A System Framework for Decision Support in Ambient Intelligence

知的環境下における意思決定支援のためのシステムフレームワーク

February 2011

Waseda University
Graduate School of Fundamental Science and Engineering
Major in Computer Science, Research on Distributed Systems

Tetsuo YAMABE

Abstract

Decision making is something indispensable to our everyday lives. In order to improve the quality of life, from individuals to top decision makers in big companies, we always try to find out the best way that would maximize profit or bring pleasant outcomes. However, people tend to take non-optimal or irrational decisions, or even make mistakes due to the limitation of cognitive capability, biases and emotional influence. Thus *decision support systems (DSS)* have been developed to aid a decision making process with computers. DSSs provide supplementary information, particularly considering complementing the limitation in decision capability. DSSs also aim at improving a novice user's skill with providing expertise knowledge, and ideally the user is expected to perform better decisions without system support in the end. On the other hand, as the term *Ambient Intelligence (AmI)* implicates, any scenes in our daily lives are gradually augmented with a variety of devices and services. Small, but powerful computers, sensors, and actuators are embedded into environments, and users can interact with its intelligence in both implicit and explicit manners. The AmI technologies empower DSSs to assist in individual consumers rather than organizations.

In this paper, we coin the term *ambient decision support systems (ADSS)*, which refers to the DSSs that support individual daily activities in AmI environments. In contrast to the traditional PC-based interaction, ADSSs convey decision support information through the interaction with environments. Momentary decision making is aided with immediate feedback, and ADSSs also train the user in a long-term feedback loop. In order to identify potential design issues in ADSS development, we developed four case studies upon an ADSS system framework. The system framework aims at supporting system developers to utilize AmI technologies to extend traditional DSSs. Based on the case studies, we identified three system design aspects corresponding to human factors in a decision making process: emotion, intention and attention. We discuss the dependency among the human factors, and clarify how they can be addressed in system implementation. As the main contribution of this paper, we illustrate the findings and experiments in the framework as a reference for future ADSS developers.

Acknowledgements

First of all, I would like to gratefully and sincerely thank my supervisor Prof. Tatsuo Nakajima, for his encouragement, guidance, support, stimulation, and patience to lead me to the completion of this dissertation. His strict, but warm mentorship always provided me with great ideas and inspiration to seek and explore the new direction of my research. It is an honor for me to study under his supervision and in this Distributed and Ubiquitous Computing Laboratory.

I am also deeply grateful to second readers, Prof. Yasuo Matsuyama and Prof. Naohisa Komatsu. Special thanks to the professors for valuable comments and advice.

Discussions and cooperation with my old colleagues helped me in this quite long and difficult journey. I would like to thank Prof. Kaori Fujinami, Dr. Hiroo Ishikawa, Dr. Fahim Kawsar, Nobuyoshi Nakamura, Tomoko Shoji, Ayako Takagi and all of my best friends who studied together during the start-up period of my researcher life.

I am also pleased to thank brilliant researchers in Nokia Research Center Tokyo for a great deal of valuable advice and guidance: Dr. Eigo Mori, Yoshiya Hirase, Hideki Ohhashi, Kiyotaka Takahashi, Toshihiro Kobayashi, Rin Nagaike and Naoto Kaneko. Thanks to the hot discussion with these geniuses, I have been able to proceed with my research with confidence.

This is also a great opportunity to express my gratitude to colleagues and friends in Finland. A part of this dissertation consists of successful collaboration with Dr. Antti Oulasvirta, Dr. Inger Ekman, Ilkka Kosunen, Kai Kuikkaniemi, and Dr. Lassi A. Liikkanen, from Helsinki Institute for Information Technology and Center for Knowledge and Innovation Research in Aalto University. They and my best friends who once studied in DCL, Aleksi Aalto and Mikko Polojärvi, helped me a lot during my one year overseas study.

I also would like to thank again recent laboratory colleagues, this dissertation would not have been possible without their help and effort: Dr. Hiroaki Kimura, Dr. Vili Lehdonvirta, Chihiro Takayama, Hitoshi Ito, Miyuki Shiraishi, Hayuru Soma, Yasuyuki Washio, Takahiro Iwata, Sota Matsuzawa, and Takahiro Shichinohe.

This research has been supported by research grants from Grants-in-Aid for Scientific Research, Core Research for Evolutional Science and Technology, and Global COE Program. I would also like to thank Funai Foundation for Information Technology for the overseas scholarship.

Lastly, I would love to show my best and heartfelt gratitude to my mother for raising me with deep and warm love. I also needed to realize great respect to my father again. I would sincerely dedicate this dissertation to him, with my deepest veneration and gratitude. Then, now I should apologize to Mariko, for leaving her alone in Japan during the overseas study. With love I thank her again for understanding and patience in the hard time, both my Finland stay and this dissertation have never been achieved without her affectionate support. I would also thank my dearest friends and all of those who supported my research work in any respects.

Contents

List of Figures

List of Tables

Abbreviations

ADSS	**A**mbient **D**ecision **S**upport **S**ystem
AI	**A**rtificial **I**ntelligence
AmI	**Am**bient **I**ntelligence
AR	**A**ugmented **R**eality
DBMS	**D**ata**B**ase **M**anagement **S**ystem
DGMS	**D**ialog **G**eneration and **M**anagement **S**ystem
DSS	**D**ecision **S**upport **S**ystem
DW	**D**ata **W**arehouse
EIS	**E**xecutive **I**nformation **S**ystem
GDSS	**G**roup **D**ecision **S**upport **S**ystem
GPS	**G**lobal **P**ositioning **S**ystem
ICT	**I**nformation and **C**ommunication **T**echnology
MBMS	**M**odel-**B**ase **M**anagement **S**ystem
MDS	**M**anagement **D**ecision **S**ystem
MIS	**M**anagement **I**nformation **S**ystem
ODSS	**O**rganizational **D**ecision **S**upport **S**ystem
OLAP	**O**n**L**ine **A**nalytical **P**rocessing
OR	**O**perations **R**esearch
SA	**S**ituation **A**wareness
SDS	**S**tructured **D**ecision **S**ystem
SRK	(Rasmussen's) **S**kills, **R**ules, **K**nowledge framework
TUI	**T**angible **U**ser **I**nterface

Chapter 1

Introduction

Life is the outcome of decisions. The will to live drives us to make decisions to build the better future. To live in a society, satisfy the desires, for any reason, anywhere anytime everyone makes decisions. Since the personal computing era came, our individual decision capability is enhanced by computation. Information and communication technologies (ICT) are indispensable to our daily lives, and at once anymore inseparable from the decision sequences. In this paper, we address that most of the present ICT service designs are still technology oriented, rather than involving decision support aspect as a primary factor. This chapter describes the background and motivation of our research, especially focusing on the notion of ambient intelligence (AmI). Then, an overview of the paper's structure is briefly illustrated at the end of this chapter.

1.1 Background and Research Motivation

In 1991, a brand-new grand vision was brought to computer science: the notion of *ubiquitous computing* was established by Weiser [125]. Whereas only the term "ubiquitous" started to widely spread over the world upon selfish interpretations, the philosophy behind the concept pointed out an important aspect on the computer-human interaction. The ubiquitous computing concept does not just aim at realizing "computation anywhere anytime" - either increasing the variety of personal devices, or establishing network infrastructures all around. The term *calm technology*, which Weiser re-coined the concept 5 years later, well indicates the most essential point in his message, disappearance beyond the awareness [126]. As we read and understand the information on a book without active attention, the information technology that does not require cognitive effort disappears into the environment. According to [125], *"such a disappearance is a fundamental consequence not of technology but of human psychology"*. The technologies, user interfaces, and devices become invisible from a cognitive perspective, upon the seamless transition of interaction between focal and peripheral areas.

During late 1990s and early 2000s, the ubiquitous computing concept had evolved to a more general paradigm by merging with the vision of social user interfaces. After originally coined in the Philips' internal workshop organized by Zelkha and Epstein, the term *ambient intelligence (AmI)* became widely recognized to share the technical issues and vision among researchers [124]. AmI aims at empowering people through the digital environment, which is aware of their presence and context. Moreover, the AmI environment is sensitive, adaptive, and responsive to people's needs and behavior. In the report to Information Society and Technology Advisory Group (ISTAG) in the European Commission, Ducatel *et al.* described the AmI concept as *"a vision of the Information Society where the emphasis is on greater user-friendliness, more efficient services support, user-empowerment, and support for human interactions. People are surrounded by intelligent intuitive interfaces that are embedded in all kinds of objects and an environment that is capable of recognizing and responding to the presence of different individuals in a seamless, unobtrusive and often invisible way"* [23]. The ISTAG report also identified five key technological requirements for AmI in 2010: very unobtrusive hardware, a seamless mobile/fixed communications infrastructure, dynamic and massively distributed device networks, natural feeling human interfaces, and dependability and security. This dissertation has been written throughout 2010 and thus it is the best time to review the proposed requirements.

Recently our daily lives have been surrounded and augmented by a variety of mobile and fixed devices. Based on the rapid progress of miniaturization trend, powerful processing power is implemented into small devices, such as smart phones. Furthermore, as represented by Apple Inc. products[1], the shape and size of mobile devices become diversified to adapt to a variety of scenes. Especially in advanced countries, high-speed broadband infrastructures are equipped and those devices are connected over both wired and wireless networks. Home appliances such as TV and game consoles are designed to coordinate with web services e.g., bring new user experiences by gaming with remote online players. The notion of augmented reality has been realized based on the advances of visual recognition technologies and small actuators (e.g., projectors). Sensor-based context recognition for primitive activities, such as walking and running, is getting commercialized in the market. Even though some of the requirements are still in the development phase, the AmI technologies have made steady progress for the past decade. However, most of the envisioned AmI services such as smart homes are not realized or actively used yet. One cause of the situation is that the intelligence is not as smart as the user expects. Automated system behavior and its service are often intrusive, annoying, and disruptive to the user's task. People are more smartly adapted to the context and decision environment than computers do, then the most of "useful" functionalities provided by AmI services are disposed.

We argue that most of current AmI services lack the understanding of decision support. As aforementioned, our life is built upon the plenty amount of decision making results, and useful ICT services more or less should involve the decision support aspect in their design. People

[1]Apple Inc.: `http://www.apple.com`

use technologies to enhance their decision capability, in order to shorten judgement time and collect large amount of information, for example. Decision support needs to well align with their cognitive activities as the original concept of ubiquitous computing proposed, otherwise it disrupts the flow of daily activities and the scheme of the user's mental model. Therefore, it is necessary to design the services with careful consideration of human decision making mechanism, rather than simply improve the smartness of environment. Decision support should be achieved through the communication and collaboration between the services and a user. If they successfully decrease the cognitive effort in a decision making process, their existence will fade away into the background of perception.

In this paper, we coin the term *ambient decision support systems (ADSS)* that refers to the decision support services in the AmI environment. In order to develop the common understanding among developers, we also propose a system framework for ADSS development. This framework aims at abstracting functions and components in ADSSs so that newly developed systems can be designed upon the decision support systems (DSS) perspective. Even existing AmI services also can be recomposed into the DSS architecture. More importantly, the framework refers to two psychological models to guide the developers to understand a human decision making process from the cognitive perspective. We believe that this attempt contributes to realize the truly helpful AmI environment and services. The next section explains the structure of this dissertation.

1.2 Structure of This Dissertation

Figure 1.1 illustrates the structure of this paper. In the next chapter, we explain how the notion of ADSS can be mapped into the DSS research history. Upon the explanation of traditional DSSs, we identify the characteristics and requirement of ADSSs. Then in Chapter 3, we propose an ADSS framework based on the theories that model a decision making process from a cognitive perspective. Following chapters introduce four ADSS case studies: *Cognitively lightweight interaction for mobile decision making* (Chapter 4), *Decision training with augmented traditional games* (Chapter 5), *Decision inducement with activity-based micro-incentives* (Chapter 6), and *Decision support by crowd knowledge aggregation* (Chapter 7). As a part of ADSS system infrastructure, the Citron framework is also introduced in Chapter 8. Each case study focuses on the different aspects of human factors and key technologies. Thus the experience and findings in the case studies are helpful to review the ADSS framework and identify practical issues from multiple aspects. In Chapter 9, we discuss the ADSS framework based on the case studies to summarize design issues. Lastly in Chapter 10, we conclude the paper with implicating future work.

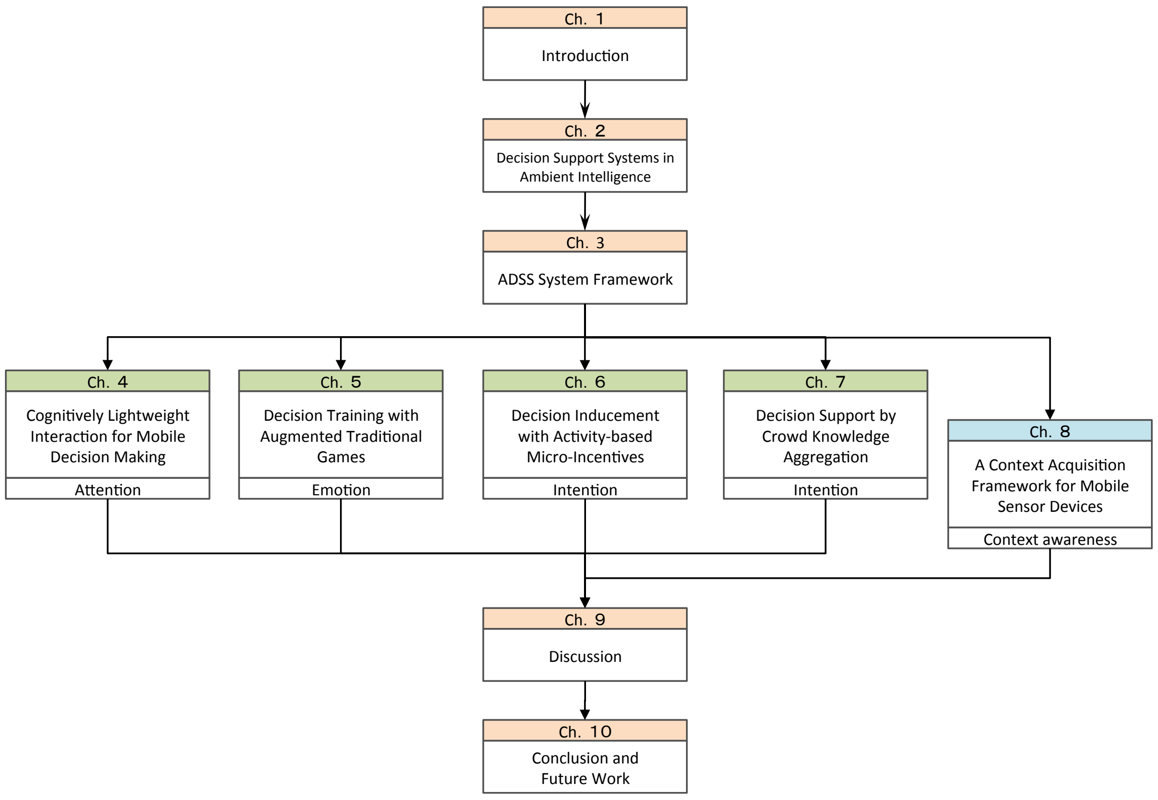

Ch. 1
Introduction
Ch. 2
Decision Support Systems in Ambient Intelligence
Ch. 3
ADSS System Framework
Ch. 4
Cognitively Lightweight Interaction for Mobile Decision Making
Attention
Ch. 5
Decision Training with Augmented Traditional Games
Emotion
Ch. 6
Decision Inducement with Activity-based Micro-Incentives
Intention
Ch. 7
Decision Support by Crowd Knowledge Aggregation
Intention
Ch. 8
A Context Acquisition Framework for Mobile Sensor Devices
Context awareness
Ch. 9
Discussion
Ch. 10
Conclusion and Future Work

Figure 1.1: Structure of this dissertation

Chapter 2

Decision Support Systems in Ambient Intelligence

This chapter explains where our research can be mapped onto the DSS history by introducing the background, traditional taxonomies and system architectures. Originally a DSS aims at supporting business decisions especially for managers in an organization. The progress of computing technologies have realized a variety of decision support styles, and as a result individual consumers are empowered to utilize a DSS for personal use in their daily lives. We also coin the term *Ambient Decision Support Systems (ADSS)*, with clarifying the characteristics that differentiates them from traditional DSSs.

2.1 The Brief History of DSSs

As the term *decision support systems* literally indicates, the concept of a DSS is extremely broad. Whereas Keen claimed that:

"there can be no definition of Decision Support Systems, only of Decision Support" [58],

according to Power, the notion of a DSS still remains a useful and inclusive term for many types of information systems that support human decision making [89]. In fact the definition of a DSS varies depending on the author's viewpoint and main interest [6, 22]. For example, in 1978, Keen and Scott Morton stated that:

"a DSS couples the intellectual resources of individuals with the capabilities of the computer to improve the quality of decisions. It is a computer-based support system for management decision makers who deal with semistructured problems" [59].

In 1992, Adelman defined a DSS as:

"interactive computer programs that utilize analytical methods, such as decision analysis, optimization algorithms, program scheduling routines, and so on, for developing models to help decision makers formulate alternatives, analyze their impacts, and interpret and select appropriate options for implementation" [2].

On the other hand, Poch *et al.* defined a DSS as:

"an intelligent information system that reduces the time in which decisions are made, and improves the consistency and quality of those decisions" [88].

According to Power, DSSs date back to 1960s [94]. Researchers started to study the use of computerized quantitative models to assist in human decision making and planning [31]. In 1964, Scott Morton first articulated the concept of a DSS in a discussion about his dissertation [72]. He worked on business planning support with management decision systems (MDS). At the same time, the development of powerful mainframe systems made management information systems (MIS) more practical and cost-effective. Early MISs mainly focused on providing managers with structured and periodic reports, but without interactivity [94]. Then around 1970, business journals started to publish articles on DSSs such as MDSs. For example, in 1971, Gorry and Scott Morton first used the term *decision support systems* in the Sloan Management Review article, and clearly defined the DSS concept:

"a DSS is an interactive computer based system that helps decision makers utilize data and models to solve unstructured problems" [37].

They integrated Anthony's categories of management activity [8] and Simon's description of decision types (Figure 2.1) [110]. Anthony categorized managerial activities within organizations into three types: strategic planning (executive decision regarding overall changes in the objectives of the organization, available resources, and policies), management control (resource management for efficient and effective use that guides the organization to goals), and operational control (the process of assuring specific tasks are carried out effectively and efficiently) [8, 106]. Operational control is concerned with tasks that relate the performance (such as manufacturing a specific part), whereas management control is most often concerned with people. On the other hand, Simon described decision problems as existing on a continuum from programmed (routine, repetitive, well structured, easily solved) to nonprogrammed (new, novel, ill-structured, difficult to solve) [106, 110]. Gorry and Scott Morton combined these Anthony's managerial activities and Simon's decision problem description, with replacing the labels *"programmed/nonprogrammed"* with *"structured/unstructured"*. As aforementioned, a DSS was defined as a computer system that deals with unstructured problems. A computer system could be developed to deal with structured problems, but usually a decision maker's judgment is brought to the unstructured part. Thus, in this case, the system was recognized as a structured decision system (SDS) rather than DSS.

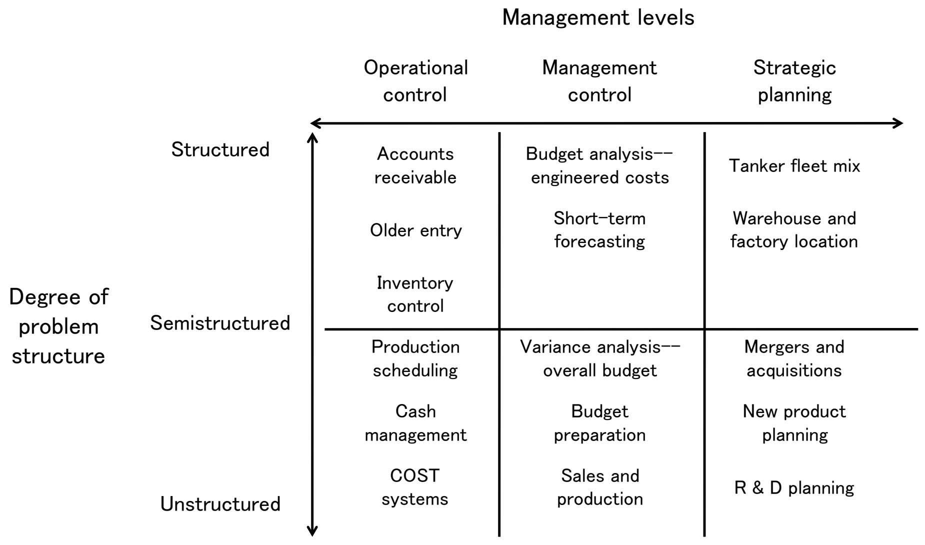


FIGURE 2.1: Gorry and Scott Morton's DSS grid

According to Keen and Scott Morton, the history of DSS started in 1950s [59]. They argued that two main areas of research were the origin of DSS concept: theoretical studies of operations research (OR) and decision making by Carnegie Institute of Technology, and technical work on interactive computing by Massachusetts Institute of Technology. These two researches were carried out independently, but gradually the DSS concept emerged on the top of them in the middle of 1970s. Anyhow, based on the classic DSS concept, a variety of specialized forms of DSSs were developed in the middle and late 1980s. For example, executive information systems (EIS) were designed to assist in senior executives with easy-to-use graphical interfaces. Group decision support systems (GDSS) aimed at supporting meetings and group work whereas classic DSSs mainly targeted a single decision maker. Organizational decision support systems (ODSS) also cover a group of people to support the goal of an organization. ODSSs provide mechanisms for ensuring that decisions are consistent with each employee and overall organization goals [122]. Beginning in around 1990, data warehouse (DW) and online analytical processing (OLAP) systems started to spread in the market. The progress of database technology realized rapid analysis of large volume and complex data. Data is repeatedly gathered from various sources, and stored into a database for further analytical use such as market research by managers or other business professionals.

During 1990s, personal computers drastically evolved with miniaturized processing units, graphically manipulatable user interfaces, and communication capability based on both wired and wireless networks with the TCP/IP protocol. The Internet/Intranet and web browsers brought new style of decision support, namely, web-based DSS [90]. A web-based DSS usually forms a server-client communication model and delivers decision support information to a user via a web browser. Web technologies can be used to implement any categories of DSSs that are explained in Section 2.2. Figure 2.2 illustrates aforementioned brief history of DSSs based on Keen and Scott Morton's argument, and positions our ADSS concept as an extension to traditional DSS research work.

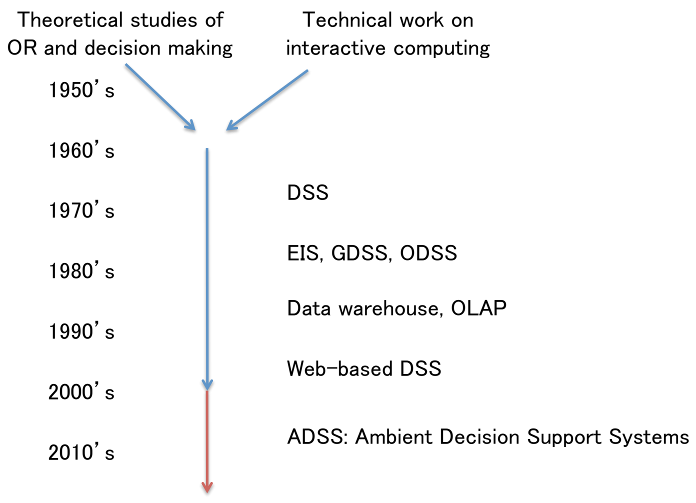

Theoretical studies of OR and decision making
Technical work on interactive computing
1950's
1960's
1970's
1980's
1990's
2000's
2010's
DSS
EIS, GDSS, ODSS
Data warehouse, OLAP
Web-based DSS
ADSS: Ambient Decision Support Systems

Figure 2.2: A brief history of DSSs and the position of ADSS research

During 2000s, the heterogeneity of computer devices increased in the market, for example smart phones and tablet size devices were popularized particularly in advanced nations. Moreover, ambient intelligence (AmI) environments are gradually realizing based on the maturity of wireless networks and sensing infrastructures. We argue that nowadays our daily lives cannot be separated from such technologies, and meanwhile the DSS concept should be revised to adapt to individual consumers' use in various contexts.

2.2 Traditional Decision Support Systems

In 1980, during the first wave of DSS study, Alter created a taxonomy based on an empirical study of 56 DSSs [6]. The taxonomy categorizes DSSs in terms of the generic operations it performs; and distinguishes between data-oriented and model-oriented types (Table 2.1). Alter's idea mainly focused on the output from DSSs and separated the characteristics of a decision problem from DSS classification.

However, as there has been no unified definition of a DSS, various taxonomies and classification approach have been proposed so far. One of the noteworthy work is Power's expanded framework in 2002 that was built on Alter and Sprague's frameworks (Table 2.2) [6, 112]. Power stated that *"Both managers and DSS designers need to understand categories or types of decision support systems so they can communicate better about what needs to be accomplished in supporting decision-makers. Researchers need a framework for categorizing DSS so that hypotheses and theories can be tested meaningfully"* [91–93]. The framework reconstructed Alter's

Table 2.1: Alter's seven types of DSSs (adapted from Power's articles [93, 94])

Data-oriented types	(a) **File drawer systems** that provide access to data items (*data retrieval*).
	(b) **Data analysis systems** that support the manipulation of data by computerized tools tailored to a specific task and setting or by more general tools and operators (*data retrieval*).
	(c) **Analysis information systems** that provide access to a series of decision-oriented databases and small models (*data analysis*).
Model-oriented types	(d) **Accounting models** that calculate the consequences of possible actions (*simulation*).
	(e) **Representational models** that estimates the consequences of actions on the basis of simulation models (*simulation*).
	(f) **Optimization models** that provide guidelines for action by generating an optimal solution consistent with a series of constraints (*suggestion*).
	(g) **Suggestion models** that perform the logical processing leading to a specific suggested decision for a fairly structured or well-understood task (*suggestion*).

seven categories into five primary dimensions, and specified other attributes such as target users with three secondary dimensions. Table 2.2 also indicates correspondence between Alter's DSS types and Power's expanded framework with an alphabetical index.

Table 2.2: Power's expanded DSS framework [93]. The alphabetical index represents correspondence to Alter's DSS types shown in Table 2.1

Dominant DSS component	**Target users**	**Purpose**	**Deployment/enabling technology**
Communications **Communications-driven DSS** (newly added)	Internal teams, now expanding to external partners	Conduct a meeting or help users collaborate	Web or client/server
Data base **Data-driven DSS** ((a)+(b)+(c))	Managers, staff, expanding to suppliers	Query a data warehouse, Monitor performance indicators	Mainframe, client/server, web
Document base **Document-driven DSS** (newly added)	Internal users, but the user group is expanding	Search web pages or find documents	Web or client/server
Knowledge base **Knowledge-driven DSS** (g)	Internal users, expanding to customers	Management advice or help structure decision processes	Client/server, web, stand-alone PC
Models **Model-driven DSS** ((d)+(e)+(f))	Managers and staff, expanding to customers	Crew scheduling, financial planning or decision analysis	Stand-alone PC, client/server or web

Communication-driven DSSs target multiple users' decision support, and thus it emphasizes communication and collaboration aspect of DSSs. This category was not included in Alter's classification, but advances in networking technologies realized such cooperative work with computers. Various technologies could be included in this category: GDSSs, electronic white boards, computer-based bulletin boards, and distributed collaborative environments.

Data-driven DSSs include three data-oriented DSS types in Alter's classification: file drawer systems, data analysis systems and analysis information systems. DWs, EISs and OLAP systems are also included in this category. A data-driven DSS puts a large database of structured data at the center of the system. For example, large collections of historical data are accessed by query and analyzed for future decisions.

Document-driven DSSs seamlessly integrate a variety of storage and provide an unified access to unstructured documents and web pages. For example, search engines are a powerful decision aiding tool supporting various contents retrieval from the web. Product specifications, corporate historical documents, and meeting minutes would be example document types in business context.

Knowledge-driven DSSs was originally termed as suggestion model DSS in Alter's classification. In Power's framework, this category mainly focuses on artificial intelligence (AI) knowledge base component. A knowledge-driven DSS has specialized problem-solving expertise that consists of knowledge about a particular domain and skill at solving decision problems. It also uses special heuristic models called inference engine, and suggests or recommend actions to decision makers based on processed rules.

Model-driven DSSs emphasize the model aspect of a DSS to support various specific purposes and application domains. Each DSS is designed for a specific set of purposes, and different models are needed accordingly. Thus very large databases are usually not needed for a model-driven DSS, and choosing an appropriate model is more important design issue.

As well as a variety of DSS definitions, different authors identify different components in a DSS [9, 20, 43, 83, 106, 111]. In 1982, Sprague and Carlson identified three fundamental components: (a) database management systems (DBMS), (b) model-base management system (MBMS), and (c) dialog generation and management systems (DGMS) [99, 113]. A DBMS stores large amount of data in logically structured format. It separates database management part from other components. A MBMS is an analogous component to DBMS: specific models in a DSS can be separated each other, and also from other system components. A MBMS processes data that are fetched from DBMS into decision support information. A DGMS handles interaction between a DSS and decision makers. As well as displaying suggestions, a DGMS should support intuitive model building throughout interaction with the user since decision problems are often unstructured.

In 1999, Haettenschwiler distinguished from another aspect with five components: (a) users with different roles or functions in the decision making process (decision maker, advisors, domain experts, system experts, data collectors), (b) a specific and definable decision context, (c) a target system describing the majority of the preferences, (d) a knowledge base made of external data sources, knowledge databases, working databases, data warehouses and meta-databases, mathematical models and methods, procedures, inference and search engines, administrative programs, and reporting systems, and (e) a working environment for the preparation, analysis, and documentation of decision alternatives [38]. In the same year, Marakas proposed a more generalized architecture with five DSS components: (a) the data management system, (b) the model management system, (c) the knowledge engine, (d) the user interface, and (e) the user(s) [70]. Then in 2002, Power identified four major components in a DSS: (a) the user interface, (b) the database, (c) the models and analytical tools, and (d) the DSS architecture and network [92].

