7-19-2012

The Utilization of Context to Adapt the Interface of a Computer

Ludovic Delaveau
Purdue University, ldelavea@purdue.edu

Graduate School ETD Form 9
(Revised 12/07)

PURDUE UNIVERSITY
GRADUATE SCHOOL
Thesis/Dissertation Acceptance

This is to certify that the thesis/dissertation prepared

By Ludovic Delaveau

Entitled
THE UTILIZATION OF CONTEXT TO ADAPT THE INTERFACE OF A COMPUTER

For the degree of Master of Science

Is approved by the final examining committee:

Eric Matson Chair	
Brandeis Marshall	
J. Eric Dietz	

To the best of my knowledge and as understood by the student in the *Research Integrity and Copyright Disclaimer (Graduate School Form 20)*, this thesis/dissertation adheres to the provisions of Purdue University's "Policy on Integrity in Research" and the use of copyrighted material.

Approved by Major Professor(s): Eric Matson

Approved by: Jeffrey L. Whiten 7/19/2012
Head of the Graduate Program Date

Graduate School Form 20
(Revised 9/10)

PURDUE UNIVERSITY
GRADUATE SCHOOL

Research Integrity and Copyright Disclaimer

Title of Thesis/Dissertation:
THE UTILIZATION OF CONTEXT TO ADAPT THE INTERFACE OF A COMPUTER

For the degree of Master of Science

I certify that in the preparation of this thesis, I have observed the provisions of *Purdue University Executive Memorandum No. C-22,* September 6, 1991, *Policy on Integrity in Research.**

Further, I certify that this work is free of plagiarism and all materials appearing in this thesis/dissertation have been properly quoted and attributed.

I certify that all copyrighted material incorporated into this thesis/dissertation is in compliance with the United States' copyright law and that I have received written permission from the copyright owners for my use of their work, which is beyond the scope of the law. I agree to indemnify and save harmless Purdue University from any and all claims that may be asserted or that may arise from any copyright violation.

Ludovic Delaveau
Printed Name and Signature of Candidate

7/16/12
Date (month/day/year)

*Located at http://www.purdue.edu/policies/pages/teach_res_outreach/c_22.html

THE UTILIZATION OF CONTEXT TO ADAPT THE INTERFACE OF A COMPUTER

A Thesis

Submitted to the Faculty

of

Purdue University

by

Ludovic Delaveau

In Partial Fulfillment of the

Requirements for the Degree

of

Master of Science

August 2012

Purdue University

West Lafayette, Indiana

ACKNOWLEDGMENTS

First of all I would like to thank Prof. Eric Matson, my adviser, for accompanying me during my Master's degree began in April 2011, and for his precious advices and encouragements.

I would like to thank the co-authors of the paper, which serves as basis for the context analysis: Benjamin Loulier, Prof. Eric Matson, and Prof. Eric Dietz.

I am grateful to my committee members, Prof. Eric Matson, Prof. Brandeis Marshall, and Prof. Eric Dietz, for their time spent in reading and correcting this thesis.

I could not have achieved this degree without the help of Prof. Olivier Boissier and Prof. Eric Matson, who offered me the opportunity to study in Purdue University, and Prof. Eric Dietz, who supported it through a grant of the Department of Energy.

Finally, I would like to thank Benjamin Loulier, who transmitted me his passion for Computer Science, Christabelle Bosson, who stands at my side helping me gain confidence and succeed in what I start, and my parents, my brother and sisters, who form this loving family which I belong to.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

AFP	Apple Filing Protocol
DNS	Domain Name System
DTD	Document Type Definition
GPS	Global Positioning System
IP	Internet Protocol
SSID	Service Set IDentifier
XML	eXtended Markup Language

ABSTRACT

Delaveau, Ludovic M.S., Purdue University, August 2012. The utilization of context to adapt the interface of a computer. Major Professor: Eric Matson.

This research examines the possibility for a computer to adapt its interface and its settings to the current environment, through the notion of situations and corresponding profiles. These situations are group of context elements and tasks or activities performed in that operating conditions, such as the activities conducted at the workplace or the relaxation time at home. The profiles are set of configuration values for the computer, such as the network services. Adapting the computer serves the user as it takes care of putting his computer in the right mode to be efficient in the activity performed. In order to demonstrate the feasibility of this adaption, a prototype is implemented, using a context analysis and the activation of corresponding settings.

CHAPTER 1. INTRODUCTION

This chapter introduces the problem with an overview of why it is significant, and identifies the scope and the goals of the study. A few definitions of generic concepts that are used widely in this report are also provided, before providing a generic frame of the study.

1.1 Statement of the problem

Computers, and particularly laptop tend to be confronted to different scenarios when used. Especially because of the intrinsic mobile nature of a laptop meant to be carried in different places, these situations can be very different, ranging from a presentation, a class, a laboratory, a public WiFi hotspot, or home. Depending on each situation a different configuration of the computer is needed. For example what is shared at home and at work aren't usually related, whereas sharing should be deactivated in a public hotspot. Or when traveling, one wants his computer to lock itself after a short period of inactivity, whereas at home the time could be different, because the risk of a stranger using the computer is lower.

A first identification of the issue acknowledges that for different settings the user wants to activate for a specific situation. Regrouping these settings in a profile, adapted to the activity of the user, will simplify the effort he needs to provide to find and modify each of the different settings.

But having these profiles is only part of the answer to the problem, because the user will still have to analyze his environment and select the appropriate profile from his list. Performing this task needs to become an habit for the user, *i.e.* he needs to adapt himself to its computer, rather than having the computer adapt itself to the user. The risk is therefore high to forget that a profile exists for the current situation.

Therefore automating the steps needed to perform the adaptation is an improvement for the user. Technologies from context-aware computing can be used to acquire information about the environment and process it, before taking the decision to activate the best profile corresponding to the current situation.

1.2 Research question

The question of the study is:

How can the physical and digital environment be processed for an adaptive interface?

1.3 Scope

The study first focuses on defining the different concepts of context, situation, and how the user and its computer are included in these terms.

The study also describes the implementation of a prototype gathering information on the context that activate predefined behaviors on the computer, as Schmidt, Takaluoma, and Mäntyjärvi (2000) did on mobile phones. However in the proposed study the elements of the context perceived by the software triggers these different profiles. Finally an evaluation of the software is conducted.

1.4 Significance of the problem

Schmidt et al. (2000) worked on the implementation of behaviors configuring how telephone notifications are presented to the user, and how the activation of behaviors can help choose the appropriate way to communicate depending on the current activity of the correspondent. But the result was limited by the intervention of the user to activate the behaviors appropriately. The need for context-awareness becomes more important, in order to automate this decision process.

Studies on context-awareness in computing began work done at Olivetti in 1992 (Want, Hopper, Falcão, & Gibbons, 1992), with the implementation of a system of badges to provide location-aware applications. Two years later, another project was reported by Schilit, Adams, and Want (1994) developed in the Palo Alto research Center in Xerox Corporation. The device called PARCTAB, a wireless palm-sized computer, was used to demonstrate four applications of context-awareness: "proximate selection, automatic contextual reconfiguration, contextual information and commands, and content-triggered actions" (pp. 86-88). Fifteen years after, context-awareness is a recurrent theme in research, aiming to improve the interaction between the user and its machine.

Indeed, computers in a generic sense lack what makes the interactions between humans easy: the ability to perceive the context. Therefore, studies aim to reduce the attention needed for the user to interact with a machine, because of the raising number of devices inserted in our environment (Kim et al., 2004). This issue becomes more and more important with the multiplication of devices in our context, through the research on ubiquitous computing.

Most of the research conducted in the domain was applied to mobile devices such as smartphones, because of their good ability to retrieve information from the environment with numerous embedded sensors. The availability of software development kits (SDKs) for the different platforms also helped researchers to build prototypes demonstrating the applications of context-awareness.

As the research in the domain is focused on mobile devices, the researcher wants to explore the possibility of improving a simple laptop and see how its limited capacities compared to the wide context-awareness of smartphones could help users in their everyday activities.

1.5 Purpose of the study

The purpose of this project is to test the possibility of using the information of an environment to adapt the computer interface. It describes the creation of a prototype of a software implementing context awareness in order to adapt the settings building the digital environment. As part of the development of the prototype, two tasks need to be conducted. One is the identification of the different settings accessible on a computer, which together, compose the interface of the computer. The second is to determinate which information is available for use by the computer, and to establish ways to process it, to detect the situation of the user.

The users of such a prototype will benefit from the possibility of adapting the interface to their current context. It can be especially useful for users of mobile work environment to help separate the different contexts between the places where they use their computer, such as work, home, transportation, etc.

