonable assumptions, predictions, and judgment calls. The methods of the classical approach furthermore make use of tried and tested methods of logics and reasoning to make sure that this knowledge of the domain specialists is properly processed by the context-aware application[17].

The classical context awareness has methods, and assumptions that are not shared by the phenomenological approach. Its goals are no more specific than then goals outlined for all of context awareness:

CG1. To improve the usability of the applications

CG2. To create applications that pursue the goals of its users by initiating activity and adapting activity

CG3. To create applications that require less attention of its users

Their methods do not include the study on context itself, but the study on the domains in which the applications will function. The cooperation between computer application designers and end-user experts results in a set of contexts that the application should

[15] Haseloff, S. (2005). *Context Awareness in Information Logistics.*
[16] Korpipää, P. and J. Mäntyjärvi (2003). An ontology for mobile device sensor-based context awareness.
[17] Mccarthy, J. (1986). Notes on formalizing contexts.

recognize to improve the usefulness of the application. They do this by building a dataset with potentially relevant information.

CM1. The applications abstracts from what it reads about its environment

CM2. The applications processes the abstractions with formal tools in order to decide what actions to perform and how to adapt its activity

CM3. The application designers cooperate with end-user experts to determine the proper abstractions and logical procedures to connect to these actions

The underlying assumption is that these end-user experts will be able to specify all the relevant knowledge: the knowledge on the specific contexts that should be processed by the application. Another assumption is that logic will bring the necessary tools to translate this knowledge to the applications. So, the generic assumptions of context awareness A1, A2 and A3, the classical approach adds the operationalized assumptions CA1, CA2 and CA3:

CA1. It is possible to make abstractions that retain enough relevant information

CA2. It is possible to predict enough situations to actually improve the usability of applications

CA3. There are people who can predict what will help future users of applications (end-user experts)

### 2.3.2 The phenomenological approach

The phenomenological approach holds that the classical approach is wrong in thinking that it can use rules and observations to create good context-aware applications. Phenomenologists state that scientists are likely to miss some crucial information. Science has as its goal to formalize as much as possible about the world, but according to phenomenologists, it offers only one of the many possible views. An assumption of phenomenology is that any particular worldview might result in its own particular scientific conclusions. For example, in the Aristotelian worldview it was common to state that an apple falls down because it “wants” to pursue its teleological goal. This view had its own explanatory power, and only after a cultural change people discovered that apples follow different rules when falling down. This is a first reason not to rely solely on scientific facts: we might think that they are correct, but history might prove us wrong. The second reason is that many phenomena have no satisfactory scientific explanation. Intelligence is a good example; although we know about many elementary mechanisms that underlie intelligence, we are still not able to reproduce the human brain. Phenomenology set out to create an alternative methodology for discovering truths,

particularly about human consciousness. The method is based on thoroughly examining our experiences and drawing conclusions on them.

Great philosophers who used such methods to increase our understanding of the world include Husserl, Heidegger, and Merleau-Ponty. Because of their extremely thorough descriptions and analysis of human experience, they are popular sources of inspiration for some application designers. Given their focus on experiencing and on human consciousness, it is not surprising that they have a long-standing history of being applied in the computer science areas of artificial intelligence and in human-computer interaction. They are also referred to by the researchers on phenomenological approaches to context awareness. In all three of these research fields, it is argued that there is a promising future for so-called embodied applications: systems that are tangible the same way that people, animals, and objects are tangible. Phenomenologists have two reasons to involve themselves in context awareness: 1) the classical approach falsely assumes that designers can use simple rules and facts to accommodate people in their activities and 2) context awareness is a suitable research field to serve as a testing ground for creating tangible and embodied applications according to phenomenology based principles. This section will demonstrate the implications for these assumptions for the methods of the phenomenological approach by analyzing statements of three proponents of this approach: Dourish, Svanæs and Müller. The emphasis of this section will be on the first reason, while the next chapter will discuss to what extent the second reason is justified.

Dourish, one of the most influential authors on phenomenological context awareness, argues[18] that context awareness has become a popular discipline and that there are good reasons for this development. He states that the abundance of cheap computing and networking facilities has stimulated the use of computing devices everywhere in our everyday lives, as is illustrated by the trends of ubiquitous computing and ubiquitous networking. This development enabled people to use applications that take contextual factors into consideration. He notes that there is little agreement on definitions of context awareness and that the result is that research takes place in an ad-hoc manner. It is ad-hoc because application designers consult with end-user experts every time they need information about the contexts to which they want their application to respond. Context awareness would be improved by structuring the way an application processes context differently, which is an analysis that is shared by the human-computer interaction community. This opinion of Dourish is complemented by Svanæs[19], who states that context awareness needs to look outside their own research fields and use

[18] Dourish, P. (2001). Seeking a foundation for context-aware computing.
[19] Prof. Dag Svanæs has been active in human-computer interaction since 1980 and is mainly involved in ubiquitous computing

these fields to rethink their methodology. Both Dourish and Svanæs state that phenomenology is particularly well suited to help researchers of context awareness with this task. And there are researchers from other domains than human-computer interaction who share this particular opinion. Müller, for example, is a theoretical computer scientist with an affinity with artificial intelligence. Müller[20] argues that people take contextual factors into consideration almost effortlessly and that context awareness can benefit from insight into how people do this (and vice versa). This is another additional assumption of the phenomenological approach: if context-aware applications process context similar to how humans process context, context awareness loses its ad-hoc character and applications will be better. This assumption supports the methods of the phenomenological approach: to consult theory on (human) context processing to improve design of context-aware applications.

A possible result of this method is that context-aware applications are no longer allowed to make choices for its user. This is a direct result of the analysis of how people behave: people do not always follow predictable rules[21] and if an application cannot predict the behavior of its user then it cannot know what it wants the application to do. The phenomenological approach also offers an alternative: the application should present the user the right options at the right moment, and it should provide the user with the right information at the right moment. Svanæs gives the following example of a simple context-aware application that does this. He describes a “magic light”, which switches on when it reaches a specific location. The magic light does not tell the user what he should do; it merely gives him an indication that he has reached a specific area. A classical context-aware application would probably try to give more information about this location, such as whether or not it is near to the final destination of the user. The phenomenological approach would disapprove of this since it denies that the application can really know what the final destination of the user is. This approach states that it is enough to give an indication of these location aspects so the user can decide how to interpret it, and how to act upon this information. It is in line with the phenomenological argument that interpretation is impossible because meaning is only revealed in the action that is performed by an actor.

Svanæs takes this argument further than Müller. Svanæs states that from the human perspective there is no difference between context and direct interaction: all input to an application is interaction, and it is all equally important. His conclusion is that context awareness rests on a too narrow view of intelligence and that the term itself is “absurd”. This conclusion is an indication that we should be careful with blindly adopting

[20] Müller, M. E. (2007, June). Being aware: where we think the action is.
[21] This hypothesis will be explored and explained in chapter 4

the phenomenological view. Here Svanæs not only broke with the classical approach to context awareness, but he is also in conflict with the general goals of context awareness. The breach with the classical approach consists of the full elimination of involvement from the end-user experts: only the user of the application and the application itself get to see anything from the usage domain. The breach with context awareness in general consists of denying that an application actually can use contextual information to pursue the goals of its users; instead, Svanæs states that useful applications should process all information equally, without differentiating between context information and direct interaction with users. The next chapter will discuss why we should not follow all of Svanæs' arguments. I will not consider Svanæs' conclusion that end-user experts have no relevant knowledge and that a distinction between contextual information and interaction is absurd to be typical for the phenomenological approach. Instead, I will consider his first conclusion to be an extreme version of the phenomenological approach' assumption that context processing theory is more relevant, and I will assume that other phenomenologists see the difference between contextual information and interaction as an axiom within context awareness, similarly to how other context awareness researchers take this distinction for granted.

The phenomenological approach also presents a viable alternative: context-aware applications should "reveal" the world. The underlying assumption is that "meaning" cannot be created by interpreting sensor data, but that all interaction between the user, the world, and the application already is meaningful. The idea that an application must "decode" its input to create something meaningful is, according to the phenomenological approach, wrong. Even more, it is impossible to program a computer to understand all situations, because every situation is unique. The conclusion of the phenomenological approach is that context-aware application must presuppose meaning instead of create meaning. Their new, additional goal is to create applications that show the relevant information to the user and present him with a choice on how to proceed. However, the user should himself decide upon what information should be acted and how.

The example of the projector, which we have seen earlier, can explain how this is different from the classical approach. In the classical approach, a projector would adjust its brightness settings according to the illumination level in the room. A really smart projector might also incorporate other information, such as whether or not the presentation has started. The phenomenological approach would design the projector differently. Ideally, the projector would not adjust its brightness automatically, but it would somehow inform the presenter that the presentation will not show properly. The presenter is then able to choose to increase the brightness of the projector or to make the room darker. The question that separates the classical and the phenomenological

approach is if this really makes an application that much more useful and if it would not be better if the application would just act without the interference of the user.

The phenomenological approach shares aim A1 and A3 with the classical approach, but they do not want their applications to pursue the goals of its users directly: they want their applications to provide their users with information and choices.

PG1. To improve the usability of the applications

PG2. To create applications that present information and choices to its users that helps them achieve their goals.

PG3. To create applications that require less attention of its users

The phenomenological approach has its own interpretation of how to make M1 and M2 more concrete. Not only has one of the goals changed from taking actions towards presenting options, at the same time the emphasis has also shifted from action design to interaction design.

PM1. The applications select features of the situations to present to its users

PM2. The application designers design interfaces that take into account how people process contexts

PM3. The application designers cooperate with context processing theorists to learn about how people process contexts

The need for creating a design that can be reused for different applications has, in the case of the phenomenological approach, resulted in an effort to design applications that can process all contexts instead of only a set of contexts that is determined in cooperation with end-user experts. The following assumptions underlie the methods that are used to create such applications:

PA1. All situations are unique and only in these situations can a person decide what is relevant

PA2. Ad-hoc design is bad and it is the result of not finding generic procedures that the applications can use

PA3. Context processing theorists can help designers with finding such generic procedures

When comparing the goals, methods and assumptions of both approaches, we can note the following tensions. The classical approach wants to pursue the goals of its users by actively performing actions (CG2), while the phenomenological approach has a more modest goal, namely to perform interaction instead of actions (PG2). The methods of

both approaches seem to be able to co-exist, but the assumptions that underlie them again present a tension. The phenomenological approach rejects that it is possible to make good enough abstractions and assumes it is better to find good enough procedures, based on theories on human cognition. Chapter 3 will test these assumptions.

## *2.4 Conclusions*

Context-aware applications are made to improve the usability of computer applications and they do this by assisting its users to achieve their goals. While classical context-aware applications can do this by making choices for its users, their phenomenological counterparts will only provide information and present choices to its users. This is a result of the shift from actively involving end-user experts in the design to relying on generic knowledge about human context processing. End-user experts typically have experience with the questions that are presented to them by the computer application designers who are creating the context-aware applications. Together, they feel confident that they can gather enough information to create applications that make good choices. Theory about human context processing state that processing context is so complicated that the ad-hoc approach of the classical approach is insufficient. When computer application designers started to rely on context processing theory, they started to develop generic procedures to process context. In the phenomenological approach, the choices are no longer predetermined by the computer application designers and the end-user experts; the computer application designers determine, based on context processing theories, which choices the application user can make and when he can make them. This approach can lead to the conclusion that context awareness does not need the end-user experts at all. The next chapter will explore the down sides of that conclusion.

# 3 The phenomenological approach: an evaluation

Context awareness is, as we have seen, a complex research field. The design of computer applications is always a complex task, and this specific subfield of computer application design has brought it upon itself to make the task even more challenging, by combining traditional computer technologies with sensor and actuator technologies, and, now, by incorporating theories from human sciences. The phenomenological approach is based on the assumption that it is not effective to perform the abstractions that are foundational to the classical approach. Instead, it proposes a focus on theories from human sciences, such as cognitive science and psychology. This chapter will argue that this would result in an approach that has a lot in common with artificial intelligence, a branch of computer science that is also interested in the link between computer applications and human cognition. The current chapter will analyze the implications of a phenomenological approach in artificial intelligence, and I will use this analysis for a recommendation for context awareness. I will also argue that, although phenomenological reflection has its merits for artificial intelligence and context awareness, phenomenology does not necessarily have to be the primary means of philosophical reflection on context awareness.

The classical approach is similar to ICT's subfield of software engineering. This is illustrated by the medical application that was discussed in section 2.3.1. That application does not only interact with its users (the patients that might have epileptic seizures), but also with other computers and it uses contextual information during both activities. However, more and more context-aware applications started to change in intelligent applications that focus on the interaction with its users, which is the result of the involvement of researchers from human-computer interaction, and reliance on human context processing theories instead of on cooperation with end-user experts. The phenomenological approach asks the new question "how can we design context-aware applications without the ad-hoc approach that involves end-user experts?" and they give two answers to this question. The first answer is that applications should not perform actions, but they should only ask for additional input in a suitable, action-related manner. The second answer is that contextual information is exactly the same as any other information, and context-aware applications should have "intelligence" in order to deal with this information. This leads to the introduction of theories that stem from the AI tradition. The next chapter will present an approach to context awareness that explores the first answer. This chapter will discuss the second answer by explaining a phenomenological critique on artificial intelligence and what this means for the role of intelligence in context awareness. This discussion will lead to the conditions under which the first answer, the answer promoting the presenting of choices, will be embraced. The

first step in this discussion is to show the relevance of the AI tradition by giving an overview of the histories of computer application engineering and AI. We will see that AI has a more turbulent history than application engineering and that its goals and methods differ a lot. An especially harsh critique on AI came from Hubert Dreyfus, who himself is a phenomenologist. Dreyfus argued that computer cannot be intelligent. A discussion of this argument will display to what extend a phenomenological approach to context awareness will be possible.

## *3.1 Traditions and their results*

Each research field has its own unwritten rules about how to perform research. These rules enable a researcher to use all the knowledge that is available on his particular subject. Other researchers can give assistance, they can criticize his work, and they can evaluate it. And the rules function as guidelines that help him not to make mistakes that others have made before him. This is particularly true in ICT, the research field that started research on context awareness. From the early days of ICT until today, research on ICT progressed steadily. It has developed many best practices to put new technical solutions from, for example, physics and electrical engineering to practical use. The handbook *Philosophy of Technology and Engineering Sciences* has a chapter on the philosophy of ICT[22] that has as one of its topics the different fields of ICT as they are discussed by philosophers. Here, Brey and Søraker analyze six subfields of ICT in terms of their goals, methods, and assumptions. I will suggest a seventh subfield, networked application engineering, as a variation to their subfield “software engineering” and I will offset this to artificial intelligence. A subfield is similar to what Van de Poel noted as the key point of a technical regime: “(interaction) rules which are actively shared by the actors, enable coordination and so result in regular patterns of technological development”[23]. We will see that context awareness originally fit to the tradition of engineering, holding dear the concepts of usability, verifiability, and reliability, and that conformance to this tradition is at stake because of the introduction of the AI tradition to context awareness, where it is equally or more important to create new scientific facts. The remainder of this chapter will present an analysis that results in the conclusion that context awareness is best served by a phenomenological approach that makes the involvement of context processing theories subordinate to the involvement of the end-user experts, which means favoring the goals of an engineering paradigm over those of the AI paradigm.

[22] Brey, P. and J. H. Søraker (2009). Philosophy of computing and information technology
[23] Van de Poel, I. (1998). Changing Technologies. A Comparative Study of Eight Processes of Transformation of Technological Regimes. p. 17

### 3.1.1 Application engineering

Information and communication technology as we see it in our society today is a combination of computer science and communication technologies. Societal forces that operated upon science that stimulated the development of our computers, were the belief in logic and the need for automation. These two principles are also reflected in the computer's central operating principles and the first use of computers. The central principle by which a computer works is the manipulation of zeros and ones. Boolean algebra, named after its inventor, Georges Boole, is a type of "formal logic" that uses two values (true and false, or one and zero) as the basis for a complete mathematical system. Our current computers function according these principles: binary logic[24]. Another central principle is the formalization of the actions that have to be performed. This corresponds to the initial use of computers[25]: to automate the calculations that previously were performed by human clerks, initially to benefit cartographers and astronomers. Even then, the people creating the computing applications needed to possess domain-specific knowledge. After that, step-by-step, computers became machines that were intended to take over more and more complex tasks and more of this domain-specific knowledge was required. Context-aware applications can be seen as an exponent of this tradition.

This is context awareness' historical background. Brey and Søraker zoom in on ICT by dividing ICT in the following subfields that are subject of philosophical reflection: computer programming and software engineering, data modeling and ontology, information systems, computer simulation, human-computer interaction, and artificial intelligence. This division fits the philosophical literature on computing science; however, context awareness originates in a subfield of computing science that has not received a lot of philosophical coverage, and consequently, this subfield does not fit to any one of the categories. Context awareness, and especially the classical form of context awareness, has its roots in the telecommunications field. This field is closely related to software engineering as described by Brey and Søraker, with the difference that it also involves hardware engineering. A telecommunications device is hardware and software that uses sensor and actuating technologies, including networking technologies. Telecommunication is often said to enable the transfer of information between a person and a system or between two systems. When we refer to it, we mean any electronic communication. Again, a historical view teaches us a valuable lesson. The first electronic signal was sent in 1830, the first wireless signal was transmitted in 1894; and almost

[24] Morris Mano, M and C. R. Kime (2001). *Logic and Computer Design Fundamentals*, pp. 27-39
[25] Aspray, W. and M. Campbell-Kelly (1996). *Computer: A History of the Information Machine*, pp. 9-15

another half a century later the regulation of telephone, radio and television broadcasting started to take place[26].The growth of computing technology is linked to its introduction to communication technology, creating a trend in which context awareness fits. At first, computers were large devices with a single user, but soon these centralized machines were followed by machines that were shared by several users. When less expensive computers became available, computing became more decentralized, and this was followed by the networking of these decentralized computers[27]. Computing devices became smaller, and communication technologies, which now include technologies for data transfers, became more widely available. Nowadays, computing devices and communication technologies can no longer be seen as separate. Even more: they are everywhere and they are quickly becoming the basis for our social and professional lives. In addition, its history still rings through in the way networked applications are being developed. There is a strong focus on making sure that all applications can work together. There is also a strong bias towards formalization, standardization, and regulation: the way applications work together must be well documented. The importance of formalization of all parts of a design is hard to overestimate, and is widely recognized by the telecommunications community[28].

Traditional communication technologies depended on carefully planning all parts of its design. Computing technologies traditionally were created more creatively, as if researchers were improvising: researchers looked at what was available and used that to create something else. Either inventions came from other fields of science or they were the result of constructing new technology that filled an immediate need. Both the planning and the improvisation approaches were of a strongly practical nature: the most important thing was that the technology performed its function, and people should learn to use it. Only recently, it has become common to design applications around what users need instead of around what is technically possible; a strategy that is also deployed successfully in other engineering fields. Instead of applying a technology-centered approach, the applications are designed with a user-centered approach, which is an inclusion of practical knowledge from application domains.