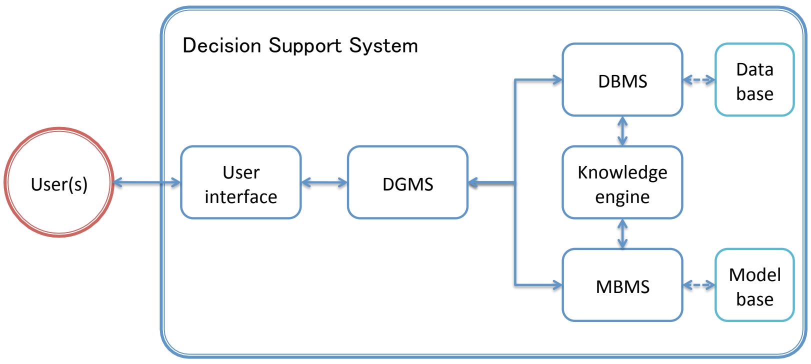


FIGURE 2.3: A conceptual DSS architecture based on [38, 70, 113]

As this research addresses the importance of human factor issues in decision making support, we illustrate a DSS architecture based on Marakas's work that recognizes the user(s) as a part of system (Figure 2.3). The user interface provides decision support information that is generated by the DGMS, while it also handles input from the user. The knowledge engine represents any intelligence in the DSS that realize reasoning and suggestion from previous outcomes. Such heuristics will be programmed by its designers or acquired by the DSS through repeated use.

In the next section, we describe the details of ADSSs and clarify how they differ from traditional DSSs.

2.3 Ambient Decision Support Systems

AmI consists of a variety of key technologies including embedded operating systems, context awareness, augmented reality, sensor networks, and etc. Miniaturized processing units, sensors and actuators are distributed into an environment and connected over the network. User interfaces appear anywhere around the user, with varying its shape including mobile devices and daily objects [57]. The AmI environment involves great potential to extend DSS application domains. As mentioned above, traditional DSSs have been developed particularly for business decision support, and correspondingly the main users are managers in organization.

As technologies are pervasively wove into daily lives, however, DSSs would also benefit general consumers and each individual.

An individual's decision support can be roughly divided into two types: decision aiding and decision training. In contrast to one time decision aiding, decision training aims at a decision maker's growth in a long-term feedback loop. Zachary and Ryder summarized main limitations in an individual's decision making and skill acquisition as shown in Table 2.3 [136]. They emphasized the importance to model a decision maker's cognitive process, and identified functional requirements from human centric perspective. For example, a decision maker would need to predict the outcome or trajectory of some external processes, in order to improve the decision performance. In decision training, the person may need practice in understanding the process and reasoning method for future prediction. DSSs should provide corresponding functions to assist in decision aiding and training.

Table 2.3: Summary of main limitations in decision making and skill acquisition (upper), and summary of feasible DSS functionality (lower). Both tables are adapted from [136].

Decision Making Difficulties and Problems	**Training/Skill Acquisition Difficulties and Problems**
process prediction	process understanding and practice
combining decision attributes	learning situations cues and case attributes
information organizing, access, monitoring	distinguishing important from irrelevant information
limits in visualizing/representing	testing/evolving representation
applying reasoning methods	acquiring and chunking reasoning strategies
attention management, identification of appropriate heuristic/domain-based knowledge	acquiring attentional strategy; developing metacognitive knowledge; mapping situational cues and case attributes to case-based heuristics and knowledge

Decision Aiding Functions	**Decision Training Functions**
process prediction	environment for practice/exploration
choice modeling	performance feedback and assessment
information management	problem query/data scaffolding
representational support	conceptual training
automated reasoning/interpretation	performance advice/tutoring/scaffolding
attention cueing/partitioning of problem space	metacognitive training/cognitive diagnosis

Ambient decision support systems are the DSSs that support individual daily activities in the AmI environment. Target activities could be either trivial or significant, such as decision support for dinner choice would be one example. As illustrated in Figure 2.1, decision problem in our daily lives can also be mapped onto the dimension of problem structure. Dinner menu planning is a repeated activity and it could be structured, unless extra factors need to be taken into account (e.g., have a guest for the dinner). On the other hand, there are unstructured problems that involve higher risks and they could affect one's life path. For example, usually people do not experience the entrance examinations to a university so many times. There are lots of uncertain factors in an unstructured problem and people try to make it clear as much as possible by collecting information (e.g., question trend, competitive rate). As Figure 2.1 indicates, strategy is often required to determine the future direction, and we need to make a big decision at a such turning point of the life. However, in this paper, we mainly consider more (semi-)structured and repeatable problems, rather than completely unstructured big decision problems. Below list summarizes the criteria of ADSSs that this paper mainly focuses on.

- ADSSs target individual consumers as a main user, and mostly support (semi-)structured decision problems in their daily activities.
- ADSSs convey decision support information through both the explicit and implicit interaction with environments.
- ADSSs aim at aiding momentary decision making with immediate feedback (or feedforward), but also training the user in a long-term feedback loop.

Whereas traditional DSSs assume to support a user with a conventional PC, ADSSs provide information in the manner that is seamlessly integrated into the environment. For example, projection is one useful approach to put digital annotation to the real world object. People can recognize the digital information that is directly superimposed onto the object, without effortfully switching attention to an extra display. AmI environments also actively identify the user and recognize the meanings of their behavior through continuous monitoring. Thus ADSSs tailor decision information based on the implicit adaptation to the user's request. On the other hand, instead of the such direct interaction with the environment, mobile and wearable devices can mediate by providing a personal user interface. For example, some services such as route navigation require detailed input for specifying destination, arrival time, preferred means of transportation and etc. In some cases, such a mobile interaction is still handy and helpful to precisely convey the user's intention to an ADSS. Thus, ADSSs should allow both explicit and implicit interaction to users.

Another distinguishable aspect of ADSSs is the stickiness of feedback. As people has started to interact with mobile phones throughout the day, nowadays services can convey information almost anywhere and anytime. The AmI concept augments the user's surrounding environment itself so that feedback can be provided in a variety of forms, even though mobile devices are not carried. As mentioned above, conventional mobile interaction requires explicit attention to the device. However, ADSSs directly provide feedback to the user's activity and its results. For example, if sensors and actuators are installed into a kitchen environment, an ADSS can support decisions in cooking activities. Traditionally, a recipe book or a corresponding mobile service is used to guide cooking process. In contrast, the cooking support system would implicitly recognize the cooking context with sensors and cameras, and spotlights the place where the user should pay attention. Such decision information could be either feedback to each action or the feedforward, which brings the vision of possible near future. In AmI environments, the timing and place that decision information appears are tightly linked to the user's activities.

As AmI technologies enable continuous user monitoring, ADSSs can provide periodical feedback to train decision capability through daily activities. For example, smart electrical meters transmit the electricity usage information so that the user can realize which activity consumes much energy. An ADSS can provide feedback to the power consuming activities such as keeping a refrigerator open, and provide tips towards energy efficient behavior. Traditionally, decision training was conducted only during the interaction period with PCs. In the AmI environment, ADSSs follow the user and provide feedback, even though they do not explicitly direct attention to the service. As far as aligned with the user's intention and will (e.g., want to decrease unnecessary power consumption), ADSSs can try to induce or even persuade them to change their behavior towards more desirable patterns.

Based on Figure 2.3, Figure 2.4 illustrates an extended ADSS architecture. The figure addresses the differences in a decision support environment.

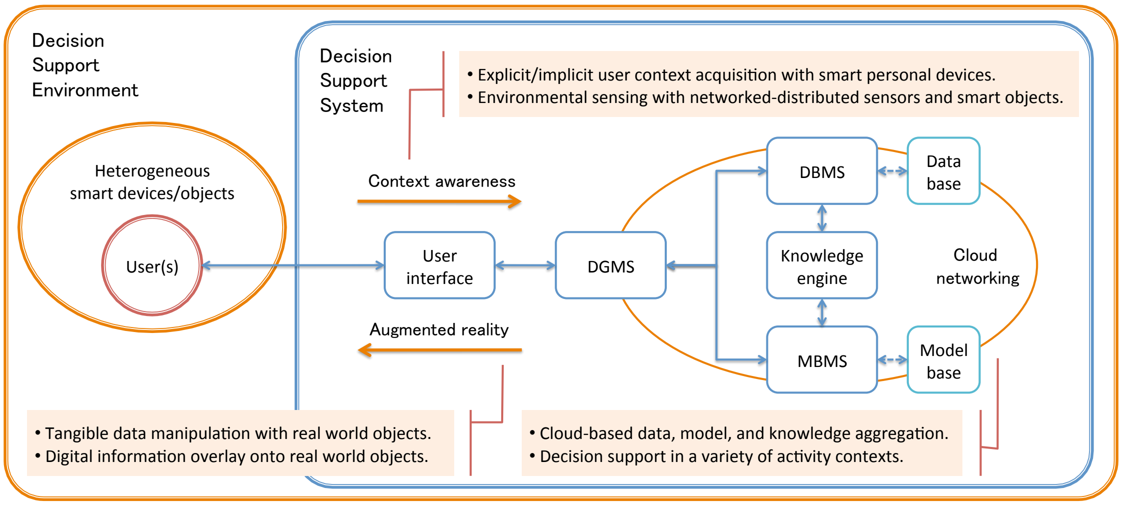

Decision Support Environment
Decision Support System
• Explicit/implicit user context acquisition with smart personal devices.
• Environmental sensing with networked-distributed sensors and smart objects.
Heterogeneous smart devices/objects
User(s)
Context awareness
User interface
DGMS
DBMS
Data base
Knowledge engine
Cloud networking
MBMS
Model base
Augmented reality
• Tangible data manipulation with real world objects.
• Digital information overlay onto real world objects.
• Cloud-based data, model, and knowledge aggregation.
• Decision support in a variety of activity contexts.

FIGURE 2.4: An extended ADSS architecture based on Figure 2.3

Heterogeneous smart devices and objects surround a user. In addition to traditional GUI-based decision support styles, the user can access an ADSS through the interaction with the environment. As the notion of augmented reality represents, data can be manipulated through physical interaction, and feedback can be directly provided onto real world objects [21]. Information for decision support also varies upon the increase of data source such as mobile and environmental sensors. For example, context information, such as location of the decision maker and their emotional states, is useful to structure a decision problem and provide appropriate support. Context acquisition could happen both explicitly and implicitly, thus it should be controllable by the user to disclose such information associated with privacy. Lastly, most components excluding the user interface can be remote and hidden from the decision maker. ADSS functionalities and data would be transferred beyond the cloud network so that the accessibility and persistency of decision support service can be improved.

Since any situations in AmI environments could be a decision support target, an ADSS takes any DSS types in Table 2.2. As the web technology is cited as a deployment/enabling technology across all categories, nowadays a decision maker can interact with a DSS through a unified user interface. As the complexity of a decision environment increases, however, it becomes more important to understand cognitive processes and design requirements as shown in Table 2.3 [111]. In the next chapter, we propose a system framework for ADSS development. The framework refers two psychological models that illustrate decision processes from the cognitive perspective.

Chapter 3

ADSS System Framework

In Chapter 2, we introduced the notion of ADSS and made a comparison with traditional DSSs. Whereas the concept envisions the evolution of DSS, the definition still covers a wide range of application domains. Indeed most user-oriented services would involve the decision support aspect and thus they can be regarded as a DSS. The AmI environment has been gradually but steadily realized, and the early stage of ADSS has already came to the market. For example, today's smart phones come with a variety of sensors such as global positioning systems (GPS) and accelerometers. Mobile services provide the user with the information that would be helpful for decision making, based on the context information detected by the sensors. The services still could develop as AmI technologies become mature. For example, mobile pedestrian navigation service would become more useful and acceptable for the people who are not familiar with technology, if the interaction happens directly with the surrounding environment without any mobile devices. However, currently the ADSS aspect is not sufficiently addressed in the system design. We argue that one reason is the lack of conceptual framework that illustrates the DSS core components with AmI technologies. Moreover, it should remind the developer to understand each individual's cognitive process at decision moment.

In this chapter, we propose an ADSS system framework. The framework aims at supporting system developers to design an AmI service from the decision support aspect. It also assists in the decomposition of existing services so that the components to which AmI technologies should be applied can be identified. As explained in Figure 2.4, each components, such as user interface and DGMS, are extended with a variety of functions and devices. It is important to consider how the system incorporates the ubiquity of sensors and actuators into the design. The framework is useful to recompose the existing system and re-design the architecture from the DSS perspective so that AmI services' decision support functions can be extended by traditional DSS techniques and design principles. Moreover, the framework also accentuates the need for understanding of human decision making process from the cognitive perspective. The processes should be properly modeled as a component so that the system behavior can continuously align with the user's mental model. System developers can expect that the existence of ADSSs fades out from the user's focal attention upon the successful alignment. The framework is designed based on traditional DSS components shown in Figure 2.3.

3.1 Framework Overview

Figure 3.1 illustrates the overview of our ADSS framework. The framework extends the typical traditional DSS architecture (Figure 2.3) with a variety of AmI technologies. For example, the user interface component involves newly added interaction techniques, such as augmented reality and mobile interaction. As mentioned in Section 2.3, ADSSs convey decision information through the interaction with environments. System developers should firstly consider how the decision support happens in an ambient way, and then select the interaction technique that suits the service. For example, if the ADSS does not accept any explicit input from the user and just provide feedback to a particular action, mobile devices are probably not necessary. In Chapter 5, we introduce the series of augmented traditional games. They detect game events by monitoring relevant context information with sensors. Then, they provide feedback onto the game items so that players can recognize the information without splitting their attention. Such a passive monitoring and simple feedback combination would directly superimpose the feedback towards real-world object, or change the smart object's behavior. Then the user need not worry about troublesome setup (e.g., establish the connection between mobile device and service). The semantics of feedback is also easy to understand, since it immediately appears just after the user's action (e.g., put Go stones onto the board).

Below explains the detail of extended components.

User interface: To improve the mobility of users, user interface is extended with mobile interaction and augmented reality technologies. Mobile devices such as mobile phones can be used to send any detail requests and information to the services in the surrounding environment. The display size is small so that the services also can provide feedback to individuals without disclosing to other users. As smart phones equip sensors, gesture-based interaction is possible as well as traditional GUI-based interaction. The sensors also enable continuous user monitoring: mobile phones are carried by the owner throughout a day, thus the services can expect to extract the user's context information from the device. Mobile devices are a handy control point of ADSSs and can be used for both explicit and implicit interaction.

Another technology is AR (augmented reality) and it composes of TUI (tangible user interface)-based interaction and information overlay. AR is one of core technologies that materialize the interaction with environments. Objects in the real world become able to show information onto

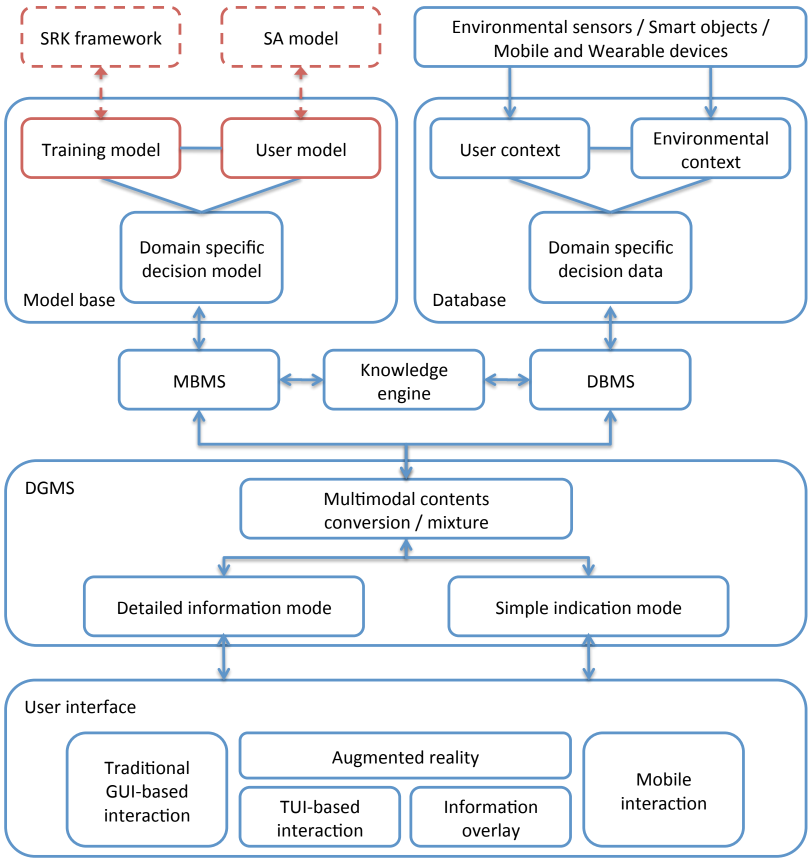


FIGURE 3.1: Overview of the ADSS system framework. Two cognitive decision making models accompany the framework: Endsley's situation awareness (SA) model [27] and Rasmussen's skills, rules, and knowledge (SRK) framework [96].

them, and also allow the user to manipulate digital data through physical interaction. AR deploys mobile devices if necessary, as mobile AR is often used for location-based information tagging services. The user can see the information linked with specific location through the mobile display. In contrast, it is also possible to conduct interaction in a device-free manner with the sensors and actuators embedded in the environment. For example, hidden cameras for gesture recognition and large display projected onto the wall would bring the sense of direct interaction with the environment.

DGMS: The framework addresses the cognitive resource limitation issue since the user often needs to handle multiple tasks in the AmI environment. In contrast to traditional DSS that requires focal attention to a stationary display, AmI environment is highly dynamic and context changes every moment. Thus ADSS should not load the user with the large amount of unstructured information, otherwise enough cognitive resource cannot be allocated for other tasks. The DGMS component in ADSS extends dialog management function in terms of the representation format of decision support information. Modality, such as visual and auditory, is an important attribute of information as well as its data size. The mobile environment often requires visual attention to maintain the comprehension of current situation (i.e., situation awareness, explained in Section 3.2.1), thus in some cases other modalities, such as auditory and haptics, are more easily recognized.

The DGMS component converts decision support information into other types of format (e.g., text to speech) according to the user context that is managed in the database. The volume of information is also an important factor to enhance the contents' adaptability, as the AmI environment deploys a variety of devices. For example, ambient displays are the notification devices that convey information in an inconspicuous way. In order to adapt to the ambient displays, decision support information should be also represented in a simplified manner in addition to the original detailed format. The DGMS component prepares two modes for detailed and simple interaction modes, but there could be even more to support the continuum with fine granularity.

Database: The database in ADSS specifies two different context information (i.e., user and environment) as primary data in addition to conventional domain specific decision data. ADSS should decrease the user's effort to specify the decision problem by inferring current situation from sensor information. For example, pedestrians often want to know the route to their destination from current location. A mobile navigation service should suggest or even preliminary input the current location detected by GPS so that interaction time and step can be shortened. As explained with the DGMS component, the context information is also used for the adaptation of decision support information. We will explain more detail of the contents' simplification, multimodality, and adaptation in Chapter 4.

Model base: Two cognitive process models of human decision making are emphasized in the model base, training model and user model. As mentioned above, it is important for ADSSs to focus on each individuals and design the system based on the understanding of decision making process from the cognitive perspective. The two models correspond to main usage of ADSSs explained in the previous chapter, decision aiding and training.

The user model describes the mechanism how people arrive at the answer of each decision problem, and what factors affect the entire process. Decision aiding is supported mainly by providing decision support information including feedback and feedforward. The decision support information could be simply a data source, such as the map of present location and current stock price of a company. These information is often tightly associated with the fickle interest, which changes as the user's context changes. Thus an ADSS should provide temporal, but realtime information access anytime they want. Feedback is another information type that enhances the perception phase. For example, the map on a mobile phone should follow the user's move so that the sense of current position can be kept updated in their mental model. In contrast, the feedforward enhances the projection phase by showing possible futures that are automatically calculated based on the current situation. For example, a mobile navigation service suggests multiple route options with showing estimated time to the destination. Then the user can develop the awareness of current situation with more accurate vision of the future. Based on the model, decision can be aided by reinforcing a particular stage or tailoring the decision support information to complement the lack of cognitive functions.

On the other hand, the training model mainly focuses on long-term decision support. Decision training is achieved by repeated use (i.e., decision aiding) of ADSS. In addition to the one-time decision aiding, training aims at developing individual cognitive factors, such as information processing mechanisms and automaticity. The model describes how the user builds up skill based on their experience so that ADSS can change the behavior according to their skill level. A variety of decision problems happen simultaneously in the AmI environment and some of them are not well structured. With fragmented attention, it will take longer time to acquire skill and expertise knowledge. Thus decision training program should be designed with consideration of step-by-step behavior change. For example, a running support service assists in choosing training routes (decision aiding). Moreover, it is important to cultivate the sense of motor control and mental strength for longer running (decision training). To achieve this, the service should iteratively set the goal higher with providing meta cognitive cues, such as consumed calorie in numerical format presentation. Since limited resources can be allocated during the training process, decision information should require lower cognitive effort. More importantly, it should motivate the user to aim for the next training, in order to involve them into a longer feedback loop.

In the next section, we introduce two decision theories referred in the framework: Endsley's *situation awareness* and Rasmussen's *SRK (skills, rules, and knowledge) framework.*

3.2 Decision Making Models from the Cognitive Perspective

3.2.1 Situation Awareness

Situation awareness is a human ability to be aware of environmental situation for achieving a task. According to Endsley, the term is defined as *"the perception of elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future"* [25]. The concept of situation awareness mainly focuses on a decision maker's cognitive process behind an action. Such theoretical framework of human decision making would identify human factor issues and assist in developing better decision support systems [46]. In 1995, Endsley illustrated the model of situation awareness with three different cognitive levels (Figure 3.2) [26–28].

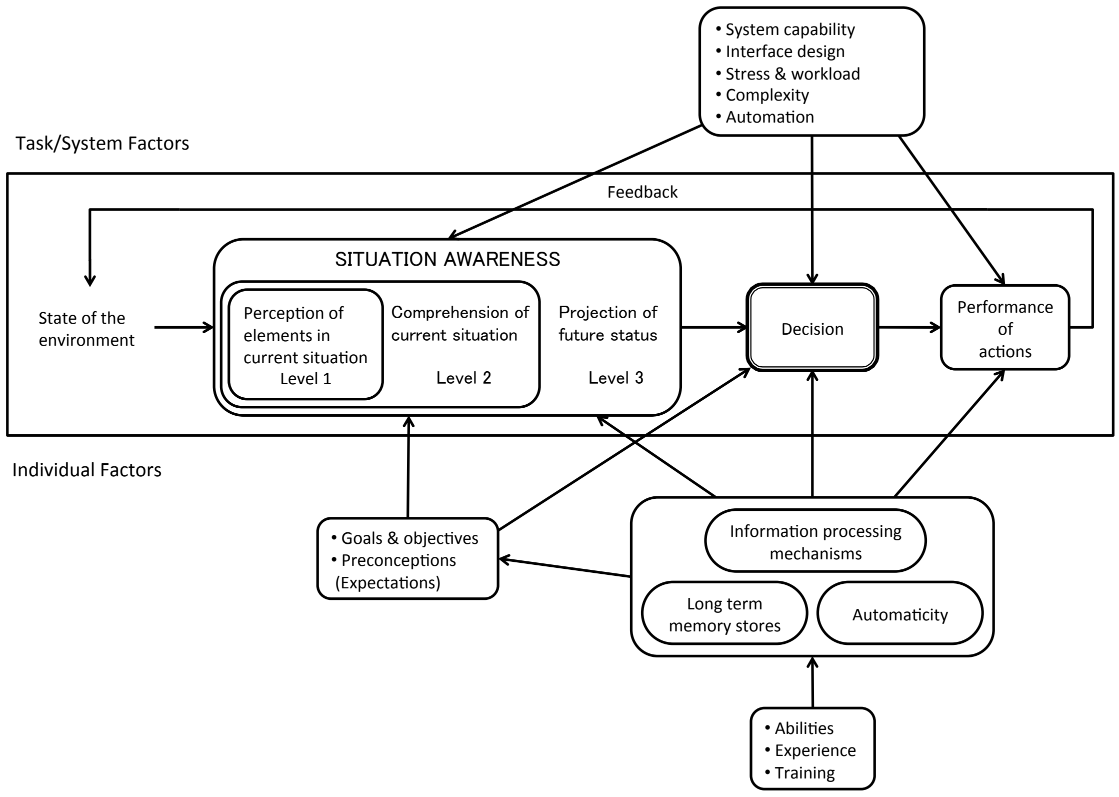


FIGURE 3.2: Model of situation awareness in dynamic decision making, adapted from [27]

Level 1. Perception: This perception phase is the fundamental part of any decision making and situation awareness. Important cues for recognizing environmental elements, such as people and objects, are perceived through sensory organs. This level involves the processes of monitoring, cue detection, and simple recognition.

Level 2. Comprehension: In the next comprehension phase, gathered cognitive cues are processed into more meaningful information. The entire process consists of pattern recognition, interpretation, and evaluation. Then collected information is used to develop a comprehensive image of the world, or it could be just a personally interested part of the world.

Level 3. Projection: The last phase in situation awareness development is projection the future. This phase is achieved through knowledge and comprehension of the current status. Based on them, it could be extrapolated and determined how a decision would affect future states.

Endsley also emphasized that several variables, such as individual factors, could influence the development and maintenance of situation awareness. For example, stronger motivation towards objectives would direct attention to the information especially relevant to the goal. On the other hand, other information could be filtered out by cognitive bias, even though they are also important to achieve a task. The volume of available cognitive resources, such as working memory, determines the quality of situation awareness [36]. However, it depends on individuals and expertness. As explained in the next section, an experienced decision maker can perform a task without allocating a large amount of resources.

Situation awareness also involves the temporal aspects: time, space, and the dynamic nature of real-world situations. Since situation awareness is a dynamically changing construct, time awareness is an important component to extrapolate the impact of decisions in the near future. The sense of available time also affects decision making process. Limited time would elicit mental pressure and stress the decision maker. Events relevant to the decision could be not controllable and happen as the time passes. Thus the decision maker needs to have comprehensive sense of time, especially for both recognition (level 2) and projection (level 3) phases.

As well as time, space is an elemental factor of activities and events that are relevant to decision making. The degree of interest towards particular events would be determined according to the physical distance. Moreover, a decision strategy would be selected based on spatial relationship between objects. With an ADSS, such spatial sense is augmented and physically remote information can be instantly presented to a decision maker. The last aspect, the dynamic nature of real-world situations emphasizes that the person's situation awareness needs to adapt to the changing environment, since it could be instantly outdated. Particularly in a highly dynamic environment, such as driving situation, a decision maker needs to change the cognitive strategy to keep the quality of situation awareness.

3.2.2 Skills, Rules, and Knowledge Framework

As shown in Figure 3.2, automaticity is a cognitive mechanism developed with experience and it influences situation awareness [28]. With sufficiently repeated experience and well understanding of the environment, a decision requires lower attention but provides good performance. For example, there are lots of decision points on the way to school, such as a subway gate to exit and where on a platform to wait for a train. In a couple of first months, a decision requires higher attention and elaboration. However, as the process gets routine, a decision becomes made unconsciously and other tasks can acquire cognitive resources. In 1983, Rasmussen proposed a model to illustrate this automaticity with identifying three behavior modes: skill-based, knowledge-based, and rule-based behavior. [96, 97]. Every behavior starts from perceiving sensory input, but the degree of consciousness is different between the modes.

Skill-based behavior: This skill-based behavior mode provides smooth and unconscious actions such as a reflex to a certain stimulus. A particular behavior pattern is well shaped based on practices and experiences, and often does not require special attention or conscious control. Thus the volume of cognitive resource consumption is relatively lower than other behavior modes.

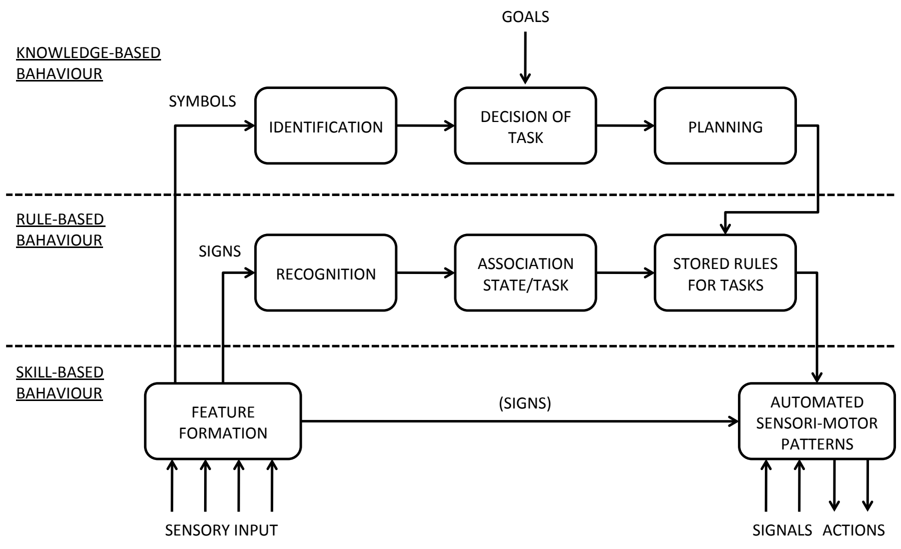


FIGURE 3.3: Rasmussen's SRK framework adapted from [96]

Rule-based behavior: In this rule-based behavior mode, the entire process is not completely automated as the skill-based behavior mode and still requires conscious attention and recognition of the environment before any actions start. This behavior mode uses explicit know-how and rules such as a cookbook recipe for decision making, which are stored based on the previous experiences.