In this study an important part is to consider which relations a user has with his machine, and how to improve the interaction usually involving disappointment. The machine indeed lacks what makes the interaction between humans work, the ability to perceive how the other feels about what is exchanged, relying on non-verbal communication. This process is very interesting, especially the design of an interaction where the user takes advantage of the abilities of the machine without loosing control of his digital environment.

The combination of context awareness computing, machine learning, different Mac OS X technologies and the creation of software centered on users is also a motivation for working on this project.

1.6 Definitions

Context. The context, as seen from Dey and Abowd (2000) 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 of between a user and an application, including the user and the application themselves” (p. 4). Ranganathan and Campbell (2003) describe the context in a similar way, adding the concept of “interaction between the user and the ubiquitous computing environment” (p. 353).

Context-awareness. Context-awareness in computing applications is “adapting to context” and “using context” (Kim et al., 2004). Gellersen, Schmidt, and Beigl (2002) listed various applications of context-awareness such as “mobile computing, wearable computing, augmented reality, ubiquitous computing, and human-computer interaction” (p. 341).

Environment. The environment is a concept frequently used in Computer Science, and can be defined from a multi-agent system point of view as everything but the agent itself (Roche, Blunier, Miraoui, Hilaire, & Koukam, 2010), including all the information that will be useful to the agent.

Situation. The situation is an intersection between a context and activities performed inside this context. For example, surfing the web at home is a situation.

Agent. An agent is the elementary unit in the multi-agent paradigm. It interacts with its environment through messages, and is capable of reasoning from predefined rules, using usually first-order logic, to create new knowledge and act. McArthur et al. (2007) did a review of the various definitions of an agent, and two common characteristics are its autonomy and its ability to perceive its environment, react to it, and modify it.

Ubiquitous computing. Ubiquitous computing was first described by Weiser (1991), when he was working in the Xerox Palo Alto Research Center, as a paradigm consisting of the seamless integration of computers into everyday's life. It "does not just mean computers that can be carried to the beach, jungle or airport" (p. 94).

Wearable computer. The concept of wearable computer refers to a device carried by the user, providing him with computing resources everywhere. Pascoe (1998) defined it in his article as a device providing "a set of core services constantly at the user's disposal, whilst also contextually binding to devices in the user's environment [. . .] that can complement or augment the wearable system". (p. 93)

1.7 Assumptions

The assumptions for this study are:

- The object of the study is to design a software, in a rather controlled experimental environment explaining the lack of risks on the study.

1.8 Limitations

The limitations for this study are:

- The application is targeted for Mac OS X 10.7.
- No additional sensors are used, in order to be actually usable on any Apple computer without additional requirements.

1.9 Delimitations

The delimitations for this study are:

- No reasoner is implemented due to the constraints in time.

1.10 Summary

This chapter introduced the subject of the thesis, demonstrating its significance. It also defined generic concepts, which are used in the rest of the report. The next chapter assesses these definitions through a review of the literature about context-awareness in a chronological order, before classifying the different tools used to work with this context data.

CHAPTER 2. LITERATURE REVIEW

The study conducted in this report tries to answer to the following question, *How can the physical and digital environment be processed for an adaptive interface?*

The emergence two decades ago of ubiquitous computing, which corresponds to the pervasive deployment of computers in our environment, raised several issues, like the need for a better interaction between the user and the machine, as more devices requiring his attention are deployed.

Through a chronological overview of the research conducted since the work of Want et al. (1992), a characterization of context-awareness and the challenges it brought will be reviewed. Then, in another section, different methods to use context and create data that applications can use will be studied.

2.1 Chronological overview

In 1992, Want et al. (1992) described a system seen as the first project involving ubiquitous computing and context-awareness, in the attempt to create a location service inside the laboratories of Olivetti Ltd. The project consisted in the interaction between active badges worn by the employees of the company, and an infrastructure allowing the calculation of the position of an employee inside the building. The badge used infrared to communicate with a cellular network, which clustered the buildings and identified each member of the corporation. One of the applications was to be able to route effectively from the reception the calls to the nearest line of the recipient.

The concept of context-awareness was not described, but Want et al. (1992) stated that "a signal produced by an Active Badge can have different effects at different locations" (p. 102), which described the effect of location on the system. The system was a benefit for the employees, allowing more efficiency for the work of the receptionist, and making informal meetings between employees easier. Perspectives for the system included the integration of activity dependent signals to refuse a call during a meeting for example, or the prediction of a destination through the history of location to route the calls in the final destination.

In 1994 with the publication of *Context-Aware Computing Application*, the concept of context-awareness was exposed by Schilit et al. (1994) to describe the research conducted by a group of researchers in Palo Alto Research Center. A first definition of context was given, describing the context as the answer to the questions, "where you are, who you are with, and what resources are nearby" (p. 85). The objective was to use the context to adapt the applications to the different locations and devices used, in order to improve the user's experience, defining therefore the concept of context-awareness.

Three categories of objects/services were then identified for use with the ParcTab, a prototype of palm-sized computer, featuring a touch screen and a mobile network access through infrared:

- physical resources, such as printers;
- abstract services, accessed usually in special locations (a library catalog, a menu in a restaurant);
- places, such as bars, restaurants.

The authors created a "yellow pages" service using the location to propose contextual results, with the position determined indoor through a cellular network and outside via Global Positioning System (GPS). Other applications of context adaptation were also described such as the activation of configuration specific to a situation, sharing of physical and abstract resources (a digital whiteboard in a meeting, files common to a group), or the adaptation of the interface to accentuate specific features.

Hull, Neaves, and Bedford-Roberts (1997) presented their work on the Ultra-Portable Computing (UPC) platform, a platform made for ubiquitous computing, which they see an effort to "add value to existing uses of computers and to create new types of application" (p. 146). In the paper they identified the constraints generated by context-awareness:

- the need for an abstraction from the sensors, to give an interface to query contextual data, and the independence between the different sources;
- scalability and extensibility, to avoid having a closed system and to allow concurrent use;
- reactivity to changes;
- fault-tolerance, as the data sources cannot always be trusted.

In their paper they tried to resolve the issues described above, and identified potential applications of context-awareness such as augmented reality, localized information (reminders, tagging of information), appropriate behaviors, or monitoring with appropriate response. Other issues can also slow down the adoption of context-awareness, first because of privacy concerns—contextual data such as location can be especially sensitive, or due to the investment in an infrastructure.

Pascoe (1998) presented, through the description of a system developed to help an ecologist in the field, a study of context-awareness. His work was based on wearable computer, defined in the article as a device providing "a set of core services constantly at the user's disposal, whilst also contextually binding to devices in the user's environment (i.e., ubiquitous computing) that can complement or augment the wearable system" (p. 93), and provided to the user a system to help acquisition of contextual data, as opposed to the traditional approach of presenting data.

Through the definition of context, "the subset of physical and conceptual states of interest to a particular entity" (p. 98), and context-awareness, "the ability of a program or a device to sense various states of its environment and itself" (p. 92), he was able to identify applications for the latter such as providing more information to the user, adapting the device or the software and automate actions, discover resources, and connect real and virtual information.

The project successfully helped the user for her fieldwork, facilitating the work of recording data, because it was integrated in the context of utilization without demanding a special attention for the interaction.

This idea of transparent interaction is reinforced by Abowd (1999) and Kim et al. (2004), and is especially necessary with the explosion of devices around the user, who cannot dedicate attention to everything. This created the need for natural interaction.

Franklin and Flaschbart (1998) identified three steps necessary to make a system context-aware through the description of what enriches the interaction between humans. This is mostly because of the ability to perceive the implicit context. A machine must perceive the initial context where the action happened, the goals the user pursues, and must give an adequate response. For the authors sensing must be tightly integrated with the reasoning and acting, in order to know what and how to sense. They gives the example of movement prediction, which can activate specific sensors to follow the user.

Dey and Abowd (2000) conducted a review of context-awareness, leading to definitions used as references in the research that followed. The authors first described the factors of success for human-to-human interaction: “richness of language”, “common understanding of how the world works”, and an “implicit understanding of everyday situations” (p. 1). The goals sought by making computers more aware of the context are to “increase the richness of communication in human-computer interaction and make [. . .] possible to produce more useful computational services” (p. 1). This process has to automated for the computer, to avoid the difficult task for the user to identify the contextual information needed and enter it.

After this description, Dey and Abowd (2000) provided a generic definition of the context, as a synthesis of the different meanings they found in the existing literature:

> 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. (p. 4)

The nature of information can change according to the situation, creating the need for relevance. They identified then two class of contextual data, the second being a more precise characterization of four initial sources forming the first class, “location, identity, time, and activity” (p. 3).