Our current mobile phones are a good example of the combination of planning, improvising, and formalizing. Phones from all manufacturers must be able to work with all phones of other manufacturers. This is made possible by strict formalization of how the phones should communicate with the phone network. However, there are multiple phone networks and, for example, not all phones that are created for the European market can be used in the United States. Manufacturers must plan their design efforts in

[26] Tomasi, W. (2004). *Electronic Communications Systems: Fundamentals Through Advanced*, p. 3
[27] Messerschmitt, D. G. (2000). *Understanding Networked Applications: a First Course*, pp. 2-5
[28] Idem, pp. 4-5

such a way that they can market their product as quickly and economically as possible in all regions of the world. On the other hand, the manufacturers can build their phones any way they want, and a Nokia phone initially was constructed completely different than, for example, a Siemens phone was constructed. However, a phone is build from many small components, and a Nokia phone might have some of the same components as a Siemens phone; this is yet another example of improvisation of the manufacturers. What all phones still share, is that they use the same logic system and they have the same core functionality: to enable people to exchange messages with each, replacing human messengers.

Nowadays, almost all computer applications are networked and there is a convergence towards integrating sensor and actuator technologies in generic devices such as ultra-portables (very small and lightweight laptops) and smartphones (mobile phones that act like computers), and to create dedicated sensor based devices based on generic components. As a result, most retail applications are at least able to run on sensor and actuator equipped devices, and many successful applications are the result of aimed efforts to optimize the application for such devices. Not only designers of networked applications need to make use of the hardware's interaction capabilities as good as possible: all computer application designers get to design their software to benefit from the hardware's interaction capabilities. Consequently, there is a convergence between software engineering and networked application engineering. For the remainder of this thesis, I will refer to design of networked applications that have sensors and actuators by just writing application design. This implies that the description of Brey and Søraker of software engineering is also valid for context-aware applications. What they state about software engineering is that is valued to create a clear specification of what its applications should do, to create applications that conform to these specifications, and to verify, according to well-thought-through procedures, that the application conforms to their specifications. [29]

To summarize, the design of applications has a foundation in using logic and formal specification and verification procedures, and it is rooted in a tradition of automation of human tasks, decentralized communication, standardization, and regulation. There is a strong connection to the application domain and both improvisation and careful planning is important. But, as indicated before, context awareness can no longer be satisfied with only looking at the traditions of application engineering. Many of its current researchers are also connected to AI, and in the next section we will see that

[29] Brey, P. and J. H. Søraker (2009). Philosophy of computing and information technology, pp. 1364-1365

AI is the result of a turbulent history. AI has matured and it has learned many lessons, which should be incorporated in the traditions of context awareness.

### 3.1.2 Artificial intelligence

As Brey and Søraker argue, artificial intelligence is a science that has as its aim to study intelligence in a structural manner, which includes studying reasoning and learning. [30] It is obvious that reasoning is relevant to context-aware applications, but we will see that it is complicated to combine this aim with the aim of engineering reliable applications. Artificial intelligence is a research field that quickly developed a profile that is quite distinct from application engineering. One of the driving forces of most emerging industries and scientific fields is the promise of early wins [31]. It is a self-reinforcing process: the successes foster the thought that the development will be fruitful, what attracts people who want to invest time, effort, and money in it, what might result in more successes. This, according to Dreyfus, appears to be also what stimulated research on artificial intelligence. The technology claimed to be able to unravel the mysteries of the mind by producing computer programs that are as smart as humans are. This would give us insight in the structure of thought processes and in the functioning of the human brain, which was seen as so promising and important that huge budgets were allocated for this kind of research. Applications that were successful in the early stages of AI are, for example, chess playing programs and conversational agents. They are examples of applications that use logic to create intelligent applications. While computer application engineers cooperate with physicists and electrical engineers, the AI researchers cooperate with cognitive scientists. One branch of AI wants to understand human cognition by recreating similar intelligence in computers (strong AI), another branch wants to know how the human brain works so they can create smarter computer applications[32]. But up to today there are neither fully intelligent computer applications nor do we know exactly how the brain works.

As Dreyfus argues, strong AI wants to discover the rules by which humans process information and that thus they see humans as information-processing systems. While doing so, they falsely assume that human intelligence is the result of manipulation of symbols in the brain using formal rules. Convincing theoretical counterarguments however, such as Searle's Chinese Room argument, make success of that approach highly unlikely [33]. Alternate theories suggest that there are at least also different processes at work, such as those based on what are called "connectionist theories".

[30] Brey, P. and J. H. Søraker (2009). Philosophy of computing and information technology, pp. 1355-1357
[31] Brey, P. (2001). Hubert dreyfus: Humans versus computers.
[32] Idem.
[33] Cole, D. (2004). The chinese room argument. In Stanford Encyclopedia of Philosophy.

These theories state that human behavior is the result of connections being made and broken in the human brain, but that these connections do not necessarily represent actual information. This marks a point where formal logic is no longer seen as the only possible means for a system to reason. Apparently, while humans do use logic, ultimately their reasoning is not based on logic but on something else. AI researchers should now inter-operate more with scientists who perform other kinds of research towards the brain in order to benefit from these new insights, and cooperation with logicians and mathematicians becomes less relevant.

Therefore, there are at least three fundamental differences between engineering and AI. First, AI let go of the idea that formal logic is the best or only way to achieve results when designing systems. Their mission includes finding a different way to decide what actions an application should perform. Second, AI does not want to create applications that take over human tasks, but they want to create applications that can reason similar to how humans reason. This does not necessarily mean that these applications should also actually take over any human tasks. Third, AI cooperates more with cognitive scientists than with end-user experts.

Context awareness is in a position that it can benefit from the spectacular and impressive contributions to computer science that AI has achieved. However, it also has an obligation to its computer science roots. Furthermore, one of its design goals is to create reliable applications and a research field that receives so much critique might not be the ideal candidate to contribute to the creation of reliable applications. Now we know more about the tradition of application engineering and AI, we can see how the classical approach to context awareness fits best to the engineering tradition. We can also see that it is tempting for the phenomenological approach to context awareness to feel connected to the AI tradition. The next section will discuss the critiques of Dreyfus, a famous phenomenologist, to the AI tradition. He will not only give an argument that supports the view that computers cannot be intelligent, he will also outline a framework that I will argue to offer an opportunity to unlock human intelligence within context awareness.

## *3.2 Hubert Dreyfus and expertise*

Dreyfus' famous work on the limits of artificial intelligence was prompted by claims from the computer science community that humans were no longer the only ones who could process information intelligently[34]. The claim was that computers could compete with humans on this point. Dreyfus used phenomenology to argue that this claim was false

[34] Dreyfus, H. L. and S. E. Dreyfus (1986). *Mind over Machine: the Power of Human Intuition and Expertise in the Era of the Computer*, p. 1-11

and that there were severe limitations to the capabilities of computers. If context awareness would want to build on the claim of the phenomenological approach that their applications should be intelligent, there should be a way to address Dreyfus' critiques to the limitations of computers.

Dreyfus argues that intelligent behavior is the result of the physical learning process a person goes through. He analyzes our experiences of our behavior and he combines this with the basics of physiology: the study of the functions of our body. More specifically, he analyzes the human body as a complex feedback system that extends our nervous system. He states that this combination is a stimuli-response system that is required for intelligence. The brain constantly receives stimuli from its entire body, such as by sweating, having muscle tensions, soreness and many other sensations. The body itself is influenced by the brain too. The complexity is extended by motor functions that work at the level of the spine, sometimes even bypassing the brain as a central regulator: the autonomic nervous system handles bodily reactions to stimuli that are triggered without "orders" from the brain. And some of the motor functions result in externally observable behavior, which might eventually result in new stimuli from the environment.

Dreyfus' phenomenological account of how people acquire expertise in what they are doing in their daily lives will explain why computers cannot achieve a same level of expertise as people can. Dreyfus started voicing his suggestions to designers of intelligent applications as early as in 1964. These suggestions are primarily about people and about what we should know about them, and not about the quality of artificial intelligence applications of these days. This thesis is not so much concerned with intelligent applications as with applications that take away some of the burden of reasoning from the user. That is why this section will discuss two of Dreyfus points that can be constructively incorporated in context awareness. The first point is that there are five stages in becoming an expert, all of which depend on human physiology. The second is that digital systems can only progress to an equivalence of the third level, competence, due to their lack of a human body. If Dreyfus' argument is sound, it will disqualify the second answer of phenomenological context awareness, which stated that context context-awarene applications should and can be intelligent.

### 3.2.1 Acquiring expertise by going from novice to expert

Dreyfus discerns the following five stages in acquiring a new skill: novice, advanced beginner, competent, proficient, and expert. During these five stages, a person

progresses from responding to easily recognizable objective features according to simple rules, towards automatically and without intellectual effort doing the right thing.

First, a novice learns simple rules and he learns to apply the rules as they are explained to him. He is given a set of objective features of the world to pay attention to, and he is to act on them. Dreyfus illustrates this by giving an example about driving a car: one of the first you learn is to shift up when you have reached a specific speed, where the speed is the objective feature.

In the second stage, that of the advanced beginner, you learn, for example, that when you are driving up a hill, you should shift when your car is at a higher speed; the speed is contextualized, and you could say that a speed that is normal on a flat road is too low for shifting up when you are on a steep road. Normal rules for performing actions become guidelines that are adjusted to the circumstances of that moment. Dreyfus states that the features here are fundamentally different from the features in the first stage. While the speed is an objective feature, the steepness of the road is relatively subjective and so is, for example, the engine sound that is used to decide if you should shift up or not: both require the driver to make a judgment call. Initially, the advanced beginner misses the amount of experience needed to make such judgment calls and he needs to learn when and how it is appropriate to deviate from the rules. Step-by-step, he will learn to recognize more subjective features of the world and how they relate to the rules he has learned.

The third stage, “competent performer”, begins when the number of relevant features has reached a specific threshold, namely when the person starts to get overwhelmed by the number of features he might consider relevant. As he encounters more and more unique situations, the number of relevant and discernible features increases and after a while he is going to have to make a choice in which features to take into account and which to ignore. The learner creates a plan, which includes a purpose of the action and a choice of features to consider. It is *his own* plan, which means that he is fully responsible for the plan and the actions he bases on this plan[35]. As Dreyfus argues, it is possible and even likely that when the learner makes these choices he also makes mistakes. The first two stages were analytical and straightforward: every aspect that is encountered is assessed and weighs in the conclusions. But as soon as he decides that some of the encountered aspects are not relevant, he makes a decision, for which he is responsible. It is this responsibility that for the human actor appears to be crucial in progressing to being a better performer.

[35] This is relevant for ethical discussions of intelligent applications.

According to Dreyfus, something special happens to a learner when he starts taking responsibility for his action. Linking his conclusions to neurological research[36], he states that this strong involvement starts a stage where the analytical steps taken are reduced and the mind starts responding more and more using holistic and emotional processes. In other words, the reasons for choosing a specific action are determined less by which features of the environment the learner can identify and more by which feeling he has while he experiences the environment.

Now we arrive at stage four, proficiency, where the learner's situation is characterized in a way that has nothing to do with the theory he started out with. When the person reaches the stage of being proficient, he intuitively knows what he wants to achieve and what the important aspects of the environment are. However, he still needs reason and rules to determine how to get from the current situation to the situation where his goals are achieved.

The final goal is stage five, expertise, where the person not only immediately knows his situation, but also the proper response. There is no reasoning, no rule following or induction, and no mental representation of the world or of the goals; the learner is by now neurologically wired to respond in a specific way. A car driver is constantly aware of his speed and almost automatically shifts gears depending on factors such as speed, slope, weather conditions, et cetera, without consciously considering them, performing calculations, or applying rules. The difference in results with stages two, three and four is that the expert also responds appropriately to stimuli that are new in combination with the current activity, since these stimuli are nothing more than part of a pattern that influences the process.

To summarize, Dreyfus discerns the stages of novice, advanced beginner, competent performer, proficient performer, and expert. With the competent performer, emphasis shifts from rule-based acting to holistic coping. Feedback plays a role in all stages, and in this stage, the performer is for the first time both able and forced to take responsibility for his actions. As a person starts to make choices in what he should or should not ignore, he is not only responsible for his actions but his actions are now subjective instead of objective action. This subjectivity is a form of unpredictability: the person who gave the initial rules that should be followed is no longer in full control of the action of this learner. This would be a threat to the reliability of context-aware application, since the original application specifications no longer apply, and thus it would be in conflict with the goals of context awareness. It can be concluded that context

[36] Dreyfus cites the following publication: Amidzic, O., Riehle, H. J., Fehr, T., Weinbruch, C., Elbert, T. (2001). Patterns of focal y-bursts in chess players: Grandmasters call on regions of the brain not used so much by less skilled amateurs.

awareness should not want its applications to be anything past the level of a competent performer.

### 3.2.2 Everything is connected: three criteria of embodiment

Not only is Dreyfus' theory useful for determining if context awareness wants its applications to be intelligent (which it should not), it can also be used to determine if they can be intelligent at all. The claim of the phenomenological approach is that context-aware applications are good candidates for intelligent applications because they are "embodied": they have means to sense the world and to act in the world. Dreyfus' phenomenological description of human embodiment as compared to computer embodiment will help us assess this claim.

According to Dreyfus, it is a lot simpler to create a computer application that behaves like a novice than on that behaves as an expert. He states that it depends on the task if a computer even can reach the level of an expert. In addition, he claims that our computers can never reach the level of expert in all areas of human life. The reason is that computers do not have the physical body that humans have. All the behavior that we have learned depends on behavior that we have learned earlier. As a result, a person growing from a baby to an adult step-by-step reaches a level of complexity that cannot be matched by a computer even when comparing them on performing basic and relatively simple human activities such as having a conversation about the weather. Dreyfus argues that there is more than a "simple" complexity problem in the rules that we should program into this computer: a person's growing process plays a necessarily constituent role in him developing human intelligence and an artificial system would need to go through exactly the same process and have the same bodily feedback system as a human person does. This is, which he claims is in line with what Merleau-Ponty would have argued, due to how the human body "constrains the space of possible generalizations"[37]; the types of similarities we can see result from the characteristics of our bodies. Dreyfus discerns a) the specific functions of specific brain regions, b) ordering of new stimuli because of limitations of bodies and c) success criteria for modifying neural pathways.

Because of the way the brain is organized, several limitations are already imposed on what we can perceive. Stimuli received are grouped in a specific way. Dreyfus argues that interpreting visual stimuli, for example, allows for discrimination in size and

[37] Dreyfus, H. L. (2002). Intelligence without representation – merleau-ponty's critique of mental representation the relevance of phenomenology to scientific explanation, p. 375

distance, which are perceptual constants. He states that these discriminations are made possible by the innate structure of the brain: our brain processes sizes and distances based on light waves as opposed to, for example, on sound waves, as bats do.

He further notes that the order and the way in which we experience things and get to know the world influences the similarities the brain will learn to discern. This sounds reasonable: new stimuli affect neural paths that have been created by old stimuli and it is possible to imagine how accidental or recurring “choices” can result in habits. The body limits what new stimuli a person can receive; a baby receives different stimuli than a child or an adult. Besides, factors such as “what can I reach” and “what is close enough to see” affect the phenomena experienced. The role of ordering and stacking of what we learned is important.

As a final influence, Dreyfus states we need criteria to define what counts as a successful event. Based on if something is a success or a failure, our neural connections will be changed so that we will act similar or differently in future situations. The first and the most intuitive mechanism for this is one that evaluates if a goal has been achieved. But Dreyfus notes a similar mechanism, one that constantly feeds information to the brain about how it is doing. The first mechanism can be compared to how a novice learns: there is a goal, there is an action, and the novice either succeeds or fails. The second mechanism compares to how the expert functions: stimuli and actions are always being evaluated, where the performer tries to achieve what Dreyfus describes using Merleau-Ponty’s concept of “maximum grip”.

This has the following implications for this thesis. Not only does a person learn to embody an activity by going through five stages, during his lifetime he also creates the neural pathways that are part of the system that he uses to go through these stages. The three essential elements of learning how to behave intelligently are having a human-like brain architecture, human life cycle ordering of learning with new inputs, and constantly adapting based on how well you are doing. A context-aware application has some similarity to the human body; it can sense light, sound, heath, and touch, and it can grab things, move around, et cetera. But this is completely different from the embodiment that is described above, and by far not similar enough to form a good basis for human intelligence. However, it does fit to the AI tradition and for AI researchers there are clearly opportunities here. Instead of creating applications that simulate the human brain, they can create applications that simulate the human brain and body (in a simplified form). AI wants to learn about how human intelligence works, and by experimenting with these applications they might find out more. Interesting opportunities

can be found in researching what Collins calls “interactional expertise”[38]. The idea is that a simplified body might be enough to develop part of human expertise, such as, for example, his ability to have a conversation, to play chess, or to determine the right level of illumination of a room.

### 3.2.3 Conclusions

One of the goals of context awareness is to create robust applications that do what we want them to do. According to Dreyfus, being unpredictable is part of being intelligent. While this would not be a problem if context awareness were part of the AI tradition, it is a problem since context awareness is part of the application engineering tradition, which values the creation of reliable applications that they can validate for correctness. But even if context awareness would want intelligent applications, these applications would still not be universally intelligent: they would never reach the same level of expertise on all subjects as humans. This is because their embodiment differs too much from human embodiment. But it might still be possible to create applications that are intelligent on a narrow domain and show an understanding of a selected part of our human lives. And it would fit the goals of AI to experiment with context-aware applications to see if this indeed is possible. The opportunities to do this in cooperation with the application engineering tradition can be seen when we continue with Dreyfus’ critique and look at his theory on holism and intelligence in the following section.

## *3.3 Holism and intelligence*

Dreyfus used phenomenological means to demonstrate that logic cannot be the direct foundation for artificially intelligent applications. It is interesting to note that the introduction of phenomenology to context awareness leads some to argue that a good context-aware application is intelligent. Sadly, Dreyfus’ argument also demonstrates that it is doubtful that this will be possible if the aim would still be to create applications that fit the engineering paradigm. But we have seen that phenomenological context awareness can also be satisfied with applications that have expertise in selected areas. The previous section only discussed Dreyfus’ analysis of how humans acquire expertise. The next section will discuss a crucial complication for creating artificially intelligent applications in a broader sense. This complication is the frame problem, which stems from the argument that since all definitions need to be defined themselves, understanding intelligence in terms of logic creates an infinite regression problem. With traditional software engineering techniques, which are based on logic, this problem

[38] Collins, H. (2004). The trouble with Madeleine. Phenomenology and the Cognitive Sciences 3 (2), pp. 165-170

cannot be solved. However, the end-user experts mentioned in the previous section have no difficulties with solving the frame problem in their daily lives. The current section will analyze the frame problem and how it is related to human actors. The intention is to look for an opportunity to use human expertise to circumvent the need for context awareness to solve the frame problem, such as delegating responsibility for taking decisions to end-user experts or to end-users themselves.