Knowledge-based behavior: This knowledge-based behavior requires the highest attention and elaboration for problem solving. Especially this mode is used to deal with an unfamiliar problem that is not well structured in the decision maker's mental model. Typically the problem itself needs to be identified at the early stage of information processing, and afterwards possible plans are considered in a trial and error feedback loop.

In fact, there is no clear distinction between three behavior modes, and a decision can be placed on a continuum over two extreme representatives (i.e., knowledge-based and skill-based) described in Table 3.1. Reason addressed the types of human error based on the Rasmussen's SRK framework in [98]. For example, the knowledge-based mode directs higher attention and consciousness to a task, but a beginner decision maker tends to miss the awareness of consequences due to lack of experiences. On the other hand, skill-based unconscious action could be inappropriately applied to a problem, since such a heuristic approach often involves cognitive bias on problem identification.

As highly practiced with an sufficient repetition of task achievement, the decision maker's behavior changes from the knowledge-based mode to the rules-based one, and finally reaches the skills-based mode. Rasmussen illustrated this transition process as a step ladder with identifying cognitive workflow (Figure A.3). There are some divergences and the horizontally across

Table 3.1: Modes of interacting with the world (adapted from [24], the table based on the Reason's article [98])

Knowledge-Based Mode **Conscious**	**Skill-Based Mode** **Automatic**
Unskilled or occasional user	Skilled, regular user
Novel environment	Familiar environment
Slow	Fast
Effortful	Effortless
Requires considerable feedback	Requires little feedback
Causes of error: - Overload - Manual variability - Lack of knowledge of modes of use - Lack of awareness of consequences	Causes of error: - Strong habit intrusions - Frequently invoked rule used inappropriately - Situational changes that do not trigger the need to change habits

stereotypical shortcuts enable to perform a quick action, instead of going upwards to higher elaboration modes.

3.3 Summary of Case Studies

Upon the framework, we developed four ADSS case studies to identify practical issues and promote further discussion. This section summarizes research context and focus points in each case study.

Cognitively Lightweight Interaction for Mobile Decision Making (Chapter 4): Current mobile interaction is not well designed with considering mobility. Usability of a mobile service is degraded while on the move, since users can not pay enough attention to the service in such a dynamic and complicated mobile context. In this chapter, we propose the mobile service design framework, which improves the mobility by decreasing the user's cognitive load. Our approach provides two interaction modes (i.e. simple interaction mode and normal interaction mode) to mobile services so that the user can retrieve important information with less attention. Moreover, the service's events are simplified to support several modalities, and thus the user can be notified in the most suitable way according to the situation. In order to evaluate the feasibility of our approach through field experiments, we have developed a pedestrian navigation service as a part of the framework. The results showed that the simple interaction mode successfully decreased the user's attention to the service.

Decision Training with Augmented Traditional Games (Chapter 5): Traditional game augmentation aims at adding new value and playful features to a traditional game with keeping its original look-and-feel. Players can concentrate on playing the game as usual, without paying extra attention to the service. Context monitoring enables to automatically record game events so that players can review a gaming process, strategy and excited scenes after game finishes, for example. Multimodal feedback also provides supporting information to understand the mechanism of games e.g., visualize threatened territories where beginner players tend to fail to notice. Moreover, players can dynamically and seamlessly turn on and off such pervasive computing features while gaming. In this chapter, we develop the concept of augmented traditional games with two case studies: Augmented Go and EmoPoker. We also report the preliminary user study results and identify design issues in augmented traditional game development.

Decision Inducement with Activity-based Micro-Incentives (Chapter 6): Economic incentives are a powerful way of shaping consumer behavior towards more commercially efficient and environmentally sustainable patterns. In this chapter, we explore the idea of combining pervasive computing techniques with electronic payment systems to create activity-based micro-incentives. Users who consume additional resources by e.g., occupying an air-conditioned space instead of a normal space are levied additional micro-payments. In an alternative approach, consumers who choose to save resources are rewarded with micro-rebates off the price of a service. As a result, the cost of using a service corresponds more closely with the resources used, leading market mechanisms to allocate resources efficiently. A key challenge is designing incentive mechanisms that alter consumer behavior in the desired fashion. We introduce four incentive models, and present evaluation results suggesting that consumers make different decisions depending on which model is used.

Decision Support by Crowd Knowledge Aggregation (Chapter 7): It is expected that public displays compensate the shortcomings of mobile web search. Public displays have a big advantage in its large display size and touch-based interaction for multi-player's use, but sometimes end-users such as elderly people do not have enough skill to make sufficient queries that derive interested information from the Internet. In this chapter, we point out an important aspect of public displays: an interaction medium in social communication. Since the public display users are expected to have similar interests about the local area, search queries and results could be reused in a future search. Our approach aims at assisting end-users' point-of-interests search by providing the local contents, which are composed of the knowledge mashed up from web services and local search history. As a proof of concept, we developed a map-based tourist navigation service and performed preliminary experiments. In the experiments, we elaborately analyzed subjects' interaction process and figured out design issues for further improvements, such as search history presentation.

Following Chapter 8 introduces a software framework for context acquisition in AmI environments, then in Chapter 9, we will discuss further design issues acquired from the case studies.

Chapter 4

Cognitively Lightweight Interaction for Mobile Decision Making

4.1 Background

The rapid progress of mobile computing in this decade has drastically improved user experience on mobile services [60]. Today we have smart software keyboards on touch screens, high resolution displays and reliable networks for mobile devices. The growth and maturity of the Internet have accelerated the progress - various attractive services and contents provide rich user experiences to mobile users. Moreover, context-awareness (e.g. location-awareness with GPS, posture-awareness with 3D accelerometer) has partially been realized with sensor equipped devices [133].

However, the mobile services still frustrate mobile users while on the move. To fix the position of a small mobile user interface, users are forced to stop walking and pause frequently in order to access information. Moreover, mobile users are often situated in multitask contexts [117] and the user's cognitive performance would be drastically degraded when frequent interrupts and task switching occur. Even though problems of mobile interaction have been addressed in previous research [50], the mobile service design is still pursuing desktop-miniaturization trend and has not been adapted to the real mobile computing environment. Most of the services are designed on the assumption that they are used in stationary situations, so *mobility* becomes restricted as a consequence.

In this chapter, we propose a dual-mode approach to mobile service design. The dual-mode approach provides a simplified interaction style (named *simple interaction mode*) to a mobile service, in addition to a conventional user interface (named *normal interaction mode*). Our approach aims to decrease the user's cognitive load with the simple interaction mode so that he/she can retrieve important information with less attention while on the move. While some of the research activities aim to realize unobtrusive user interfaces for moving users [100], their advantages from conventional user interfaces have not been evaluated sufficiently. Therefore, we prototyped a mobile pedestrian navigation service and performed field experiments in order to clarify feasibility of our approach. Below list summarizes main contributions of our work:

- Design issues of mobile services were discussed and identified from a human factor aspect.
- The advantage of the dual-mode approach was then evaluated. The simple interaction mode successfully decreased the user's cognitive load to the service in the field experiments, compared to a conventional user interface. Also, the dual-mode approach enabled all users to complete tasks, even though some users could complete only with the simple interaction mode.
- Valuable findings for further improvements are finally summarized from the experiments and feedbacks.

In the next section, we focus on the cognitive perspective and discuss how usability degradation is caused in the mobile computing environment. Also, related work are introduced in Section 4.3. In Section 4.4, we propose a mobile service design framework based on the discussion in Section 4.2. A prototyped mobile pedestrian navigation service is introduced in Section 4.5, and experimental results of the fieldwork is shown in Section 4.6. In Section 4.7, we discuss future directions based on some findings and given feedbacks.

4.2 Usability Degradation while on the Move

User attention is one of the important factors to design user interfaces. Fine usability allows users to perform a task with less attention. For example, if a mouse cursor creeps by itself or is not calibrated appropriately, the user becomes frustrated as he/she must take care of fixing the position while manipulating. User interfaces should require minimum attention to allow the user to concentrate on the task (e.g. elaborate sentences in a mail, browse web sites).

Since mobile devices are not statically placed on a desk as ordinary desktop PCs, application scenarios vary tremendously. Mobile users and their attention could be affected by more kinds of factors than with desktop PC environments. In [50], Johnson presented three scenarios and addressed problems of usability caused by mobility. He focused on the complexity of mobile users' activities and pointed out the difficulties of mobile-world modeling in interaction design. For example, the user using a mobile guide service would be in a multitask environment and how much he/she can pay attention to the interface changes according to the tasks (e.g. crossing a busy intersection, browsing dresses through glass windows). Mobility breaks the traditional interaction model and degrades the usability of mobile services.

Current mobile service design is too attention-consuming for moving users to perform their tasks. For example, most of the existing researches aim to improve information retrieval efficiency by changing the layout and size of user interface components [13]. They assume the device to be operated in stationary situations (e.g. “sitting on a chair”, “standing on the street”), and thus it needs to keep the user’s attention all the time. They are not truly helpful in real mobile environments, since such situations are ideal and most of the time mobile users are performing action.

We have analyzed how the user’s attention is affected in the mobile computing environment, and identified two important issues: *Situational Disabilities* and *Fragmentation of Attention*. Below sections give detailed explanation about each issue.

4.2.1 Situational Disabilities

Due to the user’s postural and environmental changes, sometimes the physical condition makes it hard to perform interaction. Usually, users can set up or adjust the working environment in stationary situations (e.g. adjust light position in a room). However, in mobile computing scenarios, users move around and the surrounding environment dynamically changes accordingly. For instance, the display of a mobile device is frequently shaken in walking motion and text messages on the display can not be recognized well from the user’s eye. Also, ring tone notification from a cell phone can not be heard in noisy places. In such situations, users have to pay more attention to sense the information or to catch missed information.

Regarding visual tasks, Mustonen et al. studied legibility of texts on a cell phone display while walking [77]. They found that walking condition affects processing speed significantly in text reading tasks and the performance suffers from increasing walking speed. Also, the walking effects differ between pseudo(random)-text and real(normal)-text cases. Mizobuchi et al. studied the effect of key size on text entry on a handheld device while walking in [73]. They found that text input speed while standing is faster than in walking situations, even though they did not find any evidence that walking speed can be used as a way of assessing the difficulty.

4.2.2 Fragmentation of Attention

Mobile users are often placed into a multitask environment, because the moving activity is a task by itself. Unlike stationary situations, users have to interact with mobile devices while performing their main tasks (e.g. move their legs to walk and look around to evade roadblocks). For instance, some mobile services (e.g. route navigation service) aim to interactively support the user on the move, but he/she can not or should not look at the display carefully. Also, mobile services tend to interfere with main tasks, since users can access information or can be accessed by others anytime and anywhere. When the user is performing prior tasks, the degree of attention to the service can be decreased due to the capacity limitation and frequent interrupt handling.

In [117], Tamminen et al. monitored human actions in social life and discussed how mobile contexts are characterized for context-aware interaction design. They found that navigating through an urban environment requires paying constant attention to surroundings and limits attentional resources available for interacting with a device. Also, they pointed out the fluctuation in importance of time and place that is called *temporal tensions* affects task scheduling strategy for occupying the user's attention. When people are hastened, they have to perform multiple tasks more or less simultaneously. The priority of the tasks changes and some of the pre-scheduled tasks become impossible at that moment. Instead, they start to direct their attention to space and time in order to perform urgent tasks.

As shown above, these two issues are related to each other and not clearly separated. Even though these issues are not major concerns in stationary situations, mobile services should be designed to handle them appropriately in order to provide fine usability. In the next section, missing points in related work are discussed, and then we introduce our mobile service design framework in Section 4.4.

4.3 Related Work

Kristoffersen and Ljungberg are the early researchers who addressed shortcomings of traditional mobile user interfaces [60]. They argued that direct manipulation interaction style is unsuitable for mobile work contexts, since small keyboards and screens require a high level of visual attention. Therefore, they proposed MOTILE interaction system for operating mobile devices. MOTILE requires less or no visual attention (e.g. rely on only four buttons for user input and audio output for feedback) so that the user can concentrate on performing main tasks.

Their work well pointed out practical issues of mobile interaction. However, they mainly focus on decreasing user's cognitive load by replacing visual modalities with audio. As discussed in Section 4.2.1, situational disability is a possible problem for all modalities, so appropriate modalities should be selected according to the situation. Furthermore, how MOTILE has an advantage over conventional user interfaces was not sufficiently evaluated in the paper.

In [82], Pascoe et al. identified four requirements in extremely mobile and dynamic work places: $Dynamic\ User\ Configuration$, $Limited\ Attention\ Capacity$, $High-Speed\ Interaction$, and $Context\ Dependency$. They proposed Minimal Attention User Interface (MAUI) for fieldwork in Kenya. MAUI realized one-handed operation to allow the user to keep watching animals while recording logs on a mobile device. Moreover, context-awareness supports instant input by automatically filling fields (e.g. put GPS location information to the "recording point" field).

In their work, the user's input was mainly discussed rather than output, since main tasks were animal observation and data recording. MAUI was designed based on modeled operation sequence of tasks so that the user can manipulate the device with eyes-free. In this sense, more or less the user has to keep attention to the service, but it is difficult in most of cases due to the fragmentation of attention as discussed in Section 4.2.2. Users should be able to distract the attention and be notified when interesting events occur.

Sawhney and Schmandt presented their work, Nomadic Radio in [100]. They implemented audio interface on a wearable device to allow the user to access information services by speech and audio while on the move. Scalable auditory presentation mechanism allows for change in abstraction level of incoming messages so that Nomadic Radio can provide enough information to the user with minimum interruption. Also, Nomadic Radio can catch the user's attention or interests to a message by monitoring actions to adapt notification weights.

However, in some cases, interaction on single modality requires more attention and cognitive load to users. For example, with only audio interaction, sometimes the user loses where him/herself in a message space. Visual representation is required to understand currently pointing message. In addition to the issue, the single modality approach is susceptible to the situational disabilities. Therefore, multimodal interaction style is needed to realize robust interaction.

4.4 Design Issues

In Section 4.2, we have pointed out that the user's attention to mobile services is not stable in a mobile computing environment. Therefore, we propose a mobile interaction design principle as follows:

Keep it simple, to not require too much of the user's attention while on the move.

Our approach aims to minimize interaction cost (i.e. required attention) by simplifying the interaction method and maximizing the benefit of mobility. While moving, users have to handle multiple information and events, so they can not or should not actively interact with mobile services. On the other hand, users should be interrupted when important events are monitored by the service. For example, a pedestrian navigation service should guide the user to the destination without frequent manipulation. Once the destination is set, the users should be freed from operation and concentrate on walking, enjoying the sight, chatting with friends, and

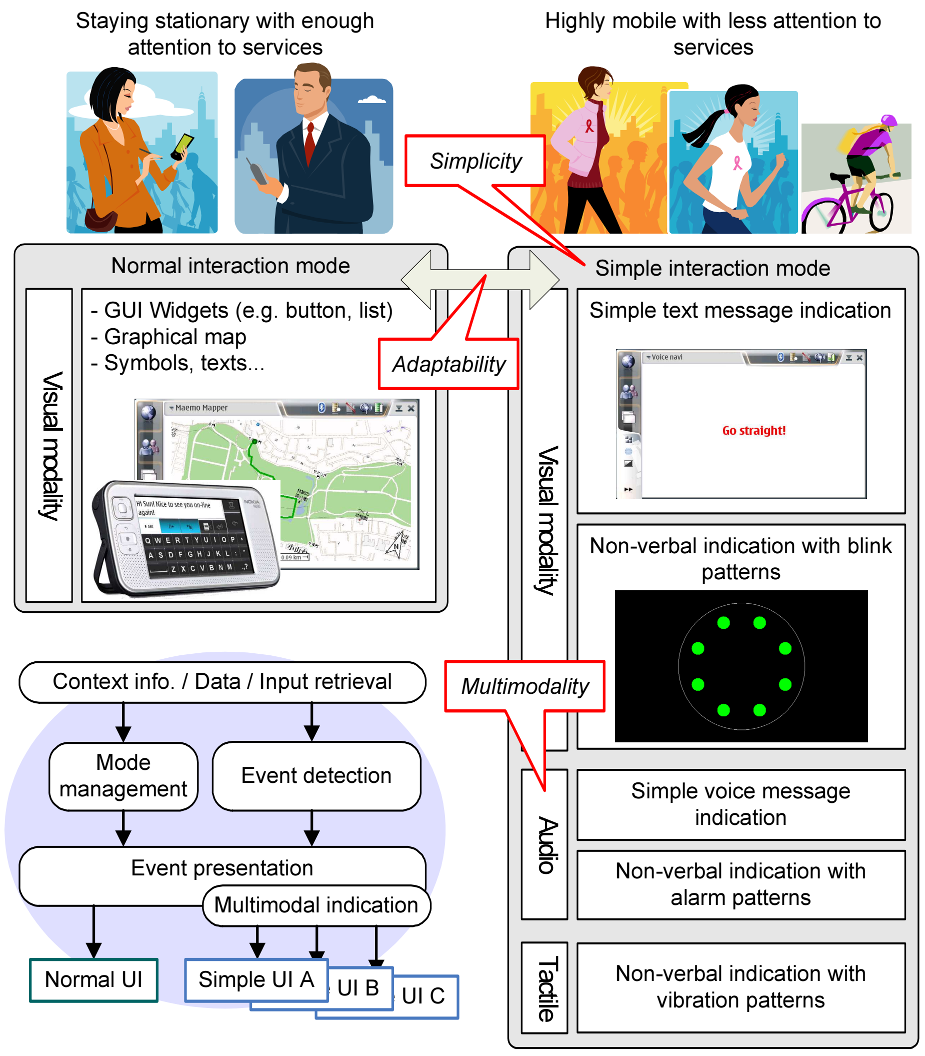

Staying stationary with enough attention to services
Highly mobile with less attention to services
Simplicity
Normal interaction mode
Simple interaction mode
Visual modality
- GUI Widgets (e.g. button, list)
- Graphical map
- Symbols, texts...
Adaptability
Simple text message indication
Go straight!
Visual modality
Non-verbal indication with blink patterns
Context info. / Data / Input retrieval
Multimodality
Mode management
Event detection
Audio
Simple voice message indication
Event presentation
Non-verbal indication with alarm patterns
Multimodal indication
Normal UI
Simple UI A
UI B
UI C
Tactile
Non-verbal indication with vibration patterns

FIGURE 4.1: Overview of the mobile service design framework

so on. Only when important events are detected (e.g. "the user is following a wrong way", "the user should turn right this corner"), the user should be notified in an appropriate way.

To realize this concept, mobile services should represent minimum, but important information effectively. In Figure 4.1, an overview of the design framework is shown with a pedestrian navigation service as an example. Logical service components are identified at the lower left of the figure. Also, we point out three important characteristics from the discussion above and explain details of each in the below sections.

4.4.1 Simplicity

Our framework makes the mobile services non-interactive to decrease the number of interaction cycles. In the framework, mobile services provide a *simple interaction mode* in addition to the conventional user interface, named *normal interaction mode* in the figure. In the example, the pedestrian navigation service provides five kinds of user interfaces in the simple interaction mode. Like status indicators, the simple user interface allows users to check status of the service and him/herself at a glance. Duration of an interruption caused by the service is minimized, and thus the service can work even if the user's attention is fragmented in a multitask environment.

In the simple interaction mode, the mobile service mainly focuses on tasks that do not require complex interaction for the user. This means that the mobile service should be designed to keep its semantics as simple as possible in the simple interaction mode. In the example, one essential point of the route navigation service is "to guide a user to the destination". Needless to say, most of the information that the map includes is excluded from the simple user interface. However, it still aims to achieve the same task without explicit inputs from the user. (If the user would like to operate the service, the mode can be switched to the normal one.)

Also, the interaction style changes to an event-driven model in order to achieve the task with minimum interaction. This means that the mobile service should be designed to detect important events for the user and notify them in a smart way. In the figure, the event detection component retrieves context information (e.g. location data), several kinds of service-related data (e.g. route information) and the user's input to detect events. Detected events are represented by the event presentation component.

In the simple interaction mode, information is mainly pushed from the mobile service to the user. Therefore, to not disturb the user, only necessary information should be sent at the right timing. Usually mobile devices can not perform complex interaction due to its tiny display and keypads. Therefore, this push-type service may fit to mobile devices.

4.4.2 Multimodality

To perform efficient event notification, multimodality is important for moving users. As discussed in Section 4.2.1, the user's perceptual ability could be limited according to physical limitations. To notify events, mobile services should select the appropriate modality that can reach the user's perception. In Figure 4.1, the multimodal indication component selects an appropriate modality channel and assigns a corresponding simple user interface. Availability of modalities is managed in the mode management component and is determined based on the input (e.g. context information of the surrounding environment).

As explained above, this service design framework forces services to be simple and task oriented. Instead of limiting freedom of interaction, information important to the user can be determined and simplified. The simplified information is easy to convert, and it can be represented on various modalities. In the example, three kinds of modalities are shown: visual modality, audio modality and tactile modality. Also, even in the same modality, the representation method varies according to the abstraction level of the information. In the example, two types of indication methods (simple text message indication and non-verbal indication with blinking patterns) are represented in the visual modality.

The simplicity and multimodality allow information to appear on various types of devices by changing its form. For example, a simple text message can be shown in the sub LCD on a cell phone. Also, it can be shown as a ticker tape message flowing on a part of LCD, while other services occupy most of available display area. Non-verbal information does not require rich display devices and it allows various forms and shapes in the device design. If a wrist-watch-type device is equipped with eight LEDs in the frame, it could be used as an event indicator of the route navigation service on our framework. Also, tactile modality based route guiding with waist-belt-type devices were introduced in previous research [29, 118]. Wearable devices are important actuators to effectively notify events to a user. Since our framework supports multimodality based on the event simplification, coordination with such devices could be realized easily.

This design approach is a kind of modality abstraction, which is a concept proposed by Gellersen in [34]. He defined modality abstraction as a concept for capturing parts of the user interface that abstracts its appearance into logical interaction. Thus, modality abstraction provides a common ground for user interfaces that may differ according to used representational media. Our approach also abstracts output to the user, and the modality adaptation is realized in this sense.

4.4.3 Adaptability

Since the framework provides two kinds of interaction styles and multimodal input/output, adaptability is required to control the modes [63]. As discussed in Section 4.4.1, mobile services should occupy the user's attention as short periods as possible. Otherwise the fragmentation of attention may get worse. To decrease unnecessary interaction, the service should autonomously change the mode or modality according to the situation.

Several approaches could be considered by varying the intelligence level. For example, time-out is one basic approach to switch the interaction mode [71]. In this case, user interface changes from the normal interaction mode to the simple interaction mode when the specified time has passed (like a screen saver). This approach enables mobile services to show the simple user interface to the moving user without explicit operation.

As shown in authors' previous work [134], smarter adaptation can be implemented based on context-awareness [109, 133]. Since the simple interaction mode aims to be used while moving, context information which relates to the user's moving status (e.g. whether walking or standing, moving speed) is an important element for this framework. Also, *attentive user interfaces* (AUI) technologies can be applied to acquire the user's interests from his/her behavior (e.g. eye movements, hand gestures) [66, 121]. For example, if the user starts to look at the display for a while, the service notices that the user has lost the way and switches to the normal interaction mode in order to show detailed information.

In this chapter, we mainly focused on the simplicity, since we have to evaluate how our dual-mode approach affects the user's behavior as the first step. Some simple user interfaces, the event detection component and the event presentation component are implemented for a pedestrian navigation service (even though they are not clearly separated as components). Detailed explanations about our prototype are presented in the next section.

4.5 Prototyping

To evaluate the feasibility of our approach, we chose pedestrian navigation as the mobile service to be implemented on our framework. Our brain can not store complete maps, and we can not find fine routes to a destination immediately in unfamiliar places [54]. The digital route guide assists us in finding the route by accessing enormous digitally stored knowledge. This is one of the location-aware services which are successfully realized and deployed in the market. However, further improvements are still required as described in Section 4.2. Also, we believe that empirical studies on the service could be a basis for other mobile services for moving users.

We prototyped the pedestrian navigation service as an extension to Maemo Mapper[1] that is a geographical mapping visualization software on the Maemo application development platform[2]. A route matching algorithm is newly developed and the simple interaction mode is added to the Maemo Mapper. In Figure 4.2, service flow of the pedestrian navigation and examples of event indication patterns are presented.

The pedestrian navigation service consists of two modules, *Vector Comparing Module* and *Navigation Interface Module*. The Vector Comparing Module retrieves the user's location information and calculates moving direction so that the service can handle events by comparing the user's track and predefined routes. The Navigation Interface Module receives parameterized context information (i.e. angle and distance between the user's track and the route) from the Vector Comparing Module. The module determines the user's current status according to the state calculation rule and makes events according to the results. Detected events are notified to the user with pre-configured modality. Detailed explanations about these modules are given in below sections.

[1]Maemo Mapper: `https://garage.maemo.org/projects/maemo-mapper/`
[2]Maemo.org: `http://maemo.org/`

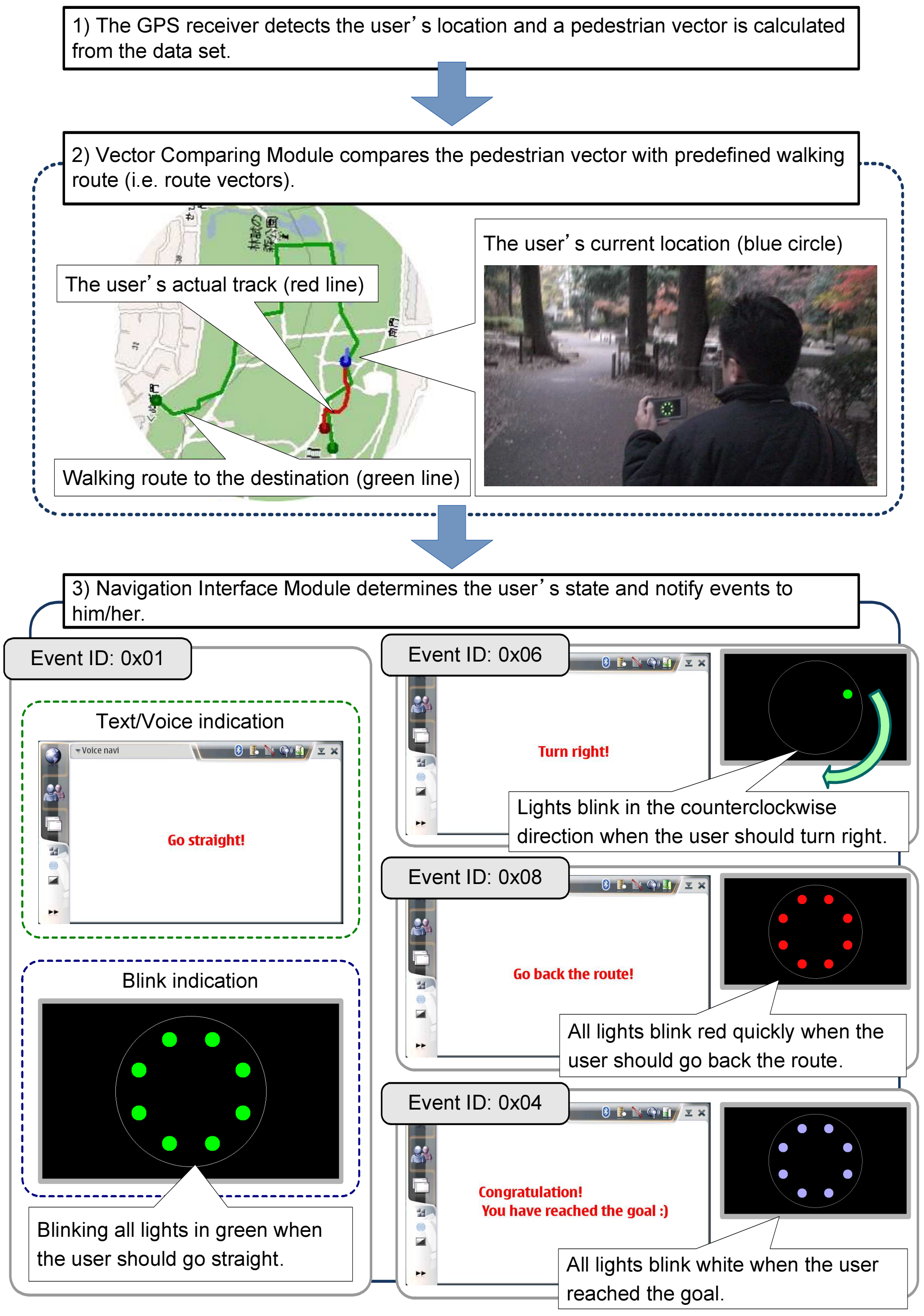

1) The GPS receiver detects the user's location and a pedestrian vector is calculated from the data set.
2) Vector Comparing Module compares the pedestrian vector with predefined walking route (i.e. route vectors).
The user's actual track (red line)
The user's current location (blue circle)
Walking route to the destination (green line)
3) Navigation Interface Module determines the user's state and notify events to him/her.
Event ID: 0x01
Text/Voice indication
Voice navi
Go straight!
Blink indication
Blinking all lights in green when the user should go straight.
Event ID: 0x06
Turn right!
Lights blink in the counterclockwise direction when the user should turn right.
Event ID: 0x08
Go back the route!
All lights blink red quickly when the user should go back the route.
Event ID: 0x04
Congratulation! You have reached the goal :)
All lights blink white when the user reached the goal.

FIGURE 4.2: Service flow of the mobile pedestrian navigation and event indication pattern examples

4.5.1 Vector Comparing Module

Two kinds of vectors are defined in the Vector Comparing Module. *Pedestrian vector* represents the user's current location and the moving direction, which is calculated from time-series of GPS data. *Route vector* represents the predefined route information which the user should follow.