After defining the context, the authors then defined a context-aware system: “a system is context aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user’s task” (p. 6). Context-awareness can then be sorted in two different categories, “using context and adapting to context” (p. 6). It can then offer three different features, the “presentation of information and services to a user”, the “automatic execution of a service”, and the “tagging of context to information for later retrieval” (p. 8). According to them, the main goal of context-awareness should be the simplification of interaction with computers.

Abowd (1999) then published an article describing ubiquitous computing and its relationship with context-awareness. For the author, the "ubicomp [ubiquitous computing] vision pushes computational services out of conventional interfaces and into the environment in increasingly transparent forms" (p. 75). Ubiquitous computing relies, according to him, on an empirical process, needing experimentation and evaluation to meet the users requirements. Good ubiquitous computing research needs to be focused on applications, scalability, reliability, targeted to real scenarios, and constantly evaluated. The field introduces challenges, and Abowd explained that "software engineering advances are required to support the research endeavors" (p. 79).
He then identified three common features, linking ubiquitous computing with the context:

- Transparent interaction, relying on more natural input interfaces such as voice, gestures, drawings, and tangible user interfaces;
- Context-awareness, defined as "information about the environment associated with an application" (p. 79);
- Automated capture, to give to the user a continuous snapshot of the environment to keep a trace of what is happening.

Gellersen et al. (2002) inferred through the description of different projects they conducted an architecture for multi-sensors awareness. The first one aimed to activate different profiles on a mobile phone, depending on the context. A more detailed description is given in Schmidt et al. (2000), where they explained a project having similar goals as this study, activating predefined behaviors depending on the context, but targeted here to mobile phones. The second project was focused on embedding in everyday-life objects communication and sensing abilities, with the goal of extending the capacities of these objects and trying to integrate them in the context-awareness process.

After these descriptions of projects, they listed possible applications for the context-awareness concept, combining “mobile computing, wearable computing, augmented reality, ubiquitous computing, and human-computer interaction” (p. 341). They considered two ways of sensing, through direct gathering of data or with the request to an external entity collecting and offering contextual information. Early work such as Schilit et al. (1994) relied on indirect collection, because of the inherent computing issues.

They insisted on the utilization of location data to deduce situation, through the combination with other contextual sources, considering the location in itself not informative enough on what the user is doing. Therefore they developed an architecture, the TEA project, to combine multiple sources to build a representation of the situation. They considered that “audio, motion, and light sensors contribute to context-awareness in most settings” (p. 347).

Kim et al. (2004) presented steps to design a context-aware object, with the example of home appliances developed in their experimental group at Samsung. Using the definitions of Dey and Abowd (2000), they classified context-aware objects inside three categories:

- “non-computational contextual design” (p. 185), when the initial design takes context in account;
- “non-computational context-aware design” (p. 186), if the object can answer mechanically or by other means to context;
- and “computational context-aware design” (p. 186), when the object has computational abilities and answer to context changes.

Being a team of product designers, their process insisted on building transparent interactions to avoid the user being submerged by the devices deployed by the ubiquitous computing technologies. An object should be integrated in the usual interactions of the user with its environment, enabling him to focus his attention on more demanding tasks. They then described a project of gate reminder, where they tried to apply the design process they built.

Zhou et al. (2011) focused on the composition of pervasive web services, web services targeted to help the user in everyday situations, using context-awareness. Context-awareness occurs in three different steps of the web services. In the peer coordination, where a peer is a service provider, they used it to regroup the providers and to answer to changes in the organization. Then, in the composition of web services, when the context change an adaptation of the organization of web services has to be performed. And, at a lower level, each of the services is able to respond to changes in the context.

The authors redefined the context to target the specificity of web services:

> Context is any information characterizing the situation of a task session or interaction between a user and his/her service world. Context is categorized into user context, peer context, process context, physical context, and service context. (p. 292)

The steps to build a service is then similar to the previous works in the literature, going from sensing, modeling, reasoning, to finally achieve the composition.

Ranganathan and Campbell (2003) developed an infrastructure for context-awareness, using first-order logic. They define the context as:

> any information about the circumstances, objects, or conditions surround a user that is considered relevant to the interaction between the user and the ubiquitous computing environment. (p. 353)

They described the context as either physical, environmental, informational, personal, or social, related to an application, or to a system. After these definitions, they described the features of their system which gathers data, reasons on it (possibly in a distributed manner), and specifies behaviors. It is viewed as a framework for the developers to design context-aware applications.

2.2 Methods to reason on the context

Different processes are used to translate the data collected by different sensors into useful material for the applications. A layer of abstraction between the physical (or virtual) source is first needed to define a common interface to multiple methods responsible for the acquisition of data (Dey & Abowd, 2000; Gellersen et al., 2002)Hull1997Pascoe1998. According to Abowd (1999), the abstraction is needed to separate context acquisition, processing, and adaptation, and also to help writing applications targeted to different devices, introducing reusability.

Pascoe (1998) used the object-oriented programming paradigm to encapsulate data with utility resources specific to each source. The architecture of the system uses then sources described as artifacts, with synthesizers generating higher level contextual data, with the use of a monitor services responsible for the acquisition and the coordination of the interpretation. Relationships between the artifacts, defining group of data, resulted in the creation of abstract information which can be updated according to the context.

Zhou et al. (2011) listed in their paper different possibilities to model the context: key-value, markup scheme, object-oriented, logic-based, and ontology-based. Wang, Zhang, Gu, and Pung (2004) also made a review of the various methods to represent the context, such as attribute-values tuples, web descriptions of each object, Unified Modeling Language (UML) and Entity Relationship (ER), and first order-logic predicates.

An ontology, as defined in the review made by Guarino (1998), is originally a concept from philosophy, and is an "engineering artifact, constituted by a specific vocabulary used to describe a certain reality, plus a set of explicit assumptions regarding the intended meaning of the vocabulary words" (p. 4). An ontology is a specific implementation of a conceptualization, described as a language-independent "system of categories accounting for a certain vision of the world" (p. 4).

Guarino (1998) continued the explanation of the ontologies with the identification of three categories:

- top-level ontologies, describing generic concepts;
- domain ontologies, relative to a field (medicine for example);
- application ontologies, relatives to a specific task.

An ontology relies on first-order logic to be expressed, using unary predicates to express a concept and binary predicates for the relations. It can be used to describe a vocabulary used inside an application, during the development or at run-time, with for example the message exchanged between the agents.

Once a representation of the context is built, different methods can be used to interact with the model and build new knowledge, before triggering answers in the application side. Franklin and Flaschbart (1998) described two models to reason upon this material, each one with specific advantages and flaws:

- procedural reasoning, in the form of a decision tree. Each context information is matched against a leaf of the tree, creating a simple yet efficient system to build higher level contextual information. Schilit et al. (1994) used this system to analyze the context, in the form of IF-THEN statements;
- declarative reasoning, where operations are performed on the representation of context and actions through data structures. The separation of reasoning and representing allows to build an extensible system.

Declarative reasoning is widely used in artificial intelligence, in the form of descriptive logics or first order logic. Zhou et al. (2011) described in their article the use of semantic technologies to compose context-aware services, using technologies such as Web Services Description Language (WSDL, standard promoted by the World Wide Web Consortium) and Prolog to reason on the web services using first order logic.

Franklin and Flaschbart (1998) used in his article two methods to reason on the context. The first one is plan recognition, in order to infer from multiple actions the most relevant explanation. First order logic is used, to refute inconsistent models, and combine the representation of context. He also used probabilistic methods to find the most likely explanation, or the combination of many to build a more generic model.

Ranganathan and Campbell (2003) identified the benefits of using first-order logic to represent the context. First-order logic has a great expressiveness through the use of ontologies. For example, it can perform various logical operations on the predicates, create rules and queries, and allows to build easily higher level knowledge from the initial environment description. They also envisioned the use of first order temporal logic in future developments to take in account a sequence of events.

Korpipää, Koskinen, Peltola, Mäkelä, and Seppänen (2003) implemented a machine learning approach to process data. They provide a list of resources, which can be used to perform operations on the information:

- machine learning, with hidden Markov models, Kohonen maps, K nearest neighbor classification, Markov chains;
- data mining, with minimum variance segmentation, k-means clustering, Principal Component Analysis (PCA) and Independent Component Analysis (ICA), Bayesian networks.

Ranganathan and Campbell (2003) also used machine learning techniques to synthesize the context and to reduce the involvement of the user in the initial configuration. They used the Na Bayes algorithm to deduce mood from the context of the user, with a true positive rate of 74%.

Wang et al. (2004) used ontologies to represent the context and perform operations on its representation. They justify their choice by the inherent advantages of ontologies: easy knowledge sharing, which is the purpose of ontologies, logic inference use to check consistency and reason, and the reuse of existing ontologies describing applications.