### 3.3.1 The frame problem: what humans can that computers cannot

The frame problem is an issue from the area of artificial intelligence that is closely related to a recursion problem that arises when the meaning of concepts is to be formalized. The general idea is that an infinite amount of information needs to be available when determining what is relevant in a changing situation. When a system is in a particular situation, and it wants to predict what would be the result of an action, how would it determine what has changed and what has remained the same? This is related to the infinite regression problem: if a phenomenon needs to be defined and that phenomenon consists of other unknown phenomena, these phenomena must also be defined, and the phenomena of which they consist must be defined, et cetera. Consider giving a definition of a table. You would probably define a table as a horizontal, flat surface that is supported by one or more pillars. A pillar is made of a hard material such as wood, metal or plastic, and often its color is relevant. Plastic is a material that consists of a particular type of molecules and a color is light that has a specific wavelength. But what are molecules, what is a material, what is light and what are wavelengths? In an infinite regression problem, the question of relevance is not important: all characteristics can be relevant. But when the table is “framed”, when it is placed in a specific scenario, some characteristics can be more relevant than others. For example, in a whiskey-tasting gallery the smell of the table can suddenly become relevant.

The frame problem is similar and according to the previous paragraph its name suggests that we should or can at some point state that enough detail has been given. The following scenario is based on a classical example, and it illustrates what the problem is for computer applications. Imagine a system that controls the movements of a car that has as a task to move this car while keeping everybody involved up to date on any changes by communicating with their Bluetooth devices. This system knows about two persons: John, who is standing next to the car, and Jane, who is sitting in the car. The system will not only move the car, it will also move Jane, and it should reflect this in its representation of the predicted results of moving the car. Moreover, if it moves the car far enough it can probably no longer connect to Jane’s phone using Bluetooth. This in

turn might result in more changes, all of which must be preprogrammed. This system needs explicit rules to express such effects, to express what does *not* result in any effects, and to express the exceptions. The result is that the number of formal rules the system should know will grow exponentially. In a world where an unbounded amount of elements can perform an unbounded amount of actions, the number of rules such a system should store is infinite. The frame problem calls for finding a mechanism to deal with this complexity. The central issue of the frame problem for a formal system is that it must explicitly state what is relevant and what is not, which is at least a daunting task and probably ultimately impossible[39].

Solving the infinite regression problem requires specifying all characteristics of everything in the world, and solving the frame problem requires either that or considering only the relevant characteristics. But you can only know if something is not relevant after you have considered it explicitly. So, this also implies that you have to consider everything in the world. The beauty of human cognition is, if we interpret Dreyfus' theory, that humans do not consider everything all the time, but that they learn from their mistakes and they use this to improve their guesses about when something is or is not relevant. They process all information they receive immediately and simultaneously, and, almost effortlessly, they discard some information and process other information.

Attempts to address the frame problem are based on reducing the complexity of the problem. Designers typically inspect a specific problem and after a while, they decide that they have accounted for the entities and actions that are most relevant. They can manage the number of required formal rules and procedures for decent results by carefully inspecting which relations should be affected[40]. This solves the problem for designers when they see it as an engineering problem, but it is not a real solution to the frame problem, since there can still be unexpected exceptions to the rules the designers deduced. Reducing these problems to simpler problems is too difficult for computers. Eliminating humans from this process seems impossible, and even if context awareness wants to create applications that are intelligent on a selected set of domains human intelligence is still needed to decide when something is relevant and when it is not.

### 3.3.2 Reductionism versus holism

Apparently, for context awareness holism is not only a source of solutions. When context awareness wants to use holistic principles, it also has to use reductionist principles.

[39] Dreyfus, H. L. and S. E. Dreyfus (1986). *Mind over Machine: the Power of Human Intuition and Expertise in the Era of the Computer.*
[40] Shanahan, M. (2004). The frame problem. In *Stanford Encyclopedia of Philosophy*.

According to reductionism it is possible to reduce phenomena, such as objects, systems, theories, et cetera, to (a combination of) smaller or simpler phenomena. Breaking up a system in smaller parts and studying how they interoperate is supposed to give more insight into the working and constitution of the system. The characteristics of the phenomenon are then determined by the characteristics of the constitutive elements. For example: if we want to understand how a car works, we can disassemble the car and analyze the functions of all its parts. If we know how they fit together, we will know how that car works. This reductionism fits to what is common in application engineering.

The idea of holism is that it is not sufficient to study the parts of a system in isolation: the behavior of a system in its totality must be analyzed and the system as a whole is greater than the sum of its parts. A typical reason for this is that the system is too complex to break up in smaller parts as a result of unexpected behavior of the combination of simple constitutive elements of the system; any element can be significant. Because of this, reduction of the complexity of the model is computationally unattractive since we cannot know what can be neglected and thus nothing in the world can be neglected. In the example of the car, the reductionist approach did not show why some people drive their car faster than other people do. This could be because some cars have a bigger radio. But why does a bigger radio make people drive faster? Apparently, we still do not understand the car after disassembling it.

Both principles have their own advantages and disadvantages. Reductionism simplifies complex systems by giving its primitives, but it might be blind to complex behavior that results from combining the primitives. Holism acknowledges the possibility of ‘surprising’ features of complex systems, but the theory makes predicting the results of combining the primitives more complicated

We do not usually consider ourselves as holistic systems that do not follow rules; we all know that to some extent our behavior *can* be described by rules. This is also reflected by philosophical debate. One of the philosophers who addressed this issue is Searle, a philosopher of mind and language. Searle argued that the mind is irreducible to the elements of the brain, while mental features are caused by neurobiological processes. Surprisingly, Searle leaves space to argue that combining holism with reductionism is not as problematic as would be thought at first glance. Context awareness can make use of this by pragmatically borrowing from Dreyfus’ theories on acquiring expertise. Dreyfus is a strong proponent of holism and states that this is the main cause that prevents mechanical systems from getting human intelligence. If creating intelligent applications is not possible when we accept that the human mind is a holistic system, we should determine what *is* possible and focus on that. In terms of Dreyfus’ five stages of expertise, we should focus on designing in a way that acknowledges the difficulties

context-aware applications will have with surpassing the third stage: learning to disregard irrelevant options and starting to create plans, supported by taking responsibility for the choices that have to be made. We know by now that this can only be done by real people: by end-user experts, by the users of the applications and by the designers of the applications. The phenomenological approach already focused on the responsibility of the user by designing applications that offer choices to its users. However, this still requires the designers of the applications to make their own choices and create their own plans, with or without the help of end-user experts.

### 3.3.3 What is undefined about context

A solution to cope with the frame problem that fits the engineering paradigm is to create applications that function as well as human experts in specific tasks. These artificial experts should, since they are context-aware applications, function in a way that helps its users attain their goals. However, during the design stage, humans should decide what is relevant and what is not; this is the responsibility of the application designers and the end-user experts. This section will validate this solution by analyzing the context definition that has been given earlier and demonstrating why it is perfectly acceptable for the classical approach to incorporate holism as long as the end-user experts are not excluded from the design process.

Svanæs stated that context awareness' distinction between context and non-context is "almost absurd", leading to the conclusion that context awareness needed a new conception of intelligence. The assumption that context awareness actually needs intelligence would be new to context awareness, and was introduced by researchers from the phenomenological approach. We have discussed how it does make sense for the phenomenological approach to pursue this direction: they draw from the AI tradition. We have also discussed how this ultimately is not possible to achieve without the help of the end-user experts, who were forgotten by the phenomenological approach. It is surprising that the classical approach itself raised no effective objections to the claim that their applications would need intelligence in order to be context aware. To understand the inability of the classical approach to defend itself to the phenomenological criticism on this issue, we can turn to context awareness' context definition, which appears to have an "operability deficiency": the definition does not allow itself to be translated to actual design principles for context-aware applications.

In context awareness, the statement that it is hard to give a suitable definition of the word context is widely acknowledged. As mentioned earlier, the most widely used definition is the one provided by Dey, which equated context to pieces of relevant information. Let us compare this to the interpretation that is given to context by

Merriam-Webster. This definition illustrates that it is apparently not possible to define context, but only to describe the relation between the context and that of which it is the context.

1. the parts of a discourse that surround a word or passage and can throw light on its meaning
2. the interrelated conditions in which something exists or occurs ***:*** environment, setting <the historical *context* of the war>

The Merriam-Webster dictionary, which does not have Dey's need to give a definition of context that should be useful in context awareness, gives an additional interpretation: context as part of discourse analysis. When context is used to interpret a word or passage, it is a hermeneutical act; the context can throw light on it and a person can use this to see what information is revealed. This more specific interpretation of context fits to the call for intelligence that was introduced by the phenomenological approach. However, context awareness, from its inception, prefers the second definition, where context is reduced to characteristics: both Dey and the second interpretation given by Merriam-Webster see context as a set of statements. Dey gives us slightly more information about the structure of context as it is used in context awareness. While the dictionary states that the context is constituted by *all* the statements that hold in relation to the phenomenon's existence, according to Dey *any* statement by itself can be context. Furthermore, he adds "relevance" as a restriction. But overall, the definition is still too general to be of any help for designers when constructing applications, since, for example, all applications that are adaptive would be context aware[41]. This is by itself not necessarily a problem, but indirectly, a potential problem is created. This definition is the only widely adopted pointer to what constitutes "good" design of context-aware applications, but it is far too general to do really give direction to context awareness research. This means that if context awareness only looks at this definition, it does not have an argument against the phenomenological approach, because the definition does not exclude a possible need for being intelligent in order to deal with "context".

However, if we look at what Dey had to say about the role of his definition in the design process itself, the part of his work that explained his definition, we see that he intended his definition to be used in a way that does *not* allow for such an interpretation. For Dey, the structure of "context" was important because it would help the design

[41] Zimmermann, A., A. Lorenz, and R. Oppermann (2007). An operational definition of context.

process, not because it would help with creating good design specifications. Context awareness is and was a rapidly changing and growing research field with many researchers who wanted to do something with context. Dey gave an explanation of context that simplified the difficulties of context awareness in a way that made it all easy to understand. However, Dey and Abowd added two important pointers to their efforts to understand what context is[42]. Firstly, they are *purposely* stated in general terms, resulting in a wide view on which applications can be called context-aware. This is to say: they explicitly rejected approaches that tried to formalize how context should be handled. Secondly, and this is the crux, their definition of context is supposed to enable designers to more easily create a complete overview of what contextual information an application might encounter[43]. So initially, the definition was not intended to serve as a guide for actually designing context-aware applications: it was intended to serve as a guide for modeling contexts during the development of context-aware applications.

While the definition was later used by Dey as the basis for the design of context-aware applications[44], it remains the question if the definition is suited for that task. Now the research field has finished trying to understand what context and context awareness is, the definition's generic nature is no longer seen as a feature that helps to bind context awareness together, but as one that makes it actually unfit to create designs[45,46,47]. The definition determines what context is, but not what to do with it. The phenomenological approach tried to define what to do with it, but we have now seen that they did not do this properly. The phenomenological approach's claim that it could offer a solution because it could pinpoint the problem was false. A phenomenological approach is not better fit to solve the frame problem than any other approach, it is only better able to explain why the frame problem is problematic for designers of computer applications. However, the AI research field did make progress with creating applications that incorporate features of holism. For instance, they taught us that computers can implement neural nets. While AI might not have given us a panacea, it has given us tools that both advance context-aware applications and fit to the engineering paradigm. The next section will discuss these neural nets briefly before the final section will tie together the arguments given up to this point.

[42] Dey, A. K. and G. D. Abowd (1999). Towards a better understanding of context and context-awareness.
[43] Idem, p. 4
[44] Dey, A. K., G. D. Abowd, and D. Salber (2001). A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications
[45] Zimmermann, A., A. Lorenz, and R. Oppermann (2007). An operational definition of context, p. 559
[46] Benerecetti, M., P. Bouquet, and M. Bonifacio (2001). Distributed context-aware systems
[47] Korpipää, P. and J. Mäntyjärvi (2003). An ontology for mobile device sensor-based context awareness.

### 3.3.4 From AI to context awareness: neural nets

Neural nets are sometimes said to be a universal solution for computer intelligence, but this section will argue that neural nets are, at least in context awareness, best seen as a particular solution for specific problems. The foundation for neural nets as used in computer science is the current view of the human brain: a complex network of neurons that has the rest of the human body as a feedback and sensor system. The brain consists of neurons that fire signals to other neurons, based on incoming signals. There is a simple mechanism to determine when a neuron should fire to which neuron and the result is a complex system whose main characteristics cannot solely be explained by the characteristics of neurons and this mechanism: the system is holistic. Information is, here, not represented in a formal manner as rules and data in the brain; the brain is said to be able to process information by using "superpositions", a way of transforming signals in a way that their significance can only be found by relating them to states of other signals[48]. When a signal enters the brain, for example, as a result of sensomotoric actions, certain neurons are affected, and they in turn might affect other neurons, following and creating a path that might even contain loops. Many signals stimulate the brain, from many different sources, resulting, again, in a complex pattern. This results in the idea that if anything meaningful is represented in the brain, it is distributed over many neurons, it can be accessed in many different ways, and its meaning automatically depends on all other stimuli, current and past.

Artificial intelligence has achieved good results with reusing this principle in computers. Their "neural nets" display intelligent behavior by mimicking the brain's neuron firing behavior. Neural nets can be trained to perform specific tasks. Despite their progress, it appears that it is easier to use neural nets in applications that need a certain, specific type of service, such as performing pattern recognition tasks like voice recognition, or implementing motor function, such as creating a walking robot dog. It appears more difficult to achieve results on creating systems that display behavior that is seen as higher-order intelligence, such as inference or applying rules[49]. It is however one of the things that context awareness can learn from AI: neural nets can help context awareness to solve a particular type of problems.

An earlier example that was used in this thesis was about a bus application. The application uses detectors in the mobile phone to determine if the person wearing the phone is running or walking. This particular application analyzes how the height of the application chances over time and neural nets are one of the methods used to analyze

[48] Gelder, T. (1998). Cognitive architecture: What choice do we have?, pp. 7-9
[49] Brey, P. (2002, Unpublished). Symbol systems versus neural networks.

these measurements and recognize patterns[50]. The designers do not have to understand the exact rules of running; they can be satisfied with an application that just instantly knows that the user wearing the device is running because the neural net says so. This is an example that demonstrates that the phenomenological approach can offer solutions to a particular type of problems, in this case feature recognition. This is not the role the phenomenological approach had foreseen for itself; the approach started out with the goal of finding a new foundation for context awareness and now the phenomenological approach is used to define the constraints of context-aware application and to deliver specific and specialist solutions to engineering challenges.

## *3.4 Conclusion*

This chapter set out to evaluate to what extent the phenomenological approach can improve on the perceived weakness of the classical approach to context awareness: its difficulties with making sure that context-aware applications are sensitive to all relevant contexts. The phenomenological approach suggests that context awareness is best served when application designers cooperate with researchers of human context processing, and that they together can create applications without the involvement of end-user experts. The advantage would be twofold: this approach is less labor intensive and it addresses the classical approach's shortcoming that it does not know how to create applications that can recognize all possible contexts. The phenomenological approach's alternative was twofold. First, context awareness should focus on the choices it offers to the application users. Second, context awareness can only understand contexts when it can understand all information. This chapter focused on this second answer, creating the background for the next chapter for discussing the first answer. However, I have also presented arguments for the objection that the application users are not the only ones who provide the application with relevant information: the end-user experts can often also be involved.

This chapter explored why the phenomenological approach can only identify the difficulties with recognizing all instead of a selection of relevant contexts, and why it cannot give the solution to this challenge. No computer application can understand all information: they lack the needed human physiology. While their account of intelligence is helpful, the notion to create systems with generic algorithms for intelligence that is based on this account proved misguided. Like classical context awareness, they can only give particular solutions to context awareness' problems. They are however a different type of particular solutions than these of the classical approach. The classical approach specializes in analyzing an application domain in cooperation with end-user experts and

[50] Mäntyjärvi, J. (2003). *Sensor-based context recognition for mobile applications*, p. 66.

together they can simplify the problem in a way that it can be programmed into a computer. The phenomenological approach gave us insight in what people can do that computers cannot always do: decide what is relevant and take responsibility for this action.

We can conclude that the added value of the phenomenological approach is that it adds nuance to the assumptions of the classical approach. While the classical approach assumed that end-user experts have all the relevant knowledge, we now see that researchers of human context processing have relevant knowledge too. The challenge however is to translate that knowledge from other domains, such as AI, to context awareness while staying within the application engineering bounds. This means that the primary goal should remain to be creating reliable, efficient, and effective applications according to verifiable and well-defined specifications. Dreyfus has given us an initial understanding of the limitations that must be remembered: computers only have a simple form of embodiment and they can only become experts on some particular domains.

The other assumption was that computer scientists only need logic to create their applications. We have now seen that this can be complemented with other mechanisms. Technologies that are developed by AI and that are ready to be used in real applications can definitely add to context awareness. One example is the use of neural nets. But there are probably many more examples. Searle, for example, demonstrates that humans have an understanding of logic while their brains work like holistic systems. The next chapter will explore this issue further.

Finally, while there is no reason to adopt the aims, methods, and assumptions of artificial intelligence in favor of these of the application engineering paradigm, artificial intelligence does give context awareness some interesting considerations. A short exploration of the origins of context awareness and the purpose of the used definition of context demonstrated that the main challenge of phenomenology should be to help the designer with specifying contexts and options, not to specify his designs. It is argued that applications at this point cannot be responsible themselves, and that the responsibility of the users is somewhat limited by the choices the applications present and the actions the applications perform. The designer has the responsibility to create applications that present the right choices to the user and at other times take the right choices to achieve the user's goals. To assist the designer in this task, the next chapter will discuss a phenomenologist, Ihde, who has a very useful theory on the relation between persons, technology and the world. We will see that this of course corresponds to the application user, the context-aware application, and its contexts. We will also see how Ihde moved away from the traditional phenomenological view that we should stick to our own

experiences. The focus of a purely phenomenological approach resulted in the view that context awareness should focus on interaction and not on action. To reflect the conclusion that any phenomenological approach should be subordinate to the classical goals, methods and assumptions, I will argue that context awareness needs to go past simple phenomenological reflection. To this end, the philosophical theories of Searle will be introduced.