4.5.1.1 Pedestrian Vector

To calculate the pedestrian vector, the method of least squares is applied to the most recent GPS data set (e.g. a series of the most recent five GPS data). Figure 4.3 illustrates the calculation method of the pedestrian vector. The number of data could be flexibly configured according to GPS sampling rate and acceptable delay.

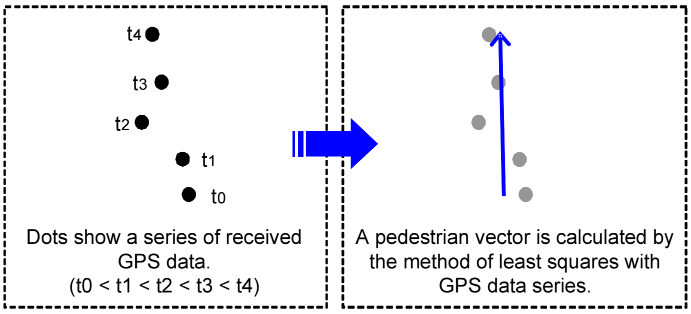

t4
t3
t2
t1
t0
Dots show a series of received GPS data.
(t0 < t1 < t2 < t3 < t4)
A pedestrian vector is calculated by the method of least squares with GPS data series.

FIGURE 4.3: Pedestrian vector calculation

4.5.1.2 Route Vector

The route information includes multiple geographical coordinate values from a starting point to a destination. Entire route information is divided into the route vectors, which connect adjacent two coordinate values, so that the service can simplify vector comparing process.

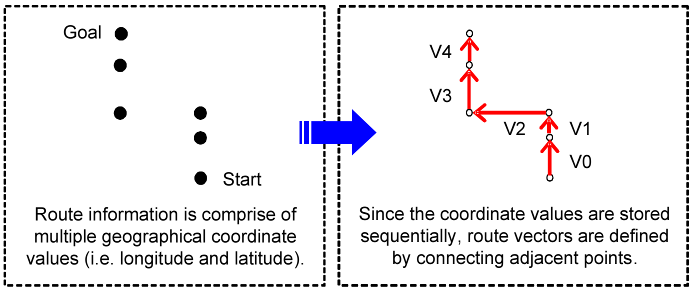

Goal
Start
Route information is comprise of multiple geographical coordinate values (i.e. longitude and latitude).
V4
V3
V2
V1
V0
Since the coordinate values are stored sequentially, route vectors are defined by connecting adjacent points.

FIGURE 4.4: Route vector calculation

Figure 4.4 illustrates an example of five route vectors calculation. In a set of route vectors, sequential ID numbers are assigned to each vector. When the set is composed of N vectors, numbers from 0 to N-1 are assigned (e.g. numbers from 0 to 4 are assigned in the figure). The vector comparing process would be performed to one route vector, which is identified as the nearest to the pedestrian vector. As shown in Figure 4.5, candidate vectors are defined by checking whether the route vector crosses a circle drawn with radius R from the user's location or not. If only one route vector is found in this process, the vector is identified as the nearest route vector.

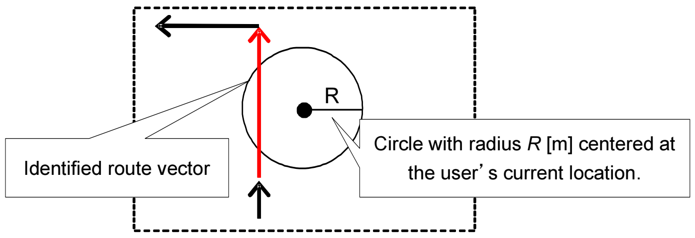

R
Identified route vector
Circle with radius *R* [m] centered at the user' s current location.

FIGURE 4.5: Route vector selection

If multiple vectors are found, the below equation is applied to calculate the vector's ID ($Vidx$):

$Vidx = Vid_{min} + \lceil (Vid_{max} - Vid_{min})/2 \rceil$.

The Vid_{min} represents the smallest ID number and the Vid_{max} represents the largest ID number among the candidate vectors. If no route vector is newly identified, the Vector Comparing Module will keep the current vector as the nearest one.

4.5.1.3 Parameterization Method

After one route vector is identified, the vector comparing process is performed to calculate contextual parameters. In this implementation, angle θ and distance λ between the pedestrian vector and the route vector are sent to the Navigation Interface Module as parameters. In order to simplify the explanation, we define P as a pedestrian vector and V as a route vector as shown in Figure 4.6.

To calculate the *theta*, an inverse function of tangent with an inner product and outer product between these two vectors is applied:

$\theta = arctan2(x, y)$.

The x represents outer product of P and V, and the y represents inner product of P and V. To calculate a distance λ, whether a perpendicular line can be dropped from the endpoint of P to V is checked. If possible, length of the perpendicular line would be calculated as the distance. If impossible, distance between the endpoint of P and the endpoint of V will be used.

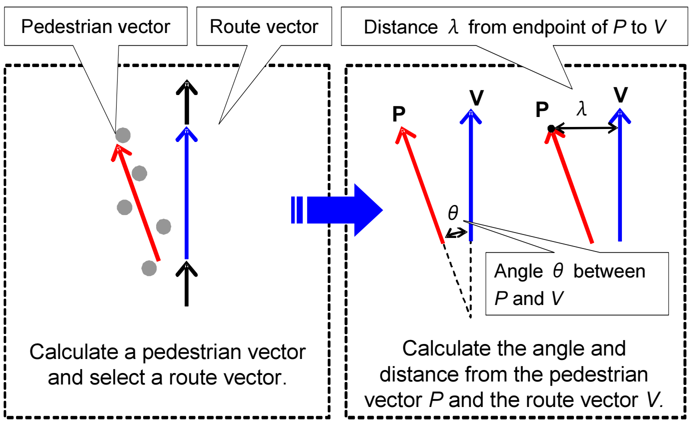

Pedestrian vector
Route vector
Distance λ from endpoint of P to V
P
V
P
λ
V
θ
Angle θ between P and V
Calculate a pedestrian vector and select a route vector.
Calculate the angle and distance from the pedestrian vector P and the route vector V.

FIGURE 4.6: Parameter calculation from the vectors

TABLE 4.1: User's state and distance level calculation rule (x[degree]: Angle between the pedestrian vector and the nearest route vector, d[m]: Distance from the user's current position to the destination)

User's state			
	$d < 10$	$10 \leq d$ ¡ 20	$20 \leq d$
$0 \leq x < 30$	state_0	state_0	state_1
$30 \leq x < 60$	state_1	state_2	state_3
$60 \leq x < 100$	state_2	state_3	state_4
$100 \leq x$	state_3	state_4	state_4
0 ¿ $x >$ -30	state_0	state_0	state_M1
-30 $\geq x >$ -60	state_M1	state_M2	state_M3
-60 $\geq x >$ -100	state_M2	state_M3	state_M4
-100 $\geq x$	state_M3	state_M4	state_M4

Distance level			
$d < 10$	$10 \leq d < 50$	$50 \leq d < 100$	$100 \leq d$
dist_0	dist_1	dist_2	dist_3

When the user reaches the destination, the service will be terminated. The termination condition is defined as "when the user gets within R distance of the destination". Distance R should be set with an appropriate length (e.g. 10m), since GPS data could contain some errors in general.

4.5.2 Navigation Interface Module

Calculated parameters are used to define the user's current state in the Navigation Interface Module. Table 4.1 shows the user's state and distance level calculation rule. In this implementation, nine kinds of user's states (*state_∗*) and four kinds of distance levels (*dist_∗*) are defined. For instance, where the distance d is less than 10m, the user's state would be defined as shown in Figure 4.7.

Event type is determined according to the combination of the user's state and the distance level as shown in Table 4.2. The table also shows how the events are handled and indicated in each

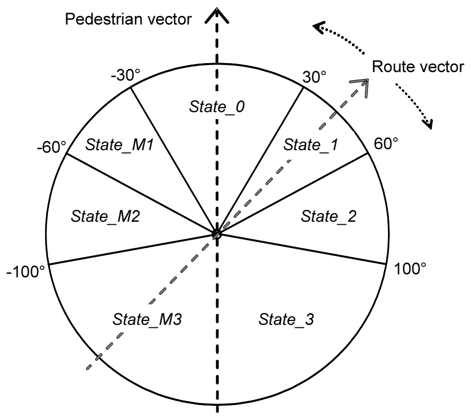


FIGURE 4.7: User's state definition where the distance d is less than 10m. (In this case, the user's pedestrian state is defined as $State\_1$ according to angle of the route vector)

TABLE 4.2: Event types and corresponding indication patterns (Asterisk "*" represents a wildcard character)

ID	Condition (*state*, *distance*)	Text/Voice indication	Blink indication
0x01	(state_0, dist_3)	"Go straight!"	All lights blink green slowly.
0x02	(state_0, dist_2)	"Go straight!"	All lights blink green.
0x03	(state_0, dist_1)	"Go straight!"	All lights blink green quickly.
0x04	(*, dist_0))	"Congratulations! You have reached the goal!"	All lights blink white.
0x05	(state_1, *)	"Go forward left"	Lights blink green slowly in the counterclockwise direction.
0x06	(state_2, *)	"Turn left"	Lights blink green slowly in the counterclockwise direction.
0x07	(state_3, *)	"Go backward left"	Lights blink yellow quickly in the counterclockwise direction.
0x08	(state_4, *)	"Go back the route"	All lights blink red quickly.
0x09	(state_M1, *)	"Go forward right"	Lights blink green slowly in the clockwise direction.
0x0a	(state_M2, *)	"Turn right"	Lights blink green slowly in the clockwise direction.
0x0b	(state_M3, *)	"Go backward right"	Lights blink yellow quickly in the clockwise direction.
0x0c	(state_M4, *)	"Go back the route"	All lights blink red quickly.

modality. Our prototype provides three kinds of indication methods: simple text messages, simple audio messages, graphical signals with blinking circles on a display. The simple audio messages are generated from the simple text messages by text-to-speech (TTS) engine[3], and thus the same information is given to the user. Blink indication pattens are defined subjectively in this experiment. In the next section, evaluation method for our prototype and results are shown.

4.6 Evaluation

To evaluate the feasibility and usability of our prototype, we performed a field experiment. This experiment aims to observe how the dual-mode approach works and affects to the user's

[3]Flite: `http://www.speech.cs.cmu.edu/flite/`

behavior while on the move. Therefore, we compared the normal interaction mode and the simple interaction mode with the same modality (i.e. visual modality). The below section explains the experiment method and derived results.

4.6.1 Method

Two different tasks are given to participants and each task is performed once.

TaskA : Participants are instructed to walk a route only with the normal interaction mode (i.e. Maemo Mapper). The normal interaction mode shows the user's current location and route to the destination. The user can reach the destination by following the route, but the shown area on the display is small enough so that the user can not remember the entire route. Users are not allowed to manipulate the device to scroll and zoom in/out the map.

TaskB : Participants are instructed to walk another route with a combination of the simple interaction mode (i.e. blink indication) and the normal interaction mode. In this case, the simple interaction mode has to be mainly used and the normal interaction mode is used auxiliary when the user has lost his/her way. Users can switch modes by manually pushing a hardware key.

Figure 4.8 shows the routes which we used for this experiment. Route A is used for task A and route B is for task B. Course length and the number of corners are almost the same and it takes about 5 or 6 minutes to reach the goal by walking as shown in Table 4.3.

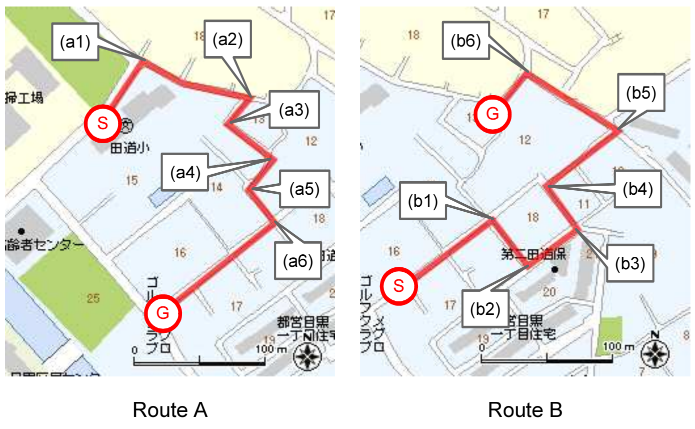


FIGURE 4.8: Routes and corners in the field experiments

5 participants from Nokia Research Center Tokyo joined this fieldwork (male:4, female:1) and 2 observers followed the participants. The participants were not familiar with the area where this experiment took place. In the tasks, participants hold an N800 Nokia Internet Tablet with the hands and wore a GPS receiver attached helmet on their head. Observers monitored and recorded participants' behavior with two video cameras as shown in Figure 4.9. Observer A recorded a participant's face to analyze whether he/she pays attention to the display or

Table 4.3: Distance between each corners in the routes

Route A		Route B	
Start - a1	72m	Start - b1	79m
a1 - a2	81m	b1 - b2	43m
a2 - a3	20m	b2 - b3	47m
a3 - a4	38m	b3 - b4	39m
a4 - a5	32m	b4 - b5	66m
a5 - a6	33m	b5 - b6	81m
a6 - Goal	109m	b6 - Goal	29m
Total	385m	Total	384m

surrounding environments. Observer B monitored observer A and the participant from behind so that situational information (e.g. walking time, obstacles) could be recorded. After the fieldwork, we asked the participants to answer some questions.

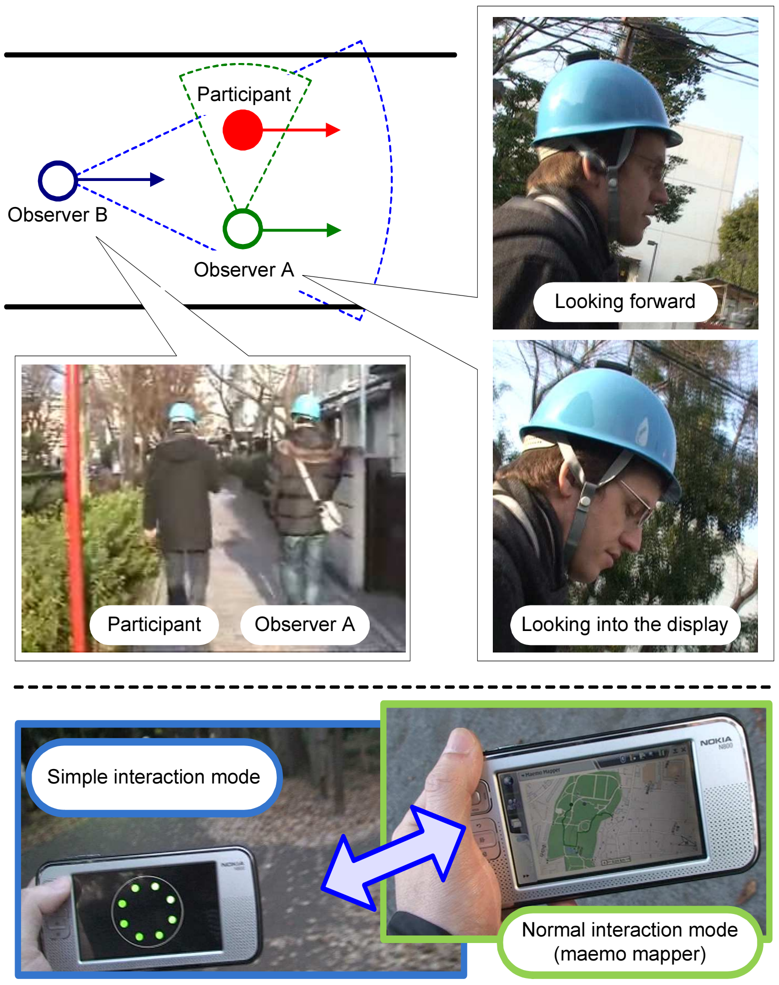


Figure 4.9: Two observers recorded the participants' behavior with video cameras (above). Interaction modes used in the Task B (below)

Table 4.4: Execution time and standard deviation (sample data = 1816). Parameter calculation function including location data retrieval is measured in the Vector Comparing Module. Also, event determination function is measured in the Navigation Interface Module (event indication on user interfaces is not included).

	MET	*SD*
Vector Comparing Module	376usec	134usec
Navigation Interface Module	16usec	17usec

Table 4.5: Experimental results from the five participants' trial

Task A	*TT*	*ST*	*LT*	*LD*	*SP*	*GR*	*MT*
User A	276	26	27	24	5.33	52.17	-
User B	303	0	18	16	4.76	44.55	-
User C	283	0	0	0	4.89	62.19	-
User D	308	4	30	24	4.78	38.96	-
User E	233	0	41	70	6.13	46.78	-

Task B	*TT*	*ST*	*LT*	*LD*	*SP*	*GR*	*MT*
User A	258	7	0	0	5.35	50.00	0
User B	381	53	150	124	4.80	33.07	39
User C	267	9	0	0	5.17	62.17	12
User D	266	9	0	0	5.19	32.70	0
User E	274	57	141	168	6.07	22.26	54

4.6.2 Experimental Results

4.6.2.1 System Performance

To evaluate the system performance, we measured execution time of the Vector Comparing Module and the Navigation Interface Module. N800 Nokia Internet Tablet was used as the evaluation platform (CPU: TI OMAP2420 330MHz, RAM: DDR 128MB), and we calculated the values using actual samples acquired from the fieldwork (1816 samples of data). Table 4.4 shows the results of mean execution time (*MET*) and standard deviation (*SD*). Our navigation service is driven by GPS data input and measured overhead is added for every updates. However, the overhead was quite small and thus our algorithm did not critically affect the system performance.

4.6.2.2 Route Navigation Results

From the recorded videos, we measured and analyzed the below factors for each tasks: *TT* [sec]: Total time spent to reach the destination, *ST* [sec]: Total time of stationary state (i.e. just standing without walking), *LT* [sec]: Total (loss) time spent for walking incorrect route, *LD* [m]: Total (loss) distance of incorrect route walking, *SP* [km/h]: Average speed, *GR* [%]: Proportion of time that the participant paid attention to the display to *TT*, *MT* [sec]: Total time of the normal interaction mode (i.e. Maemo Mapper). Table 4.5 summarizes the analysis results.

TaskA : As shown in LT and LD, even though the map information is provided, 4 participants followed a wrong route. This is due to both individual map reading ability and fluctuation of GPS data. However, soon afterwords they noticed the situation and went back to the original route. Indeed, all participants reached the destination.

TaskB : As for Task A, 2 participants lost the way and asked the normal interaction mode to help. This is due to notification delay caused from the fluctuation of GPS data and lack of attention. Contrary to Task A, the fail was critical since the participant could not understand the situation without graphical map information. Moreover, our prototype does not support rerouting features that updates and rearranges the route with the user's current location information. Our prototype continued to guide even though the participant left away from the route, and it resulted in greater loss than Task A as shown in LT and LD. However, at last, they could reach the goal with checking the map in the normal interaction mode.

On the other hand, 3 participants could perform the task without time losses. The user C checked the map information once, since the system behavior seemed to be incorrect. However, other 2 participants reached the goal only with the simple interaction mode.

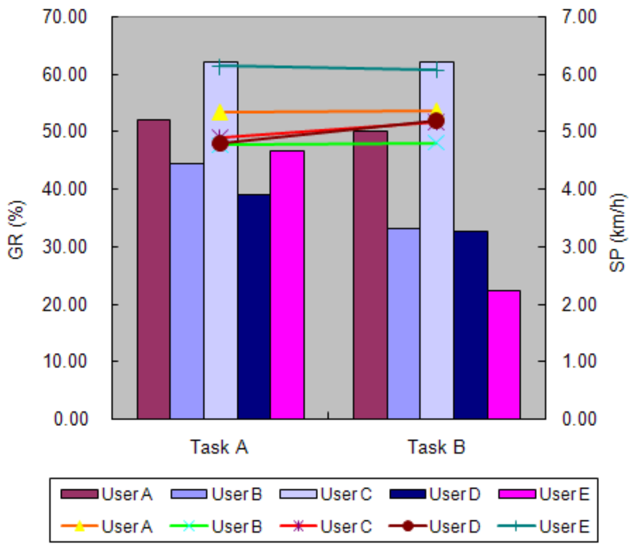


FIGURE 4.10: Chart of SP and GR value comparison

Also we found two important factors SP and GR, which tightly relate to a user's cognitive load, show interesting trends in the results. As shown in Figure 4.10, most of participants' GR value decreases in Task B. This means the participants paid less attention to the display than Task A case. On the other hand, some of SP slightly increases and it indicates participants could walk faster than Task A case. We acquired some of the reasons from questionnaires. In the next section, we discuss future directions for further improvements based on participants' comments and our findings.

Table 4.6: Mean subjective scores for the simple interaction mode

Question		Score (MAX: 5 - MIN: 1)	SD
Q1	Compared to map UI, did simple UI lighten the load to be guided by the navigation service?	4 (better than map UI)	0.63
Q2	Is the variation of shown information (blinking pattern) enough to be guided?	2.6 (almost enough)	0.48
Q3	Is the meaning of shown information (blinking pattern) intuitive and comprehensible?	4 (comprehensible)	0
Q4	Is the information shown at the right timing?	2.6 (slightly inadequate)	0.8

4.7 Discussion and Implications for Future Work

Table 4.6 shows questions given to the participants and subjective scores on the evaluation. The subjective scores show that the simple interaction mode could keep service semantics useful enough to complete tasks.

Also, we acquired useful comments for further improvements. Below, we discuss about identified issues and possible solutions to them.

- Users would expect the next event and attempt to know when it will occur. For example, users can recognize the next corner to turn from map information. It seems that users temporary remember route information with point of interests (e.g. landmarks, corners). Actually, in some cases, GR decreased for a while after checking map information when the participant lost the way. It can be said that the participant remembered the next corner and used the simple interaction mode to check whether the memory was correct. In the navigation service case, the timing of next event can be estimated according to the pedestrian's moving speed and direction. If such information (e.g. time bar which indicates when the next event will occur) could be added, users do not have to wait the indication carefully, and usability might be improved as a result.
- Some events and indications seem to be unnecessary. In our prototype, the "Go backward left/right" indication was a bit confusing and the yellow light indication was not appropriate to the meaning. Also, timing to show the indication was slightly inadequate due to fluctuation of GPS data. Since the simple interaction mode does not provide any further information, unnecessary or incorrect warnings are critical. Service designers should consider possible errors and allow the system to propose to check the status with the normal interaction mode, when handled events seems to be doubtful.
- The blinking pattern was comprehensible, but no indication seems to be needed while the user is walking on the right route. Also, this indication pattern does not support stationary situations. Once the user stops, he/she could not recognize the right route

since the indication remains green blinking. The simple interaction mode should indicate the direction to go when the user stops.

- Some users commented that audio indication is preferred while on the move. In the experiment, we did not allow to use the audio modality, since this experiment aims to simply evaluate whether the simple interaction mode decreases the user's cognitive load. Also, it is obvious that audio indication has advantage in terms of decreasing visual attention. However, current implementation does not support some of proposed important features including the multimodality management. The modality management feature is also needed in order to coordinate with wearable devices, so we will implement them in the future work.
- Since the user interface on the simple user interaction mode was not interesting, users started to look the scenery. This is an interesting point, since the simple user interface affected users and motivated them to pay attention to the surrounding environment.

Also, we have found that sometimes the user has strong confidence or relies on the service too much. However, such users tend to be stuck and be thrown into utter chaos when the service behaves in an unexpected way. Therefore, recovery methods should be provided to the user, even in the simple interaction mode. This issue was also pointed out by Jones et al. [51]. Their system named ONTRACK guides a pedestrian to the destination by continuously adapting the spatial qualities of listening music. ONTRACK is similar to our work in terms of the simplicity, since it uses only audio modality with non-verbal information. They experienced that sometimes visual landmark became a stronger attractor than the audio cues and misled users.

Finally, the user's input should be allowed even in the simple interaction mode, in order to provide feedbacks to mobile services [44, 119]. One reason for this is that explicit commands from the user is useful to detect the user's attention in addition to implicit information (e.g. eye movement, ambient noise).

4.8 Conclusion of This Chapter

In this chapter, we proposed a mobile service design framework that aims to improve usability of mobile services on the move. Main causes of usability degradation have been discussed and we took an approach that allows users to perform interaction with less attention. We implemented the simple interaction mode on a pedestrian navigation service and evaluated its feasibility through field experiments. The result has shown that we have successfully decreased the users' cognitive load by adding a simple interaction mode. We also evaluated that the service semantics was successfully kept even in the simple interaction mode, and users could perform tasks with it. Even though there is room for further improvements, we received positive comments and identified feasibility of our approach.

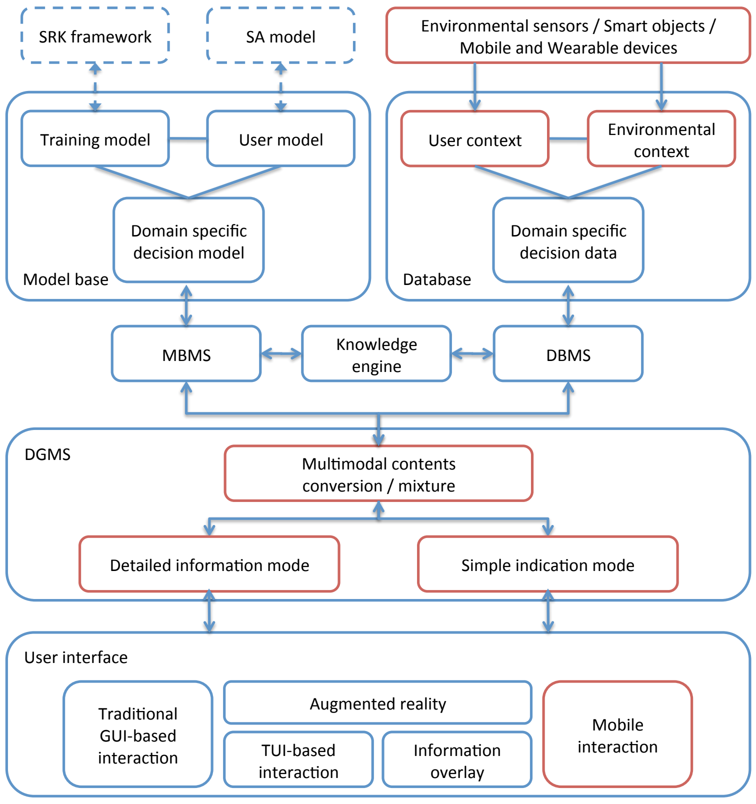


FIGURE 4.11: Focused components in this chapter

Figure 4.11 shows the focused ADSS framework components in this chapter. This case study mainly focused on mobile interaction, and emphasized the importance of adaptation based on the user and environmental context information. In order to address the situational disabilities and fragmented attention issues, the DGMS prototype that supports simplification of multimodal contents is implemented.

Chapter 5

Decision Training with Augmented Traditional Games

5.1 Background

While newly developed game consoles explore brand-new gaming style, traditional games, such as tabletop games, billiard and darts, are still appealing to us with various irreplaceable tastes. Spatial and physical interaction with tangible game objects enhances momentary, but emotional user experience in gaming. Players unconsciously use the all senses to perceive not only necessary information for the game play, but also the one that provides additional tastes or clues. For example, players need to read the face of a card to play poker. In addition, auxiliary information, such as shuffling sound, opponent players' facial expression and shakes caused by thrills, helps to play the game well and also simply makes game fun. Moreover, these advantages of traditional games cannot be reproduced in completely digitalized games, such as video poker.

On the other hand, the concept of mixed reality is getting realized today. For example, these days AR (augmented reality) toys are commercialized in the market. Users can interact with a virtual character by capturing a matrix code with a web camera. The character is superimposed on the code and animated as preliminary programmed. Real and virtual environments are united into a hybrid world. Then digitally augmented features provide new user experience to consumers and assist players in gaming. For example, pervasive game is one domain in the mixed reality games [41, 65]. The rapid progress of pervasive computing technologies enable games to be seamlessly integrated into our daily lives.

Such technologies often deserve attention because of the potential to invent brand-new games. However, they can also be used to augment existing games, and traditional games as well. In [42], Hinske *et al.* clarified a concept of augmented traditional game environments and pointed out the socializing aspects of the games. To play traditional games, usually players gather around a single table and directly share experiences in real-time. It smoothens social communication and amplifies the excitement caused by the game. Online games also enable users to simultaneously experience events, but the aforementioned additional information is lost through the indirect communication. Thus the augmented traditional game environments are basically designed with focusing on a single location gaming style; players are not supposed to move around, while most pervasive games adopt a time and location independent style [48, 65].

Hinske *et al.* also proposed guidelines for the design and implementation of augmented traditional game environments. In this chapter, we proceed with the discussion originally argued by them, with introducing three case studies of augmented traditional games. As they pointed out, it is challenging to augment an already existing game without changing its original look-and-feel. Moreover, in order to support a game flow with mixed reality technologies, we need to carefully design interaction: for example, context information should be observed passively so that players can concentrate on a gaming activity. Mixed reality technologies are expected to be applied to future products. Thus we believe that our findings can be also utilized in a wider range of applications as well as augmented traditional games.

In Section 5.2, we explain the concept of our work with several use cases. Then in Section 5.3, we introduce two case studies of augmented traditional games: Augmented Go and EmoPoker. Based on the game development work, we discuss the design issues of augmented traditional games In Section 5.5.

5.2 Augmented Traditional Game Environments

5.2.1 Attractiveness of Traditional Games

Traditional games offer physical interactions to game objects (e.g., darts, playing cards, Go stones). Physical interactions enable players to sense several kinds of information, even though they are not necessarily relate to the game itself. For example, one of important factors to evaluate other players' mental condition is pace. In digitalized games, a player's actions, such as selecting cards in poker and putting Go stones on a board, can be instantly performed as a single mouse click. There are certain distance between objects and players in physical games, thus it allows players to observe the duration spent for an action and get an idea of what the other player is thinking.