Then, once a representation of the context is built and operations on this representation can be performed, two models can be used by the applications to acquire the context and answer to the changes: a query interface, allowing the applications to request specific data, and the subscribe/notify model, where an application registers itself with the source to be informed when new contextual data is available (Abowd, 1999; Ranganathan & Campbell, 2003).

2.3 Summary

The context-awareness and context definitions among the literature seem to be rather consistent with the ones provided by Dey and Abowd (2000) being the more generic and underlining the need for the relevance of information to characterize a context. Various techniques can therefore be used to operate on the information collected by the sensors, with the emergence of first-order logic models as prevalent method.

After the modeling, reasoning using first-order logic or machine learning algorithms allow to build higher level context description, which can be queried through a common interface as described by many authors.

CHAPTER 3. METHODOLOGY

As stated before in this report, the aim of the study is to build a software as part of the answer to the question *How can the physical and digital context be processed for an adaptive interface?*

In the first part of this chapter, the specifications from the systems are presented. The second part acknowledges the ways to assess the success of the research.

3.1 Specifications

The software designed as part of the study runs on OS X, due to the researcher's available equipment and personal interests. It targets version 10.7 released in July 2011. A prior study on the platform conducted by the author revealed several tools that could be used to customize elements of the interface. In order to interact with the Operating System (OS), the tool is written in Objective-C, C, and uses also external tools accessible through Command Line Interface (CLI). The elements of the interface, as explored in the preliminary inquiry are accessible through commands calls, or with the activation of configuration files, like traditional UNIX text files or Mac OS X plist, which are XML documents.

The steps involved in the process followed by the application in order to perform its goals include the following:

- Define behaviors. The profiles are configured by the user, capturing snapshots of the current configuration or defining manually specific options. The goal is to have a user friendly interface, summing up the different options chosen, with the possibility to have a fine control over it. The settings that are used are identified, before the description of the implementation.

- Acquire the context. The context here is accessible only through built-in sensors and system information, and targets the context of the computer more than the context of the user.
- Model it. To make it available for further use by the software, a representation has to be chosen. As seen in the literature review, several models exist to represent the context, using for example predicates from first-order logic, or ontologies.
- Reason on the context. Using the context as expressed by the model, make use of different techniques to analyze it and build higher level abstractions, to match the situations as described by the user in the initial configuration. The techniques as revealed by the literature review can be simple decision trees, first-order logic, ontology reasoning, data mining and clustering, or machine learning methods.
- Activate the behaviors. With the situations inferred by the reasoner, the profiles are activated, in an non-intrusive way.

3.2 Methods of evaluation

3.2.1 Criterion of performance

Different elements can be measured to determine the performance of the application:

- Time needed to acquire the context: retrieving information about the network configuration, the applications opened, or the calendar can cost time.
- Time needed to reason on it and to activate a profile that seems to meet the current context.

- Accuracy of the choices, through user's evaluation: after a profile has changed a notification to the user is shown, and allows the user to rate the decision made (yes it is accurate, no, or other context information not detected by the application prevents from being a good answer).

3.2.2 Population used for the evaluation

The evaluation is conducted by the researcher to test the application and generate the required data. The different parameters evoked in the previous subsection are collected during the execution of the application, and then analyzed.

CHAPTER 4. ENVIRONMENT, SITUATIONS, AND USER

The study conducted in this report tries to answer to the following question, *How can the physical and digital environment be processed for an adaptive interface?*

In the question defining the research, the information is described as coming from the environment. The following sections refine this affirmation by clustering the environment into situations, which are the activities where the interactions between the couple formed by the user and its computer with its surroundings occur. First a description of what builds these situations is given, before identifying which elements can characterize from an observer point of view these activities.

4.1 The creation of a situation

A situation, as defined earlier in this study, is an activity or a set of tasks performed by the user with its computer, in a given context. The context here qualifies the current activity by providing a specific answer to the how, when, where, and what questions from the state of the environment. The situation can therefore be qualified in term of actions and surroundings. In the first paragraph, the activity component is described, before studying what can be observed externally to identify the situation.

4.1.1 Building the situation

A situation as stated before is qualified in term of context and in term of tasks performed. These parameters taken separately tend to only form a part of an activity, without creating it certainly. As an extreme example, a person could use personal files to illustrate a work-related document, pointing first towards a situation related to the private life of the user, but also towards his work life because of the edition of the work document. A situation is therefore described by the association of multiple parts of the activity, creating a coherent meaning.

The context of the situation is a collection of parameters characterizing the environment where it happens. It can be related to the physical world, with precise information like the location of the person, or more abstract by associating these locations with the notion of home and workplace. It can also be related to the digital world, and can then designs what applications are in use, what files are browsed and if they relate to work or to the personal life of the user, what are the characteristics of the network, etc.

Three generic situations can be defined, when analyzing the day of an employed person. These situations serves as basis for the rest of the study, as they expose in a precise manner three distinct elements. In the following description, the assumption is made that the user and its computer are always confronted together to the situations. It can be viewed as a limit as work and personal computers can be two different things, because for example of the policies of the company or of the own choices of the person. The first one is the work situation, where the person performs its main occupation of the day. It is usually characterized by being at a specific location, the workplace, even if this definition presents immediate limitations for people working from home, or consultants visiting different workplaces. In this situation, the person studied interacts with other fellow employees, exchanging work related files, or showing progress to others during meetings. He also uses resources belonging specifically to his workplace, such as file servers, applications, or printers.

A second situation occurs when the person comes home with its computer, and migrates from a work status to a more relaxed status. Its utilization of the computer changes, and it now serves for leisure, browsing the web, interacting with people from his family or his friends, listening to music or watching videos. It is usually used in a smaller network that has well-known characteristics in term of configuration and services. The work is not the main focus anymore, whereas the personal life of the user drives the activities performed.

A third situation for the user can be described as the interaction with the computer in a public area. Examples of this situation can be using the computer waiting for transportation such as a flight, spending some time in a coffee shop offering internet access, etc. It is more difficult to precisely define than the two previously described, as the person can be found in this shared area for work as well as for personal reasons. The activity performed can be responding to emails, working on some documents related to a coming meeting, planning a future activity with friends or family, or just enjoying some free time.

These three situations can be viewed as purely theoretical, as work life and personal life tend to collide. Connections between the two can happen when the person has to solve a personal problem at work, for example through a phone call to an agency only open during work hours, or when important work-related mails arrive, needing an answer from the user although he is currently home.

4.1.2 Identifying this situation

A situation, being a collection of tasks in a given context, present observable aspects making it identifiable. This identification is necessary in order to be able to take advantage of them and propose an adapted interface to the user. The identifications can be operated at three levels:

- An internal observer, *i.e.* the user himself. It has the characterization of internal as this level has access to the knowledge of the user. The user is supposed to know in theory in what situation he currently is, and what he is achieving, providing an immediate and accurate answer to the question. The major constraint with this level is that the user needs to still provide an effort to do the identification, asking to step back from the activity and observe what its status is, what parameters are in use.

- An external observer, not the user, participates to the study and tries to identify what the person is currently doing, by "looking over his shoulder". Obviously this way of identification does not present any particular advantage, other than liberating the user from the task. But a major constraint of this method is the need for this observer to learn what the user usual situations are in order to be able to recognise them.

- A third one, which seems well fitted to perform this task, is an agent inside the user's computer. Having access to the virtual world of the computer where the user is performing his activity, and to the physical world through the help of sensors, he can build the situations at no cost for the user, once a learning period is achieved. The learning faculty is also a strong point playing in favor of the agent, with machine learning field in computer science allowing the agent to learn from the user, be trained to the different situations it might encounter, and correct its mistakes.

This third observer model is the one chosen for this study, as it answers the needs for an user to perform actions silently, without additional overhead. The learning capacity of this model allows to define initially simple situations, that can be then modified according to the specificity of the user. Indeed, the division between work, home, and public life is often difficult to see for the common person presented to these three situations. To add difficulty to the process, many users don't have these three classics situations, for example someone working from home, or a student doing research.

An immediate limitation that pops out because of the usage of a computer program to identify these situations it the dependence to what the agent is capable of sensing inside its environment. The limitation can be related to the physical world, because of the lack of sensors causing the agent to be aware of certain aspects of the context, or related to the digital world, because mainly of what is offered to the program in term of information by the operating system, for example caused by a lack of public Application Programming Interface or a limitation in term of privileges attributed to the program.

In the three situations described in the previous sections, many physical and digital elements can be observed that characterize them. For the first situation, workplace, the first obvious context elements in the location. Being in the company walls is a certain criterion of being at work. Other parameters related to the physical world are the date and the time, as usually the work hours are determined —taking the holidays into account. The agenda of the person can also provide useful information, telling for example that the user has a scheduled professional trip. Regarding the digital environment, many elements can help determine the current activity. The network in a company usually exhibit specific characteristics in term of address ranges, and other configuration information such as the gateway or the DNS (Domain Name System) servers. Services on the network are also important, like the presence of different servers, printers, etc. The applications running on the computer, as well as the files used inside those applications are also very specific to the current frame, even if as stated before false positives can occur when multiple activities collide.