# 4 A philosophical view on context awareness

Chapter 3 argued that the phenomenological approach as outlined in chapter 2 does not fit to the application engineering paradigm, and that it has too many unsolved challenges to pose a threat to that paradigm. However, the original reasons for adopting a phenomenological approach are still valid, and the promise of phenomenology is still the same: it can bring about improved cooperation with researchers who know a lot about human context processing and it will result in better coordination within context awareness. The previous chapter argued that the methodology of the phenomenological approach, the adoption of generalized procedures that reflect how humans process context, should be abandoned. It also gave an alternative: to structure the support to designers in creating applications that make use of human expertise. This is in line with typical middleware approaches: to create reusable designs that can be customized for specific needs. This chapter will argue that context awareness needs such middleware designs that can be optimized by incorporating knowledge from both end-user experts and context processing theorists. The difference with the phenomenological approach is that context processing theorists should be consulted in an ad-hoc manner. Phenomenology is still able to provide at least part of such a solution, since it is able to give a good initial analysis of how people will use an application. However, this chapter will go beyond typical phenomenology, because a purely phenomenological approach appears to restrict itself to interaction design, while the goal of context awareness is also to improve on activity design. Ihde's post-phenomenology fulfills this need because it is not only about experience and perception, but also, more general, about the relation between humans and the world. Searle's logical analysis is particularly relevant because it, as he argued several times in his discussions with Dreyfus, sometimes leads to the same result as phenomenology[51], but more often is able to explain a lot more than phenomenology[52]. The proposed approach is partially phenomenological because the goal is still to benefit from the experience of human actors (the end-users) and our knowledge of this experience (the end-user experts) and our ability to unlock similar knowledge (the context processing theorists). However, the approach goes beyond typical phenomenology because it does assume that a good design of context-aware application should be prepared to incorporate scientific knowledge.

This chapter will zoom in on theories from Ihde and from Searle that will help a designer understand how his application will function within its contexts and in relation to its users. To do this, I will present theories that discuss the relation between people,

[51] Searle, J. R. (2005). The phenomenological illusion, p. 322.
[52] Searle, J. R. (1999, January). Neither phenomenological description nor rational reconstruction, p. 10

technology, and the world they operate in, as outlined by these two philosophers. While this chapter will stay close to the theories as they are presented by Ihde and by Searle, they are directly applicable to context awareness, which will be demonstrated in the next chapter. In chapter 2, we saw that the classical approach assumed that end-user experts have knowledge that is essential to context awareness and that logic can be used to create applications that incorporate this knowledge. Chapter 3 demonstrated that a phenomenological approach cannot replace this knowledge but that there are other ways than building logic devices to benefit from this knowledge. It also demonstrated that the designer should decide on the distribution of expertise: the designer decides which decisions are taken by the application and which decision the application delegates to its user. Ihde's theories will explain how technologies impact on its users' experience of the world. Searle will give an explanation of rational behavior that both stays close to the holistic nature of people and can be used to create computer applications. Together they will empower the designer to create applications that can take decisions where possible and ask for additional information where necessary. After these theories have been presented, chapter 5 will demonstrate that the concepts that are introduced in chapter 3 and 4 can effectively be used to describe and discuss elements of context awareness and that such a discussion might be used to feed new insights back to scientists who reflect on human context processing.

## *4.1 Ihde and a relativistic approach*

Ihde's phenomenology can be a great help to designers of context-aware applications who want to understand why interpreting the data gathered by their application's sensor needs to require so much of their attention. He does this by making a distinction between simple perceptual acts, which are similar to those of sensors (microperception) and meaningful and interpreted observations, describing how people understand their world (macroperception). Ihde's phenomenology will help us to understand how a person will use a context-aware application to do something in the world. Phenomenology is the study of phenomena as they appear to us, of how we experience the world around us from a first-person perspective[53]. Ihde states that his version of phenomenology binds together phenomenological approaches that focus on direct observations and observations that put the observed in a wider perspective, and gets its added value there[54]. He sees the first type of association as being part of a microperceptual domain and the second as part of the macroperceptual domain.

[53] Smith, D. W. (2008). Phenomenology. In *Stanford Encyclopedia of Philosophy*.
[54] Ihde, D. (1990). *Technology and the Lifeworld: from Garden to Earth*, pp. 74-84

It appears to be this microperceptual domain that is at the center of the phenomenological approaches to context awareness. The idea here starts with the statement that people observe the world with a specific perspective from a specific location and that they use their bodily senses for this observation. The phenomenologist's task is to explore what happens when the perspective and location would be changed. This will help him to understand observations better and gain new insights from them that do more justice to the bodily limitations of people as observers. Macroperception is also based on the principle of taking a stance and adding variations to this position. However, now it is not about taking a physical stance but a cultural stance. When observing a phenomenon, the phenomenologist needs to take a step back and find a comparable phenomenon in a different cultural setting. By varying the cultural parameters this way, the phenomenologist can explicate interesting elements of the situation: he can hermeneutically interpret the phenomenon. Ihde's post-phenomenology is build on the thesis that when both approaches are used in tandem the phenomenologist will be able to get grip on what is happening: the one is used to understand the other and vice versa, and meaning arises as a result of both. A direct implication for context awareness is that it will always be a person who performs the macroperception, either in the person of the user of the application or in the person of the designer or end-user expert who anticipates on a specific macroperception by the user. The microperception is done by the application, possibly in cooperation with the user, when the application uses its sensors and actuators. Again: the interpretation of these observations depends on typical human capabilities.

Ihde provides an interesting example of how such interpretations work when he explains how, when using his methodology, experience is constituted in micro- and macroperception when we see the nighttime skies[55]. Microperspectivally spoken, one sees something dark with dots in varying degrees of brightness. But he proceeds and argues that nobody would state that this is all that he sees: there is also a macroperceptual element. Some might see the sky "*as* a dome, black at night, blue in the day, but obviously 'solid'". The lights could be holes through which we see fire, or they could be travelers across the dome. In other cultures, the sky could be considered to be infinitely stretched out as opposed to a finite dome. And one can see a bounded but large space with a sun, a moon and stars at varying distances, as a Western, naturalistic world view would dictate.

This macroperceptual perspective, which is culturally determined, is as technologically determined as is the microperceptual perspective. Consider, for example, how cultures divide the seemingly random patterns of the lights in groups. Agricultural

[55] Idem, pp. 42-43

societies could group the stars in patterns that relate to “the various cycles of the seasons and reproduction” and cultures that are actively sailing often relate stars to destinations. Here, the macroperceptual setting is determined by technology as a force that has steered the direction of the culture. If we look at our own culture, we see that both the micro- and macroperceptual setting is influenced by technology. We have various means to help measure and visualize the distance to the moon and stars, such as telescopes, observatories and satellites, which influence our microperspective. And at the same time our culture itself is interested in measuring and calculating everything, macroperspectively spoken. Context-aware applications are also bound to play their own role in this process.

Ihde states that humans never participate in only microperception or macroperception but always in both at the same time. He also states that a subject always experiences something, which is what he means when he says human experience is intentional: it is experience *of* something. This something is the world, and that intentional relation of the subject to the world is in what Ihde is interested. He furthermore states that this relation is always influenced by technology. While Ihde describes these as elements that mutually influence each other, the different elements have different emphases and the table below captures these emphases. They allow for application to the context awareness scenario, and furthermore, the division between a subject (the application user), technology (a context-aware application) and the world (the domain in which the application is used) will prove to be another powerful way for an application designer to understand his challenges better.

| | Microperception | Macroperception |
|---|---|---|
| Subject | Use senses | Use the sensed |
| Technology | Mediate sensomotoric activity | Influence interpretation |
| World | Spatiotemporal objects | Meaningful objects |

*Table 1: A schematic view of Ihde’s perceptual roles*

### 4.1.1 Relations, translations and context awareness

Ihde claims that his approach is *relativistic* but that it is not a *relativism*[56], with which he means that the statements his approach wants to make can be defended and are non-trivial. Consider an object A that is heavier than an object B. It would be a relativism[57] to say that object A is heavy, since the object is merely heavy by comparison. Focusing on

[56] Idem, p. 23
[57] In Ihde's terminology

the observation that both objects are (and can be) compared however is not dubious in this way; this implies a focus on the relation between both objects, which is defensible. Furthermore, Ihde argues that it is possible to learn to see other interpretations; statements would be trivial if they would only hold for one specific observer, but apparently, in Ihde's eyes they are not.

Ihde's approach becomes relativistic because we are constantly reminded that the observations that are discussed are observations that originate in a subject. This subject serves here as an indexical: a pointer to one side of the relation. Ihde informs us of this side of his approach because he needs to mount a defense to possible allegations of mistaking subjective observations with truths; in a relativistic approach like this one, all phenomena have a sense of absoluteness when the point of departure is fixed. This is linked to the other characteristic of Ihde's phenomenology: he presupposes that the world is experienced by us as subjects, possibly through technology. It is this role of this technology that Idhe is interested in, and it is this relation between a context-aware application and its user that a designer is interested in..

In Ihde's view, the technology is experienced with a certain translucency and the technology mediates by translating the subject's sensomotoric activity: what the subject normally can sense or do, he can sense or do differently when technology is involved. The way the technology changes how a subject can act or sense is what Ihde calls its translation. For example, a microscope translates what we see by making things bigger and a hammer translates what we can do by exerting force. A more complex example is a VCR, which is part of a system that translates where and when we see something. The message is clear: the subject is Ihde's origin of all relations and the technology translates his sensomotoric activity. This is to say that ultimately, the user is the start of the analysis, and the designer should investigate what the user can do when he uses the context-aware application.

### 4.1.2 Relations between subjects, technologies and the world

Experiencing the world through technology is Ihde's entrance to phenomenological reflection. Ihde uses a specific notation for describing this, with the following base format: "subject – technology – world". All this says is that a subject has a relation to a specific piece of technology, and this technology has a relation to the world, giving the subject an indirect relation to the world, namely through this technology. This fits closely to the view that was formulated on context awareness in chapter 2: a user of a context-aware application wants to achieve some goal in the world, and the context-aware application plays an assisting role in this. Ihde's next step makes us able to express even

more about this. He adds a sign that denotes intentionality or aboutness: when a relation is given an arrow instead of a dash, the element that is pointed to is at the focal point of the element that points to it. Consider the relation "subject → world"; here, the subject has directed his attention to the world, which is more than just an unspecified relation: Ihde states that in this case, the subject has a relation *to* the world instead of a relation with the world. The final part Ihde adds to his notation is the use of parentheses; when two elements are grouped together by parentheses, they are, to some extent, seen as a unity. We can, for example, write "(subject – technology) → world", which implies that the subject and the technology have a similar intentionality towards the world.

The dash implies another consequence: at this point, unexpected results can occur when the technology fails. Ihde calls this "enigmatic", because it is unclear what exactly might happen. For example, glasses might break, or a thermometer might for unclear reasons display a wrong temperature. Obviously, the interaction with the context-aware application might also be impacted by enigmatic behavior, and a designer should consequently always consider the possibility that a context-aware application's sensors or actuators fail or that a person cannot use the application properly for one reason or the other.

Furthermore, Ihde considers a set of combinations of elements of special relevance when considering how we experience the world. The first is the relation that was mentioned above, "(subject – technology) → world", which he calls the embodiment relation. A typical example is wearing glasses; a person who wears glasses perceives the world *through* these glasses; the glasses are transparent and the subject's intentionality is *through* the glasses *to* the world. However, the glasses are not completely transparent: the subject, and others, notice that they are there. If the technology would completely blend with the subject, this can be seen as a cyborg relation, as proposed by Verbeek[58]: "(subject/technology) → world". An example would be a person who takes anti-depressants: he would be a different person than without the anti-depressants. A person using a mobile wearable device would also fit to this relation.

The next type of relation Ihde mentions is a hermeneutic relation. Here, the subject has an intentional relationship to the technology, and it is this technology that has a relationship with the world: "subject → (technology – world)". In this case, the world itself is only indirectly accessible to the subject: he knows what is represented by the technology. An often used example is a thermometer: it represents the temperature in degrees Fahrenheit or Celsius, and while we know whether it is cold or warm, we have to

[58] Verbeek, P. P. (2008). Cyborg intentionality: Rethinking the phenomenology of human–technology relations, p. 391

interpret the number that is given to us by the thermometer to sense the world (the temperature).

This relation also has a more extreme version, as the cyborg relation can be seen as a more extreme version of the embodiment relation. Verbeek introduces the composite relation, where the technology has an intentional relation to the world, and the subject has, in his turn, an intentional relation to this intentional relation: "subject → (technology → world)". An abstract painting would be an example: the art represents some aspect of the world, and the person who sees the art is interested in how the art relates to the world.

When the technology recedes into the background, there are no explicit intentional relations. This is Ihde's background relation, which is described as "subject ( – technology – world)". An example of such a relation is air conditioning; a person is only marginally aware of the presence of air conditioning: the air is cooler than he might suspect and he might hear a humming noise. However, interaction between the air conditioning device and the person is limited to the point of being nonexistent.

The next variant is the alterity relation, where the focal endpoint of the attention of the subject is the technology itself and the relation of the technology to the world is less important to the subject. This relation is depicted as "subject → technology ( – world )". The difference between the alterity relation and the hermeneutic and composite relations is that here the relation between the technology and the world is not relevant: the technology is not only the focal endpoint, it is all the subject is interested in. A computer game can be a good example: the person playing the game is, at least with classical computer games, not interested in the relation the game has to the world. An artificially intelligent agent can have an alterity relation with its user, although it can also have a hermeneutic relation.

The last variant I would like to mention is the unidirectional relation that was introduced by Heersmink[59]. This relation describes a situation in which the technology has intentionality towards the user, while the user does not actively affect the technology or the world through this technology. The relation is depicted as "subject ( ← technology/world)" and it expresses that the technology, for example, might observe the behavior of the user and try to influence it. An example Heersmink mentions is a hearth-monitoring application used with hospitalized persons. It can also apply to ambient intelligence systems, since they often disappear into the background while they keep monitoring the user and performing actions based on their behavior.

[59] Heersmink, R. (2009, unpublished). Ghost in the machine: Brain-computer interfaces in postphenomenological terms.

For application of this terminology in the context awareness domain, two more characteristics of the three-part relation can be explicated. Firstly, there is the relative positioning of the technology to the subject: the technology's *locus*. Secondly, there is the element that is typically observed by the subject: the focal endpoint. This way Ihde's vocabulary, like Heidegger's *vorhanden* and *zuhanden*, enriches our frame of understanding of what exactly we are doing. Heidegger's famous example of using a hammer is often used to illustrate these concepts. When somebody uses a hammer to drive a nail into a piece of wood, this person is focused on the nail and the hammering, but not so much on the hammer itself: the hammer has become an extension of his hand and it is *vorhanden*. But when something goes wrong, the hammer breaks, he misses the nail or worse, his attention is redirected from the hammering towards his hammer, and now the hammer is *zuhanden*. At the same time, the focal endpoint of the user is no longer the world (the nail and the wood), but the focal endpoint has become the technology (the hammer). Equally interesting is the technology's *locus*: as the hammer was *vorhanden*, the hammer was quite near to the person hammering. But when we remember the example of the thermometer, we notice that the technology is quite close to the world and distanced from the person looking at the thermometer. These concepts apply directly to the three types of concept-aware applications: the mobile wearable device (the *locus* is near the user and the focal endpoint depends on the user's activity), the artificially intelligent agent (the *locus* is away from the user and the focal endpoint depends on the user's activity) and the ambient intelligence system (the *locus* is away from the user and the user itself is the focal endpoint of the system).

When the relations between subject, technology and world are considered, we see that the subject's attention is either directed at the technology, the world, the relation to the technology and the world or to none of them. We also see that the technology can be seen as either more or less close to the world or to the subject. The different notations of the subject-technology-world relations that are discussed fit to this distinction, as is outlined in Table 2.

| Relation | Notation | Focal endpoint | Technology's locus |
|---|---|---|---|
| Cyborg | (subject/technology) → world | World | Part of subject |
| Embodiment | (subject – technology) → world | World | Close to subject |
| Composite | subject → (technology → world) | Technology/world-relation, world | Intentional to world |
| Hermeneutic | subject → (technology – world) | Technology | Close to world |
| Background | subject ( – technology – world) | None | Close to world |
| Alterity | subject → technology ( – world ) | Technology | Independent |
| Unidirectional | subject ( ← technology/world) | User | Close to the world |

*Table 2: subject-technology-world relations*

### 4.1.3 On quasi-otherness and quasi-me-ness

Ihde introduces an interesting concept when he explains the alterity relation, which has an attractiveness that is reinforced when we analyze Verbeek's composite relationship: the concept of "otherness". He states that we perceive some technologies as if they are actual other beings, other subjects like all of us. As an example, he wants us to consider how we talk about a) riding a horse and b) driving a car. We might say about both that he is fast, he does or does not do what we want him to do, and he has a spirit of his own. We can note the two different kinds of anthropomorphism: a mild one when we talk about the horse and a strong one when we talk about the car. The mild one manifests itself in the fact that we act as if the horse really knows what we want, what can be debated. The strong one manifests itself in us calling the object a "he" instead of an "it" and us perceiving the car as having a spirit and a will.

While the horse is not a human being, it is a subject, and in Ihde's terminology, we perceive him as another and the horse has otherness. The car, however, is not a subject, and we should know that it is not a subject. That we perceive him as an "other", while it is in fact a construction, is a reason for Ihde to state that it has "quasi-otherness".

The term "quasi" indicates not only that the application is not a "real" other, but it also reminds us that the suggestion of autonomous behavior that makes us think it is an

"other" originates in the person interacting with the artifact. The person operates in tandem with the artifact, or as Brey describes[60] it: a coupled system. Going back to Ihde's terminology, we perceive the artifact as an other because it displays behavior that "fascinates" us and because it appears to be autonomous. Likewise, it appears to have subjectivity. When Ihde labels it quasi-otherness, he is dismissed of the need to discuss the level of autonomy and subjectivity further: the suggestion in our mind is triggered, and this makes us threat the car, the horse, et cetera as if it really has autonomy and subjectivity. While Ihde introduces this issue when he introduces the alterity relation, quasi-otherness can be seen in technologies that are in other relationships too. When the technology is the focal endpoint of the user's attention this can be understood easily. For example, the thermometer, which is part of a hermeneutic relation, can almost just as easily be seen as "stupid", "friendly" or any other human qualification as any "real" other, depending on for instance whether or not we like the description of the weather that "he" gives us.

However, when not the technology but the world is the focal endpoint of the user's attention, the technology is typically not seen as an other: the glasses that someone is wearing are not, as long as they function correctly, fascinating us in the way that they seem to have a will or a spirit of its own. In fact, they appear to do what we do and what we want them to do and they are noticed the least when they appear to share our will and spirit. In addition to Ihde's quasi-otherness, technology that has a locus close to the subject has what I will call "quasi-me-ness": it appears to share the subject's subjectiveness.

In both cases, as well with quasi-otherness as with quasi-me-ness, the technology translates the subject's sensomotoric range, and in both cases, the technology is in some aspect seen as a simplification of a subject. In the first case, the technology appears to us as a subject despite the fact that we know the autonomy and the mind are missing. In the second case it appears as if the technology is similar to the subject who uses the technology, despite that we know that it neither shares the autonomy and mind of the real subject nor has an autonomy or mind that is similar to his. This has implications for designers of context-aware applications, since it allows him to be more explicit and more nuanced about the attitude that the users of his applications will have towards these applications. The designer should, for example, find a balance between equating a person to his mobile phone ("I will call you tomorrow") and realizing that they are not really a unity ("I cannot hear you" versus "my phone is having bad reception"). A designer that performs such a balancing act is more likely to design applications that can cope better with failing technologies or other enigmas.