Moreover, tactile interactions amplify expectation to uncertain things and the excitement caused by achieving a goal. For example, in mahjong, players cannot know what kind of mahjong tile will be drawn in the next turn, since the tiles are ordered on the table as walls. During tile drawing motion (i.e., moving a tile from the wall to the player's area), the player's attention is drawn to the tile and expectation gets increased as the mark of the tile is gradually revealed. Skillful players can even recognize tiles by just feeling the surface of the tile. Therefore, the distance between the wall and the player's area can be considered to make the most exciting moment of the game play. These advantages of physical interaction cannot be reproduced in digitalized mahjong games.

We consider this most important factor of traditional games as *Ma*; Japanese word that represents distances among objects in a space. *Ma* is originally used in art and design, but it also can be used in a wider range of domains, such as martial arts and theatrical performances. In martial arts, players adjust distance to an opponent in order to keep their own territory to take good offense and defense. In theatrical performances, actors and actresses coordinate the timing of actions in order to make greater impression to audiences. *Ma* includes concepts of spatial distance, gap and pause, so both physical and mental distances among game objects including human players can be represented in this single word. At the same time, it is hard to reproduce *Ma* in digitalized games, since *Ma* is total information that players perceive in physical interactions.

5.2.2 Digital Augmentation with Pervasive Technologies

While digitalization causes the loss of the aforementioned advantages, pervasive computing technologies can support various aspects of traditional games [65]. There are roughly two styles to apply the technologies to the domain of play and games: creation of novel forms of play and games, and augmenting existing forms of play and games. In [42], Hinske *et al.* mainly focused on supporting the players by providing services in digitally augmented traditional game environments, instead of integrating virtual play and game elements [108]. They also proposed two important aspects of the augmentation: *the game flow virtualization of the game* and *the physical augmentation of the game.*The game flow virtualization enables an augmentation system to deal with a game rule so that it can check the rule consistencies and violations. Moreover, the system can provide information that is relevant to the game at appropriate timing. The physical augmentation also identifies two main objectives: unobtrusive technology integration and non-negative influence on the original game's rich social interactions. Table 5.1 quotes some of design guidelines they proposed in [42].

As the guideline indicates, ideally technologies support a gaming process with minimum interference. Moreover, technologies are supposed to be cognitively disappeared into background [125]. As Hinske *et al.* pointed out, however, game objects are often small and will influence the choice of technologies. For example, RFID (radio frequency identification) tags are convenient to identify the objects. However, in order to detect the location of such small objects on a small

Table 5.1: Design guidelines for physical augmentation (quoted partially from [42])

1.	The technological enhancement should have an added value.
3.	The focus should remain on the game and the interaction itself, not on the technology.
4.	Technology integration should be done in a way that is unobtrusive, if not completely invisible.
5.	The game should still be playable (in the "traditional" way) even if technology is switched off or not working.
8.	Players should receive simple and efficient access to information. Feedback should be immediate and continuous.
12.	Secondary user interfaces should be minimized.

tabletop, readers also need to be small or other positioning techniques are required. Flexibility is also a problem. If the tags are attached to cards, they should be enough thin and bendable so that players can shuffle the cards without breaking them. Regarding to RFID tags, anti-collision mechanism is another issue. If a reader does not support it, the objects cannot be piled up and thus possible use cases will be limited. However, currently such anti-collision readers are still expensive. Thus physical augmentation could change game objects' appearance, but players still should be able to play a game in the traditional way (even augmentation support is switched off).

Unobtrusive feedback and minimized (secondary) user interface are also important design issues to keep original look-and-feel. People often concentrate on game playing and experience a deep involvement that removes awareness of everyday life. According to [49], *"during play, distractions from major game tasks should be minimized by reducing nongame-related interactions and reducing the game interface to maximize the amount of screen taken up with game action"*. Interaction with an augmented game should follow this principle, in order to let a player experiences flow. However, it is difficult to automatically and perfectly recognize game relevant information. Thus players should be able to explicitly interact with the system to switch modes or correct detection errors for example. This user interface should be designed to naturally fit to original interaction process, rather than checking a display and configuring with a keyboard.

5.2.3 Use Cases

Augmentation enhances the value of traditional games, and it also allows using the game for the various purposes for which they are not originally designed. In order to proceed with the discussion with more concrete scenarios, we roughly categorized possible use cases into three domains: *Fun*, *Practice*, and *Research*. Table 5.2 shows example use cases from the domain and functionality aspects.

We focused on two main functionalities of augmented traditional games: *Context monitoring* and *Multimodal feedback*. In order to support gaming process, an augmentation system needs to monitor the context information of game objects (e.g., cards' position). Context information is

TABLE 5.2: Example use cases of augmented traditional games

	Context monitoring	Multimodal feedback
Fun	(a) Automatic game event recording	(b) Handicap making, visual effect
Practice	(c) Game review with skillful players	(d) Showing invisible information
Research	(e) Players' behavior analysis	(f) Feedback design

recognized from sensor data, such as location information given by RFID tags and images captured with a web camera. Players' context information is also interesting source to elaborately analyze human factors in game events. For example, how game events (e.g., betting money) affect emotional state can be analyzed from physiological sensor data.

While context monitoring process is passively conducted and hidden from players, the multimodal feedback augments games in the way that players can perceive. With the feedback, players can recognize information that is invisible in conventional games (e.g., an opponent players' heart beat). Several actuators are used to provide feedback, and its modality varies accordingly. For example, our applications use a projector to show visual feedback, and a mobile device to play audio feedback. These functionalities realize different value according to the domains. Below we explain the detail of use cases that are shown in Table 5.2.

5.2.3.1 Fun

One main purpose of traditional game augmentation is simply to enhance the pleasure of gaming and provide better user experience. For example, Ishii *et al.* augmented a ping-pong table with a projector and sensor modules that detect a ping-pong ball's position [45]. The system called PingPongPlus supports several modes and some of those transform game playing style from competition to collaboration. While such new gaming styles are invented with the augmentation, conventional ping-pong play is still enhanced with attractive visual effects such as virtual ripples on the table. Players do not need to newly learn how to use the system. The system adds multimodal feedback corresponding to game events, and players realize they are interacting with the game (system) itself as well as opponent players (Table 5.2-(b)). Multimodal feedback also can be used to make handicap between beginner and skillful players. For example, providing warning information prevents a beginner player from a careless mistake. We will explain more detail handicap making scenarios in Section 5.3.1 and Section 5.3.2.

One use case featured by context monitoring is automatic game event recording (Table 5.2-(a)). The augmentation system automatically monitors context information such as game events, and players can review a process after finishing the game. Unless recording equipments (e.g., video camera) are preliminary set up, conventional gaming process is not recorded and cannot be replayed. However, players sometimes encounter a miraculous happening and it becomes unforgettable experience. We often regret that nobody has started filming, and actually we have missed lots of opportunity to look back on such memorable scenes. With the augmentation system, however, ordinary people can record their game play as a TV program relay a professional game match. Since we use original game objects, players can enjoy gaming as usual If multimodal feedback is turned off. It is also interesting that the system automatically digests a recorded video from players' context information (e.g., a shout of joy), and propose the "highlight of this game" after the game play.

5.2.3.2 Practice

While the fun domain mainly focuses on short-term (i.e., one game play) user experience, we regard the progress of gaming skill as another important factor to enrich longer-term user experience. As the concept of game flow indicates, a gaming process consists of multiple tasks and it has people concentrate on game play. A task has a clear goal and provides immediate feedback so that the player can feel satisfaction when the task is achieved. Every game player starts from a beginner level. Game players can overcome difficult tasks and situations as master technical knowledge and skill. However, beginners tend to spend lots of time and effort for practice to advance their skill. Moreover, they need a good trainer and iterative success experiences in order neither to be frustrated nor stop playing during a growth phase. Video and online games are one good way to practice game playing, but still it is difficult to check the point of strategy and understand the difference with skillful players. Moreover, as aforementioned, real atmosphere gives extra, but essential information such as facial expression to game players. We believe that real game experience with human players is important for a beginner player to learn game play.

Therefore, one possible scenario of the practice domain is the game review with skillful players (Table 5.2-(c)). Automatic recording allows reviewing a finished game process with upper grade players. Particularly it is hard to review imperfect information games, such as poker and mahjong, since a round finishes without disclosing most information to players. Moreover, multiple players join the game, thus it is almost impossible to manually track such randomly dealt cards and tiles. However, the recording system allows replaying the play with showing all players' hand. This "making invisible visible" is important when a player learns game mechanism by experience. In a conventional way, a player can get only results to their decision: for example in Texas Hold'em poker, when an opponent won the round without showing hand (i.e., you decided to go "fold"), you cannot have information to evaluate whether the strategy was good or not (i.e., the opponent's hand keeps disclosed). Moreover, a player tends to be biased due to mental status during game play (we explain detail in Section 5.3.2). Thus increasing the amount of information helps to notice own habit of decision-making and leads a player towards better understanding of the game.

Moreover, by experience, usually skillful players have learned useful intuition to insight the course of a game. Thus they can shorten the time to recognize a situation and quickly process acquired information to make possible strategies. Beginner players often take longer time, since they cannot filter out unnecessary information and do not have knowledge about the patterns that frequently happen under a certain situation. If such invisible experience could be visualized, a beginner player could see what upper grade players are seeing. We explain this idea with Augmented Go system in Section 5.3.1. In this case, such invisible information is shown rather in real-time than after game play (Table 5.2-(d)). Players should be able to control the timing, types, and level of visualization so that feedback can appear only when it is necessary. Moreover, feedback could be multimodal.

5.2.3.3 Research

Lastly, we emphasize that the augmentation system also contributes to academic research as well as individual game players. For example, Nacke *et al.* proposed to use a game monitoring system to analyze players' physiological responses [78]. Games have drawn considerable attention to understand human activity. Gaming process requires intelligent brain activities and also useful to know how people reacts to a game event and behaves under a certain situation (Table 5.2-(e)). Moreover, while playing a game, people concentrates on a task and do limited interactions. It means that an observer can eliminate unnecessary factors in the experiment and expect tighter correlation between a game event and player's behavior than other types of wild experiments. Thus additional sensor devices, such as brain wave monitor, are also interesting to be supported by the system, even though they are not directly used to enhance gaming experience. In addition to the activity monitoring, it is also possible to make a stimulus and observe how people's behavior changes. As we study with prototyped augmented traditional games, identifying a guideline for feedback design is one big challenge that contributes to future product design (Table 5.2-(f)). We discuss the findings in Section 5.5.

5.2.4 Related Work

In [33], Christian et al. introduced a computer-augmented card game. RFID tags embedded into cards are used to capture the game state; and players can check scores and advice with their mobile phone. Since card games are incomplete-information game, personal devices such as mobile phones are useful for giving each user individual feedback. However, as discussed in the paper, mobile phones also require visual attention and it leads to distraction from the game since players wanted to check system behavior and possible errors. Thus audio feedback might be useful to make lightweight feedback when simple notification should be given (e.g., confirmation of object tracking status).

In [19], Cooper et al. augmented one traditional game called Chinese Chekers to investigate user interface issues for tabletop augmented reality entertainment applications. Original game objects are completely replaced with digital devices and players manipulate virtual objects on a large display with an augmented reality marker. Even though their approach loses the advantages of traditional games we pointed out, unified interaction style for a wider range of applications is useful in some cases. Moreover, they introduced Passive detection framework, which is an object recognition infrastructure for pervasive services in a meeting room environment. Having a single setup for multiple games is important for decreasing the installation cost and increasing the availability of the framework.

5.3 Case Study

In this section, we introduce two case studies of the augmented traditional games: Augmented Go and EmoPoker.

5.3.1 Augmented Go

Go is a traditional board game for two players, where the objective is to occupy a larger portion of the board than the opponent. Black and white stones are used to control the territory and a board with a grid of 19 x 19 lines is used as the game field. The rules of Go are relatively simple, but the underlying strategies are extremely complex and rich. As in chess and reversi, numerous set sequences and strategies have been invented to reduce the complexity, but studying them requires the player to actually understand the strategic concepts. Thus it takes a long time for beginners to do well against experienced opponents.

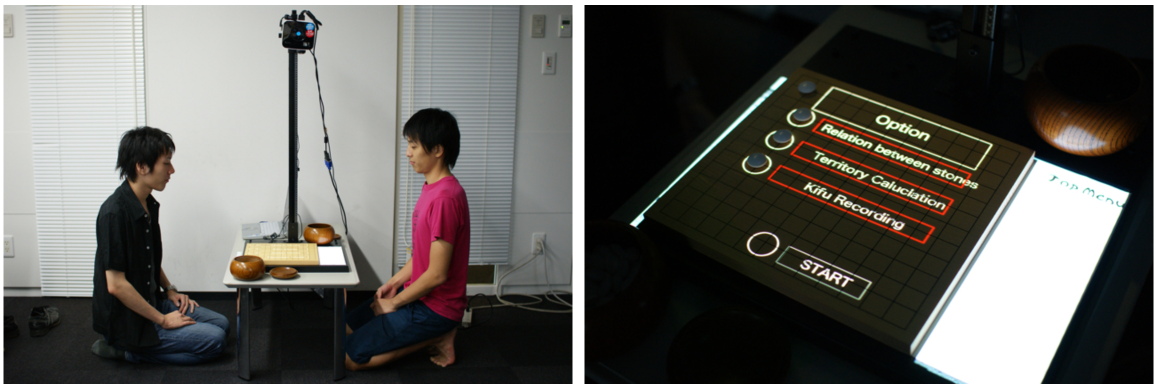


FIGURE 5.1: Augmented Go system

Augmented Go system supports several gaming modes in addition to normal play (Figure 5.1). The basic idea is to provide valuable information to beginners without additional interactions and devices. Figure 5.2 shows the system architecture of Augmented Go system. In this prototype, feedback is provided visually by superimposing guiding information onto the Go board with a projector. A web camera connected to a PC is used to detect the position of the Go stones. OpenCV library is used for visual analysis and the logic engine determines the information presented to the players about the current game situation[1]. The sequence of stone moves is recorded into the database, which facilitates replaying the game for self-training. Finally, a visual animation is created by a flash application, and directly projected onto the Go board.

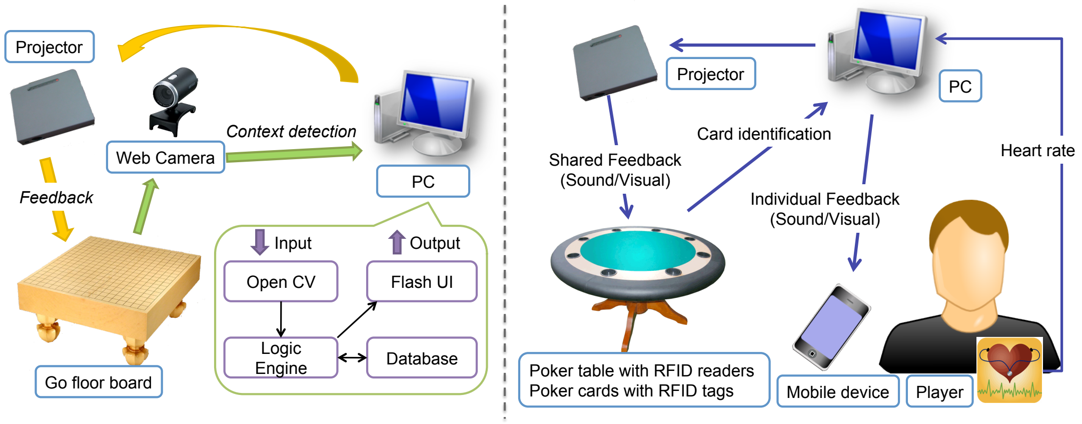


FIGURE 5.2: System components of the augmented traditional games (left side: Augmented Go, right side: EmoPoker)

The system supports several gaming modes. As shown in Figure 5.3-(a), players can interact with the system by putting Go stones on a menu that is projected onto a board. In the following sections we explain some of the modes and how players interact with the Go application.

Match mode: Match mode represents the original concept of Go augmentation. In this mode, two players play Go as usual, but valuable information appears on the board to help beginners recognize the situation and make better decisions. The rules of Go are simple, but the vast number of possible moves in each turn makes it hard for beginners to make decisions. Moreover, on the large 19x19 board, beginners tend to concentrate on localized fighting and overlook the big picture in the process. It is difficult to recognize invaded areas, since an invasion process gradually progresses as new stones are put on the board. For choosing good offense and defense strategies, recognizing the links between the Go stones is important, but it requires experience. Moreover, the match mode visualizes the strength of links between the Go stones. As shown in Figure 5.3-(b), same-colored stones are connected with a line. Moreover, if a dangerous situation occurs somewhere on the

[1]OpenCV: `http://opencv.willowgarage.com/wiki/`

board, a warning message appears to draw the player's attention there to avoid losing the area. In addition, this mode supports stone position recording, so the players can review the match and discuss the efficiency of their strategy.

Kifu mode: Kifu means the record of stone moves, usually made in professional matches. One good self-training method is to repeat the sequences on the board and study professional players' thoughts and strategies. In the conventional way, players have to hold a Kifu book to check the record, which makes it more difficult to concentrate on the board. Moreover, Kifu are usually written in a compressed format, which makes it difficult to recognize intuitionally. Thus in Kifu mode, the application visually indicates the position of the next stone, along with comments on the strategy. Players can follow the sequence without other materials, and train themselves by just interacting with the board.

Joseki mode: Joseki means a set sequence or common pattern that is frequently used in matches. Joseki awareness helps the players to shift the set sequence to fit one's own strategy or improvise when the sequence seems to favor the opponent. In this mode, a player can learn Joseki by interacting with the board. At first, the application show virtual stones on the board, but gradually decreases the number of visible stones as the sequence proceeds. This helps the player to remember the pattern and use it in actual matches.

Tsumego mode: Tsumego is a type of exercise where the player is presented with a game situation, usually with the objective of finding the best sequence of moves in the given situation. In this mode, the positions of the stones are visualized on the board. Players can try out different moves by placing stones on the board, whereas the result and comments explaining key points are displayed as visual feedback (Figure 5.3-(c)). Tsumego mode prepares different levels of questions, and a player can select in the menu (Figure 5.3-(d)).

As shown in these modes, the advantage of our approach is to allow players to get information through the original interaction offered by the Go board and the stones. By superimposing information onto the board, players can concentrate on the match at hand or self-training without fragmenting their attention towards an instructional book and etc. This is important to make it possible for the players to allocate enough cognitive resources for recognizing the situations in the game. Using original game objects as the basis preserves *Ma* and traditional look-and-feel, such as distance between players, touch of a wooden board and sound of stones.

5.3.2 EmoPoker

Poker is one of the most popular card games and it is often played for gambling. It maintains an element of chance because it is an imperfect information game: except for own hand cards, most of the cards are hidden from a player. A poker player needs to make decisions

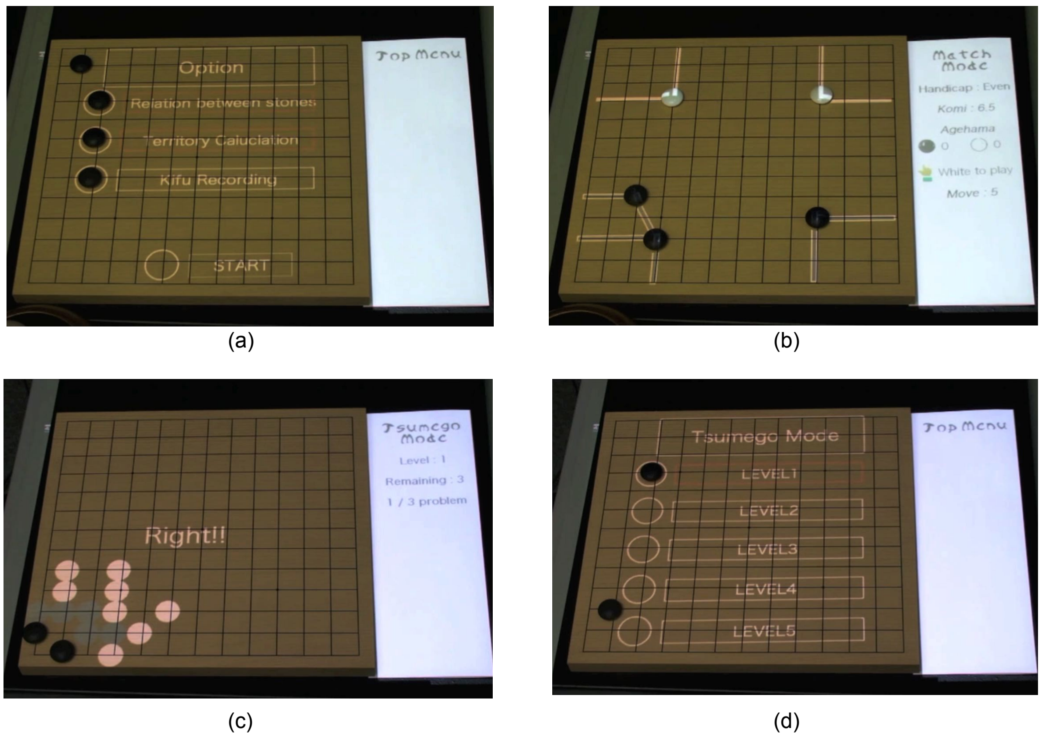


FIGURE 5.3: Feedback examples of Augmented Go system

under uncertainty, such as changing cards, raising a bet and discarding a hand. Like other gambling games, this uncertainty can evoke the mixed feelings of excitement and anxiety. This makes poker play tightly linked with emotional arousal. For example, Kallinen *et al.* examined psychophysiological responses to a poker game [53]. They measured the level of arousal from SCL (skin conductance level), attention from HR (heart rate), and positive/negative emotion from facial EMG (electromyography) activity focusing especially on differences between specific events in Texas Hold'em poker (e.g., "All-in", "Fold", and "Win"). Their experimental results showed that specific game events elicited significant psychophysiological responses. For example, positive emotion, higher attention and higher arousal were observed after a player went All-in (i.e., bet all the money).

Beyond luck, poker requires skill. For instance, understanding probabilities and logical strategies facilitate effective poker play. However, as the common term "poker face" indicates, controlling ones own emotions is also an important skill. In addition to the information apparent on a table (e.g. discarded cards, amount of tips), the players behavior can give clues as to the contents of his or her hand. Moreover, as learned from studies in behavioral psychology and behavior economics, emotional arousal increases the amount of irrational decision making [11, 102]. This is a remarkable fact to consider when looking at gambling, since we also easily get emotionally excited in unpredictable economic activities [131]. Thus even if a player prepares a well-formulated strategy, the strategy might fail as the consequence of the emotions elicited by the game. Moreover, even when a poker player understands the importance of arousal control, it is difficult to objectively and calmly assess how aroused one is while playing. Especially novice players may have a hard time paying enough attention to their emotions, as their cognitive resources are occupied by the demands of the game strategy.

As mentioned above, techniques inherent in human communication such as a bluff are important in poker play, as well as calculating the probability of winning hand. Therefore, instead of targeting video poker, we referred to conventional tangible poker play. For example, game objects (e.g., dealt cards in poker, and chessmen in chess) can be tracked by adding RFID (radio frequency identification) tags. An object tracking system also enables recording game events automatically so that players can review the gaming process, strategy and excited scenes after the game finishes. In this case, object monitoring is conducted automatically. Thus, players can concentrate on playing the game as usual. EmoPoker is also designed to be an instructional aide. Another merit of this approach is that the players can dynamically and seamlessly turn on and off the feedback support while gaming. It is desired that the self-training effect remains after removing the system support. Since the EmoPoker system works based on an original poker gaming environment, players can seamlessly proceed to a practice phase from the training phase, and vise versa. As illustrated in Figure 5.2, EmoPoker consists of four main components: a PC, sensors, actuators, and conventional poker items (e.g., table, chips, and cards).

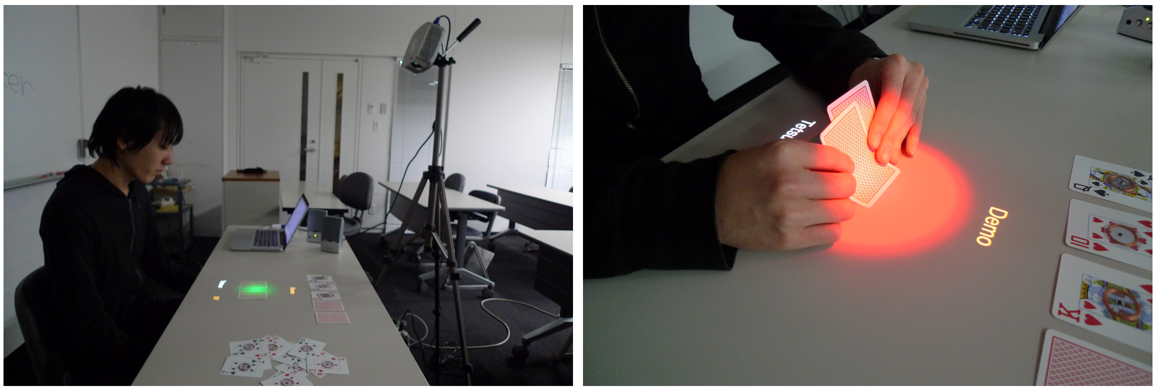


FIGURE 5.4: EmoPoker system and visual feedback example. The player's hand cards are spotlighted by the projector and the color indicates arousal level (e.g., blue: calm, red: aroused). RFID tags are attached to each card in an unobtrusive way so that poker game events can be implicitly tracked and recorded by the system.

Wearable sensors, such as a wireless heart rate sensor, are used to monitor players physiological state. RFID readers and tags are used to track card moves on the table. The readers are embedded into the table, and thin tags are put on the cards face side (i.e., face-down cards are indistinguishable). Actuators are used to provide auditory and/or visual feedback to the poker players. For example, the heart beating sound is played by a mobile device. Such audio feedback could be shared among players, but in the self-training scenario, an individual player is assumed to be listening to the sound through earphones. It is also possible to provide individual visual feedback on the mobile phone (e.g., heart rate transition as an animation graph).

A projector is used to display shared visual feedback to all players (Figure 5.4). In addition to audio feedback, arousal level can be conveyed visually. For example, when a player gets excited, EmoPoker spotlights them with ambient red light. In this case, players share the visual information, so this feature simply invents another poker playing style (i.e., system reveals poker face). This could also be a way to introduce a balancing factor between skillful players and novices. For example, if only the skillful players arousal levels are disclosed in the shared visual feedback, novices can use this hint to distinguish whether the experts are bluffing or not. Visual feedback is also used to guide the gaming process. Poker rules are sometimes complicated for novices. EmoPoker can project guiding information onto the table, such as the dealer and the rounds minimum bet, etc. The EmoPoker system runs on a conventional PC, and it receives and analyzes physiological data transmitted from the physiological sensor. The PC is also connected to RFID readers so that identified card position is restored to a database. We use an open source poker software PokerTH as the games logic engine.

5.4 User Study

As a preliminary user study, the usability of augmented traditional game systems is evaluated. We also compared user experience between an augmented traditional game and a conventional digital PC game. We chose Augmented Go game for the study and prepared an extra PC game mode. The mode supports exactly same contents and functionalities, and the only difference is that physical interaction is replaced with digital one such as mice, keyboards, and displays. 12 subjects participated the user study and training tasks to learn the rudiments of Go game are assigned in each mode. The subjects are asked to answer a questionnaire after each task to subjectively rate user experience in 5-point Likert scale. We also collected comments and feedback from the subjects to identify future work towards more practical game augmentation. For example, most subjects answered stone detection error was not obstructive to complete the tasks (*Mean = 3.91*, 5: strongly agree, 1:strongly disagree). As for the time lag in stone position recognition, it was also enough applicable (*Mean = 3.58*). Visually projected information could be intuitively recognized and understood (*Mean = 4.91*). The user interface of augmented Go game could be used as conventional PC games (i.e., there were no significant differences and difficulties from the extra PC mode) (*Mean = 4.75*). On the other hand, the physical stone manipulation frustrated the subjects especially in repeated tasks, since they needed to physically remove stones to clear the board status and begin a new task (*Mean = 4.00*). The most notable result in the questionnaire was that the physical interaction (i.e., putting stones onto the board) in augmentation mode promoted deeper elaboration rather than conventional mouse-click interaction (*Mean = 4.25*). Some subjects answered that the transaction cost of redo and undo actions in physical interaction is relatively higher than digital interaction. Thus they stopped instant try-and-error learning approach, and started to choose the best hand with careful consideration. Also, the original game items such as Go stones and board developed the sense of real game, and feelings of actual match with human players.

5.5 Discussion

In this section, we summarize findings in the augmented traditional games development. Based on the game development work, design issues are identified from two functional aspects: context monitoring and multimodal feedback.

5.5.1 Context Monitoring

5.5.1.1 Tradeoff between Accuracy and Cost

As described in Section 5.2.2, the accuracy of game object tracking is one important issue that tightly links to a game's characteristic. For example, in Augmented Go system, we applied camera-based visual recognition to detect Go stones' position. Since there are two types of stones (i.e., white and black), it is cheaper than embedding tiny RFID tags into them. However, we have experienced that a player's shadow (e.g., when looking into the board) affects its accuracy and results in a mis-detection. We designed the system as cheap and portable as possible to keep commercialization in sight. It makes the criteria of system harder since robust context monitoring is required to work under various types of environmental settings (e.g., light condition). On the other hand, most of the imperfect information games must identify each game object without changing its appearance. Moreover, the objects in such games are often piled on a table before being dealt to players. For example, in mahjong, where the tiles are piled as a wall, tag-based positioning analysis is not useful. Visual analysis might be an effective approach, but unique visual tags cannot be printed since players should not be able to identify them from their appearance. Thus invisible markers are expected to be common, but it increases setup cost since the number of mahjong tiles is 136 in total and they are too small for printing distinguishable markers on the side.