The second situation corresponds to the person and its computer in a personal environment. Again, the location of both is an important element in determining which situation they are in. In a similar way, the personal life is affected to a certain date and time, and events can be scheduled on the person's calendar to highlight specific events related to personal activities such as holidays with family members or friends, trips, appointments, etc. The smell is also different at home, with some people sensible to this odours that are specific to certain areas. When thinking of the digital environment, the network again comes first. The name of the wireless network is somewhat unique, and can help determine the context. Indeed, SSIDs (Service Set IDentifier) are used by companies to geolocate users of mobile devices or computers, once they have been collected with their approximative positions. It allows to speed up the time needed to determine the position of the user compared to using GPS (Global Positioning System), or even to enable geolocation on devices disposing of no GPS. The applications used can also be specific to the personal activities, such as listening to music with a player browsing pictures, or surfing the web.

The third situation identified in the previous section is related to the use of the computer by the user in a public place, such as a coffee shop or an airport. The location alone is not really useful, but can be matched against available databases to identify the place. Foursquare, Yelp, or Facebook all use this kind of databases to propose geolocated content, and it is possible to use this data externally to match a location with a restaurant or a shop. The date and time are only useful if they can be matched to an event set up in the user's calendar. For the digital elements of the context, the SSID is again a good information as they tend to be consistent in public places, because they are operated by the same companies. The situation can also be inferred by changes, activated if no usual elements are found in the context, as a protective reaction.

4.2 Adapting to a situation

The previous sections were focused on defining what creates a situation, and how to identify it. In this section the focus is on the description of what makes a computer adapted to a situation, what are the dependent settings and how they affect the digital activity of the user.

The activity of the user on its computer can be centered around different poles corresponding to certain groups of applications and files. This separation also depends on the user, what his professional occupation is: a journalist does not present the same usage pattern as a software engineer or a biology researcher. The learning ability of the program adapting the computer to the situation manifests here its utility. Beside this classification of the usage of the computer, a common adaptation can be performed on the machine depending on its operating environment, related to simpler settings. The network configuration is dependent of the situation, with parameters such as using a proxy, or other specific settings. OS X provides already a way to gather these settings related to network in a profile, that can be easily activated in two clicks. The organization of the workspace is also important, with parameters such as desktop background, choice of screensaver, running applications, shortcuts, bookmarks, and icons presented to the user. The computer can also be adapted to limit the distraction of the user, enabling notifications, defining priorities in the mailboxes, hiding some unnecessary information, reducing the volume, etc. The energy profile is another aspect of the configuration, to set the display and computer sleep times, the screen brightness.

In the three situations defined previously in the chapter, we can identify the values these settings should take when the user and the computer are presented to the operating environment. At the workplace, the user might need to share work-related files on the network, or to be able to log in from another computer via remote access. The personal content of his computer, depending on the user's choices, might not be relevant for the task performed and then more confined. The network configuration might need some changes to conform with the company's policies, such as the activation of a proxy setting. Applications that will be used by the user can be started, resuming the files opened in the previous session. The arrangement of bookmarks and shortcuts can also be altered to present in priority relevant aspects to the user.

The second situation is related to the personal life of the user. The goal for the user is to relax, performing activities related to amusement. New groups of applications can be started, proposing to the user certain categories of files in priority. The network configuration is simpler, with a more direct access to internet. The files shared won't be the same, offering pictures, music, or videos rather than work files.

The third situation is the public case. In the public case, the user wants to keep the maximum privacy and protection on its computer, shutting down the services he offers on the network such as shares, or remote access. Other security measure such as a password lock after a certain idle time can be activated, to offer a better protection of the computer against unwanted access.

In this chapter the concept of situation was further developed, overseeing what elements of the context build a situation and how it can be recognize. In the last section of this chapter elements of configuration were described, associated with three different situations that are professional, personal, and public life. In the following chapter, this generic discussion is applied to the development of a prototype showing an example of how the interface of a computer can be adapted to the situation.

CHAPTER 5. IMPLEMENTATION OF THE PROTOTYPE, AND RESULTS

The study conducted in this report tries to answer to the following question, *How can the physical and digital environment be processed for an adaptive interface?*

The physical and digital environment creates situations as described in the previous chapter, that are a set of activities performed by the user and its computer in a specific context. The elements of the context that can be useful to determine the situations the user is in, are multiple and do provide a good understanding of what is happening in the environment. But the richness of all this available information is limited by the possibility to acquire it and to process it. In this chapter, the development of a basic and limited prototype is described, serving as example that a computer interface can be adapted to the context.

5.1 A prototype for OS X Lion

As exposed in the limitations of this study, the prototype designed is built for OS X Lion, running on a computer that is not equipped with any specific awareness other than the stock sensors. OS X Lion has been chosen for practical reasons, as it is the operating system of the author. The available information is described in the subsection 5.1.3. The application demonstrating the capacity of a computer to be adapted is composed of three different parts, divided as represented in the figure 5.1. The first part, the preference pane described in the subsection 5.1.1, is responsible for creating the profiles according to the three situations described in the previous chapter. It defines profiles in an XML file stored globally for the user, which is then used by the center piece in the diagram described in the subsection 5.1.2 and 5.1.3,

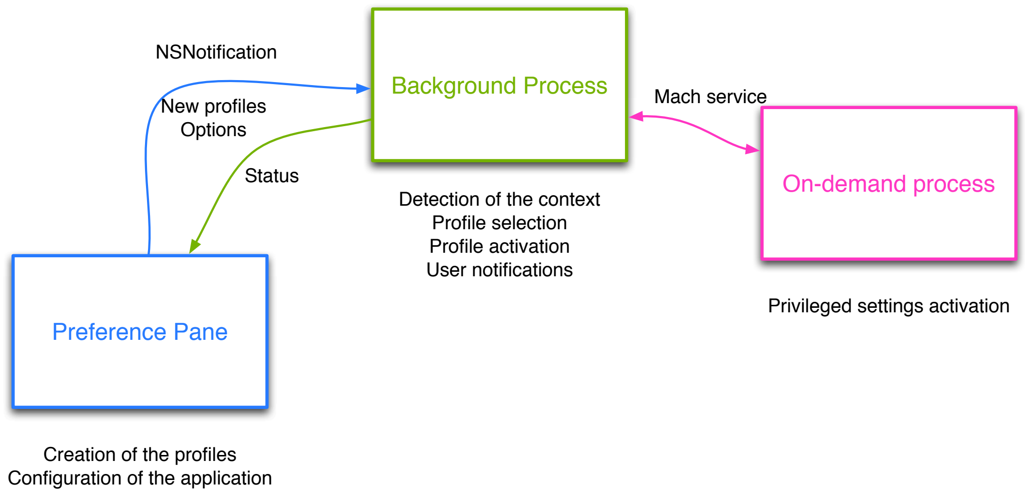


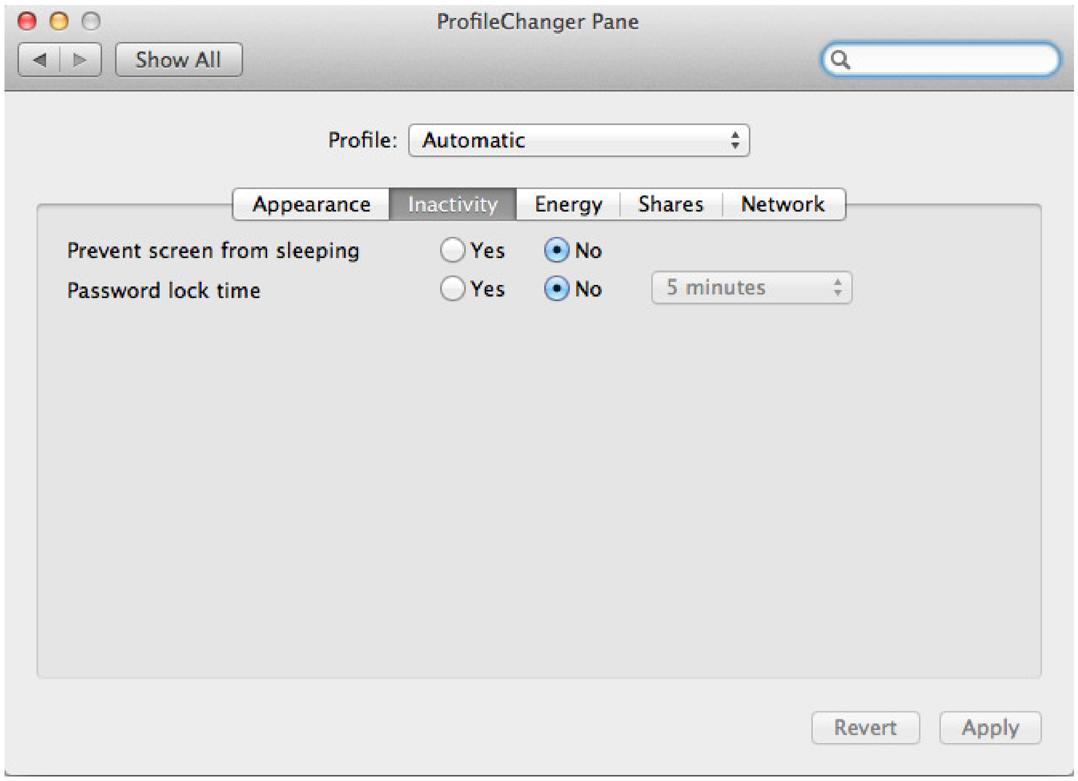
Figure 5.1. The architecture of the prototype, describing the three components and how they communicate with each other