[60] Brey, P. (2005). The epistemology and ontology of human-computer interaction, p. 392

### 4.1.4 Summary

Ihde teaches us that a person's sensomotoric activity both is influenced by and influences his wider perspective: there is an interaction between microperception and macroperception. A context-aware application influences its user's microperceptual activities and its designer should investigate any macroperceptual consequences. Furthermore, all behavior is directed at something. Technology mediates this directedness, which can be a translation of what is sensed or of what actions are performed. When the technology is the object of the user's attention, the technology can be a quasi-other. It can also be close to the person; this is when it acts as a quasi-me. The difference between the two becomes sharper when we state that the technology can be close to the person or close to the world, and that the user can direct its direct attention to the technology itself or to the world. Either way, the technology affects the user's sensomotoric capacities.

## *4.2 Searle, choices and structures*

As Ihde describes that technology impacts the relation between a person and the world, Searle presents a structure that describes how people act in this world. Dreyfus taught us that it would not be possible to design computer applications that act intelligently the way people act intelligently. Searle's description of people's rational behavior should thus not be understood as a guideline for designing a rational application, but it should serve to help the designer understand where cooperation between the application user and the application is necessary. What should be kept in mind is that there are things that a computer cannot do; a designer should not be tempted to design applications that do them anyway. It is usually the expert choices that cannot be made by computer applications. Searle will help a designer to decide when the designer should have his application ask for input from its user.

The parts of Searle's theories that are relevant to these issues from a context awareness perspective are the following. First, he gives a logic structure for discussing causation and reason in relation to how and why people act. Second, he explains the distinction between first- and third-person ontologies. This distinction gives us a platform where human behavior, such as seeing something as a cause, takes place. Together they help a designer understand rational user behavior better, which will allow him to design applications that perform actions that are better aligned with user expectations.

### 4.2.1 Searle's points of view

The third-person point of view is about the tangible world that we all know: there is grass, there is a sky, people produce sounds, et cetera. The first-person point of view is about what goes on in a person's head: he is happy, understands that it is warm outside, and intends to have a chat about the weather. This distinction has its relevance in Searle's discourse on where we can find *consciousness* and later *rationality*, where he argues that consciousness is only one of the prerequisites[61] of rationality[62]. The point Searle wants to get across is both interesting and provocative: what happens in the first-person point of view is caused by phenomena from within the third-person point of view, but the former cannot be reduced to the latter. It is interesting because we can apparently do something with first-person point of view that we cannot do with third-person point of view and it is provocative because Searle knows that we probably consider causation to imply the possibility to reduce the one to the other. And it is relevant for designers of context-aware applications because apparently a person's moods, emotions, et cetera cannot be deduced from a sensor's observations in a straightforward manner. We will see in the next two sections that Searle at the same time does provide us with suggestions for dealing with this complication. First, we should take a closer look at what Searle means with this irreducibility.

When Searle argues that events within first-person point of view are caused by events within third-person point of view, he also argues that this is equivalent to saying that what happens at the psychological level is caused by what happens at a neurobiological level: the firing of neurons causes us to have some specific thought of feeling or something else. But his conception of "causing" is slightly different than what we expect it to be; when Searle states that A causes B, he states that B happening is the result of A happening and he does not say that since A happened, B necessarily had to happen. He does this because he has no suitable answer to how to deal with (non)determinism[63]: he states that both determinism and nondeterminism have unlikely results and have no suitable known physiological foundation, so we cannot realistically say that real randomness exists, thus whether determinism or nondeterminism is true. He "solves" this problem by stating we have a psychological level, where only *relevant* events can be causes[64], thus restricting what is included in accounts of causal events.

So we have two problems in the system of causation; according to Searle: 1) it is not necessarily impossible that random (nondeterministic) things happen, allowing

[61] Again, my argument is not that designers want or need to create ration actions, but they do need the proper vocabulary.
[62] Searle, J. R. (2001). *Rationality in Action*, p. 143
[63] Idem, pp. 276-298
[64] Idem, p. 280

unforeseeable influences on causation processes, and 2) it is possible that irrelevant events, which he does not want to see as causes, influence the process somewhere. While this might seem as a sketchy justification, it is clear that Searle does not want to use a logic system that invites to make faulty arguments. His system is congruent with holism, which will be discussed later, and with Ihde's relativistic account, that we have seen in a previous paragraph, and it adds the following statement: a first-person point of view that cannot be reduced to its constitutive elements. And while Searle states that we do not know what the underlying mechanism is[65], we do know that we have experiences and that we have the experience of free will. The first-person point of view is where this takes place, and this is also where our rationality is located. Rationality in this way is a mechanism that comes into play in how we act and we experience this from our first-person point of view. Both these first-person points of view as a platform and these mechanisms of rationality are concepts that have a structure that for Searle is worth analyzing on a level that is separate from the neurobiological level. When we have the first-person point of view as a platform, we can say more about the structure of causation (on the psychological, not the neurobiological level) and about rationality.

One more note must be added. While the views of Ihde and Searle on subjectivity are at least to some extent congruent, there are subtle differences. This might lead to confusion when using their theories together. Ihde postulates that we as experiencing entities can only experience the world from our subjective perspectives. We are related in that way to the world, and the technology we have is part of this relationship. All that is said about the objective world, a world that is only knowable through experience, is open to debate and results from what we have learned with this subjective perspective. In Searle's account, the first-person point of view is a construct we use to describe *ex post* what we, as experiencing entities, know, while he states that the third-person point of view causally brings about this first-person point of view. As such, it is an apparatus that abstracts from the debatable nature of facts. That note aside, they both give us a means to discuss what happens when a context-aware application and its user are involved in performing an action. The next section will zoom in on the points in time where a person who is performing an action can choose how to proceed.

### 4.2.2 Causes and gaps

Having this first-person platform, Searle argues it is not possible to predict a person's desires and intentions, but that it is possible to pinpoint moments where he forms his desires and intention. To arrive at this conclusion, we will look at how Searle compares logic of action to logic of thought. Logic of thought is straightforward and comparable to

[65] Idem, p. 298

formal logic: if we think that such-and-such is true, we are committed to this statement and the rational thing to do is to believe it. If we think that the statement that grass is green is true, we should believe that grass is green. Logic of action is less straightforward: if we think we should do something and we think that a particular way of doing it would be good, we are not actually committed to doing that thing and there are other rational things to do than that specific something. As Searle argues: if we think that traveling by plane would get us to our destination, we can still rationally choose to go by car. A factive statement is a proposition about the world that is either true or not true; according to Searle, there is no system of formal logic that describes what a rational person will do when he holds an opinion about a set of factive statements, while there is a system of formal logic that describes what a rational person should believe when he has an opinion about a set of factive statements. This is clearly a complication for designers who want their applications to perform actions that get them closer to its user's goals, since the application cannot know what its user would normally do. However, the application can choose an action and then verify if its user agrees with that action. To get more grip on the nature of this complication, I will now further explore Searle's view.

The best that Searle can do when describing a person's possible behavior is to give a logic structure that can express what a person should want to do based on his beliefs, "*all things considered*"[66]. This "all things considered" means that what this person should want will depend on the result of weighing all his relevant beliefs, such as what he thinks is important, how decisions should be taken, what is true, et cetera. The criterion of relevance makes this an unworkable structure, since it leads to infinite regression: explicitly assessing the relevance of one phenomenon requires explicitly investigating all other phenomena that might be relevant and since all phenomena are potentially relevant this is an infinitely large task. Searle suggests a way to avoid with this problem: it is too hard to determine all these things and that is why we cannot do more than assume the person already has considered or processed them, instead of giving an account of how he does this. So while all this gives us little predictive powers, we do have a system of logic that can express (again *ex post*) what a person wants to do and what he should want to do "all things considered" and given some assumptions, which is an improvement over a system that predicts what a person will do by using induction where induction cannot be used.

But a person who intends to model and actually predict user behavior has another challenge. Not only is such a person interested in what a person wants to do when he knows this person's beliefs, which he cannot; he also needs to know what this person will

[66] Idem, p. 225, his italics

believe when some actions occur, which he also cannot. According to Searle, the structure of causation is thus that it is not possible to state what a rational person will believe when some action occurs when we look at it from a third-person perspective. This is again due to the nondeterministic character of causation in the first-person point of view.

So we cannot know what a person's intentions and beliefs are. With what can Searle help us? What Searle provides are structures that describe where decisions get taken: points in time and points in lines of thought and points in action patterns where a person can or even must choose, because there are several future decisions and actions causally open to a person. When such things are experienced by a person, Searle states that he experiences a "gap", referring to the feeling we get that at these points not all is determined by rules of cause and effect.

Searle distinguishes between three different gaps during rational acting. The first occurs between deliberation and a decision, where the person forms a "prior intention" to do something: a person has beliefs, desires and other reasons to use as the basis for a decision, but all these reasons do not force him to take one specific decision. The person experiences a sense of freedom and Searle argues that he rightly does so, because neither at this psychological level or at a neurobiological level proof exists that the deliberation contains causally sufficient conditions for the decision. A similar gap in causal sufficiency occurs after we have made a decision and before we are actually acting on this decision. Searle calls this the gap between the prior intention and the intention-in-action: first we make up our mind to do something, such as lifting up your arm, and then you actually start to lift up your arm by controlling your muscles. Searle's point is not that you first need to make up your mind and only then you can act; his point is that *if* you act on this specific intention, you have had the choice *not* to act on that intention. Similarly, where Searle notes the third gap, it is not necessarily to finish the action that was initiated: it often requires effort to finish what was started. This is the gap between the prior intention and intention-in-action on one side and the continuation of the action on the other side.

To conclude, Searle presents the following structure for (rational) actions: deliberation → prior intention → intention-in-action → action continuation, with the arrows denoting the gaps. What is on the left side of the arrow is said to cause that what is on the right side of the arrow, however, the cause must be seen as an explanation and not as "causally sufficient" to make the part on the right side happen. What is more: the person as a subject who is performing the action experiences the parts at the arrows as moments where he has the freedom to act and choose: he experiences these points as gaps in causality. These gaps are especially interesting since they are experienced in the first-

person perspectives and it appears there is no third-person perspective account that can explain what happens in the gap in such a way that causal sufficiency, and with that determinacy, can be achieved. Any model of how people think, deliberate, act, feel, expect or how they have any other mode of intentionality in the world must do justice to this structure. The implications for context awareness are that causes have an explanatory function and that they do not actually determine what will happen next. When a causal process is going on, the decision to initiate or stop the next step in the process takes place in a black box which is a gap in the model.

While Searle's rejection of the idea that a designer can predict what a user wants to do next would seem like a step back, this account of rational action also gives us some good starting points for improvements in the design of applications. Obviously, it is always better to design an application according to a correct theory, but the rejection of an incorrect theory is not the only thing that Searle can do for us. He demonstrated that an application can do more than present choices to users: it can make choices and start acting, and in parallel inform its users and ask questions. Furthermore, Searle gave pointers to the designers on the moments to ask for input, namely at the gaps that are foreseen by the designer.

### 4.2.3 The disclosing of motivations and intentions

Searle argued that it is impossible to predict with certainty what a person wants. The reasons are all variations to the statement that what counts as a cause for acting at the first-person point of view is not causally sufficient for acting. The moments where freedom is experienced, are the moments where actual gaps in the chain of events can occur, the outcome of which cannot be predicted. However, when a person partakes in social interaction, something interesting happens: he gives insights in his reasons for acting. More specifically: we can learn to what this person is committed, and these commitments will function as a motivator to this person. This way, Searle in the end salvages attempts to systematically use observations of behavior to say something meaningful about a person's intentions: behavior indicates a commitment and this suggests a possible course of action.

Crucial to Searle's line of reasoning here is that he states that almost all social interaction is filled with norms, and these norms are held dear to the person interacting[67]. When I say "it is raining outside", I typically understand that you will assume I think it is raining outside and that I should be truthful in this. When the question of the weather conditions is raised, my believe that it is raining outside motivates me to say that it is raining, in line with my commitment to the truth of my

[67] Idem, p. 182

statement that it is raining. My conviction of the truth is implied in me making that statement: otherwise, I would be telling a lie. Similarly, when I order and drink a beer in a bar, I have also communicated my intention of paying for that beer: I made this proposition. I committed myself to paying for it in a similar fashion to committing myself to the belief that it is raining. But more importantly, I have now given insight in one of my future actions, I have made it likely that, in the future, I will pay a beer.

Searle criticizes what he calls “classical theories of rationality” amongst others as leaning too much on thinking in terms of means to an end. He states that we do not think or behave rationally based on how we induce proper means to an end. According to Searle we can be committed to certain statements and this can motivate us to start acting in line with this commitment, even though this action is not necessarily something we desire. This is one mechanism that describes getting a reason for acting and the good thing about this reason is that it includes publicly announcing this reason. Information is communicated on an end that a person wants to pursue and even though it is not part of a simple construct of a “means to an end”, there is a correlation between the end and the proposition. So although the naive interpretation of Searle’s work is that he proves that it is impossible to predict human behavior, a closer reading teaches us that Searle proves that people are perfect judges of when something is relevant. Instead of focusing on what cannot be done, designers should focus on what can be done: spotting events that people think are relevant. This also holds for designers of context-aware applications. If they want their applications to learn what their users want, they should use their sensors and actuators to investigate to what its users are committed and then direct their behavior in line with these commitments.

### 4.2.4 Summary

Searle argues[68] that rationality is not simply a matter of following a special set of rules. Even more, he argues that merely following rules is an activity that lies outside the domain of potentially being rational. In order to be rational, it must be possible to be irrational; one must have a choice where he is free to choose between options. This freedom occurs within the gap, and according to Searle, the most likely but also unsatisfying scenario holds that even at the most fundamental level, that of neurobiology, we still do not know what happens in this gap. As a theoretical fundamental philosopher however, his task is to shed light on what we do know about the structure of rationality and some elements of that endeavor have been introduced in the previous paragraph. Firstly, there is the distinction between the first-person point of view (that what happens at the psychological level) and the third-person point of view

[68] Idem, p. 8

(that what happens at the neurobiological level). While the human brain brings forth rational behavior, this behavior cannot be reduced to mere neural patterns. We knew this, because this is in line with what Dreyfus argued about holism. But Searle takes the argument one step further; he states that we should look at the psychological level, because it has more explanatory power than the neurobiological level. Searle warns us that rational behavior is to some extent unpredictable, and he pinpoints where this unpredictability occurs. First, a person has freedom of choice, which manifests itself as three gaps in the stages of rational behavior: deliberation → prior-intention → intention-in-action → action continuation. Second, a cause on the psychological level is more like an explanation of an action than a causal sufficiency for an action. These two concepts should be used by the designers of context-aware applications to get their applications to move forwards to the goals of its users. The concepts allow us to explicate suggestions to a designer of context-aware applications. The gaps are moments where his application can make a choice itself or can opt to ask for input from its user. And when the application detects a cause, it should consider this cause to be information about what has happened before and be very careful when using this information to determine what should happen next. This is because it is information about to what the users of the application are committed, it is not information that defines what they are actually are going to do or what they are thinking.

## *4.3 On combining the theories*

We have seen parts of the theories of Dreyfus, Ihde and Searle, and in this section, I will argue that they can be combined for use within the limited scope of assisting designers of context-aware applications. The phenomenological approach to context-awareness as it was discussed in chapters 2 and 3 used phenomenology to argue that design of a context-aware application should be based on generalized, universal procedures, namely those presenting choice to users because only this way justice can be done to the view that only people can reveal the meaning of anything. After arguing that this is not in line with what context awareness should want in chapter 3, I argued that context awareness should want to do more than design a system that interacts with its users based on generalized, universal procedures: it should both want to act, react, and interact, and its designs should be such that they can easily be adapted to any scientific theory on human context processing and to all end-user expert input. This implies that designs should have core functionality – middleware – that is based on concepts that do not have to be changed when the scientific domain in which it is used is changed, which is what phenomenology is about, and what is in line with what Searle's logical analysis claims to be able to do: to provide concepts that describe human experience of the world and to give a logical analysis of actions within this world. However, combining the theories of

these three philosophers is generally agreed upon to be problematic. This section will discuss these difficulties and propose that the limited scope that results from application of the theories to context awareness is the reason that they can be combined here.

### 4.3.1 Phenomenology and logical analysis

In his book *Consequences of phenomenology*[69], published in 1986, Ihde starts with giving an overview of how phenomenology started developing its own profile in America. The picture he paints is that of an activity that is both distinct from and similar to phenomenology as performed by continental phenomenologists, such as Husserl, Heidegger and Merleau-Ponty, Sartre, and Schutz. It is similar in that it reveres scholarly work, and thorough discussions of "giants": the big names and authorities in phenomenology. It is different in that it dares to go past the traditional themes of phenomenology: trying to find foundations for knowledge in human perception is extended with applying phenomenological methods to make new insights possible in other fields. He mentions Dreyfus' works on computer cognition as an example of a scholarly phenomenological approach on non-traditional topics[70]. Ihde also offsets the American phenomenological approach to analytical philosophy, an approach that is linked to the American geographical area. While the American phenomenological approach has with analytical philosophy in common that it can be liberal in choosing its domains of reflection, Ihde argues that its method differ in a way that can result in conflicts and misunderstanding[71]. He argues that analytical philosophy wants to be similar to a scientific paradigm, were facts can be accumulated and logic is elementary to finding facts. In the resulting adversary model, where philosophers analyze theories of contemporaries, this can conflict with the scholarly method of the phenomenologists, and particularly the American phenomenologists, since they can address the same topics. Imagine a phenomenologist who interprets a specific word as one of the giants would do, and who wants to use this word to shed light on a phenomenon, such as computer cognition. Now imagine an analytical philosopher, who uses the normal, everyday meaning of that word, detached from that scholarly interpretation, and then does not recognize "shedding light on a phenomenon" as a valid philosophical challenge. Although a lot has happened since Ihde wrote this overview, this seems to be a spot-on description of the conflict between Dreyfus and Searle.

### 4.3.2 Dreyfus and Searle

Although Ihde mentions Dreyfus explicitly as an example of an American

[69] Ihde, D. (1986). *Consequences of phenomenology*.
[70] Idem, p. 22
[71] Idem, pp. 12-13

phenomenologist who made the phenomenological cause understandable to analytical philosophers[72], the opportunity for conflict turned into a long lasting discussion and disagreement between the two. To summarize the discussion, Dreyfus accuses Searle of being a dualist, and Searle, in his turn, states that Dreyfus' approach is useless because of his skeptical attitude towards science. The falsity of the claim that Searle is a dualist has been properly addressed in Section 4.2.1, where I explained that events at the psychological level are caused by events at the neurobiological level, while the former cannot be reduced to the latter. This section will briefly attempt to make the point that Searle perceives Dreyfus as being skeptical towards science because of the reasons that Ihde pointed out, that Dreyfus' methodology might indeed be inadequate for Searle's objectives, and that Dreyfus is not irrelevant for the objectives of context awareness[73].