5.5.1.2 Cognitively Lightweight Interaction

Even though a general approach that covers several kinds of games is difficult to realize, players should not be forced to perform bothersome extra interactions in order to capture game status.

We put importance on keeping *Ma*, and passive observation with originally used game objects is preferred in our augmentation concept. Therefore, a system designer should consider cognitively lightweight interaction in a game. For example, as one guideline in Table 5.1 indicates, secondary user interface should be minimized. Even though players need to provide feedback to the system's behavior (e.g., correct mis-detection), interaction with additional device would disrupt the feeling of flow and degrade user experience. Thus, the additional interactions should be integrated into the original game playing process as natural as possible. If players learn how to manipulate the system with such interactions, the awareness of extra effort will cognitively disappear into background. They do not need to switch attention between main tasks of the game and secondary interaction. Gesture recognition with a camera is one possible approach to allow players to input feedback to the system. Boards of tabletop games could be also smart, and it is one big research question that how game objects-based interaction could be complicated as we demonstrated in Augmented Go system.

5.5.2 Multimodal Feedback

5.5.2.1 Immediate Feedback and Cognitive Load

Feedback design is also an important aspect to consider, since it is tightly linked with human factor issues. In order to prevent from interfering players' concentration, timing, modality and the complexity of information should be sufficiently considered in a design process. Immediate feedback could be frequently provided to a player in a short period, and thus cognitive load to recognize and process the information could increase accordingly. In order to support elaborated and rational decision-making, enough cognitive resource should be allocated to the main task. As we are trying to support sonificated feedback in EmoPoker and Augmented Calligraphy, increasing the variety of multimodality is one approach to avoid conflicts in a cognitive activity. Since most of the traditional games requires visual attention, other types of modalities will allow a player to interpret given messages with different cognitive resource. We believe that immediate feedback should be ambient (i.e., non linguistic) information, and verbal description should be used particularly in the case that a system needs to draw players' attention (e.g., report a system error).

5.5.2.2 Accumulated Feedback and Emotional Engagement

As we pointed out in Section 5.2.3, this augmentation changes relationship between players and traditional games: players develop their skill in a longer period rather than temporary enjoying game play. Thus the contents of feedback should be changed according to the progress of players. As a result, the playfulness of augmented games will be designed with assuming players to be involved in longer period. As conventional video games make efforts on the issue, the augmentation system also should keep players' interests not to bore them. Thus incentive mechanisms should be incorporated into the service design. For example, social competition among digitally connected players is one traditional approach to spontaneously and continuously join the game. On the other hand, if feedback is design to stimulate players' emotion, it could affect decision-making process [102]. As well as immediate feedback, stimulus that could induce emotional arousal should be carefully designed, otherwise the main purpose of the system (i.e., training players) cannot be achieved.

5.6 Conclusion of This Chapter

In this chapter, we developed the concept of augmented traditional games with two case studies: Augmented Go and EmoPoker. Traditional game augmentation aims at adding new value and playful features to a traditional game with keeping its original look-and-feel. The preliminary user study result shows the positive feedback from subjects, which implies physical interaction increases the sense of playing against human player. The transaction cost inherent in physical interaction also induces higher elaboration at a decision moment. We also summarized several findings in the traditional augmented game development. In the future work, we will break down the experiments in order to investigate in the elementary design factors that differentiate user experience from conventional digital game play.

Figure 5.5 shows the focused ADSS framework components in this chapter. Augmented traditional games utilize AR technology instead of completely digitalizing a game and allow physical interaction with original game items to the players. The system implicitly monitors user context such as emotion and environmental context (i.e., game events) with augmented game items. Then it provides decision support information based on the user and training models.

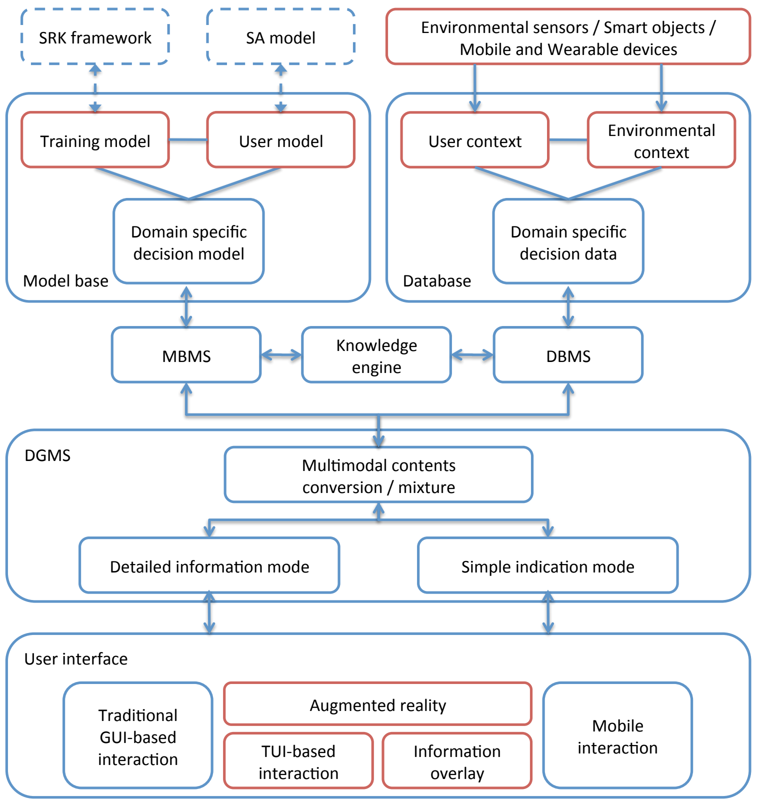


FIGURE 5.5: Focused components in this chapter

Chapter 6

Decision Inducement with Activity-based Micro-Incentives

6.1 Background

Two emerging topics in pervasive computing and HCI research are persuasive applications and electronic payment systems. Pervasive persuasive applications seek to alter user behavior through the means of a feedback loop between sensor-tracked user behavior and system output [55, 107]. In many cases the aim is to encourage environmentally responsible behavior. In electronic payment systems, pervasive technologies have been used to implement and deploy mobile payment solutions that enable small payments in a discreet and effortless manner [67]. In this chapter, we combine these two topics, exploring the possibility of using pervasive computing technologies to create small activity-based economic incentives that discreetly steer consumer behavior towards desired patterns. Applications for such technology can be found both in business as well as in resource conservation.

Free resources shared by a number of people, such as a public toilet or the natural environment, tend to be overused in a process called the tragedy of the commons [39]. This happens because each individual derives a personal benefit from using the resource, while any costs are shared between all the users, leading to inconsiderate use. An example of such behavior is the wasteful use of free plastic shopping bags that are filling landfills. A common strategy to dealing with the tragedy of commons is to impose a tax on the use of the resource. In Japan, plastic bags are usually free, but in Finland shoppers have to pay for them. This provides an economic incentive for individuals to re-use shopping bags. This kind of economic incentives have been found to be a powerful way of persuading people to change their behavior.

However, vendors and regulators are currently limited in what behaviors they can set a price on. Only a limited range of consumer activities can be feasibly observed in a limited number of locations (e.g., at the cashier in a store). Sometimes manual observation is used to enable complicated activities to be priced (e.g., using a plastic bag vs. bringing one's own shopping bag), but the cost of human resources imposes a limit on this approach. If activities could be recognized with less cost, activity-based micro-pricing would become possible in a great range of application areas. Some of these applications could be aimed towards preventing over-use of environmental resources, while others would simply enable businesses to charge for their services in a more fine-grained manner, overcoming inefficiencies and stimulating commerce[1].

On the other hand, real consumer behavior does not fully conform to the economic expectations of rationality. In our previous work on mobile payment systems, we found that the consumer's emotional response to the payment system was an important factor in economic behavior [62]. This topic area has been studied extensively in the field of economic psychology, where certain predictable patterns such as risk-aversion have been identified. In this study, we draw on findings in economic psychology to propose four basic incentive models for activity-based micro-pricing. Our experimental studies suggest that consumers make different decisions depending on which of the incentive models is used, even when the total economic impact is the same. This implies that activity-based micro-pricing systems can make services persuasive by simply changing transaction flows, without necessarily altering total amounts. In an experiment with a prototype of the billing system we also identified a number of issues that must be addressed in future research.

In the following sections, we describe the concept of activity-based billing systems and discuss how economic incentives affect a consumer's decision making process. We then present an overview of a system architecture for an activity-based billing system that supports four different incentive models. Each model corresponds to a different set of transaction flows, so that services can alter consumer behavior by choosing a different model. Then, based on user evaluation and lessons learned in the prototyping process, we discuss the feasibility of the concept and the main research challenges that remain.

6.2 Activity-based Billing System

6.2.1 Ubiquity of Payments

The rapid growth of mobile computing has transformed mobile devices to a medium of payment. Mobile devices are used to initiate, activate and confirm payment transactions in various kinds

[1]Pay-per-use: `http://www.accenture.com/global/services/accenture_technology_labs/r_and_i/payperuseobject.htm`

of services, collectively known as “mobile payments” [56]. Mobile payments are not simply an extension of normal electronic payments, as they free users from physical constraints (i.e., time and place) and allow flexible decision making that adapts to the mobile use context [68].

While new features have been added to mobile user terminals, our surrounding environments are also increasingly being embedded with ambient intelligence. To support daily tasks and events, living environments are expected to become sensitive to the presence of users. For example, elder people’s activities might be monitored with sensors in order to automatically detect emergency situations. Thus computers, sensors and network connectivity have been installed into buildings, parks, trains and everyday objects [57].

The maturity of mobile payments and the increasingly prevalent ambient intelligence technology suggest the notion of *ubiquitous payments*. As argued in [32], sufficiently fine-grained tracking of user activity makes it possible to implement accurate pay-per-use payment models in commercial services. With sufficient context information, vendors can precisely calculate the economic cost of a consumer’s action, and bill accordingly. Ubiquitous interaction techniques give consumers real-time information and control over spending. Transactions of small nominal value take place frequently, since payment is associated with consumer actions. In ubiquitous computing environments, payments become ubiquitous, too.

In this section, we introduce the idea of activity-based billing systems. Figure 6.1 illustrates an example scenario with two types of transaction flows, case A and case B.

6.2.2 Pricing Consumer Actions

A common problem for managers of busy cafe and restaurants is that customers linger in the space for a long time after their initial order without placing any additional orders, whilst taking up space from other potential customers. It is difficult to keep track of how long each customer has stayed without making some effort to monitor them. Moreover, it would be troublesome for the staff to collect a fee for the overstay even if such customers could be identified accurately. As a result, limited seats remain occupied and from the manager’s point of view, resources used inefficiently.

If the cost of customers’ time spent in the cafe could be automatically priced, the situation would be different. Consider a billing system that notifies customers how much they have to pay for spending time in the space: “An additional fee will be charged for further stay: $0.1 for every 10 minutes”, for instance. If the customer continues to stay despite the notification, the system begins to charge the time on their mobile phone. In this way, the manager can charge an additional fee from overstaying customers, improve the availability of seats with recouping the cost of lost business from the occupied seats (case A). On the other hand, it is also possible to give rebates to customers who take actions that are beneficial to the business. For example, coffee price can be reduced if a customer orders coffee to go in lunchtime (case B).

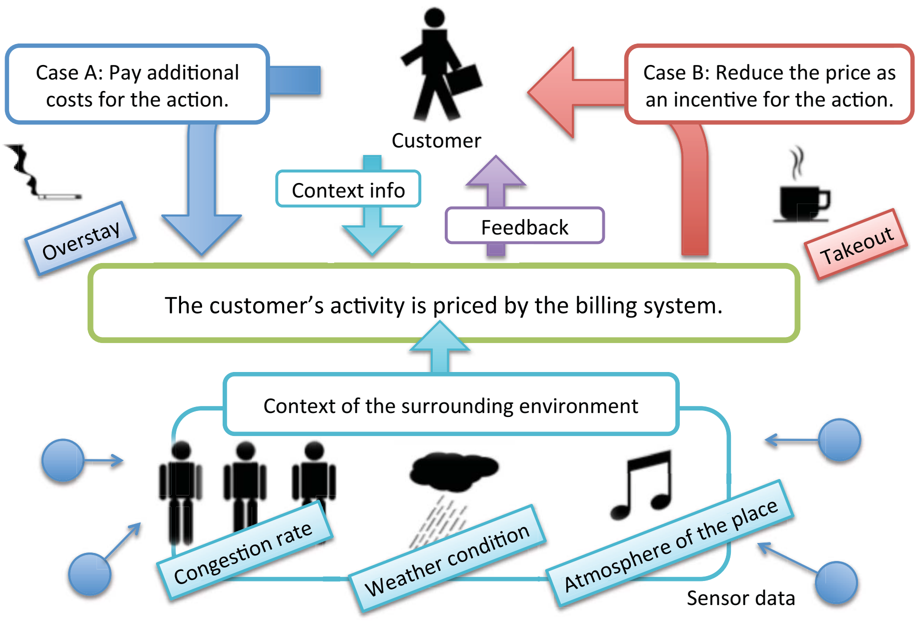

Case A: Pay additional costs for the action.
Customer
Case B: Reduce the price as an incentive for the action.
Context info
Feedback
Overstay
Takeout
The customer's activity is priced by the billing system.
Context of the surrounding environment
Congestion rate
Weather condition
Atmosphere of the place
Sensor data

FIGURE 6.1: Activity-based micro pricing

Our key idea in this scenario is coupling economic incentives with specific actions, and implementing the resulting incentive system using ubiquitous computing technologies. Context recognition techniques provide support for tracking consumers' actions. The payment or rebate can be carried out smoothly by mediating it with a mobile payment system. Furthermore, different pieces of context information can be used by the vendor to determine the current price of a given action. If the overstay charge increases as the cafe becomes increasingly packed, it will motivate customers to move on speedily. The occupancy rate of seats can be manually checked, or automatically detected with pressure sensors embedded in chairs. Methods such as discount coupons and selective taxation are sometimes used for creating economic incentives that steer consumer behavior, but compared to the approach outlined above, they are inflexible and static.

6.2.3 Psychological Factors in Incentive Design

The price of an action should be based on the characteristics of the action as well as the desired incentive effect. In the example scenario, the cafe charged an additional fee for overstay at every 10 minutes, since staying is a continuous state of action. On the other hand, taking out a coffee is a one-time action, after which the customer departs from sphere of the service. The rebate for this action is therefore only conducted once and the sum is higher than in the continuous payment case.

The prices of these actions will nevertheless be relatively small compared to the price of, for example, a cup of coffee, because the incentives are not a core part of the service. Under the notion of ubiquitous payments, transactions happen anywhere, anytime a consumer takes a relevant action. This leads to an increase in the frequency of transactions, and the price per one action will correspondingly decrease. Thus in the activity-based pricing mechanism design, we must consider how to affect consumers' behavior with stakes of relatively low nominal value.

It is well recognized that consumers' real economic behavior is emotional and sometimes leads to irrational decision making [10, 74]. In some cases this lack of economic rationality follows quite predictable patterns. For example, people tend to avoid risk associated with uncertain gains. If there are two choices, such as (a) receive a guaranteed sum of \$10, and (b) receive \$20 with a 50% probability, choice (a) is preferred even though the expected utility is same. While risk-neutral and risk-loving attributes also exist, this preference that most people have is called risk-averse. Risk-averse decision making results from so-called loss aversion bias explained in *prospect theory* [52, 120].

According to prospect theory, an individual's value function for any kind of gains and losses takes the shape of an asymmetric S, as illustrated in Figure 6.2. Variable U does not represent actual gains or losses in money, but the utility that a consumer perceives. A floating reference point divides the value range of gains and losses into positive and negative segments. If variable x is negative (representing a loss), the slope of the curve is steeper than in the positive (gains) case. Thus the absolute value of $f(-\alpha)$ is greater than $f(\alpha)$, indicating that people experience greater impact from losses than from correspondingly large gains. This leads to the observed loss-averse behavior.

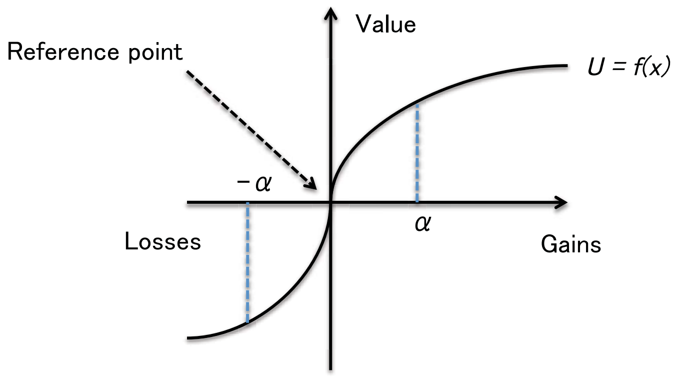


Figure 6.2: A hypothetical value function in prospect theory

The practical implication of this asymmetry for economic incentive systems is that surcharges (losses) and rebates (gains) can be used to create different kinds of incentivizing effects even when their overall economic impact is the same. For example, during busy lunchtime hours, a cafe could charge patrons in the form of a small initial fee and additional time-based surcharges, encouraging short stays. During quieter hours, when the manager wants patrons to linger for as long as possible, the equivalent sum could be charged in the form of a bigger initial fee and tiny time-based rebates. The suitable model can thus be chosen according to desired behavior. The graph also shows that the marginal increase or decrease in perceived value diminishes as variable x goes further from the reference point. This suggests that a well-designed incentive system can realize disproportionately large incentivizing effects with trivial sums of money.

In behavioral psychology, an important factor in effecting behavioral changes is the timing of feedback. Immediate feedback is superior to delayed feedback, and suitable scheduling can be used to enhance the effect further. In activity-based micro-pricing, calm feedback methods can be used to inform customers in real time of micro-payments and rebates that are being conducted as a consequence of their actions. Notifications of time-based charges can be conducted at suitable intervals.

6.3 System Architecture

In this section, we describe the system architecture of the billing system (Figure 6.3). The system is composed of user side modules and service side modules.

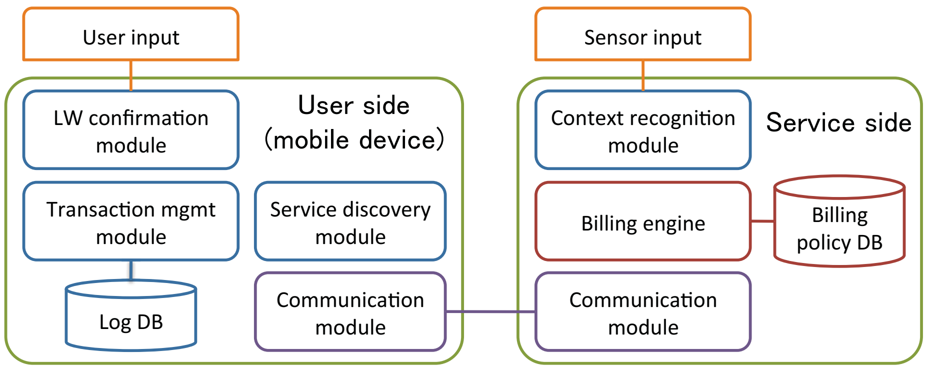


FIGURE 6.3: System architecture of the billing system

6.3.1 User Side

Consumers are assumed to have a mobile device for communicating with services and conducting transactions. The transaction management module handles transactions, such as payments and rebates. The module also records all transactions into a database called Log DB, which can be used for error recovery. The service discovery module detects available billing systems within range of the device's wireless connectivity (e.g., Wi-Fi, Bluetooth). The communication module establishes a connection to the detected billing system and encrypts data before transferring it to the service.

The LW confirmation module stands for "Lightweight confirmation". Since activity-based transactions happen frequently, each transaction should impose minimum cognitive workload on the consumer. Traditional PIN code based payment confirmation steps are too heavy; instead, the lightweight confirmation module allows consumers to approve payments using simple gestures, such as tapping the device twice, without even removing it from the pocket. Moreover, a completely automatic payment mode with no user approval steps could be feasible if the system is sufficiently trustworthy and the sums at stake minimal. We will return to this cognitive transaction cost issue in more detail below.

6.3.2 Service Side

The billing system must recognize consumer's activities within the scope of the service in order to price and bill the behavior, and provide feedback. The context recognition module handles inputs from consumer's devices as well as sensors that are installed in the environment. Context information required by the billing engine is generated by pre-configured sensor analysis algorithms. The billing engine gathers the necessary context information and consumer's personal information, where necessary. The service's billing rules are pre-configured into a database called the billing policy DB; the engine calculates the applicable payment or rebate amount according to the policies set in the DB.

The key challenges in implementing the system are designing an effective context recognition module and suitable billing policies that lead to desired incentivizing effects. In the following sections, we will build on the earlier discussion on economic incentives to introduce four basic micro-pricing models that can be used as the basis of such policies.

6.4 Four Basic Models of Micro-Pricing

In this section, we introduce four basic models of activity-based micro-pricing. The models consist of simplified transaction flows, and each one represents different incentive design. Thus it is possible to replace, combine, and switch the models according to a vendor's objectives.

6.4.1 UbiPayment Model

As illustrated in Figure 6.4, the UbiPayment model is used to charge additional costs upon a consumer's specific actions. This transaction flow corresponds to the case A in Figure 6.1.

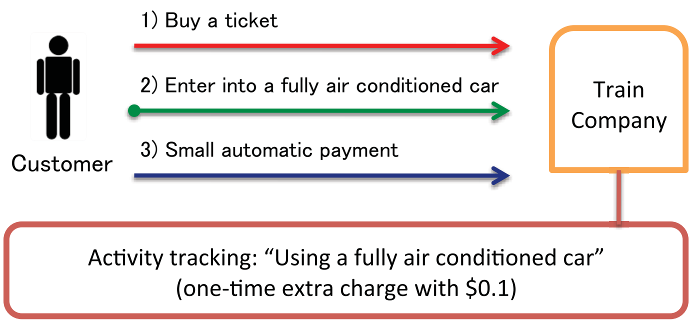

1) Buy a ticket
2) Enter into a fully air conditioned car
3) Small automatic payment
Customer
Train Company
Activity tracking: "Using a fully air conditioned car"
(one-time extra charge with $0.1)

FIGURE 6.4: Transaction flow of the UbiPayment model

In the UbiPayment model, the initial cost of a service would be smaller than in the conventional case, because vendors can expect additional revenues from the micro-payments. It also has certain benefits to consumers: they obtain the possibility of leaving out unnecessary options (i.e., actions) that are normally bundled into the price of the service, reducing costs. The UbiPayment model comes from an earlier study we conducted on the possibilities of ubiquitous payments, explained below.

6.4.1.1 UbiPay System

In [62], we developed a ubiquitous billing system called UbiPay. UbiPay consists of two components: a mobile device with an accelerometer and a server that offers services. Money can be charged into the device so that consumers can purchase items and services in a mobile payment fashion. The main difference compared to conventional mobile payment systems, such as Suica[2], is the price range of the payments. UbiPay aims at facilitating the billing and payment of services of extremely small value by minimizing the cognitive workload of the payment process. For example, in Tokyo, trains have two types of air conditioned cars: fully air conditioned cars and reduced air conditioned cars. The fully air conditioned cars are more comfortable and costly to operate than the reduced air conditioning cars, but the ticket price for both train cars is the same. The action "using a fully air conditioned car" cannot be billed separately at the moment. The UbiPay system breaks this bundle by making it possible for the train operator to charge different fees from commuters according to their choice of carriage.

The unique feature of UbiPay is that the consumer can assign three configurable user interaction modes to be invoked at different price ranges: automatic payment, light confirmation with simple gestures, and authentication with a PIN code. This feature aims at decreasing the transaction costs of the purchasing process. Transaction cost is an economic term that is used to refer to any cost, either in the form of money, time, effort or other disutility, which is incurred in the process of making an economic exchange [81]. In services with a very small single purchase

[2]Mobile Suica: `http://www.jreast.co.jp/mobilesuica/index.html`

value, payment-related transaction costs can represent a very large fraction of the total cost. It is not feasible to charge on a per-use basis for such services using conventional payment systems (e.g., stored value card, credit card), because the act of payment can incur more effort than the service itself is worth.

In the train car example, when the consumer steps into a fully air conditioned car, UbiPay recognizes the action and withdraws an air conditioning fee from the consumer's mobile device. If the air conditioning fee is in the price range that is associated with the automatic payment mode, billing will be completed immediately and automatically without any distraction to the consumer. The consumer can view the payment history later to see the total amount of money spent on air conditioning in a month, for example.

6.4.1.2 Concerns with Automatic Payments

In a user evaluation of the UbiPay system, certain technical and psychological problems were identified in implementing it in practice. We performed an experiment to evaluate the usability and acceptability of the system, where 14 participants joined the experiment, most of them were students (male: 10, female: 4). We asked them to use the system in a controlled setting and fill in a questionnaire afterwards. Results of the experiment suggested that users feel reluctance towards fully automatic payments.

Firstly, the respondents had concerns regarding unnoticed manipulation and fraud. To the question "How did you feel about letting the system conduct automatic payments without your awareness?", five participants responded "I felt reluctant" and eight participants answered "I felt slightly reluctant". This was due to the potential of incorrect payments, such as the vendor making a mistake, the consumer misunderstanding the pricing method or someone attempting to commit a fraud. In the prototype, the consumer could accidentally pay other's bills, increasing distrust.

Secondly, it was difficult for the vendor system to recognize consumers' activities correctly. In order to achieve fully automatic payments, wireless communication must be used between the handheld device and the vending machine. Even though many wireless communication technologies (e.g., Bluetooth, wireless LAN, ZigBee) are available, none of them can eliminate the potential for incorrect payments. For example, if multiple handheld devices are detected in the vending machine's communication range, it is difficult to identify who is actually purchasing the items. Therefore, new techniques for identifying the consumer are needed in achieving automatic payments.

We created the light confirmation mode to reduce the occurrence of incorrect payments, but it introduced certain new issues. In the light confirmation mode, the consumer had to make gestures with the handheld device, such as tapping it twice, in order to approve a payment request. However, the recognition system had a less than perfect recognition rate, meaning that it could be quite difficult to use, sometimes even increasing instead of reducing transaction cost.

As is apparent from the above, it is challenging to realize an automatic or semi-automatic payment system that will be perceived as sufficiently safe and secure by consumers. We therefore decided to experiment with a different approach, reversing the transaction flow and thus placing the semblance of risk on the vendor's side.

6.4.2 UbiRebate Model

As pointed out in the previous section, the psychological reluctance is an important issue in payment systems. We therefore considered how the payment system can reduce this reluctance without changing the total volume of transactions. As shown in Figure 6.5, the UbiRebate model returns some amount of money back to the consumer according to the consumer's actions.

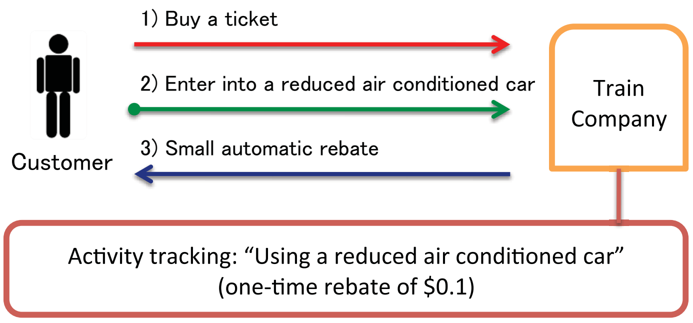

1) Buy a ticket
2) Enter into a reduced air conditioned car
3) Small automatic rebate
Customer
Train Company
Activity tracking: "Using a reduced air conditioned car"
(one-time rebate of $0.1)

FIGURE 6.5: Transaction flow of the UbiRebate model

The concern over incorrect payments or system errors comes from "paying" without awareness, as in skimming of smart cards. There are three possible approaches to solving this issue. One is adding a confirmation phase after the transaction completes. Another possibility is a tracking back feature that allows a consumer to decline incorrect transactions. It is too difficult to implement this feature, however, since purchasing items in the activity-based billing system are consumers' activities, not physical objects. In traditional payment cases, incorrect transactions can be tracked back by returning the item. It is not possible to track back consumers' activities, however, so payments cannot be easily refunded. Moreover, proving the invalidity of activity-based billing is also difficult, since human bodies are always in a transitional state. Therefore, it is unrealistic and prohibitively costly to validate system errors for every claim.

The third approach is eliminating the "paying" process from the automatic payment system. In this approach, the initial payment is set higher than in the previous example, but rebates are paid for particular activities instead of additional payments. The concept of this approach comes from psychological aspects of consumption behavior. In general, people resist paying from their wallet, but want to "get" something instead [67]. The UbiRebate model automatically gives micro-rebates that replace the automatic payments for opposite actions. The UbiRebate model enables vendors to implement an automatic billing system that overcomes the consumer reluctance issue. Contrary to the automatic payment case, consumers may feel happy when they get a rebate for their actions, even though the total economic impact would be the same. Also, there is no satiation in the pay-as-you-do model, but the amount of maximum payment is ensured in the rebate-as-you-do model.

As with the first model, UbiRebate should inform the consumer on how what kind of behaviors will lead to rebates. Then, after the transaction is completed, the system shows the results to consumers. However, the confirmation phase is not important in the case of UbiRebate, because the onus of ensuring the correctness of payments is shifted on the vendor. Rebates that the consumer feels entitled to but does not receive should create less problems than incorrect payments. Also, instead of real money, a virtual currency such as airline miles or loyalty program points could be used in the rebate transactions. This would further alleviate the reliability problem.

6.4.3 UbiReward Model

The third model is UbiReward, which removes the initial payment process from the UbiRebate model (Figure 6.6). Unlike in the UbiPayment and UbiRebate models, consumers do not pay any fees for service use. Vendors indirectly obtain revenues from the consumers' actions, which are motivated by direct economic incentives. Such actions can include watching advertisements, filling out surveys, doing actual productive work, or saving energy.