the background process, to perform the adaptation to the context. The last part is used by the profile activator to activate certain settings needed higher privileges than the one owned by the background process.

5.1.1 Creation of the profiles

A profile is a collection of settings, which are used to configure the computer for the user in presence of a situation, and a collection of context triggers, mapping elements of the context with their values in that same situation. There is therefore a profile for each situation. The creation of the profiles is made in the system preferences, a program regrouping in OS X the configuration in use system-wide, as opposed to per-application preferences configured inside the application. And as the prototype has system-wide consequences, it seemed like the natural place to put it.

To create a profile, the user has to be in the situation where to activate it. It allows to simplify the process of choosing which values should be taken for the element of the context, and a too complex step for the creation of the profile. A second prerequisite is to have the configuration of the computer in the state wanted for that profile. This step is also meant to simplify the creation of the profile as working interface already exists to configure the computer. Some elements are still modifiable in an advanced view when they are simple enough such as a boolean value. Taking a snapshot of the situation is therefore saving completely the current state, which corresponds to the situation and replies to the questions what to do and when.



Profile: Automatic

Prevent screen from sleeping ◯ Yes [✓ No]

Password lock time ◯ Yes [✓ No] 5 minutes

Figure 5.2. The preference pane to create the profiles, showing the customization for the network services

A view of the preference pane, which is a section of the system preferences, is shown in the figure 5.2. On this figure are displayed the controls for the network services, when the user wants to modify which preferences are changed when activating the profile.