Searle accumulated and organized his arguments against phenomenology, and particularly, against Dreyfus' phenomenology, in his 2005 article "The phenomenological illusion"[74]. His main point is that many phenomenologists falsely think that human experience is enough to get to know everything about the world. He argues that logical analysis is needed, because there are many things that are true but that do not directly present itself to the human faculties. Searle argues that where phenomenology gets to its limits, logical analysis is just getting started. Although I have not been able to find any response by Dreyfus that addresses and solves this issue, it might be an interesting[75] challenge to cognitive scientists. A phenomenologist might propose that logical analysis falls within the domain of experience, and he should propose an account of "coping" that includes logical analysis that is supported by neurobiological-friendly, but not generally accepted accounts such as Van Gelder's vision on connectionism[76]. Such a reconciliation is far beyond the scope of this thesis. What is more important here is what, according to Searle, logical analysis is able to add. Searle states that it is able to tell us more about reasons and causality, which are phenomena that do not present themselves to our experience or consciousness in a way that phenomenologists are able to recognize and acknowledge. Dreyfus objects that these are facts that we only believe are true, and Searle answers that this indeed is the way science works and that they indeed are our best efforts towards determining the "basic facts" of the world[77]. At the same time, Searle acknowledges explicitly that anything that follows from logical analysis should not be in contradiction with what follows from a phenomenological account[78]. The

[72] Idem, p. 22
[73] A lot more can be said about the overlap and differences between the works of Dreyfus and of Searle. However interesting such an analysis would be, it is outside the scope of this thesis.
[74] Searle, J. R. (2005). The phenomenological illusion.
[75] Or convoluted, depending on if you believe in the merits of phenomenological accounts.
[76] Van Gelder, T. (1993). Connectionism and the mind-body problem: exposing the distinction between mind and cognition.
[77] Searle, J. R. (2005). The phenomenological illusion, p. 325.
[78] Idem, p. 335

explanatory power that both approaches have, is however still different. While the phenomenological approach is, for example, able to explain why we, as people who are involved in our current world, understand the world differently from people who grew up in the Aristotalian world, Searle cannot be bothered with such an account. As Ihde already predicted: Searle is interested in the questions that can be asked and answered by us, modern people. While Searle contests that differentiating between the two is useless, Dreyfus would need to agree with all of Searle's conclusions if he assumes that the modern worldview is timeless and correct. This seems to be a reasonable assumption for context-awareness, especially given the pragmatic nature of the classical approach. So, despite their fundamental differences, given that context awareness wants to make use of modern, scientific results, it can accept both Dreyfus' phenomenology and Searle's logical analysis.

### 4.3.3 Towards a new approach

Dreyfus' methodology has been used in chapter 3 to analyze where science can be used to delegate making choices to computer applications, which is in line with Searle's view that "phenomenology sets conditions of adequacy"[79]. Searle has given designers a means to analyze rational behavior and interpret user responses. Ihde has given phenomenology a means to extend its reach beyond the human body. Technology, which includes context-aware applications, are either a means through which a person experiences the world, or it is part of the world that this person experiences. Going back to the goals, methods and assumptions that were outlined in chapter 2, I will now propose a new approach that is in line with the findings of this chapter.

The new goals are very close to the goals of the classical approach, but they have incorporated a lesson from the phenomenological approach: sometimes it is better to ask for additional input or to perform any other action that results in the application receiving the additional input it needs to progress on achieving its user's goals.

NG1. To improve the usability of the applications

NG2. To create applications that pursue the goals of its users by initiating activity or adapting activity, based on interaction

NG3. To create applications that require less attention of its users

In line with Ihde's position on how people use technology, the first methodology is more focused on the relation between the user, the technology, and the world. Furthermore, as designers will discuss the procedures that are implemented in the design more closely

[79] Idem, p. 335

with context processing theorists, they need to be able to explain their design more easily to people who are not intimately familiar with computer design jargon, a principle that was illustrated in section 2.1.2. Finally, the designers apply formal methods were possible and any other methods when available.

NM1. The applications performs sensomotoric activity that positions itself between one or more agents and the world

NM2. The design is based on concepts, or abstractions, that are recognizable to people who are not familiar with computer application design

NM3. The application designers cooperate with end-user experts to determine the proper abstractions and logical procedures to connect to application actions

NM4. The application designers cooperate with context processing theorists to learn about how people process contexts, in support of NM3

The underlying assumptions stem directly from the analysis in chapter 3 and the discussions in chapter 4. I argued that the phenomenological approach attempted to improve upon the pragmatic methodology of the classical approach by dismissing the use of abstractions and generalizations, but that its alternative was unable to produce the applications that are created in the application engineering tradition, resulting in NA1. However, Dreyfus has also shown that the classical approach was also too ambitious, and thus context awareness needs a way to deal with the issue that user behavior is sometimes hard to predict and at other times even impossible to predict, as is stated in NA2. NA3 articulates that there are three steps to approach a correct assessment of the context in which the application is used. First, context processing theorists can provide an analysis that can range between strict laws or rules, through guidelines, to indicative pointers to what can be expected. Second, end-user experts can make predictions based on human experience. Third, the application users themselves can often also give important input. NA4 is a statement that stems from the software engineering paradigm, and together with the other three assumptions it results in the view that middleware solutions should focus on assisting the designer in making the right design decisions, and should not be implemented in a too specific manner.

NA1. Although science cannot give all the abstractions and relevant information that a designer needs, it is necessary to rely on science and abstractions to produce applications that are reliable and verifiable.

NA2. Generic, universal procedures are not good enough to predict user wants and needs, people can do this a lot better.

NA3. Knowledge and expertise can be obtained from end-user experts, application users, and context-processing theorists.

NA4. Ad-hoc design must be avoided by creating designs that can at least partially be reused.

## *4.4 Conclusions*

This chapter outlined theories of the post-phenomenologist Ihde and the philosopher of mind and of language Searle. In the previous chapter we have seen the limitations that Dreyfus outlined for context-aware application. With Dreyfus' analysis of human beings as the standard of intelligent behavior, Searle's logical structure of rational acting and Ihde's framework for subject-technology-world-relations a foundational theory for a methodology for an approach to context awareness that combines the best parts of the classical approach and the phenomenological approach to context awareness.

Ihde teaches us that a person's sensomotoric activity both is influenced by and influences his wider perspective: there is an interaction between microperception and macroperception. Furthermore, this behavior is directed at something. Technology mediates this directedness, which can be a translation of what is sensed or of what actions are performed. When the technology is the object of the user's attention, the technology can be a quasi-other. It can also be close to the person; this is when it acts as a quasi-me. The difference between the two becomes sharper when we state that the technology can be close to the person or close to the world, and that the user can either direct its direct attention either to the technology itself or to the world. Either way, the technology impacts on the user's sensomotoric capacities.

While Ihde helped us understand the relation between people, technology, and the world, Searle, as a critic of phenomenology, argues that a mere phenomenological solution is not productive since phenomenology cannot "dig deeper to get at the real underlying structure"[80]. His logical account does get deeper, or more accurately: it gets us to a higher level of abstraction. His description of rational behavior at the psychological level offers direct suggestions to designers of context-aware applications. He denotes three gaps in the behavior of people who perform rational actions (before the prior-intention, before the intention-in-action and during the action continuation) that should be accounted for by applications that also have to display part of such rational behavior. Second, observations of a person's behavior have an explanatory function, but they have very limited predictive powers.

The combined theories of Dreyfus, Ihde and Searle as they are applied to the design of context-aware applications help to structure the challenges of context

[80] Searle, J. R. (1999, January). Limits of phenomenology.

awareness and they define its limits. This chapter provided designers of context-aware applications with a theory that allows the phenomenological approach to reconcile its efforts with the classical approach. Chapter 5, the final chapter of this thesis, will demonstrate that this new theoretical foundation has the power to properly address current issues of the classical approach and that it enables us to see context awareness as a field of science that gives us interesting food for thought from a philosophical perspective.

# 5 Design implications and reflection

This thesis has given an overview of how to design context-aware applications and it has given an analysis of the role of phenomenology in its design. A philosophical approach that was introduced in a technical field has been evaluated and new philosophical theories have been introduced with the aim of demonstrating their relevance to an active, lively, and promising field of computer science. After giving an overview of context awareness in general, I contrasted the classical and the phenomenological approach to context awareness. This led to the conclusion that phenomenology has a lot to offer to context awareness, but that the proposed phenomenological approach also has some difficulties. Chapter 4 demonstrated that it is possible to overcome these difficulties by outlining theories from Ihde and Searle. These theories, together with Dreyfus' theories, are a good foundation for a phenomenological approach that is reconciled with the classical approach.

The current chapter will evaluate this new approach and show that the philosophical theories that were presented led to a methodology that reconciled the classical and phenomenological approach. More precisely: this chapter will discuss how using theories on human context processing in general fits to the classical approach's assumptions that knowledge from end-user experts is required. The relevance of the new approach will be demonstrated by two discussions. First, sections 5.2 and 5.3 will discuss problems that the classical approach faced, which up to the introduction of these philosophical theories did not have a satisfactory motivation. This discussion will demonstrate the aptness of the conclusion of chapter 3 that context-aware applications should be designed to distribute the responsibility for intelligent behavior over the designer, the applications, and its users. Second, chapter 5.4 will confirm that the concepts introduced in the current chapter can be used for discussing the actions of context-aware applications. To conclude, section 5.5 will demonstrate that the new phenomenological approach is both acceptable and useful to researchers from the classical approach, and that it is relevant to philosophers who want to see their theories tried and tested in practice.

## 5.1 Summary of the findings so far

This thesis started out with a chapter that gave an overview of context awareness in general and the difference between the classical approach and the phenomenological approach in particular. Context-aware applications intend to make life easier for people by taking over some of their tasks, namely these tasks that bring a person closer to achieving his goals. The classical approach wants to do this in close cooperation with people who have expertise in the circumstances in which their applications are supposed to operate: the end-user experts. The phenomenological approach wants to do this by

incorporating theory on how humans process context. The next chapter, chapter 3, discussed the problems that arose from this initial phenomenological approach. I argued that the classical approach is an engineering discipline that wants to design applications that can be verified to do what they need to do. I also argued that the phenomenological approach introduces too many elements from the artificial intelligence paradigm, elements that do not fit to the engineering paradigm. More specifically: the phenomenological approach will lead to a situation where the frame problem needs to be solved and since no suitable solution has been found yet, this would be a problematic complication to creating context-aware applications. Context awareness needs to avoid this problem. The phenomenological approach did however point out convincingly that context-aware applications could be improved by paying more attention to theories on how humans process context. Phenomenologists involved in context awareness should realize that the knowledge of end-user experts cannot be replaced by creating applications that have smarter generic functionality. As was argued by the phenomenological approach, phenomenology should be used to present choices to the users, and to help the designer with deciding how to design applications in a way that they keep acting in order to achieve its user's goals. It is the task of phenomenologists to describe how people and technology are related to the world, and how users of context-aware applications act.

Whereas the phenomenological turn drew its inspiration from early phenomenologists such as Husserl, Heidegger, and Merleau-Ponty, contemporary philosophy has had many opportunities to improve on their theories and fit them to today's scientific challenges. The reconciliation between the classical approach and the phenomenological turn should draw from these contemporary philosophical traditions. Ihde's vision on phenomenology as a relativistic approach to "provide a perspective from which to view the terrain [and] a framework or 'paradigm' for understanding"[7] illustrates how phenomenology can offer the pragmatic methodology context awareness is looking for. Searle's theory on how people behave rationally is especially fit to link a phenomenological approach to the classical approach because he was also able to create a link between the holistic principles that operate at the neurobiological level and logical structures that describe what happens at the psychological level. The task that was set out in chapters 3 and 4 was to get to an approach to context awareness that combines the best of the classical and the phenomenological approaches. The key is to respect Dreyfus' theory on the limits of computers.

Dreyfus showed some of the limits of what can be achieved by applications, and how people achieve expertise. In some activities, computers can only become 'competent', which is not as good as an 'expert'. This is related to the simpler form of embodiment of

applications. However, the sensors and actuators in context-aware applications offer possibilities of creating applications that are better at some tasks than its non-context-aware counterparts are. To benefit optimally from these opportunities, a designer should carefully design its applications to exploit the expertise from the application users and from end-user experts. The assumption that is postulated in chapter 3 is that this can be done by structuring the design according to principles that are deduced from theories from not only Dreyfus but also from Ihde and Searle. The role of artificial intelligence itself should further be restricted to being a supplier of useful techniques such as neural nets.

Ihde teaches us that a person's sensomotoric activity both is influenced by and influences his wider perspective: there is an interaction between microperception and macroperception. A context-aware application influences its user's microperceptual activities and its designer should investigate any macroperceptual consequences. Furthermore, all behavior is directed at something. Technology mediates this directedness, which can be a translation of what is sensed or of what actions are performed. When the technology is the object of the user's attention, the technology can be a quasi-other. It can also be close to the person; this is when it acts as a quasi-me. The difference between the two becomes sharper when we state that the technology can be close to the person or close to the world, and that the user can direct its direct attention to the technology itself or to the world. Either way, the technology affects the user's sensomotoric capacities.

Searle argues that rationality is not a matter of following a special set of rules, but in making choices in all freedom. This argument leads to the conclusion that user behavior ultimately is unpredictable. However, Searle's discourse does lead to suggestions as to what can be predicted and what can be inferred from observations. First, a person's freedom of choice manifests itself as three gaps in the stages of rational behavior: deliberation → prior-intention → intention-in-action → action continuation. The gaps are moments where his application can make a choice itself or can opt to ask for input from its user. Second, what we typically call a cause for an action is more like an explanation of why we perform an action, because the causal sufficiency for actually performing that action is missing. These two concepts should be used by the designers of context-aware applications to get their applications to move forwards to the goals of its users. They should do this by asking for input from users according to the gaps that are expected to occur by the designer and end-user expert, and considering the responses to be indications of to what its users are committed. Since it is the task of this chapter to evaluate the results of this new reconciled phenomenological approach, the following

sections will discuss the findings of this thesis more loosely from and move towards a reflection on context awareness itself.

## *5.2 On holism in the classical approach*

As was concluded in chapter 3, a phenomenological approach should still respect the principles of engineering, similar to how the classical approach fits to the engineering paradigm. Chapter 4 presented phenomenological theories that can actually fit within the classical approach. This section will demonstrate that it indeed is not problematic for the classical methodologies to incorporate phenomenology's holistic principles. The way to do this is to go back to Dreyfus. Dreyfus, not quite unlike Searle, argues that people are not rule-following machines. In Searle's apparatus, a cause on a psychological level is not an event that triggers an inevitable response, but a motivator that is available to be selected as a reason to act upon. This is how Searle's "gaps" introduce the possibility and necessity of freedom of will and unpredictability of rational behavior. Dreyus in his turn pointed out that after 2000 years of scientific effort in explicating rules of nature, with the introduction of the computer, researchers are finally able to reach the conclusion that human intelligent behavior is not a matter of following rules, and that it was "merely" orderly behavior that was mistakenly interpreted as rule-following behavior[81].

According to Dreyfus, artificial intelligence's inability to create intelligent systems by explicating rules that are to be followed in order to "produce" intelligent behavior is not a failure that stands by itself. He argues that "it is not some specific explanation […] that has failed, but the whole conceptual framework which assumes that an explanation of human behavior can and must take the Platonic form, successful in physical explanation […] has failed"[82]. It is up to the phenomenological approach to give us an alternate way for us to understand human reason. This alternate way is firstly a description and secondly, possibly, an explanation[83].

Not only have philosophers such as Husserl, Heidegger and Merleau-Ponty defined this challenge and set out to tackle it, many have joined them and fruitful results have followed from it, the phenomenological turn in context awareness being only one minor example. Sadly for artificial intelligence and for context awareness it must be observed that while we, human beings, might learn to understand more about human intelligent and rational behavior, it will still be impossible to translate this understanding to rules and facts for a digital computer to use as a means to produce intelligent and rational behavior.

[81] Dreyfus, H. L. (1992). *What Computers Still Can't Do: A Critique of Artificial Reason*, p. 271
[82] Idem, p. 232
[83] Idem, p. 233

As was demonstrated in chapter 2 and chapter 3 of this thesis, context awareness did however at some point take steps towards incorporating knowledge on how people process information. Researchers proposed to change designs in a way that took into account the lessons learned by phenomenology: we should not interpret data, we should uncover meaning; abstractions are misleading because all situations are unique; context-aware applications need intelligence and embodiment. These views create difficulties: it allows for poor design guidelines, possibly because of the vagueness of the language, but mostly because it does not structure the tasks of the designer, and it misleads designers into thinking it should perform an impossible task: create fully embodied intelligent machines.

Since rational behavior and human intelligence is reserved for human beings, a clear demarcation of expectations is called for if designers are not to fall into this phenomenological trap. The pitfall is to think that context-aware applications should be intelligent, the solution is to explicate that it is not the application, but its designers and its users that are intelligent. Similarly, it is not the application that should know what the meaning of a specific context is, it is the designer, the end-user expert, or the application user that should know what it means to be in a specific context.

The situation is however even more complicated than this. The application to some degree behaves itself as if it is a real subject and moreover: it has some autonomy. It is easy for people to mistake such an application for an intelligent application. The designer should be fully aware that the application is not and should not be intelligent. An application user however will have a positive user experience if he considers the application to behave intelligently. The application might not “have” intelligence, it might not display truly intelligent behavior, but it will display behavior that is easily confused with intelligent behavior: it will suggest a display of intelligent behavior.

A context-aware application’s suggestion of intelligence is this: it is a quasi-other that takes over parts of the tasks of people and it works towards reaching its user’s goals in a way that takes into account the things it has learned from its sensors. Exploring what it means to say that a context-aware application does not exhibit human intelligent behavior is a process that has two tracks. This is due to two roles that are taken up by humans in their relationship to context-aware applications. Firstly, there are the humans that design the applications, putting effort in the design process in a way that results in intelligence-suggesting applications. Secondly, there are the humans that use the applications, interacting with them in a way that distributes intelligent behavior over the user, the technology, and the world. To summarize: not everything should be left up to the applications. But let us also look at what a computer, according to Dreyfus and Searle *can* do.

### 5.2.1 Neural nets

According to Dreyfus, the highest level a computer application can match without serious problems is that of a competent performer. Of special interest here is that a true competent performer has learned how to contextualize some features: he has learned to recognize subjective features and prioritize them in order to interpret the situation correctly. A computer cannot realistically learn how to recognize instantly and holistically which features are relevant. However, it can be programmed to investigate all possible *objective* features and thus, making use of its speed and precision, achieve the same results as a competent performer. A computer can recognize "context-free facts" and "organize [them] in terms of goals, like a competent human being"[84].