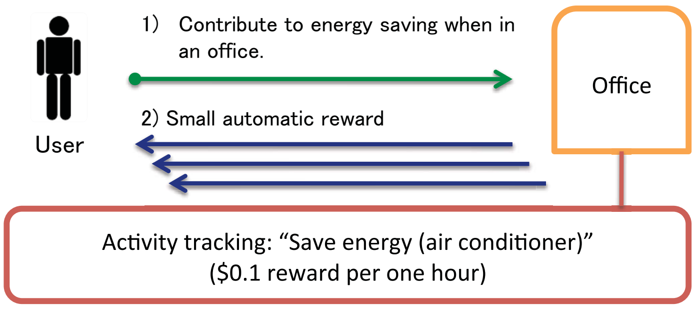

User
1) Contribute to energy saving when in an office.
Office
2) Small automatic reward
Activity tracking: "Save energy (air conditioner)"
($0.1 reward per one hour)

FIGURE 6.6: Transaction flow of the UbiReward model

A practical use case for the UbiReward model is a power saving scenario in an office environment. As many companies encourage employees to save energy, reducing unnecessary power consumption helps cutting costs. The UbiReward could be used to encourage employees and customers alike to take actions that save energy. For example, employees can get a micro-reward if they use air conditioners with unnecessarily high power for one hour. Small sensors are embedded into the office environment, so that temperature can be sensed with fine granularity. Needlessness is defined by the difference between inside and outside temperatures. The location of the employee's seat and other factors that affect the conditions (e.g., layout of the partitions, structure of the office room) are taken into account in the pricing policy.

6.4.4 UbiTrade Model

The UbiTrade model involves multiple participating consumers. One of the three models explained above is used as a basis, but the pricing of the activities reflects other participant's behavior and interests. This can be realized through a marketplace where buy and sell offers on actions combine to form market prices. Figure 6.7 shows a transaction flow of the UbiTrade model.

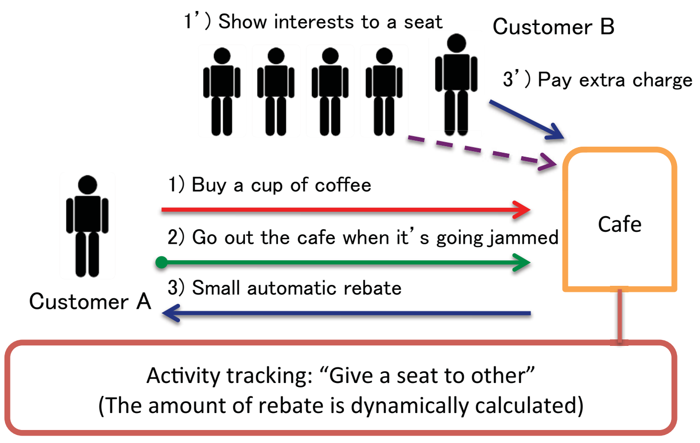

1') Show interests to a seat
Customer B
3') Pay extra charge
1) Buy a cup of coffee
2) Go out the cafe when it' s going jammed
3) Small automatic rebate
Customer A
Cafe
Activity tracking: "Give a seat to other"
(The amount of rebate is dynamically calculated)

FIGURE 6.7: Transaction flow of the UbiTrade model

With the UbiTrade model, the price of consumer actions is dynamically calculated in a way similar to a stock price. For example, in a cafe, the value of a seat changes according to the occupancy rate of total seats. At busy times, the value of occupied seats becomes higher than at normal times. If a customer is sitting on one of the seats, they have two options: either continue using the seat, or give the seat to another customer. Usually, the people who are waiting for vacant seats cannot explicitly encourage the occupants to leave the cafe. With the UbiRebate model, however, the waiting people can price the customer's voluntary action (i.e., giving away the seat) and inform the occupant of this incentive in an explicit way. The occupant can make the decision according to the amount of money at stake; whether it is worth taking the action. If the occupant decides to leave the cafe, a rebate is given to them and charged from the bidder who was waiting for the seat. Both direct and indirect negotiations among customers are conceivable. Direct communication might not be comfortable for some customers, however, so the cafe should mediate the communications in one way for the other.

In a UbiPayment model based scenario, users could make offers on how much they are willing to pay to the cafe for the seats, meaning that the seat under you may become increasingly expensive to you as more people bid for it.

6.5 Evaluation

In the above sections, we explained four basic models of activity-based micro-pricing. In order to evaluate the feasibility of these concepts, we performed two experimental studies. The research questions shown below were explored using two corresponding prototype applications.

RQ1: How strongly does the difference between incentive models affect consumer behavior?

RQ2: What is the social acceptability of activity-based micro-pricing systems?

In the sections below, we describe the methods, implementation and results of the evaluation.

6.5.1 Economic Incentives and Behavior Alteration

6.5.1.1 Method

In this study, we developed a flash application to perform an experimental study on the RQ1. This application gives the task to a participant, which requires continuous actions in a short period of time. Figure 6.8 shows a screenshot of the application.

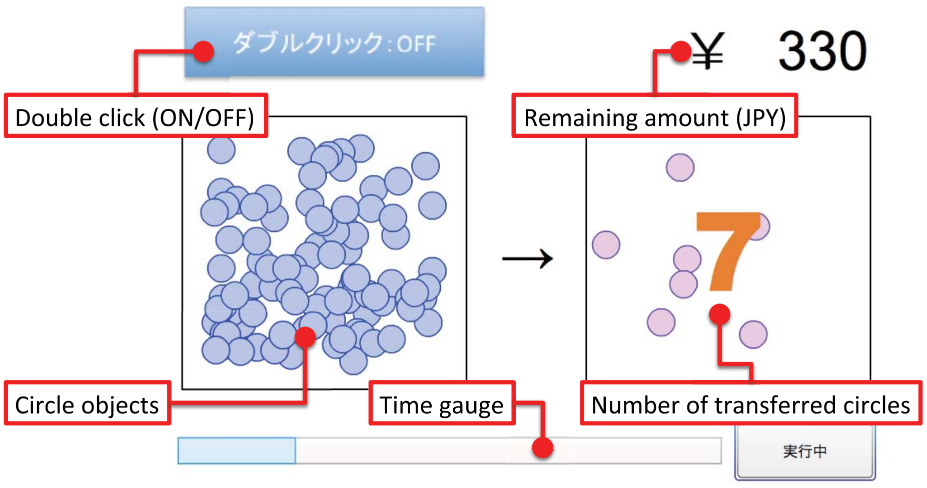


FIGURE 6.8: A screenshot of the flash application

In the application window, two boxes are shown in the right and left side. 100 circle objects appear when the examination starts, and the participant is instructed to drag-and-drop the circles from the left box to the right one. The counter in the right box shows how many circles were transferred by the participant, and the time gauge shown below the window indicates how long time remains. 60 seconds were given for each round, and the participant had to transfer as many circles as possible. Also there is another way to transfer the circle objects. When the participant turns on the button left above, a special tool is enabled and the circles can be automatically transferred by double-clicking them. Using this tool is much easier than dragging the circles, but economic penalties are given per certain amount of time.

In this study, we prepared two application scenarios that correspond to UbiPayment and UbiRebate model, in order to evaluate the difference of incentive models. At the beginning of each round, 500 JPY [3] is given to a participant. Also in addition to this base point, bonus points are given at the end of test. The bonus point is calculated from the number of transferred circles. Then, the participant starts the task with below conditions according to the scenario:

UbiPayment: No initial cost has to be paid, but 5 JPY is withdrawn per 2 seconds when the tool is enabled.

UbiRebate: 200 JPY is withdrawn at the beginning as an initial cost, but 5 JPY is rebated per 2 seconds when the tool is NOT enabled.

In order to avoid making strategies and to observe psychological effects, only 1 round test was conducted for each scenario per 1 participant. Before starting the tests, a practice phase was allowed to get used to manipulate circle objects with a laptop PC's touchpad. 12 university students joined this test as participants (male: 11, female:1, age: 22-25), and some questions were asked after finishing the test.

6.5.1.2 Results

Table 6.1 shows the experimental result of the flash application test. Total number of transferred circles (a), the remaining amount of points (b), and time taken to use the tool (c) were recorded with checking whether the tool is used or not. As shown in the table, the number of transferred circles were almost same between the UbiPayment and UbiRebate scenarios. However, as (c) indicates, participants tended to use the tool longer time in the UbiRebate model than the UbiPayment model. It means that the UbiRebate model could strongly encourage the participants to use the tool, even though a bigger amount of initial cost was withdrawn.

[3] 1 USD $\simeq$ 98.84 JPY as of 15th April 2009

TABLE 6.1: Averaged experimental result of the flash application test. (bonus point = 5 * the number of transferred circles)

	UbiPayment	UbiRebate
(a) Transferred circles (num)	60.91	61.33
- Dragged circles (num)	36.58	31.75
- Double-clicked circles (num)	24.33	29.58
(b) Total point (JPY)	762.08	702.5
- Base point (JPY)	457.50	395.83
- Bonus point (JPY)	304.58	306.66
(c) Time taken to use the tool (sec)	17.00	38.33

We also asked several questions to the participants, in order to investigate how they feel the difference of the models in the experiment. Some of the participants commented that they felt anxious or sad when using the tool in the UbiPayment scenario, because their money was lost. As we explained above, the loss aversion worked and it prevented them from using the tool. On the contrary, the rebate made the participants elevated. The UbiRebate model also led to use the tool, because no risk for losing money was concerned in the decision.

6.5.2 Acceptability of the Micro-Pricing World

6.5.2.1 Method

As a second study, we set up an experimental environment in our laboratory. This study corresponds to the RQ2, and we wanted to evaluate how consumers feel about the concept of activity-based micro-pricing in real settings. The experimental environment was presented as a Japanese-style comic cafe (*manga kissa*), and several kinds of services were installed as shown in Figure 6.9. For example, participants could play a game (Nintendo Wii), read comics, browse web sites, and use a toy gadget (Chumby). A display shows a list of possible actions where participants can check which actions are priced and how much they cost. A prototype of the billing system was developed for this study.

A mobile device (iPod Touch) was handed to participants before the study started. 1,000 JPY was virtually charged into the device and participants could freely enjoy the services using the money. The remaining amount is displayed on the mobile device (Figure 6.9), and the participant can configure feedback settings by flipping the window. Under default settings, the mobile device periodically checks whether any transactions have been conducted or not with a 10 second interval. If remaining amount is changed, sound notification is given to the participant. The participant can change the feedback interval and enable/disable the sound notification. This feature was implemented to provide insights into appropriate feedback design.

In the same as with the first study, we prepared two scenarios: UbiPayment and UbiRebate. Table 6.2 shows priced actions and corresponding price in the UbiPayment scenario. Also Table

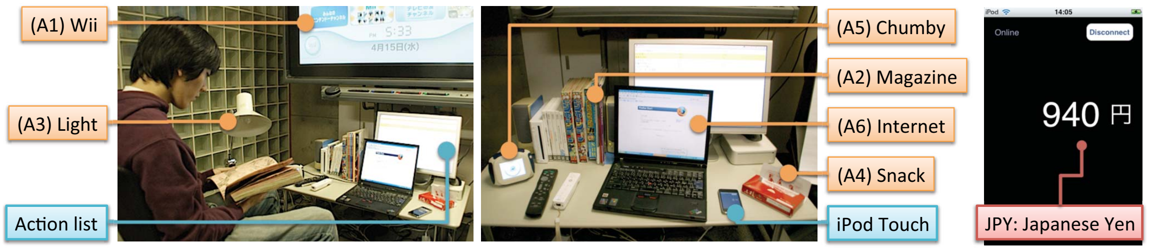


FIGURE 6.9: A comic cafe setup and a mobile device user interface for the experimental study

TABLE 6.2: Action list in the UbiPay scenario

ID	Action	Interval	Payment
A0	Initial cost	-	100 JPY
A1	Play Wii	1 min	10 JPY
A2	Read a comic	1 min	5 JPY
A3	Turn on a light	1 min	1 JPY
A4	Have a snack	(per 1 snack)	10 JPY
A5	Play Chumby	1 min	5 JPY
A6	Browse web sites	1 min	5 JPY

TABLE 6.3: Action list in the UbiRebate scenario

ID	Action	Interval	Rebate
A0	Initial cost	-	500 JPY
A1	NOT play Wii	1 min	10 JPY
A2	NOT read comics	1 min	5 JPY
A3	NOT turn on a light	1 min	1 JPY
A4	Have a snack	-	- JPY
A5	NOT play Chumby	1 min	5 JPY
A6	NOT browse web sites	1 min	5 JPY

6.3 shows the action list in the UbiRebate scenario. For example, in the UbiPayment scenario, a participant has to pay 100 JPY at the beginning. Then, if the participant plays a Wii game, 10 JPY will be automatically withdrawn from their mobile device per minute. In contrast, in the UbiRebate scenario, the participant has to pay a bigger initial cost than in the UbiPayment scenario (500 JPY). However, if the participant does *not* use services, the corresponding amount of money is automatically rebated to their mobile device. For example, if they play a Wii game, the sum of all the actions' prices excluding A1 is rebated each minute.

Participants were instructed to stay in the room for 10 minutes for each scenario. While the study was performed, we monitored the room with a web camera and used the Wizard-of-Oz technique instead of using the context recognition module explained in Figure 6.3. In the monitor's display, a check list is shown and performed actions can be checked accordingly. Then the amount of payment/rebate is automatically calculated, and the remaining amount is updated on the mobile device's display. Six university students joined this study as participants (male: 5, female:1, age: 22-25). We recorded how they configured the feedback settings in the experiment. We also asked several questions about their impressions regarding activity-based micro-pricing at the end of the test.

6.5.2.2 Results

On average, the participants performed approximately 2.5 different types of actions in the UbiPayment scenario, and 2 kinds of actions in the UbiRebate scenario. As shown in Table

6.4, they felt that the test duration was not long (Q1), and the price of payments/rebates was neither expensive nor inexpensive (Q2 and Q4). The responses to Q3 and Q5 suggest that each model affected the participants' mental state: they felt anxiety over UbiPayment model, and satisfaction in the UbiRebate model. However, some of the participants pointed out that camera monitoring was one reason for their anxiety, so we cannot assert that loss-aversion effects in the UbiPayment model are the only explanation for the responses to Q3.

Table 6.4: Questions presented to the participants and the mean responses (5-point Likert scale)

	Question	Point
Q1	What do you think about the duration of the test? (5: too long - 0: too short)	2.5
Q2	What do you think about the size of the payments in the UbiPayment scenario? (5: too expensive - 0: too inexpensive)	3.0
Q3	Did you feel nervous or anxious in the UbiPayment scenario? (5: strongly felt - 0: did not feel)	3.1
Q4	What do you think about the size of the rebates in the UbiRebate scenario? (5: too big - 0: too small)	3.0
Q5	Did you feel happy or satisfied in the UbiRebate scenario? (5: strongly felt - 0: did not feel)	3.3

In the experiments, only two participants customized the time interval of the sound notification. In the questionnaire, they commented that frequent sound interruption was annoying. One of the participant felt that even rebating becomes annoying due to frequent notification. Alternative ideas in their comments are notifications with vibration, notifications without sound, and notification with price limitation. Almost of all participants preferred to decrease notification frequency, and 7.2 minutes was the average interval that they would set if the activity-based micro-pricing got realized. Also some of the participants commented that simple authentication or confirmation is needed in the payment transactions, but it is not necessary in the rebating transactions.

6.6 Discussion and Future Work

In this chapter, we have introduced the concept of activity-based micro-pricing. Even though the concept has potential for new business opportunities and resource conservation, the cost of manual activity recognition prevents its commercial adoption. Therefore, we proposed to apply ubiquitous computing technologies to implement effective activity recognition, and introduced our idea to bill consumers' activities in a wide variety of situations. In our vision, vendors can effectively charge for users' activities on a per-action basis. At the same time, wider choice is provided to the consumer, who can avoid unnecessary costs that are normally bundled in the price of various services. In order to bring the concept into a concrete shape, we introduced four different models. User evaluation suggests that the choice of incentive model affects consumers' behavior and has an impact on the social acceptability of the activity-based micro-pricing.

In this section, we discuss in more detail some insights and challenges identified in the experiments, consider possibilities for future work, and reflect on related work in incentive design and persuasive computing.

6.6.1 The Gap between Experiments and Reality

Since we performed experiments in a controlled laboratory setting, there could be gaps between the experiments and real world outcomes. Firstly, actual money owned by the participants was not used in the experiments. Even though we could observe how the model differences affect consumers' decision making, results could be different due to, for instance, the endowment effect[4]. We believe that the trend of consumers' decision making does not drastically change, but it is difficult to confirm in a laboratory controlled experiment, because services in the space have to be attractive enough for participants to stay for a while and also enough valuable to voluntary pay their own money.

In the experiments we also used the Wizard-of-Oz technique to conduct payments/rebates, instead of sensor-based automatic activity recognition. The activities that interact with digital devices (e.g., play Wii, turn on a light) are easy to detect, if we can access and retrieve the internal state of corresponding service or device. Thus it is possible to set the transaction interval with fine granularity (e.g., an international call is charged on a minute basis). The payment scheme becomes understandable and clear to consumers, since transactions are conducted while they are actually using the service. Through direct interaction with devices, consumers can obtain a clear picture of the system's behavior.

On the other hand, activities that relate to interactions with non-digital objects are difficult to track (e.g., reading a comic, having a snack). Even though it might be possible to capture the user's motion with visual analysis (e.g., book reading), installation cost would increase because multiple cameras are needed for robust detection. Moreover, the definition of activity becomes vague, since it cannot be digitally judged. For example, from the consumers' viewpoint, the definition of "reading a comic" might be literally reading a comic, not simply holding a book or placing a book on the desk. We can monitor a bookshelf with RFID tags and charge a fee while a comic is being taken out, but the consumer might not expect to pay while they are not actually reading the book. The passive observation tends to hide the whole system behavior from consumers, but system errors and limitations should be disclosed in a human-readable way. Otherwise there could be the serious gap between the payment scheme and a consumer's

[4]A hypothesis according to which people place a higher value on objects that they own than on objects that they do not.

understandings (i.e., mental model about the system behavior). Thus the volume of inference logic should be minimized in a context analysis process.

The technology limitations directly relate to economic risks, because cheats could be possible if a complex activity detection process becomes automated. Therefore it is important to consider how manual monitoring can be supported by technologies, instead of trying to automate the whole billing process. The selection of system model is also important to decrease the economic risk. As we mentioned the impact of incorrect rebate is smaller than incorrect payment, so UbiRebate model is more appropriate when the recognition accuracy is not high. In other words, the context recognition module can be simplified in UbiRebate model and it results in cost saving. With UbiPayment model, vendors can expect additional gains, but consumers' activities have to be checked with enough accuracy. As a result, system installation cost and labor costs would increase, because man-powered checks are required as well as special installations for context recognition.

6.6.2 Lightweight Interaction

In the experiments, the automatic confirmation mode was applied and we did not allow the participants to deny activity-based billing. However, as we pointed out in [62], lightweight interaction design is an important issue if the billing system presented in this chapter is to be implemented in practice. Interactions in the system take place in two directions: feedbacks from the system to the consumer, and confirmation responses from the consumer to the system. In the activity-based billing system, transactions can occur very frequently, so notifications should be delivered with consideration to their level of importance. One of the participants commented that they felt happy getting rebates, but that the frequent notifications somewhat detracted from this feeling. One possible solution is to implement a threshold-based notification system: the mobile device alerts when the total amount of payments reaches a previously configured threshold. This decreases the frequency of interruptions and cognitive workload for handling the information. Changing the billing mechanism can also contribute to a decrease in the number of interruptions. The UbiRebate or UbiReward models do not need to notify the remaining amount at every update, so the consumers might prefer to manually check the amount by themselves.

Related to the feedback design, lightweight confirmation is also important. Completely automatic processing makes consumers anxious and leads to reluctance towards adopting the service. However, heavyweight confirmation such as the conventional PIN code is not suitable for handling frequent transactions. It is necessary to select an appropriate confirmation style according to the risk level of the transaction. In [62], we enabled the use of multiple confirmation styles to adapt to a wide range of purchase amounts. However, it is also important to consider the transaction flow in each billing model. For example, in the UbiRebate model, the initial cost is big and all transactions after the first payment give rebates. Thus it will be fine to apply secure PIN code based confirmation at first, and change to automatic processing for the subsequent rebates.

6.6.3 Incentive Design for Sustainable Behavior Impact

Consumers' decision making is strongly influenced by economic incentives, but in this study we noticed several challenges in this approach. For example, too much mental pressure makes services uncomfortable to use. Also, requiring too much attention prevents consumers from enjoying the service. One alternative approach is to apply social psychological incentives that try to effect behavioral changes by appealing to consumers' values and ethics. Social incentives are especially effective for tightly linked communities (e.g., families). For example, Nakajima *et al.* proposed an "ambient lifestyle feedback system" to effect changes in our daily behavior [79]. In their work, several kinds of pervasive applications were designed while considering emotional engagement. For example, if a child does not brush their teeth in the correct manner and frequency, a virtually presented aquarium becomes dirty and its virtual fish fall sick. The virtual aquarium is installed in a bathroom, so pleasant and unpleasant feelings evoked by the application are shared among family members. Thus social pressure provides motivation for each individual to keep the aquarium clean through proper tooth brushing behavior.

Social incentives are also expected to compensate for technical limitations. Shiraishi *et al.* developed an application called *EcoIsland*, which is a system persuading individuals to reduce CO_2 emissions [107]. In their work, economic incentives are implemented as EcoPoints, a virtual currency used in EcoIsland. Users can earn EcoPoints by reporting eco-friendly actions they have achieved, and the points can be used to purchase items to decorate their virtual island in a way that visualizes each family's contributions to CO_2 reduction. In the application window, users can see other families' islands so that they can compete and improve their behavior in a playful fashion. Moreover, this social networking feature decreases dishonest behavior that is otherwise possible due to technical limitations. Since eco-friendly actions are often too complex to detect automatically, it is hard to verify users' self-reported data. The social aspect of EcoIsland allows users to monitor each other, providing peer pressure against cheating.

A hybrid incentive approach could cover a wider range of users within an application. Each individual has their own preferences, and multiple types of incentives increase the chances of them developing interest in the application. Moreover, social psychological incentives work most effectively inside groups or communities, while economic incentives operate on the individual consumer level. In a local community, users know each other and they collaborate to maximize their total benefit, instead of taking actions that are detrimental to each other. To avoid the tragedy of the commons, it is important to try to bring together a small community and offer a virtual space (e.g., islands in EcoIsland) to smoothen its communication and collaboration.

Finally, it is recognized that once a user is motivated by economic value, previously existing social incentives become inoperative, even after the economic incentive is removed [10]. Thus we should be careful in designing the balance between social psychological incentives and economic incentives. Social psychological incentives are powerful and work sustainably in a smaller community, so economic incentives should not be introduced until necessary. In contrast, for bigger groups in which anonymized users are loosely coupled, economic incentives probably work better. This is because users start to make their decisions individually, and economic values become prioritized over other kinds of values. The activity-based micro-incentive approach presented in this chapter should thus be especially suitable for places and situations involving large numbers of general public.

6.7 Conclusion of This Chapter

In this chapter, we explored the idea of combining pervasive computing techniques with electronic payment systems to create activity-based micro-incentives. Users who consume additional resources by e.g., occupying an air-conditioned space instead of a normal space are levied additional micro-payments. In an alternative approach, consumers who choose to save resources are rewarded with micro-rebates off the price of a service. As a result, the cost of using a service corresponds more closely with the resources used, leading market mechanisms to allocate resources efficiently. A key challenge is designing incentive mechanisms that alter consumer behavior in the desired fashion. We introduced four incentive models, and present evaluation results suggesting that consumers make different decisions depending on which model is used.

Figure 6.10 shows the focused ADSS framework components in this chapter. The activity-based billing system combines a mobile payment system with context recognition techniques. Environmental context information is used to calculate the price of the user's activity. We also pointed out the irrationality of decision making in economic activities, and utilized the psychological bias to induce consumers to particular actions.

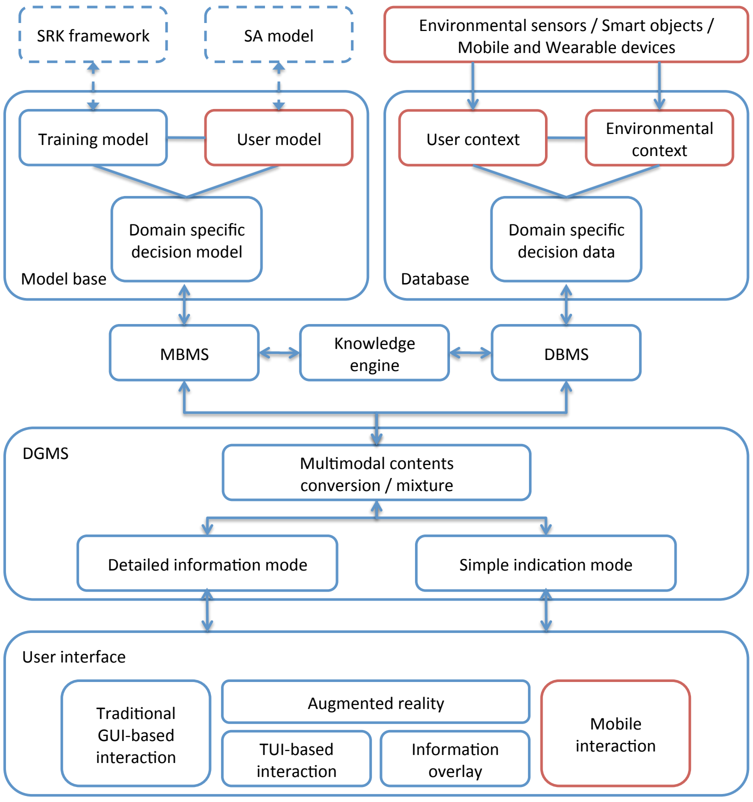


FIGURE 6.10: Focused components in this chapter

Chapter 7

Decision Support by Crowd Knowledge Aggregation

7.1 Background

Web services on the Internet provide a variety of useful information for tourists, such as train timetables, maps, and local restaurants recommendation. Thanks to the progress of mobile computing technologies, today we can access the Internet and retrieve necessary information on demand - it allows us to visit unfamiliar places without a careful preliminary survey. Sometimes mobile devices are not suitable for complex search, however, because the size of mobile display is too small to show entire search results. Moreover, frequent interaction with tiny keypads frustrates a user, since it enforces them to allocate much cognitive resources for a task [135]. Even though they are skillful to manipulate the device, it is often time-consuming to interact with large-scale contents (e.g., maps).

Using interactive public displays (e.g., public display with a touchable surface) is an alternative approach to the mobile search. Usually a public display is larger than a desktop computer's display, so a user can look and easily recognize large-scale information at a glance. Unlike conventional bulletin boards, contents on a public display can be dynamically changed according to a user's context [17]. Public displays can be also networked with the Internet and a personal mobile device so that personalized contents can be sent to/from a public display. As shown in [75, 84], potential new services on the interactive public displays have been proposed with exploring the feasibility of service deployment in the real world. Moreover, modern cities equip public displays at the important points of transportation service nowadays, and the cost of displays has been getting cheaper. We believe that it is feasible to apply public displays as a platform of web search, in a great range of application domains.

One important issue on the application design is the diversity of users. Unlike personal computers and mobile devices, a public display is supposed to be used not only by skillful users, but also the novices who do not have sufficient technical literacy. Making a search query to derive an exact answer from such a flat and boundless Internet is quite difficult for them, since the search syntax is too flexible and the vast amount of data could be hit as a search result. Search process requires both knowledge and skill. Moreover, this issue cannot be completely resolved by replacing a mobile device with the large workspace on a public display. One possible approach to support a end-user is pushing the contents that the user might be interested in, instead of increasing the flexibility of search interface.

Another issue is the locality of information. Users on the go often want to get the information that relates to their context (e.g., time and place). For example, a digital signage on the train platform indicates train arrival time and its destination. Such information is mostly important to the people who are waiting for the train, and temporary consumed before the train arrivals - they will forget the information after leaving the platform. Usually a public display is stably located at a certain place as well, and temporal and local contents should be prioritized than static contents. Stable information can be widely accessed from portable computers anytime. However, nomadic users tend to have the niche interests (e.g., near pubs opening now, good photo shooting spots for current lightning condition), which are not available on the Internet.

Our approach aims at solving these two design issues with knowledge mush up and reusable search query mechanism. We focused on public displays' role in social communication, and designed an ecosystem to realize the query reuse. Public displays are installed into popular public spaces and act as a medium of communications among general public. It means that the trend of search can be analyzed from previous queries, and then optimized information can be offered to newly joined users. For example, if an end-user can use the search query that is previously made by a skillful user, it will drastically shorten the time for information search. Moreover, the search history will be useful for a future use as well, since the people around the public display are expected to have similar interests. Our approach enables to collect skillful users' knowledge and interests from the queries, and cluster its results in order to use a public display as an access point to the regional information.

As a proof of concept, we prototyped a tourist guide service for public displays. In the next section, we introduce the ecosystem with sample scenarios and explain how a public display's knowledge is enriched. In Section 7.3, the essential parts of the navigation service's system design are explained. We describe the detail of system implementation in Section 7.4, and show experimental study results in Section 7.5. Based on the experiments, lastly we conclude this chapter with discussing the application design issues to improve our system in future work.

7.2 Public Displays in an Ecosystem

In this chapter, we chose tourist navigation as a service to demonstrate the ecosystem of public displays' contents. In this section, firstly we present a sample scenario, and after that we explain how the ecosystem works in detail.