Once the profile is validated by the user, a description including the settings to configure and the context in which to activate this profile is stored in an Property List (plist). A property list is a container organizing data in a structured way for disk storage. This data is represented through basic types of Core Foundation, such as simple collections (array or dictionaries) or numeric and string types. A file is then created to store this data, using either a XML (eXtended Markup Language) syntax with a well-defined Document Type Definition (DTD) or an opaque binary format. The information stored in the case of the application implemented for this research relies on numeric, booleans, or string types, and can therefore be organized inside the collections offered for a property list. The listing 5.1 presents an example of a profile as stored on the disk. Two sections are needed to compose a profile, the `Settings` and the `Context` keys. For each value of these keys, a new dictionary is exposed to define either which setting or context element is used, and the value it should take.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/
    PropertyList-1.0.dtd">
<plist version="1.0">
    <dict>
        <key>Current Profile</key>
        <string>Public</string>
        <key>Profiles</key>
        <dict>
            <key>Public</key>
            <dict>
                <key>Settings</key>
                <dict>
                    <key>Allow remote login</key>
                    <false/>
                    <key>Enable AFP sharing</key>
                    <false/>
                    <key>Enable WiFi</key>
                    <true/>
                    <key>Enable Windows sharing</key>
                    <false/>
                    <key>Enable bluetooth</key>
                    <true/>
                    <key>Enable password lock</key>
                    <false/>
                    <key>Enable screen sharing</key>
                    <false/>
                    <key>Password lock time</key>
                    <integer>300</integer>
                    <key>Prevent screen sleep</key>
                    <false/>
                    <key>Show desktop</key>
                    <true/>
                </dict>
                <key>Context</key>
                <dict>
                    <key>SSID</key>
                    <array>
                        <string>Boingo Wireless</string>
                        <string>Starbucks</string>
                    </array>
                    <key>Location</key>
                    <array>
                        <string>40.425869,-86.908065</string>
                        <string>40.426581,-86.911082</string>
                    </array>
                    <key>Default profile</key>
                    <true/>
                </dict>
            </dict>
        </dict>
    </dict>
</plist>
```

Listing 5.1 Example: a profile as stored in a plist for use by the different elements

Preferences are stored in a directory that is common to all the applications existing on the computer, on a per-user or system-wide basis. This directory is either ~/Library/Preferences, or /Library/Preferences for the system-wide, any user storage. Each application has its own file, stored using as name an unique application identifier, usually the domain name of the company in reverse-DNS notation such as com.apple, followed by the name of the application. The file for this application is named net.vicould.ProfileChanger, and is stored in ~/Library/Preferences as it is a per-user profiles collection.

5.1.2 Application of the profiles

In this part, an agent is a background process with a user interface, as defined by the OS X developer documentation. For this prototype, the agent plays multiple roles, that are detailed in this section and the following one. The first role presented is to apply the preferences, to adapt the computer to the situation, once this situation has been identified. First, the agent retrieves the profiles from the configuration file stored in ~/Library/Preferences. As the agent is constantly running, and changes to these profiles could be made by the user once the program is started for the first time, the preference pane is able to notify the agent that it should reload all the profiles. It is performed through the use of NSNotifications, which are delivered to an application subscribing to this type of event through a NSNotificationCenter, using the subscribe-notify design pattern. As the two applications are separated, the notification is in fact fired from a shared notification center, instance of NSDistributedNotificationCenter, available for all the applications running on the system.

Once the event has been received by the agent, it reloads the preferences using the CFPreferencesCopyAppValue function that parses and returns the content of the plist. This step accomplished, this data is only used when a situation is detected as described in the following section. Whenever a new situation is detected, the corresponding profile stored in memory is read and its preferences are applied, using separated classes specialized in one type of configuration, such as network or power management related settings. Outside of the prototype, these settings are usually managed and activated by specific programs, that also store these settings on disk using plists. Therefore, in order to activate the right configurations, each class in turn either modifies other plists, or executes external program that take as input the configuration. For example the Apple Filing Protocol (AFP) shares are enabled or disabled by using `launchctl`, which takes care of loading the daemon in the right configuration. Another example is showing the icons on the desktop, setting configured by modifying the `com.apple.Finder` plist, and restarting the Finder.

In the first example of settings activation, the target is a network service, requiring privileges to be started as it needs to open a listener to wait for connections to the low-number port. In order to perform that operation, as well as other settings activation that require privileges, a helper having this privileges has to be invoked. The recommended way to accomplish that in OS X Lion is to use a privileged helper, *i.e.* a program running with the root uid and gid, that will be able to perform calls without limitations. The helper is started on-demand by `launchd`. To establish the communication between the agent and the helper, an interprocess communication library is used, called XPC. This library allows to exchange data between two processes, using C primitive types similar to the one used in the property lists. The communication is established on top of a mach service registered by the helper tool, and to which the agent connects. When the agent tries to connect to the helper tool, `launchd` handles the connection and starts the helper if it is not running. Then the XPC message sent by the agent is received by the helper tool, and parsed to analyze what to do.

The content of this message is set by the class activating the setting requiring privileges. The message is created with at least one key: the value for this key is a space separated list of the other keys that contain preferences values. After reading that key, the helper iterates over the list of the other keys to retrieve what preferences to change and what values to set. Then the helper invokes the corresponding operation, which can be restarting a daemon in the case of the AFP shares service given previously in this section, using a `NSTask` to wrap the call. It then replies to the sender to confirm that the changes were made. At this moment a visual notification is displayed to the user, informing him which profile has been activated.

5.1.3 Inferring the situation

The other task conducted by the agent in the prototype is to acquire the context and determine the corresponding situation from the different elements available. Relevant context components were described in the section 4.1.2, identifying generic external information that could be used in a maximal awareness condition. For the prototype, as stated in the limitations of the study, the context is only acquired through the raw capacities of the computer where it is running, without the help of any additional sensors. Compared to actual smartphones that are equipped with location capacities, inertial measurement unit, light sensors, and microphones, a computer has limited abilities when asked to poll the physical environment for data. Therefore, most of the context acquired by the computer in this study is related to the digital context. Even the location data is available only when WiFi access points are in range, as it is the way Apple provides to get the location of the user.

The context gathered in this study for the prototype implemented comprises: the applications running, the date and time, the time zone, the external IP (Internet Protocol) address of the computer, the IP address of the gateway, the IP address of the machine, the type of network (ethernet or wireless), the SSID of the wireless network to which the computer is connected, the power source used by the computer (battery or AC power), and the location of the computer and its user.

Each context source that were described in the previous paragraph complies to a certain protocol, creating an abstraction between the part of the code that needs the context and the part polling it. The context acquisition is triggered by a recurring timer, allowing the application to watch the changes. Each context source then returns a vector of the context elements, that is used by the reasoner to perform the situation inference (Delaveau, Loulier, Matson, & Dietz, 2012). An example of this representation is given in equation 5.1.

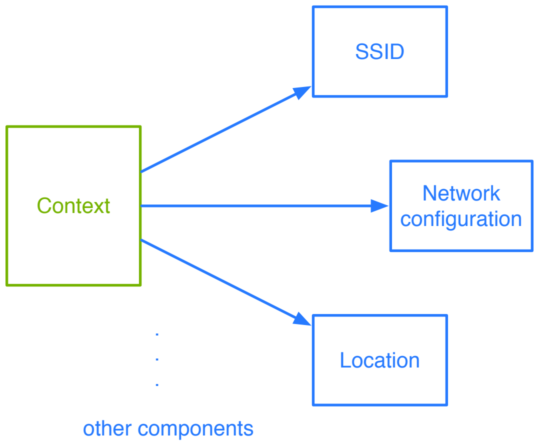


Figure 5.3. Decomposition of the context in its different aspects

$$\vec{object} = \begin{pmatrix} \vec{network} \\ \vec{location} \\ \dots \end{pmatrix} \tag{5.1}$$

$$\vec{location} = \begin{pmatrix} longitude \\ latitude \end{pmatrix} \tag{5.2}$$

Inside the system, several vector-spaces representing each an element of the context (location, date and time, etc.) are built, allowing a different internal representation of the values associated with the variants of one type of context. For example, a complete numerical information such as the location, as presented in the equation 5.2 will be represented differently than the SSID. In figure 5.3 the decomposition of the current context as detected by the sensors in its different aspects is shown.

$$\vec{context} \cdot \vec{object} = \begin{pmatrix} \alpha_1 \times \vec{noise} \\ \alpha_2 \times \vec{light} \end{pmatrix} \cdot \begin{pmatrix} \vec{noise} \\ \vec{light} \end{pmatrix} \tag{5.3}$$

$$similarity = \frac{\vec{context} \cdot \vec{object}}{\|\vec{context}\| \times \|\vec{object}\|} \tag{5.4}$$

To calculate the similarity of the query to the corpus of contextual actions, the formula given in equation 5.4 is used. It computes the scalar product between each vector of the query representing the current context and the tested possible answer from the corpus is used, following the methods from Information Retrieval (Raghavan & Wong, 1986), and divides it by the norm of the different vectors. The similarity corresponds the cosine of the angle between the two vectors, once each component of the input has been weighted. Because of the representation of the context in different vector-spaces, the final result is the scalar product between two vectors, with each component being the weighted scalar product of the input and the tested answer for each element of context. The values of the similarities are then compared to deduce the current situation.

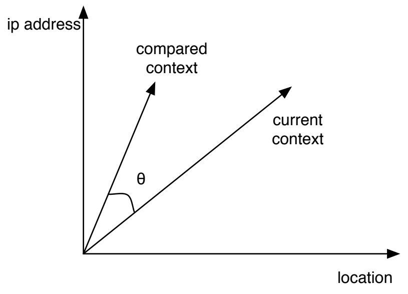


Figure 5.4. Example of similarity between two vectors, one modeling the context and the other an item from the collection

5.2 Testing the prototype

With the prototype completed, the main goal was to assess its ability to successfully detect the situations. The prototype was tested for profiles that follow the three configurations previously described: work, home, and public. The public profile is configured as the fallback if no corresponding situation is found. In the next sections, each profile are described with the elements of context that are used to trigger their activation, and the affected settings.

5.2.1 Profile 1: work at PHSI

PHSI (Purdue Homeland Security Institute) corresponds to the author's research assistantship workplace. This place is on the Purdue campus, but at one of its extrema separated from the rest of the buildings where the courses take place. In that building the same WiFi SSID as the rest of the other campus buildings is offered, "PAL2.0" or "PAL3.0". The search domain for that network is also used, even if it is shared over the campus. The other network parameters, such as the external ip address, the dns configuration or the ip address of the default interface were not used as they are common to the whole campus, with some variations that are not enough to segregate into groups. The power source is set to "AC Power", as the computer is usually plugged to the power in that situation. The last element important for that context was the date and the time, as the work hours for the assistantship were known for that experiment and used to determine the schedule.

When the situation was successfully activated, different settings were activated:

- The sleep time was set to 30 minutes, as the computer is connected to the power.
- The AFP and the Windows sharing are disabled.

- The screen sharing and the remote login are enabled.
- The WiFi and the Bluetooth are activated.
- The icons on the desktop are hidden.

5.2.2 Profile 2: work at the M2M lab