Since Dreyfus' writing of his main books on the limits of digital reasoning, much progress has been made in research on pattern recognition, especially using neural networks. The complexity and embodiment issues however still stand when neural networks are seen as an alternative for rule systems: it would take an unimaginable amount of effort to produce a suitably large and efficient net, which would still be an inadequate net since it would lack the embodiment needed for proper feedback.

However, sensomotoric tasks such as facial recognition or tactile operations such as picking up items have benefited greatly from neural nets and other learning mechanisms for digital computers[85]. Such techniques are simplifications of the processes that occur in the human mind. The neural processes in the human mind are explained to rely on superpositions, where many synchronous and asynchronous patterns are stimulated and co-stimulate each other, resulting in the complex system that we see as holistic. We are not there yet, but the local, objective features Dreyfus talks about are good targets for the artificial neural nets, since at the stage of a competent performer, where such features are relevant, no holistic effects are expected of the performer[86].

So while neural nets are until now not suited to bring a context-aware application to the level of an expert performer, they can be great tools to recognize features in the environment or to perform actions in the environment. With Dreyfus' critiques in the back of our minds, it is prudent to keep neural nets' usage for now limited to such domains. This is perfectly in line with the way many designers use neural nets now. For example,

[84] Dreyfus, H. L. and S. E. Dreyfus (1986). *Mind over Machine: the Power of Human Intuition and Expertise in the Era of the Computer*, p. 65

[85] Vlasveld, J. (2007). On context in context-aware applications. Master's thesis, University of Twente, p. 80

[86] Except for the holistic processes that are going on in the competent performer as a learner, which are ignored here.

Mäntyjärvi[87] explicitly leaves room for pattern recognition to interpret data from mobile device sensors as an alternative to following rules. For him, the sensor data “it is above 15 degrees Celsius, it is winter and we are in Holland” as read from the GPS, clock and thermometer can both be processed in a rule-like manner and by a neural net: once the contextual information (or in Dreyfus’ terms “subjective feature”) “it is warm” has been determined, it is no longer relevant how it has been determined. The complex data is once again reduced to a simple observation, reaffirming that the application does not function according to holistic principles but according to reductionist principles, making it easier to formalize and verify what the application is doing.

### 5.2.2 On making choices

Yet, when going back to Dreyfus’ five steps, we will notice the following. The context-aware application will have limited learning capabilities: its learning capabilities will be confined to fine-tuning how to recognize features. It will not become proficient or an expert, since it will not holistically determine *in a flexible manner* the current situation or in the case of an expert the next step to take. I am stressing the flexibility, because this seems to be what is at stake. Surely it would be possible for a computer system to implement a neural net that replicates some of the responses of a human actor, but this would only be a subset of all the possible responses a proficient or expert performer would be able to display.

So it is up to the designer to determine which situations a context-aware application should be instructed to recognize, and how it should recognize them. It is up to the designer to ensure that if necessary end-user experts are consulted and involved in such a way that their experience is used to “teach” an application what it should know. And it is up to the designer to gather enough data on the situations the context-aware application is likely to be put in to enable it to respond in a similar manner as a human competent performer, or in some situations possibly even better.

What is asked of the end-user expert here is to explicate the causes on which he would act, were he to behave rationally in such a situation. Such a cause is then inspected by the designer and it is used to create a rule set for the application. With a “cause”, I mean a cause on a psychological level as described by Searle. These causes do not have the function of rules, but they are the explanations given by the experts as to why they chose to act in that specific manner. Some reservation must be held towards whether or not the expert would really have the application act upon this cause, since the causes function more like guidelines. An uncertainty is introduced, and Searle’s

[87] Korpipää, P. and J. Mäntyjärvi (2003). An ontology for mobile device sensor-based context awareness.

apparatus of “gaps” is well suited to deal with this. During such a gap, a choice is made: a judgment call is made by a subject, using a mechanism that cannot be implemented in a computer application. Just like a person has to decide whether or not he intends to step on his bike, whether or not he starts stepping on his bike and possibly whether or not he continues putting effort in stepping on his bike, the context-aware application needs a mechanism in his design to determine how to deal with the range of possibilities that result from recognizing the possible causes that are presented at any moment in time.

If the user of the application has the impression that the context-aware application is in any way intelligent, it is because the designer has properly predicted which causes to act upon to consider when acting intelligently. It is the designer who really judges which causes to consider and how to consider them, not the application. But the designer is not the only one who is fit to make such judgment calls: the user of the application might also be perfectly suited for this task. The application might, for example, continuously offer the user suggestions, or ask him questions about his intentions or expectations. This modest ambition level should be intrinsic to context awareness. To conclude: the assessment of a phenomenological approach to context awareness that states that applications cannot be intelligent does not pose as large a threat as might initially be imagined, because at the same time phenomenology itself offers two possibilities to make context-aware applications better. The first is a means to incorporate technologies from artificial intelligence such as neural nets. The second is a powerful structure of distributing responsibilities for rational acting over a designer, an application, and its users.

## *5.3 On applying the theories*

There is a lot more to say about smart behavior and context-aware applications. The philosophical theories that were presented can be used to describe in more detail how the applications should be designed. This is a matter of further application of these theories. Some pointers have already been given, but a more detailed application is outside the scope of this thesis. Nevertheless, this section will make a first attempt at demonstrating the theory’s applicability. It will give a short recapitulation of parts of the theories as they were outlined by Ihde and complemented by Verbeek and Brey on the relation between people, technology, and the world. It will also refer to Searle’s statement that people’s minds are the location where judgments are made. With their vocabulary, a theory from the classical approach will be discussed. What is special about this discussion is that before the introduction of these philosophical theories there was no good way to explain why this specific classical methodology was a good methodology. After the introduction and application of Ihde and Searle we can say that it is a good

methodology because it results in applications that do not draw conclusions that they should not, and it supports placing responsibility for decisions with the right parties. In short, it results in applications that make life easier for users, but still have a modest enough ambition level.

In practice, this modest ambition level serves to prevent user disappointment when applications fail to live up to expectations on intelligence. This pragmatic point of view is also reflected in the designer's position towards how the application should respond when it recognizes features of specific contexts. The designer should see the context-aware application as a tool that has a function, and he should operate on this function in such a way that the tool's results fit to the outcomes expected by its user. As such, the context-aware application is a flexibilised translation device. This means that the designer should reason with the perspective of the user as a point of departure. However, the context-aware application also has an intentional attitude towards the world, which results in an application design that to a certain extent resembles the human mind. This means that the designer should attempt to take the perspective of the context-aware application.

Remembering that computer applications in general and context-aware applications in particular are, in Brey's terms[88], hybrid cognitive systems, we can state that cognition is distributed over the context-aware application, the application user, the application designer, and the end-user expert. The contexts that have been predefined by the designer are equally important as the feedback the application receives from the user.

As part of his efforts to design the context-aware application intelligently, the designer should take seriously the notion that the context-aware application acts as if it is intentional, and that its users to some extent consider the application to be intentional. The designer should crawl under the skin of a subject who has the capabilities of that context-aware application and mediates between the user and the world. He should pretend his perspective is the same as the perspective of the context-aware application.

This is the platform where the designer has to implement his mechanisms for making the judgment calls that are necessary when talking about Dreyfus' competence. And this mechanism can very well be similar to the logical structure for rational behavior outlined by Searle. It can simulate the human faculties and what has been learned about them by cognitive science. It can use any programming strategy that the designer wants it to, as long as the designer does not feel the urge to try to create a system that behaves intelligently. How this platform is a step towards reconciling the classical approach to context awareness with the phenomenological turn will be demonstrated by a brief

[88] Brey, P. (2005, November). The epistemology and ontology of human-computer interaction.

discussion of a strategy to context awareness that is common practice. The discussion will be held using the vocabulary that was presented in chapter 4.

### 5.3.1 Context condition correspondence

The classical context awareness designer Dockhorn explicated[89] a design strategy for context-aware applications that is implicit in almost all other context awareness design approaches and that rests on presuppositions that were until now complicated to justify. This section will set out to give that justification. This strategy's central working principle is that a context is constituted by context conditions that can be measured. The conditions are seen as a fingerprint: if the conditions are present, the context is uniquely identified. While this might be a suitable practical design rationale, we have seen that the phenomenological turn stimulated the practice of formulating a sound theory and subsequently implementing a system that does justice to this theory, which should still be done for this strategy. When fingerprinting is seen as a practice that from a practical point of view works well, the question arises if its assumptions are in line with the other assumptions of context awareness. According to the phenomenological turn, fingerprinting would be problematic since it is inherently unable to match all possible contexts that can be encountered: black swans will not be recognized. Only a truly holistic system would suffice. The new approach supported by the theories presented in chapter 4 however does not call for this restriction. Now, we are satisfied with the knowledge that the designer uses a design approach that maximizes information on all contexts the application possibly might encounter, as was required by the classical approach.

Now, let us take a closer look at Dockhorn's mechanism. In practice, according to Dockhorn, the system waits for the occurrence of a pattern to trigger an action, which is implemented using the so-called "event-control-action pattern". Subsequently, this pattern is used to trigger non-user initiated activity in the form of a modification of the actions performed by the application. This functionality has implications for the context condition correspondence: it is implied that the conditions are observed if and only if there is a certain context. If there is a different context the context condition pattern does not occur, and if the pattern is not measured, the context "is not present". The problem one could have with this strategy is that the application can misinterpret a context if it does not recognize its fingerprint. This strategy is in fact a good step towards the completeness that the classical approach wants, most notably because it a) threats fingerprints as "causes to act upon", b) leaves room to ask for additional input and c)

[89] Dockhorn Costa, P. and Ferreira Pires, L. and van Sinderen, M.J. "Architectural support for mobile context-aware applications.", 2006, p. 7

helps the designer with creating a sufficiently complete set of contexts that can be recognized.

The first issue to look at is what the application can do with the fingerprints it recognizes. The designer should by know that any input it receives can at best be explanations of its user's intentions, and the fingerprint might be an indication of an event that the user wants to act upon. For example, consider an application that sees a high illumination level, a temperature of minus 20 and an agenda that has no appointments as a fingerprint for a situation in which its user often goes out for a walk and returns for hot chocolate. When stated this way, a designer is invited to consider the weather conditions to be a motivation for the user for getting hot chocolate. Actually preparing the chocolate might be a step too proactive, because the user might choose otherwise. But preparing all the ingredients and equipment might make it easier for the user actually to get his hot chocolate. The designer succeeded in balancing between making choices for its user and leaving choices open to its user, and all that because it was able to threat the sensor input as motivations for its user.

As Searle demonstrated, the user's intentions are internal states and the only means to get any information at all about these states is through his activity. Even then, they cannot be truly known, but a correlation between displaying certain behavior and having certain intentions can be assumed. Other facts than those relating to behavioral activity that can be measured, such as the illumination level in the room, the temperature of the chicken, et cetera, are either objective facts or subjective facts and in both cases need someone to make an intelligent judgment call before they can be relied upon as input into the application's reasoning system. An objective fact needs this because according to Dreyfus objective facts are part of crude rule systems that are typically used by beginners and will only have qualitatively good results under very specific conditions. A subjective fact has already been interpreted, namely by the designer, when he designed all contexts the context-aware application could encounter. This subjective fact might simply be wrong because the application ended up in a context the designer could or did not foresee.

In short, the context-aware application's classification of the current situation as a particular context is not to be trusted unconditionally, but is to be treated as a suggestion to resolve a specific problem in a specific way. Given that a context-aware application that is designed this way is likely to default to harmless behavior, the fingerprinting strategy does not restrict the behavior of the application but is only a means for the designer to discuss the situations the application is likely to encounter in an effective manner. What is won by the designer is not only that he has this platform for making judgment calls, but that this platform allows him to gather facts from the

environment effectively and use them to simplify the way the user uses the application to achieve a goal. This implies that a reduction of complexity is achieved because part of the efforts aimed at finding out details on the current situation and incorporating them in his use of the application is moved away from being only his responsibility to being the shared responsibility of the context-aware application, its designer, and its user.

This allows us to address Svanæs' objections to using the term "context aware". With him, we ask in what sense context-aware applications can still be seen as aware of their context. They are not aware in the strict sense that they have conscious awareness and that they "know" that their context is of a specific nature. Neither are they able to determine a specific context from the infinite number of possible contexts. They cannot even absolutely select the proper context from a set of possible contexts, predetermined by a designer. We assume however that they are in a specific context for as far as its users and its designers are concerned. This context that the application assumes, which we could distrust, is a black-boxed state that the application uses to decide on how to serve us, and that serves as a means for its users to make the usage of the application easier.

The combination of the distribution of responsibility, the use of sensors and of actuators and the platform for making judgment calls make this a context-aware application. One of the early problems of context awareness was that it is hard to know when an application can recognize enough contexts: many researchers wanted to know how to create a complete list of contexts that their application should recognize. From this perspective, it could be doubted that Dockhorn's approach on recognizing contexts would not result in missing too many relevant contexts. The current approach demonstrated that these worries are correct and that this problem is intrinsic to context awareness. There is no way to avoid this problem, but context awareness can improve its strategy on dealing with the problem, namely by distributing the responsibility and thinking from the perspective of the application user in a new way.

## *5.4 On performing context-aware activities*

The previous section briefly touched upon the notion that context-aware applications mediate between the world and a person, which of course refers to Ihde's theories presented in chapter 4. Ihde presented his theory to explain the relation that people have to all technologies. We are of course interested in one specific type of technologies: context-aware applications. This section will discuss Ihde's theory in closer relation to context-aware applications than has been done in chapter 4, while it leaves enough room for interpretation to transfer these theories to more detailed architectures of applications. Ihde gave us a way to talk about human – technology – world relationships, which both

explains that the application is located somewhere on an axis between close to the user and close to the world, and that an application translates human capacity for acting. On both accounts, there is more to say about context-aware applications. Firstly, they span the range from embodied applications to alterity relations, but they are strongly "enigmatic". Secondly, their structure is explicitly aimed at transforming the user's capacities. This section will restate these theories in more context-aware friendly terms, preparing them for incorporation in a design process.

### 5.4.1 Between the user and the world

In paragraph 4.1.2 it was discussed that the relations technology can assume with the world and the person using the technology can be characterized as cyborg, embodiment, composite, hermeneutic, background, alterity, and unidirectional relations. Most context-aware applications are designed as if they fulfill an embodiment or a hermeneutic relation. But since we have seen that the designer strongly determines how the application behaves, it is fair to argue that the context-aware application often is more likely to fulfill a composite relation, where it has its own intentionality towards the world. It is my argument that both views should be considered, since they both have their merits. The first should be considered because it acknowledges explicitly the enigmas that occur when the applications are used, especially when they fail to function properly or as expected. The second should be considered because it confirms the view of an application that is designed as a quasi-other (hermeneutic) or a quasi-me (embodiment).

Looking at the three prototypical applications mentioned in paragraph 2.1.1, mobile wearable devices, artificially intelligent agents, and ambient intelligence systems, we see that the relation classification offers an interesting mapping. The mobile wearable device is a great example of an embodiment relation. More and more, people start seeing their mobile phone as part of their identity, when they get used to its interface they can work quite quickly with it, almost without thinking, and most importantly: it makes them always within reach for communications. The artificially intelligent agent on the other hand can act as a representative or an intermediary of its user, or as a representative or intermediary of "the world", putting him respectively in an alterity or hermeneutic relation. An example of the first would be our microwave oven when we order it to prepare chicken, while it would be an example of the second type of relation if we instruct an alarm to warn us when the chicken is ready. An ambient intelligence system is likely to incorporate various agents and devices in various types of relationships, making it the choice of the designers which types of relationships to prefer. Consider a smart house with various kitchen appliances, which quite possible even interact with each other without a short-term need for interaction with a real person. In this case, the

technology could be in a unidirectional relation, where the user (barely) notices the technology at work, but is not directly involved in its results or activities.

### 5.4.2 Cyborg and composite relationships

This characterization of typical context-aware applications does not mention the cyborg and composite relationships. Excluding the cyborg relation is a deliberate choice, since for now it seems prudent to postpone such efforts until there is more clarity about to what extent we should want to accept the intrusion of these particular systems in a way that "physically alters the human". As Verbeek argues, the technology can be so closely intertwined with a person that it is no longer possible to differentiate between them. Think of a pill against depression or a pace maker to support hearth activity. Such technologies are well tested and often well regulated. A context-aware application however is not a normal, simple piece of technology, but one that can be unpredictable for two reasons. First, it appears to have autonomy, it is a quasi-other and it requires much design effort to have the application behave in a way that is meaningful to the user. Second, designing for context awareness is a complicated process because application designers need knowledge that is usually only possessed by end-user experts or that is specialist human context processing knowledge. This makes designing context-aware applications so complicated that it is hard to create applications that do not at some point break. As a result, designers of context-aware applications should explicitly take the option of broken applications into account. Looking more closely at the enigmas will result in a rejection of the cyborg relationship as a candidate for context-aware applications and acceptance of the composite relationship as a way to look at context-aware applications.

The possibilities of context-aware applications performing unwanted or unexpected behavior takes us to a central element in Ihde's vocabulary: the enigmas. The point where the technology can become unintelligible to a person when it fails its function is what is called the enigma. An enigma can occur when the user cannot handle the technology as he is supposed to, or when the technology does not impact or represent the world as it is supposed to. But in Verbeek's cyborg relationship, the person and technology are represented as a unity: (subject/technology) → world. If the technology fails, the person has no opportunity to correct this, since he himself fails at this point.

Verbeek's other new relationship on the other hand, the composite relationship, fits well to the structure of context-aware applications. A composite relationship, subject → (technology → world), has a user with an intentional attitude towards the application, and the application that has an intentional attitude towards the world. More specifically, the user is concerned with how the application is related to the world. A cook is

concerned with how long it will take an oven to finish grilling; a driver is concerned with the route a Tom Tom plans to take him towards its destination. While a person can be in an embodiment relation with a mobile wearable device, this can be similar to being in a composite relation, since the mobile wearable device selects which events are relevant to the user and which not. Similarly, a person in a hermeneutic relation to a context-aware application, such as someone listening to a radio station that selects songs that he is likely to enjoy, is likely to experience the application as having intentionality too. This impacts on how the designer should design the interaction moments: both the user and the application will have three gaps where interaction about each other's motivations is warranted, and each action that is performed can break at two points, namely the user can instruct or understand the application wrongly and the application can interpret or instruct other entities wrongly.

The enigmas, the closeness of context-aware applications either to a person or to the world and, the intentionality that an application has are all very recognizable to who is familiar with Ihde's theories. We have seen how Ihde's structure explained that a designer is justified in presenting choices to its users at the points that Searle identified: the gaps. The enigmas in Ihde's structure present opportunities too for designers who want to improve their design: depending on the type of relationship, the impact of failing technology is different. And as a final consideration on this subject, the actions that an application perform should depend on how a designer thinks about this relationship. This will be the subject of discussion of the next section, where the notion of performing judgment calls will be placed at a central position.