The M2M (machine to machine) lab hosts the research group the author belongs to. It is also on the Purdue campus, at a different (about a mile between the two) location than the PHSI. It has a specific SSID, "M2M-lab", with a network segment in the 192.168.1.255 mask. The address of the gateway is 192.168.1.1, which is common for a small-sized network, and the DNS search domain was a known value, "m2m". The external ip address is also one of the element used as it is statically attributed to the gateway of the lab. Again, the computer is plugged to an electrical outlet when used inside the lab. Date and time were also two known parameters of the context as the presence in the lab was scheduled.

The settings affected were:

- Sleep time set to 30 minutes.
- The AFP and the Windows sharing are enabled.
- The screen sharing and the remote login are enabled.
- The WiFi and the Bluetooth interfaces are activated.
- The icons on the deskop are shown.

5.2.3 Profile 3: home

The profile home was set up with the location of the author's home. The SSID of the WiFi is also a used parameter, with the value "SmileApt4". Time was set up for the weekdays between 7pm and 9am, and the whole day for the weekend. The external ip address is changing with unknown values, so it could not be a reliable context element. Instead the ip address of the computer was used, which was a set value inside the router configuration on the MAC address of the computer wireless interface. The power source could be either battery or plugged, so it was not considered for that profile.

The settings' values were:

- Sleep time set to 10 minutes.
- No password lock.
- Network shares enabled.
- Screen sharing and remote login enabled.
- Items on the desktop shown.

5.2.4 Profile 4: public

The last profile used was more a fallback when no other profiles were matching the current environment. It is difficult to define a public location, so no specific elements were studied to understand the environment. It is difficult to determine the public condition of the area other than polling location and trying to match it with a directory of places around using Foursquare or Facebook. For this research this feature was not implemented.

The settings affected were aimed at protecting the user, disabling what could expose the privacy of the user:

- Sleep time set to 10 minutes.

- Password lock time to 10 minutes.
- Network shares disabled.
- Screen sharing and remote login disabled.
- Items on the desktop hidden.

5.2.5 Experiment

The experiment realized consisted in testing the four previously described situation in their respective environments. The profile PHSI was tested five times, the profile M2M five times as well, the profile home was tested 6 times, and the public 3 times in a library, and in two coffee shops. Apart from the public profile, which does not correspond to specific values of the context but is rather a fallback when no know situation is detected, the tests took place in the respective environment conditions.

The application was activated when arriving in a new situation, for example arriving at the workplace in PHSI, or returning home in the evening. Once the situation was detected and the appropriate profile selected the application was turned off, until the next change in situation.

The variable observed was the identification of the situation, which took binary values: recognized or not. No specific performance tests were conducted, as the prototype was a first attempt to demonstrate the feasibility of such adaption in response to context changes.

As it is described in the chapter 6 presenting the future research, other methods could be implemented to model and reason on the context. With the different reasoning methods benchmarks could be conducted to compare them.

5.2.6 Results

The protocol described in the previous section was applied consecutively over one week to the four different situations described by the profiles. The table 5.1 shows the accuracy results for these profiles. Those values represent the percentage of success for the identification of the profile. For example, PHSI was tested five times and successfully detected four times by the application. The lower accuracy for the M2M profile is explained by one of the test that happened outside of the configured hours, while the WiFi was disabled. It prevented the context engine from detecting the environment and identifying the situation, and activated the public profile instead.

Table 5.1
The accuracy of the profiles activation

	PHSI	M2M	Home	Public
Accuracy	80%	60%	100%	100%

The time required for the profile's activation, from the start of the application to the end of the configuration changes was less than 20 seconds, in average 12 seconds, including the time for the location to be refined. The acquisition of the location is the most time-consuming in the context acquisition, as many requests are made to external servers to transmit the data needed for the location detection and to receive the reply. In comparison, doing the changes manually requires over one minute spent in exploring the different sections of the system preferences, with the need to enter the administrator password for privileged settings. Inside the application exists the possibility to stop the context engine and select manually a profile, which takes less than a second to be applied. Doing the selection that way removes the automation part of the selection, which is the primary goal of this research.

CHAPTER 6. CONCLUSION AND FUTURE RESEARCH

The study conducted in this report tried to answer to the following question, *How can the physical and digital environment be processed for an adaptive interface?*

The research performed in the chapter 4 demonstrated that this environment could in fact be regrouped in a cohesive way by creating situations, which are meaningful collections of tasks performed in certain operating conditions, such as work, personal life, or public life. Once these situations are built, using the different context elements of the digital and physical world as acquired by a computer program on the user's machine, it is possible to act on the user's computer to adapt the interface to its situation. This adaption is motivated by creating the best conditions to perform certain tasks, such as shutting down the sources of distraction when working, or forgetting about the work when relaxing at home. It can also be motivated by the need to protect the user and its computer from various threats, going from setting up a frontier between work and home in the case of an important meeting, to security itself in the case of a public network, where attempts can be made to penetrate the computer to acquire valuable information. The automated adaption is therefore even more valuable as the application of these settings is a painful and time-consuming process, which simplification by for example grouping the changes in scripts is only reserved to power-users.

In order to apply this research to the real-world, a prototype of an application having these features has been developed and described in the chapter 5. This prototype was a first experiment with the topic, and results demonstrated the effective possibility of adapting the interface of a computer to its environment. Several simplifications were made when establishing the specifications of the prototype, in order to reach a feasible concept for this research. The context was indeed limited to what a computer equipped with no external sensors than what comes standard, setting the main source of the context as the digital environment, which can be indirectly related to the physical one. But at the same time the goal of this study was to determine if such adaptation could be performed at large-scale, *i.e.* by performing it on simple computers.

Another direction for this research is to compare different reasoners, using tools described in the literature review performed in the chapter 2. For example, rather than using similarity detection by computing the cosine of the vectors, the context elements composing the situation could be written as rules, and first-order logic could be performed on it using a reasoner such as CLIPS.

To improve the help brought to the user by the program, one way is to work on the recognition of the situations using the user's feedback. In the current prototype the situations are somewhat hard-coded once, with no place to learn. As stated before, situations tend to collide, when work is conducted inside another context such as from home. The three situations defined for this research only work for a specific part of the population, who has a very structured and segregated life, with a clear separation between work and personal life. Learning from the habits of the user with machine learning techniques could help increase the potentiality of the application. One envisioned way was to affect weights to the context sources in order to select the ones giving the most useful results, like the location or the SSID.

LIST OF REFERENCES

LIST OF REFERENCES

Abowd, G. D. (1999). Software engineering issues for ubiquitous computing. In *Proceedings of the 21st international conference on software engineering - icse '99* (pp. 75–84). New York, New York, USA: ACM Press. Retrieved from http://portal.acm.org/citation.cfm?doid=302405.302454 doi: 10.1145/302405.302454

Delaveau, L., Loulier, B., Matson, E., & Dietz, J. E. (2012, March). A vector-space retrieval system for contextual awareness. *in Situation Awareness*, 162–165. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6188372http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6188372 doi: 10.1109/CogSIMA.2012.6188372

Dey, A. K., & Abowd, G. D. (2000). Towards a better understanding of context and context-awareness. In *Chi 2000 workshop on the what, who, where, when, and how of context-awareness.*

Franklin, D., & Flaschbart, J. (1998). All gadget and no representation makes jack a dull environment. In *Proceedings of the aaai 1998 spring symposium on intelligent environments* (pp. 155–160). Retrieved from http://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:All+gadget+and+no+representation+makes+Jack+a+dull+environment#0

Gellersen, H. W., Schmidt, A., & Beigl, M. (2002). Multi-Sensor Context-Awareness in Mobile Devices and Smart Artifacts. *Mobile Networks and Applications*, *7*(5), 341–351. Retrieved from http://dx.doi.org/10.1023/A:1016587515822 doi: 10.1023/A:1016587515822

Guarino, N. (1998). Formal ontology in information systems. In *Proceedings of fois'98, trento, italy, 6-8 june 1998* (pp. 3–15). Amsterdam: IOS press. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.29.1776&amp;rep=rep1&amp;type=pdf

Hull, R., Neaves, P., & Bedford-Roberts, J. (1997). Towards situated computing. In *Digest of papers. first international symposium on wearable computers* (pp. 146–153). IEEE Comput. Soc. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=629931 doi: 10.1109/ISWC.1997.629931

Kim, S., Park, S., Lee, J., Jin, Y., Park, H.-m., Chung, A., ... Choi, W. (2004, May). Sensible appliances: applying context-awareness to appliance design. *Personal and Ubiquitous Computing*, *8*(3-4), 184–191. doi: 10.1007/s00779-004-0276-9

Korpipää, P., Koskinen, M., Peltola, J., Mäkelä, S.-M., & Seppänen, T. (2003, July). Bayesian approach to sensor-based context awareness. *Personal and Ubiquitous Computing*, *7*(2), 113–124. Retrieved from http://www.springerlink.com/Index/10.1007/s00779-003-0237-8 doi: 10.1007/s00779-003-0237-8

McArthur, S. D. J., Davidson, E. M., Catterson, V. M., Dimeas, A. L., Hatziargyriou, N. D., & Ponci, F. (2007). Multi-Agent Systems for Power Engineering Applications. *IEEE Transactions on Power Systems*, *22*(4), 1743–1752.

Pascoe, J. (1998). Adding generic contextual capabilities to wearable computers. In *Digest of papers. second international symposium on wearable computers (cat. no.98ex215)* (Vol. 44, pp. 92–99). IEEE Comput. Soc. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=729534 doi: 10.1109/ISWC.1998.729534

Raghavan, V. V., & Wong, S. K. M. (1986, September). A critical analysis of vector space model for information retrieval. *Journal of the American Society for Information Science*, *37*(5), 279–287. doi: 10.1002/(SICI)1097-4571(198609)37:5<279::AID-ASI1>3.0.CO;2-Q

Ranganathan, A., & Campbell, R. H. (2003, December). An infrastructure for context-awareness based on first order logic. *Personal and Ubiquitous Computing*, *7*(6), 353–364. Retrieved from http://www.springerlink.com/openurl.asp?genre=article&id=doi:10.1007/s00779-003-0251-x doi: 10.1007/s00779-003-0251-x

Roche, R., Blunier, B., Miraoui, A., Hilaire, V., & Koukam, A. (2010, November). Multi-agent systems for grid energy management: A short review. In *Iecon 2010 - 36th annual conference on ieee industrial electronics society* (pp. 3341–3346). IEEE. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5675295 doi: 10.1109/IECON.2010.5675295

Schilit, B., Adams, N., & Want, R. (1994). Context-aware computing applications. In *Workshop on mobile computing systems and applications* (pp. 85–90). IEEE Comput. Soc. Press. Retrieved from http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=512740 doi: 10.1109/MCSA.1994.512740

Schmidt, A., Takaluoma, A., & Mäntyjärvi, J. (2000, December). Context-Aware telephony over WAP. *Personal Technologies*, *4*(4), 225–229. Retrieved from http://www.springerlink.com/index/10.1007/BF02391563 doi: 10.1007/s007790070008

Wang, X. H., Zhang, D. Q., Gu, T., & Pung, H. K. (2004). Ontology based context modeling and reasoning using OWL. In *Ieee annual conference on pervasive computing and communications workshops, 2004. proceedings of the second* (pp. 18–22). IEEE. doi: 10.1109/PERCOMW.2004.1276898

Want, R., Hopper, A., Falcão, V., & Gibbons, J. (1992, January). The active badge location system. *ACM Transactions on Information Systems*, *10*(1), 91–102. doi: 10.1145/128756.128759

Weiser, M. (1991, January). The computer for the 21st Century. *Scientific American*, *99*(1), 94–104. Retrieved from http://wiki.daimi.au.dk/pca/_files/weiser-orig.pdf

Zhou, J., Gilman, E., Palola, J., Riekki, J., Ylianttila, M., & Sun, J. (2011, August). Context-aware pervasive service composition and its implementation. *Personal and Ubiquitous Computing*, *15*(3), 291–303. Retrieved from http://www.springerlink.com/index/10.1007/s00779-010-0333-5 doi: 10.1007/s00779-010-0333-5