## *5.5 On discussing design in the new jargon*

Chapter 2 demonstrated how difficult it is to define what context is, and consequently what a context-aware application does. Chapter 3 and chapter 4 introduced theories that came with their own vocabularies. This section will use this new vocabulary to discuss what context-aware applications do. The vocabulary is very specific for the three introduced authors and not very common outside philosophy or even phenomenology. It is thus jargon and as such not suited to use to discuss the design of context-aware applications with non-insiders. It is, however, an accurate way for a designer to express himself.

When using this jargon, it can be argued that by avoiding the word "context" and instead focusing on how the applications use judgment calls on sensed information to support its user, it becomes possible to see that the benefit of context awareness is its effectiveness in working towards a user's goals. Both classical context awareness and the phenomenological agreed that some sort of judgment mechanism needed to be

implemented in context-aware applications. The composite relation offers a platform for such a mechanism. This mechanism can be described as follows. The application uses its sensors and actuators to translate the user's ability to sense and act in the world in a complex manner. This complex translation is done in a system created by the designer that shares a feature with the human mind: it has to make judgment calls. This system should be considered to be a quasi-other or a quasi-me and the designer should design it to function as a competent performer, using his sensors to determine the characteristics of the situation, which are used to decide how to perform the commands from the user. The user need no longer worry about the "how" of his commands, and the designer still does not have to do the impossible: induce the "what" of a user's intentions.

### 5.5.1 On translating

Section 4.1.1 introduced the concept of translation of sensomotoric activities. A context-aware application translates what we see and do too. It is a specific kind of translation device: it "decides" how much it needs to translate our sensomotoric activity in order to end up with an appropriate result. This section will explore that concept further.

To start this exploration, consider driving a nail in a piece of wood. The force the hammer needs to exert depends on the wood: a nail needs more force to dig into hard wood than into soft wood, but old and dry wood is more fragile and requires more careful hammering. It also depends on how far the nail must be dug into the wood. A normal hammer is swung by a person who decides on the force of hammering and the hammer translates the force of his arm using relatively simple physical laws: the force is multiplied because there is a longer arm and because the force is concentrated at the small iron end of the hammer. A context-aware hammer would need a smart mechanism to decide on how much actual force needs to be exerted in order to drive the nail far enough that specific piece of wood, resulting in a flexibilisation of the translation function of the hammer.

While the concept of translating can apparently be used to express parts of the behavior of context-aware applications, applying the concept is not an easy task. The challenge is to design the system as having a proper platform for deciding on what the application should translate and how to make use of its flexibility. We can use the concepts of intentionality of context-aware applications, which was discussed when it was argued that context-aware applications often are in a composite relationship. However, a complication might surface from the application of this concept. Does the context-aware application create a new type of facts, facts that are outside our scope of knowing similar to that we cannot know what it is like to be a bat? This would imply that the task for the designer (thinking from the perspective of the application) would become almost impossible.

The short answer to the first question is “no”. What makes Nagel’s bat problem interesting is that it makes us aware of the shared frame of reference that people have. While I cannot really know what someone else is thinking, I can reasonably guess what his world might look like, since I have a similar physiology. The significant difference between me and a bat is that the bat uses sonar where we use vision: instead of light, he uses sound as a primary means to construct a means of spatial awareness. I cannot know or imagine what that looks or feels like, let alone what other mental states this bat might have. If a context-aware application was a real other instead of a quasi-other, I would also not be able to know or imagine what its world would look like. However, a context-aware application’s world does not look like anything. It magnifies, reduces and transposes in space and time, constructing facts that a human designer makes him construct. Even if I would not instantly be able to grasp these constructs, there would be some way to visualize them to help me grasp them. This is because, as we have learned to be axiomatic in chapter 4, context-aware applications operate on concepts familiar to humans. And these constructs are intermediate steps in the transformation process initiated by the user.

What makes computer applications special, and what separates them from many other artifacts, is the extent at which the form of the human input is decoupled of the form of the applications achievements, and it is this metamorphosis for which the designer needs the support in the form of a complex design strategy. So the complexity does not stem from an immediate unintelligibility, but from the fact that the translation includes a designer’s intentionality and because the user’s contribution to the action does not immediately correspond in form to the context-aware application’s action.

### 5.5.2 Complex translations

The concept of translating also offers an advantage for explaining what is useful about context-aware applications. Translating sensomotoric action can be seen as one of the steps an application has to take in order to achieve its user’s goals. But it is more than that: it is a smarter step than the translations we know from normal technologies, but not so complicated that we can no longer understand what the step exactly does. The translations performed by context-aware applications are appropriate steps towards a goal.

The transformation Ihde refers to is often a simple, linear transformation. A person using a hammer to drive a nail in wood simply magnifies his force. A thermometer translates temperature into a number. A camera transposes images through time or space, making it available at a different location. All three of them have a rather straightforward relation between the sensed and the outcome. Moreover, the action

performed by the user is similar to the action performed by the artifact. A person exerts force on a nail by exerting force on a hammer, the air “gives” the temperature to a person by giving hot air to a thermometer, a person sees a transposed image because the camera observes an image.

We have accepted that technology does what it does, and people using technology often quickly grow adept at using technology to achieve their goals. However, often there is a gap between what a technology does and what we want to achieve. For example, when we switch on a lamp, we do not really want the lamp to give us light, we typically want to be able to see clearly in a room, or we want to be visible, or we want a decorative shimmer. And we will choose our the lamp accordingly, for example, by buying a lamp with diffuse lighting that we hang on the ceiling, or a converging light that we shine in the direction were we want people to notice us, or we buy colored lights. All three lamps would just simply produce light, but if we would have a device that would be able to fulfill all three of these functions if we need it to, we would be able to contextualize the light production. And if it would choose the right modus after we gave the command to “give light”, it would be a context-aware lamp.

Looking at the interaction between the user and the context-aware application, we first see that the user typically gives the application a command, which is related to the goal the application is to achieve. Then, the application uses its sensors to gather information on how the goal is to be achieved. Successively, the application can use its actuators not only to achieve the goal, but also to give feedback or elicit further interaction with the user. This mechanism has as a result that the gap is bridged between a technology’s immediate use and the user’s expected outcome.

This illustrates the beauty of context awareness. It is a design strategy that supports the designer to optimize the application for usage in an as wide as possible range of scenarios. The designer can take some functionality that was already available in other applications and have the application adapt it to the current situation in order to achieve the user’s goals. Now we have reformulated the added value in terminology that fits both the new phenomenological approach and the classical approach we are ready to continue to the final reflection on context awareness.

## *5.6 Final conclusions*

The main conclusion of this thesis is that context awareness can benefit from phenomenology by improving the strategy of designers for designing: they should not only consider the context-aware applications and its users, but also the designers themselves and context-aware behavior is distributed over these three groups. Context

awareness has pragmatism as one of its key norms and phenomenological insights should be subordinate to these norms. Phenomenology demonstrates the advantages of analyzing unique situations over using generic procedures and it shows that if a system is to be truly context aware, it should understand all contexts and it should have human intelligence. Classical context awareness however dictates not to pursue such impractical goals. If context awareness wants to use phenomenological insights, they should be adapted to the everyday praxis of designing context-aware applications. This implies that a phenomenological approach is needed that conforms itself to the classical approach.

The first conclusion of this new phenomenological approach is that designers should use the knowledge from end-user experts for the context modeling, and that they should not rely solely on theories about human context processing, such as theories from philosophers, cognitive scientists, and psychologists. The second conclusion is that theories human context processing can be relevant when analyzing the relation between applications, their users, and the world: by analyzing how people reason and act in this relation, the designer will learn how the application should reason and act in this relation.

Context-aware applications can be close to the users, close to the world or in between the two, it can be a quasi-me (such as a mobile wearable device) or a quasi-other (such as an artificially intelligent agent) and it can be either transparent or the focal end point. As a quasi-subject, in translates the user's intentionality in a complex manner and it makes judgment calls. These actions it performs should be congruent with what they present or represent, which is a design challenge. Context awareness does not create intelligent applications, but applications that assist the user in making choices to achieve their goals, using all information it can gather. This information is used to determine what can and should be motivating its users. The designers can use the suggestions that are pointed out here to identify themselves with the context-aware application, but still it should be remembered that the application's abilities to reason are limited and that the application and the user as a combined actor can be imperfect: any system can break.

### 5.6.1 Afterthoughts

This thesis discussed what a designer of context-aware application can and should do. That discussion reasoned from a technical point of view: what would be an efficient approach for designers to create context-aware applications. However, the approach that was outlined might also offer some first steps for ethicists to get more on the development of context-aware applications. While ethics is outside the scope of this thesis, in this final section I would like to suggest a direction for such a discussion. These suggestions rest upon my observations that context-aware applications can fail at predetermined locations, times or activities, and that according to Dreyfus a learner

needs to be able to take responsibility for its failures to be able to learn. The responsibility question is not only important to ethicists, but also to proponents of artificial intelligence. The question is if the introduction of Ihde's concepts to context awareness offers new possibilities for both parties to come closer together.

This question has four elements. First is the need for ethicists to be able to say who, or possibly what, is accountable for any action. If a context-aware application performs actions that have negative results for a third person, who should be held accountable for this action? The second is a need for artificial intelligence to create a mechanism that is equivalent to taking responsibility – creating a plan – by human actors, as can be deduced from Dreyfus' five steps to acquiring expertise. According to Dreyfus, responsibility is, next to embodiment, key to progressing from stage two (advanced beginner) to stage three (competent performer). The third is the distribution of responsibility for taking decisions in context awareness. A designer and the context processing theorists decide which procedures to incorporate in the application, the end-user experts decide how to localize the middleware, the application user decides on the issues that are presented to him by the application and the formalization of the entire decision making process is delegated to the application. The fourth is the perspective of observers of context-aware application behavior. As argued by Ihde, it is often possible to perceive an application as a quasi-other. This is especially likely with context-aware applications, which perform sensomotoric activity and activity that is not necessarily explicitly initiated by users, and thus are to some extent embodied and can be argued to display autonomous behavior.

They are related as follows. Ethicists are facing a challenge because it is not immediately clear how the accountability is distributed, especially because it appears that some parties in some circumstances consider a non-human actor to be responsible. AI designers are motivated to find a mechanism that institutionalizes responsibility as part of an application. Context awareness offers at least two conditions that are favorable for such an approach: partially embodied applications and a framework for distributing responsibility. The users of context-aware applications might also accept a solution that institutionalizes a responsible and accountable context-aware application.

The problem of course is that we do not know what it means to hold an application accountable. Should it be imprisoned? Can it get a fine? Do we need to destruct it? Or reprogram it? Maybe it should reprogram itself. An AI designer might opt for such an option, opening up the option to a legalist to penalize applications that cannot reprogram itself, or to penalize its designers. Whatever the solution, context awareness can create favorable conditions for such an approach by continuing to engineer better applications. The ethicists can then be seen as a scientists who has authority on one of the design

aspects of context-aware applications. The argument in this section was meant to demonstrate that new options are opened, or at least are put to the spotlight, by the approach that is proposed in this thesis, to application designers, consumer, and to different types of scientists.

# 6 Bibliography

Aspray, W. and M. Campbell-Kelly (1996). *Computer: A History of the Information Machine*. New York: Basic Books.

Bazire, M. and P. Brézillon (2005). Understanding context before using it. In 5th International and Interdisciplinary Conference CONTEXT 2005, Paris, France, July 5-8, 2005. Proceedings, Volume 3554/2005 of Lecture Notes in Computer Science, Heidelberg, pp. 29-40. Springer Berlin.

Benerecetti, M., P. Bouquet, and M. Bonifacio (2001). Distributed context-aware systems. *Hum.-Comput. Interact. 16* (2), 213-228.

Brey, P. (2005). Freedom and privacy in ambient intelligence. *Ethics and Information Technology 7* (3), 157-166.

Brey, P. (2005, November). The epistemology and ontology of human-computer interaction. *Minds and Machines 15* (3-4), 383-398.

Brey, P. (2002, Unpublished). Kritieken op de symbolische ai. In *Lecture Notes on Artificial Intelligence*.

Brey, P. (2002, Unpublished). Symbol systems versus neural networks. In *Lecture Notes on Artificial Intelligence*.

Brey, P. (2001). Hubert dreyfus: Humans versus computers. In H. Achterhuis and R. P. Crease (Eds.), *American philosophy of technology*. Bloomington: Indiana University Press.

Brey, P. and J. H. Søraker (2009). Philosophy of computing and information technology. In A. W. M. Meijers, D. M. Gabbay, P. Thagard, and J. Woods (Eds.), *Philosophy of Technology and Engineering Sciences*, pp. 1341-1409.

Cole, D. (2004). The chinese room argument. In Stanford Encyclopedia of Philosophy.

Collins, H. (2004, June). The trouble with Madeleine. Phenomenology and the Cognitive Sciences 3 (2), 165-170.

Dey, A. K. (2001, February). Understanding and using context. *Personal Ubiquitous Comput. 5* (1), 4-7.

Dey, A. K. and G. D. Abowd (1999). Towards a better understanding of context and context-awareness. Technical report, Georgia Institute of Technology.

Dey, A. K., G. D. Abowd, and D. Salber (2001). A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Hum.-Comput. Interact. 16* (2), 97-166.

Dockhorn Costa, P., Ferreira Pires, L., and van Sinderen, M.L. (2006). Architectural support for mobile context-aware applications. In *Handbook of research on mobile multimedia*, pp. 456-475. Hershey: Idea Group Publishing.

Dourish, P. (2001). Seeking a foundation for context-aware computing. *Hum.-Comput. Interact. 16* (2), 229-241.

Dreyfus, H. L. (2002). Intelligence without representation – merleau-ponty's critique of mental representation the relevance of phenomenology to scientific explanation. Phenomenology and the Cognitive Sciences 1 (4), 367-383.

Dreyfus, H. L. (1992). What Computers Still Can't Do: A Critique of Artificial Reason. Cambridge, MA: The MIT Press.

Dreyfus, H. L. and S. E. Dreyfus (1986). Mind over Machine: the Power of Human Intuition and Expertise in the Era of the Computer. New York: Free Press.

Guha, R. and J. Mccarthy (2003). Varieties of contexts. In 4th International and Interdisciplinary Conference CONTEXT 2003 Stanford, CA, USA, June 23-25, 2003 Proceedings, Volume 2680/2003 of Lecture Notes in Computer Science, Heidelberg, pp. 164-177. Springer Berlin.

Haselager, W. F. G. and J. F. H. Van Rappard (1998). Connectionism, systematicity, and the frame problem. *Minds and Machines 8* (2), 161-179.

Haseloff, S. (2005). *Context Awareness in Information Logistics*. Ph. D. thesis, TU Berlin, Germany.

Heersmink, R. (2009, unpublished). Ghost in the machine: Brain-computer interfaces in postphenomenological terms.

Ihde, D. (1986). *Consequences of phenomenology*. Albany, NY: State University of New York Press.

Ihde, D. (1990). *Technology and the Lifeworld: from Garden to Earth*. Bloomington: Indiana University Press.

Johnsson, M. (2007). Sensing and making sense - designing middleware for context aware computing. PH. D thesis, The Royal Institute of Technology, Sweden.

Korpipää, P. and J. Mäntyjärvi (2003). An ontology for mobile device sensor-based context awareness. In 4th International and Interdisciplinary Conference CONTEXT 2003 Stanford, CA, USA, June 23-25, 2003 Proceedings, Volume 2680/2003 of Lecture Notes in Computer Science, pp. 451-458. Heidelberg: Springer Berlin.

Lenat, D. (1998). The dimensions of context-space.

Mäntyjärvi, J. (2003). *Sensor-based context recognition for mobile applications*. Ph. D. thesis, University of Oulu, Finland.

Mccarthy, J. (1986). Notes on formalizing contexts. In T. Kehler and S. Rosenschein (Eds.), *Proceedings of the Fifth National Conference on Artificial Intelligence*, Los Altos, California, pp. 555-560. Morgan Kaufmann.

Messerschmitt, D. G. (2000). *Understanding Networked Applications: a First Course*. The Morgan Kaufmann Series in Networking. San Diego: Academic Press.

Müller, M. E. (2007, June). Being aware: where we think the action is. *Cognition, Technology & Work 9* (2), 109-126.

Rienks, R., A. Nijholt, and D. Reidsma (2006). Meetings and meeting support in ambient intelligence. In A. Vasilakos and W. Pedrycz (Eds.), *Ambient Intelligence Wireless Networking Ubiquitous Computing*. Norwood, MA, USA: Artech House.

Schilit, B. N. and M. M. Theimer (1994). Disseminating active map information to mobile hosts. *Network, IEEE 8* (5), 22-32.

Searle, J. R. (1999, January). Limits of phenomenology.

Searle, J. R. (1999, January). Neither phenomenological description nor rational reconstruction.

Searle, J. R. (2005). The phenomenological illusion. In M. Reicher and J. Marek (Eds.), Experience and Analysis, Proceedings of the 27th International Wittgenstein-Symposium.

Searle, J. R. (2001, March). *Rationality in Action*. The Jean Nicod Lectures 2000. Cambridge, MA: The MIT Press.

Shanahan, M. (2004). The frame problem. In *Stanford Encyclopedia of Philosophy*.

Smith, D. W. (2008). Phenomenology. In *Stanford Encyclopedia of Philosophy*.

Svanæs, D. (2001). Context-aware technology: a phenomenological perspective. *Hum.-Comput. Interact. 16* (2), 379-400.

Tomasi, W. (2004). *Electronic Communications Systems: Fundamentals Through Advanced* (Fifth ed.). New Jersey: Pearson Prentice Hall.

Van de Poel, I. (1998). Changing Technologies. A Comparative Study of Eight Processes of Transformation of Technological Regimes. Ph. D. thesis, University of Twente, Enschede.

Van Gelder, T. (1993). Connectionism and the mind-body problem: exposing the distinction between mind and cognition. *Artificial Intelligence Review* 7, 355-369.

Van Gelder, T. (1998). Cognitive architecture: What choice do we have? In Z. W. Pylyshyn (Ed.), *Constraining Cognitive Theories: Issues and Options*. Stamford CT: Ablex Publishing.

Verbeek, P. P. (2008). Cyborg intentionality: Rethinking the phenomenology of human–technology relations. *Phenomenology and the Cognitive Sciences 7* (3), 387-395.

Vlasveld, J. (2007). On context in context-aware applications. Master's thesis, University of Twente.

Zimmermann, A., A. Lorenz, and R. Oppermann (2007). An operational definition of context. In 6th International and Interdisciplinary Conference, CONTEXT 2007, Roskilde, Denmark, August 20-24, 2007. Proceedings, Volume 4635/2007 of Lecture Notes in Computer Science, Heidelberg, 558-571. Springer Berlin.