7.2.1 Sample Scenario

7.2.1.1 Scene 1 (Finding a Restaurant)

User A was at station. He needed to attend a meeting at his office an hour later, and he wanted to have lunch shortly before that. He found a public display in the station, and started to look for good restaurants with the navigation service. Firstly he confirmed that the area nearby the station is shown as a map with pointing his current location in the center. On the right side of the map pane, genres of point-of-interests (POIs), such as restaurants, landmarks and shopping stores, were shown as a list. Then he touched the *"Restaurants"* button and shortly after listed items changed to show restaurant categories (e.g., cafe, pub, dining kitchen). He chose the *"Hamburger"* button, and also touched the *"Highly ranked but inexpensive"* button to screen out the results. He realized that the map indicated the restaurants far from the station, so the *"Within a 10 minutes' walk"* button was also selected and three restaurants remained in the map. He was still worried about their foods quality, so lastly touched the *"Reviews"* button to show other users' comments about the restaurants. As the navigation service gathered various types of information from the Internet and showed in a well understandable way, he could decide the restaurant for lunch without struggling with mobile web search.

These all buttons in this scene (e.g., *"Restaurants"*, *"Hamburger"*) are called *Filters* in our system, and they are used to screen out information on the map as the scenario mentioned. Filters have a hierarchy structure and available filters at a certain point change according to previously applied filters (e.g., the *"Hamburger"* button does not appear until the *"Restaurants"* filter is applied).

7.2.1.2 Scene 2 (Route Planning)

User B was on a trip and arrived at the station to see the sights of the town. She was completely a stranger there, but had not conducted enough preliminary survey since she was so busy before the trip. Shortly after, she found the public display and started to plan a tour route with the navigation service. Firstly she applied *"Sightseeing"* filter and continuously selected *"Museum"*, and also tried *"Sightseeing"* and *"Temple"* filter combinations to find POIs. Since she roughly knew the name of famous museum and historic temples, the filter based search helped to reach the exact location on the map. She wanted to check in the hotel after seeing the museum and temple, but no reservation was made at that moment. Then she started over map search with keeping the results of previous search (i.e., the museum and temple) as POI marks. The *"Accommodations"* filter was applied firstly, and *"Hotels"*, *"Hot spa"* filters followed after that. Then she chose one of the hotels appeared on the map and made a reservation on the phone.

Finally, three POIs were shown on the map: the museum, the temple and the hotel. Then she pushed *"Route"* button to find the routes to go around the town, which starts from the station. The routes allowed her to drop into both the museum and temple, and the hotel was destination. She transferred the route information into her mobile phone so that she can check the route while on the move.

7.2.1.3 Scene 3 (Reusing the Search History)

A few days after the user B's trip, user C happened to visit the town to attend a conference. Since by chance he could allocate time for sightseeing before the conference opening, he started to go around the town. Firstly he went to the station to gather information, and found the navigation service running on the public display. Then he started to use the service in order to grasp the town's overview.

When he applied *"Sightseeing"* filter, the navigation service showed the other users' search history as recommended routes. The user B's search results were also included in the history, and it met user C's interests, because he was roughly aware that the town was famous for ancient temples and its historic view. Since he had no idea about the exact name and place of famous temples, the history was helpful to initiate route planning. He checked the route's contents in detail and realized that there was a famous museum in the town. He was also interested in the museum, since lots of comments from previously visited users indicated the museum was must-to-see. He specified his hotel as the final destination, and made a route as user B did in the Scene 2. After he left the town, he added his comments by accessing the web link that remains in his mobile phone. Since the navigation service allows remote access from conventional computers such as laptops and mobile phones, users can add information from their experiments anytime.

7.2.2 Knowledge Localization Process

As introduced in the sample scenario, we propose the ecosystem that works based on CGM (Consumer Generated Media) mechanism. CGM is an approach to compensate the gap between real consumers' needs and the preliminary provided contents that are made by service designers. Moreover, it also enriches the entire contents level, since users are motivated to compete with other users by creating attractive contents. This cycle evolves a web service quickly, but on the contrary creating media itself requires a certain time and effort to amateur creators.

In our system, users' POI search composes of the combination of filters, and search histories are stored in the service's database. This means that every user plays both roles of contents creator and consumer. Filter-based search allows users to unconsciously embody their vague knowledge about POIs in a reusable manner, without coercing elaborate data input and formatting process. On the other hand, a user can obtain other users' knowledge from the history shown on the map. Unlike conventional web search, this approach works effectively, since public display users at a certain place are expected to share similar interests. Figure 7.1 illustrates the interaction process in the ecosystem.

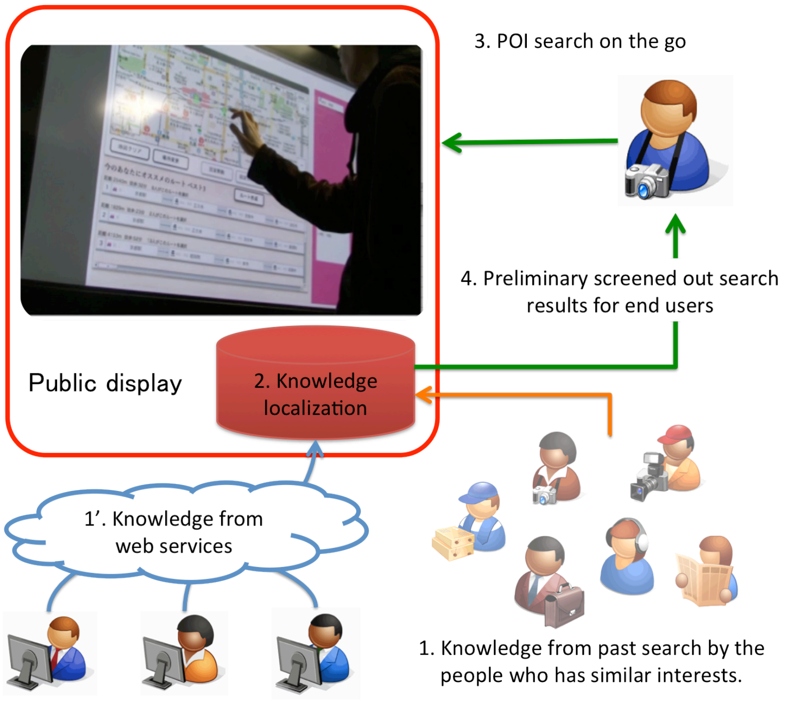

3. POI search on the go
4. Preliminary screened out search results for end users
Public display
2. Knowledge localization
1'. Knowledge from web services
1. Knowledge from past search by the people who has similar interests.

FIGURE 7.1: Public display as a knowledge collecting system in an ecosystem.

As the numbers in the figure indicate, the ecosystem roughly composes of four steps: 1. collect knowledge from search histories and web services, 2. rank locally interested knowledge from the trend of search on the public display, 3. interact with new users, 4. provide requested information with suggesting possible options from the localized knowledge.

The service accepts inputs not only from the remote users who have personal computers, but also from the users who directly interact with the service running on a public display. This mechanism allows the service to analyze the trend of interests at the area, with keeping the remote users as an information source. For example, conventional map services, such as Google Maps[1], do not take the locality of information into its system design. Thus every area is shown

[1]Google Maps: `http://code.google.com/intl/ja/apis/maps/`

in a same way, even though each has an unique trend in regard to peoples' interests. Some cities might be famous for well-designed buildings, and other some might be nice place to go hiking. Peoples' interests will differ as place changes, so contents on the map should adapt to that's trend.

However, such contents localization process requires time and human resource cost. As aforementioned in the introduction, the trend often creates niche markets; and locally interested information tends to be temporary useful for smaller communities. For example, if the area has a great view from a mountain, photo-shooting spots will be a major interest among tourists. Moreover, additional information, such as current lightning condition and sample composition described by skillful photographers will be helpful to make a plan for those who are about to have a stroll. Thus our system aims at automating the process by reflecting the real users' interests, which are obtained from the public display installed at the area. Our approach also compensates the shortcomings of the locality in the web services' knowledge, since such information is created and consumed in the local interaction loop [7].

As aforementioned, a public display gathers information and screens out it in order to shorten an end-user's interaction process. Since a variety of people use a public display, applications running on the display should cover both skillful users and the novices who lack of technical literacy. Thus our approach supports two paths for information search: flexible manipulation for the former users and selection from limited options for the latter users. As seen in conventional map services, our system allows skillful users to manipulate the service; and its result (e.g., filter combination and searched contents) is saved for future reuse. Moreover, the record is ranked according to its usefulness (i.e., how often used for search), and thus the search history that matches the local interests will remain in the end. Technically influent users will use the remained search results as the other path. The highly ranked search results are helpful to reach the localized knowledge, since sometimes it is hard for the users to appropriately combine filters.

7.3 System Design

In this section, we break down the ecosystem into core features and explain technical points.

7.3.1 Filter-based Search

As explained in Section 7.2, our system applies filter-based search. Filters have a metaphor for extraction, and they will remind users to find interesting contents from a map. Since a public display can show a large map, the amount of contents could be enormous. Users need to screen out them step-by-step with specifying category, genre, and etc. Filters also have a metaphor for combination, thus we applied this filter-based search for the first prototype.

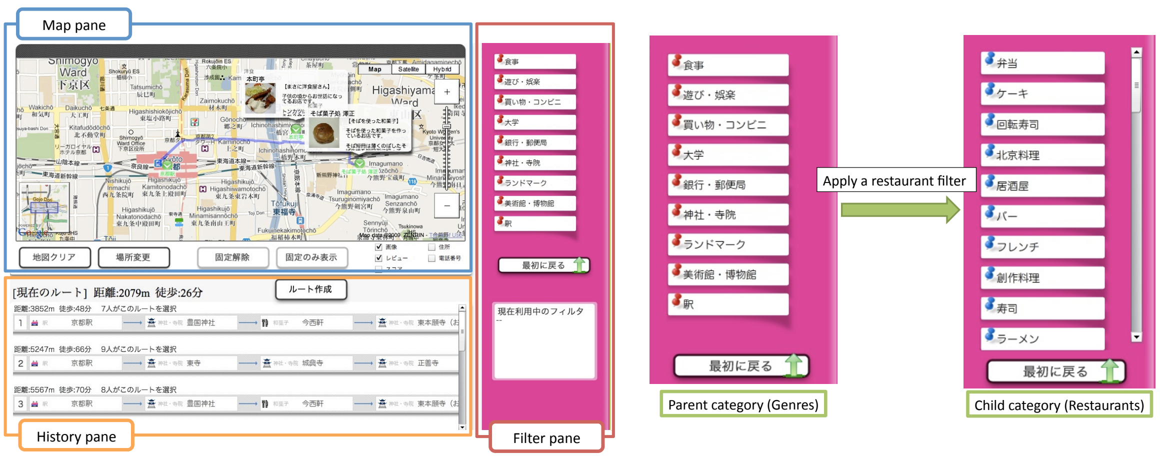


Figure 7.2: The tourist navigation service running on a public display. The user interface composes of three main panes: map pane, filter pane and history pane (left side). Item list in the filter pane changes according to a user's choice (right side).

In contrast, most conventional map applications adopt two different approaches: text search and tree-based search. The text search requests a user to input keywords or an address to find an interesting place. This approach is highly flexible, but sometimes it is difficult for an end-user to devise the keywords. It could be also considered that users feel annoying when the service returns unexpected results due to inappropriate keywords. Moreover, text input forces a user to type keys to organize a word, and as a result the user needs to keep their hands raised for contentious interaction. This is an important design issue for public display applications, if the service does not support additional devices (e.g., keyboard, mobile devices) for text input.

The tree-based search composes contents into clusters based on its attributes (e.g., category). Each cluster is linked with others, and as a result a tree-based structure is organized. A user can follow the link between the nodes (i.e., clusters) to find search results. As the filter-based search realizes a stepwise search, with this tree-based search a user can gradually get close to interesting information. However, the tree-based search requires lots of steps and it is difficult to predict results until reaching to the deepest nodes.

The filter-based search is less flexible than text input, but increasing the variety of filter can solve this problem. Moreover, there is the trade-off between the flexibility and the universality (i.e., allow end-users to use the service without sufficient knowledge). Compared with the tree-based search, the filter-based search shows its search results on a map immediately, and thus a user can confirm results at every operation.

7.3.2 Programming by Demonstrations

The filter-based search also enables to reuse the search process as a script. Once a skillful user creates a filter set, other users also can use the combination in a future search. As aforementioned, we assume that users who have similar interests will share the script; and the script will reflect the characteristics of the region where the public display is located. Moreover, since the filter set can be flexibly decomposed and reconstructed, a user can customize the script for own purpose.

This feature was designed with keeping the concept of *programming by demonstrations* in mind. It is difficult for end-users to make a complex script with a public display, even though each filter is easy to reuse. For example, either changing parameters of a filter or setting a conditional branch between combined filters requires certain programming skill and knowledge. Therefore, our approach aims at creating a script without forcing a user to write a program consciously. Our system only requires users to demonstrate search process by manipulating GUI widgets. The programming by demonstrations is realized by tracking the user's operation, and the demonstration is recorded and compiled into a reusable script [64].

7.3.3 Web Service Mash-up

Contents are another important aspect of our system. When we search information on the Internet, usually we use other web services in parallel, and compare and/or compensate the results to improve the quality of outcome. A search result in one service will be used as a keyword for other web services. Mash-up is more sophisticated style of such multi-service-based search: it combines multiple services into one service [123]. For example, a map service can show restaurant information with review results and photos, if it is mashed up with restaurant search service.

Since multiple people share a public display, the transaction time for each interaction (i.e., search) should be shorter than personal computers. Moreover, launching multiple services in one window could be confusing to end-users. On the other hand, contents should be enough attractive and consist of variety types of information. For example, in terms of format, text comments and images should be available to convey information in both verbal and nonverbal ways. Furthermore, from the aspect of variety, amateur creators' contents derived from CGM mechanism should be provided as well. Some web services only manage the contents prepared by companies and designers, but CGM is also important in order to explore niche markets and stimulate the ecosystem.

For the first prototype, we mashed up four different web services. In the next session, we explain the implementation of our system in detail.

7.4 Implementation

In this section, we explain the detail of system implementation from two main aspects: user interface and system architecture.

7.4.1 User Interface

As shown in Figure 7.2, our tourist navigation service's window consists of three main panes: map pane, filter pane, and history pane.

7.4.1.1 Map Pane

The map pane is the area for showing a map, and it also indicates POIs with corresponding information. At the default position, the public display is shown at the center of the map. Then a user can move shown area as they like by dragging the map. Moreover, the map is scalable, thus a user can zoom in/out it if necessary.

When a user applies the filter that specifies POIs' category (e.g., restaurant), corresponding points are marked as *markers*. Then, additional filters are applied to the markers on the map. Figure 7.3 shows an example of the transition of map pane. If a user selects a review filter, review comments and its rank are retrieved from a web service. Such information is shown in the markers' bubble so that the user can confirm the POI's detail at a glance. If the user applies a route filter, a route from the display's location to the point is shown as a line. This feature is also implemented with using another web service's functionality.

The map pane also has menu buttons on the lower side of the pane. These are "map clear", "location change", "unmark bookmarks", and "show bookmarks". With the map clear button, all markers on the map are cleared, so a user can refresh the pane and start from an initial state. The location change button allows a user to change the focused area. The unmark bookmarks and the show bookmarks buttons are used for bookmarking feature. In the search process, a user can partly keep markers even though the map clear button clears other search results. This feature is mainly used for route creation, since the user often needs to connect different types of POIs as mentioned in the sample scenario. In this case, the results of previously conducted search need to remain on the map, even though a new search started.

7.4.1.2 Filter Pane

The filter pane shows available filters as a list. The filters have a parent-child relationship as Figure 7.2 shows. In the example, filters in the parent category specify POIs' genres (e.g.,

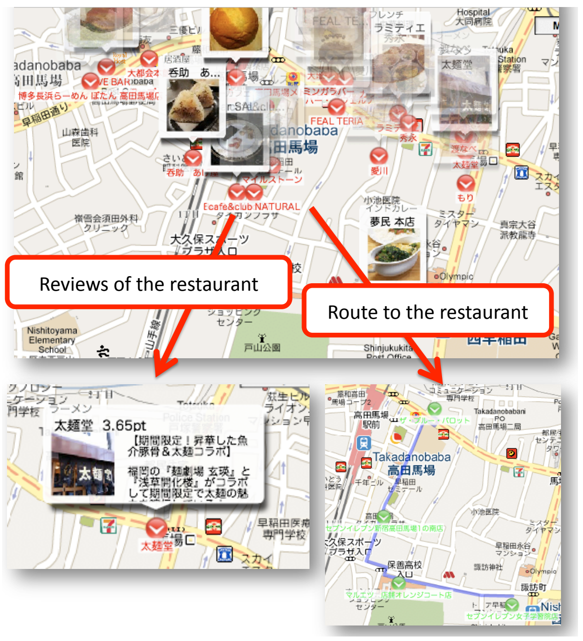


Figure 7.3: An example of the information that appears on the map pane.

Restaurants, *Universities*). If the *Restaurants* filter is applied, then the available filter list changes to specify restaurants' types (e.g., Japanese, Chinese). To apply the filters, the user has to drag-and-drop a filter from the list onto the map pane. Applied filters are shown in the window on the lower part of the filter pane. If already applied filters need to be canceled, each filter has a "close mark button" so that the user can remove it by clicking the button.

7.4.1.3 History Pane

The history pane shows the filter combinations that are previously created by other users. In Figure 7.2, the histories of created routes are shown for an experimental study. The filter combination is ranked higher as it is reused more, so that highly ranked combinations represent the locally interested points. Moreover, highly ranked combinations are shown as recommendations from the system. Thus it helps for end-users to choose an appropriate filter, since they cannot create a search query by themselves. The user can directly apply the filter combination by selecting options from this history pane.

7.4.2 System Architecture

The tourist navigation service is a client-server system. The client application runs on a networked public display and it handles a user interaction part. The user interface is implemented with Flash (Action Script 3.0). The client application communicates with a server application running on a remote computer. The server offers REST interface to receive the client's HTTP request and also to transfer stored contents as XML format. The server application is implemented with Ruby on Rails (2.3.3) and it stores application data into a relational database (SQLite 3) with O/R mapping. The stored data consists of data that is currently shown on the display (e.g., shop data, review data) and search history data. Figure 7.4 shows the detailed system architecture of the client and server applications.

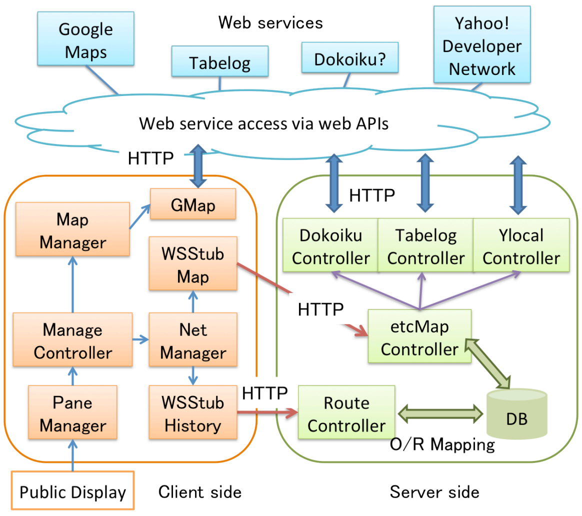


FIGURE 7.4: System architecture of the tourist navigation service.

When a user interacts with the service via a public display, the client application creates a HTTP request according to the selected filter. Each filter has an identity number corresponding to filter category (e.g., id_10: *Restaurants*, id_11: *Hamburger*). The HTTP request contains the ID number, and in some cases additional data is also included so that a query to web services can be created in a supported format. For instance, both the *Restaurants* and the *Hamburger* filters are handled as different HTTP requests with unique ID numbers. However, in the server side, they are aggregated into one query, since the web service that provides restaurant information requires a query in a specific format. After the web service responded to the query, the server application again interprets the data and sends it to the client application in XML format. At the same time, the server application stores the ID number that corresponds to the applied filter into the search history database. Following sections explain the system components that are illustrated in Figure 7.4.

7.4.2.1 Client Side

ManageCtrl class handles the events that are generated through the interaction with users. As aforementioned, a user interacts with GUI components on the panes, and PaneManager handles input events from GUI and update panes accordingly. Event type differs according to the user's input, and ManageCtrl class dispatches the event to other classes. When new information is acquired from other classes, ManageCtrl class sends a request to PaneManager, in order to update the panes.

MapManager handles all requests to the web service that provides map information. Google Maps is used in our system, thus we implemented GMap class as a proxy of Google Maps API. Map related functions, such as accessing map information, control the map (i.e., move and scale), and drawing routes, are invoked by MapManager through GMap class. NetManager handles communication between the server side applications. Two sub classes are implemented to offer HTTP access: WSStubMap class retrieves POIs' information for map drawing and WSStubHistory class retrieves search histories from the database.

7.4.2.2 Server Side

EtcMapController class returns POIs' information in response to the client application. The first prototype mainly focuses restaurants' guide, and three web services are used to fetch the relevant information: Tabelog[2], Dokoiku?[3], and Yahoo! Developer Network[4]. Tabelog provides restaurants' information, such as phone number and street address. Review comments, photos of the restaurant and ranking are available as well. Dokoiku? provides POIs in other categories, such as entertainment, shopping, convenience stores, universities, museums, and etc. By specifying a keyword from the basic information, a user can access category tags and reputation score. We also used Yahoo! Developer Network to show railway stations' information that is accessed through the *Stations* filter. The server side application has the proxy classes corresponding to each service, so three web service controllers were implemented in total. Fetched information from the web services is stored into the database as a cache, in order to decrease access overhead.

7.5 Evaluation

With the tourist navigation service, we performed experiments to evaluate the concept of *public display in an ecosystem.* This experiment also aims at identifying design issues in application development for a public display. In this section, we explain the method and results of the experimental study.

7.5.1 Method

Below list describes the experiment's procedure.

[2]Tabelog: `http://tabelog.com/`
[3]Dokoiku?: `http://www.doko.jp/`
[4]Yahoo! Developer Network: `http://developer.yahoo.co.jp/`

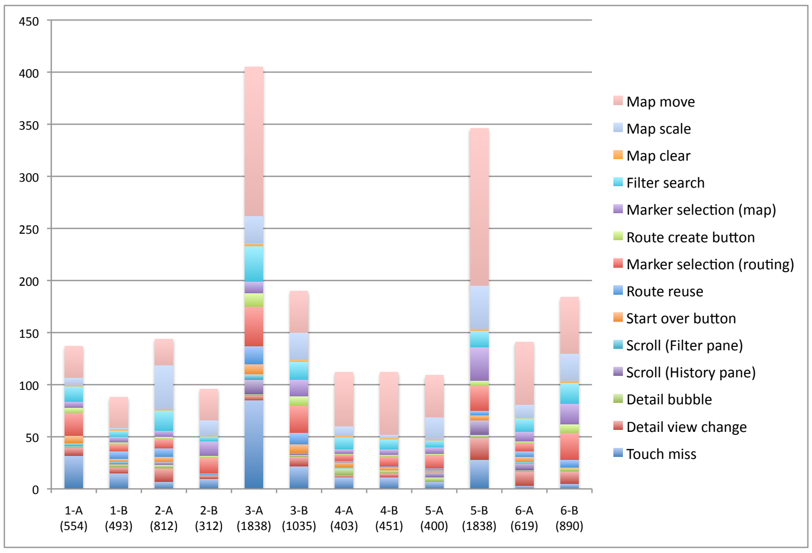


FIGURE 7.5: Classification of the touch interaction to the tourist navigation service. The X-axis shows subjects and tasks with indicating the time to complete the task (sec) in *Subject's ID-Task ID (Time)* format. The Y-axis shows the number of times the subjects touched the display in the experiments.

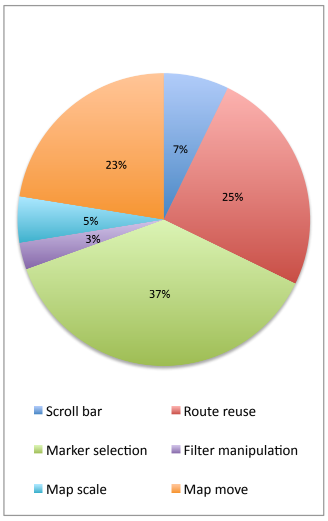


FIGURE 7.6: A breakdown of touch misses. The circle graph shows a ratio of touch miss patterns recorded in the experiments.

1. Brief explanation about the tourist navigation service was given to subjects. Then, the subjects were allowed to try all features of the service freely for 10 minutes, in order to get used to its user interface.

2. After the practice step, two tasks were given to the subjects. Both tasks requested them to create a sightseeing route under given scenarios. In Task A, the subject is assumed to be at Sapporo St. in Japan, and they are about to start sightseeing. They have 3 to 4 free hours to go around the city and it is around noon at the moment, but they have not had lunch yet. Under this condition, the subjects can freely create sightseeing routes, but they need to visit at least one special landmark (a famous clock tower in the city) on the way.

3. Secondly, Task B was given to the subjects. In Task B, the subjects are assumed to be at Kyoto St. in Japan, and they are about to start a historic temple tour. It is about 14:00 p.m., and they already had lunch at that point. They also have 3 to 4 free hours for sightseeing, and no special landmarks were specified in the task.

4. After the experiments, the subjects were asked to fill in questionnaires with a web browser.

6 university male students (23 - 25 years old) joined the study, and all subjects were not familiar with the places used in the scenarios. In order to minimize an order effect, the subjects were divided into two groups. One group performed from Task A to Task B order, and the other

group did vice versa. Experiments' process was monitored and recorded with a video camera in order to analyze how the subjects interact with the service on a public display. A Smart Board was used as a public display[5].

7.5.2 Results

Figure 7.5 shows the number of times each subject touched the display, and the ratio of its purpose. For example, even though the touch times and total time spent in the experiment differ according to individual subjects, map operations such as *"Map move"* (about 35% of total touch times in average) and *"Map scale"* (about 11%) keep the larger part of interactions. This is due to the balance of three panes. Since the history pane keeps a certain portion of area, the map pane is compressed and it requires the user to manipulate the map frequently. The resolution of used display is also problematic, since presentable information was limited. Fine-grained display could resolve this problem.

More importantly, the ratio of touch misses was also significant (about 11%). We analyzed the video and classified main results as shown in Figure 7.6. The ratio of *"Marker selection"* was relevantly larger than other manipulations. This is due to the size of interactive area, and it also becomes difficult to touch a marker precisely as the shown markers get denser and closer. This implies the filter-based approach does not always screen out information effectively, especially in the case that too much information appears in a small area. In addition to that, as the mash-up enriches contents, the volume of information that relates to one marker also increases. This is another issue of mass contents' appearance design for touch-based interactions.

Lastly, we evaluated the feasibility of the ecosystem from the experiment results. Figure 7.7 shows how many times markers in the search history were reused. In the experiments, randomly created routes were shown in the history pane and the subjects could make routes by referring the scripts. Each route has a couple of markers, so the subjects can select one of it and show corresponding markers on the map. Then with the bookmarking feature, they can create own route without removing the markers in the new search.

As the figure shows, about 50% of markers (35 out of 72) were reused in the experiments. Thus it could be said that the preliminary shown markers affected the subjects' choice in such a decision-free condition.

[5]Smart board: `http://smarttech.com/`

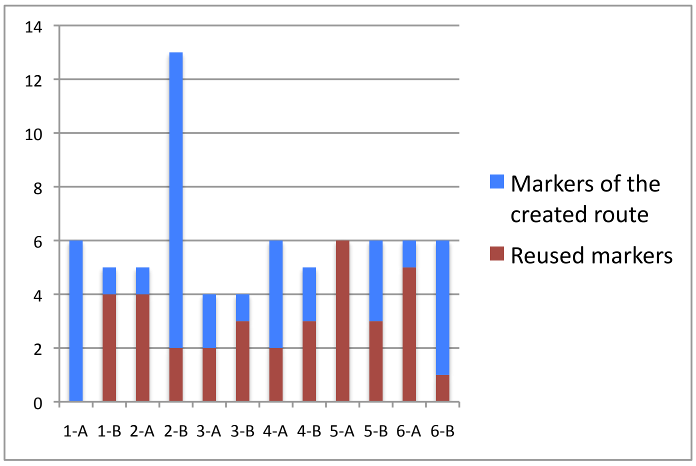


FIGURE 7.7: Number of times markers in the search history were reused in the experiments. The X-axis shows subjects and tasks in *Subject's ID-Task ID* format. The Y-axis shows the number of markers used in the experiments.

7.6 Discussion and Future Work

In this chapter, we presented the concept of *"public display in an ecosystem"*, which adopts the display to an access point to the local knowledge. By utilizing the filter-based search, users can leave their search queries and results for future reuse, and it helps end-users who do not have sufficient technical literacy to perform information search. As a proof of concept, we developed a tourist navigation service for an interactive public display. Then we evaluated the feasibility of the concept, from both user interface and ecosystem design point of views. In the following sections, we discuss design issues found from the experiments and identify future directions to improve the system.

7.6.1 Improvements in User Interface Design

As explained in Section 7.5, there are two important design issues to improve the usability: the balance of the three panes and the capable volume of information for touch-based interaction. Furthermore, we realized that conventional GUI design does not well fit to public displays, since the focal area is limited in the interaction process. Even though the public display can be larger and larger, a user needs to stand at the position where they can reach the display. Therefore, the display gets closer, and as a result only limited area in front of their head becomes recognizable. Actually in our preliminary experiments, the history pane could not draw a user's attention enough, because it is shown on the lower part and out of their sight. Even though the histories are useful, it does not work if the user cannot notice it. One possible approach is to directly overlay recommendations onto the map pane so that the user can recognize it without switching focus to other parts. However, the amount of information and presentation timing should be well considered, otherwise it will also drastically increase cognitive workload.

Another issue is the consistency of metaphors. In the experiments, some subjects commented that the filter metaphor is a bit confusing since some other functions require pushing a button instead of drag-and-drop. Moreover, a user needs to keep raising their arm while the drag-and-drop gesture. It makes the user tired and degrades the usability. In addition, the animation effect of marker presentations did not remind the filters. This linkage between the metaphor and presentation is important for a user to understand how the script is composed, and manually improve the scripts that is automatically generated by their demonstrations.

7.6.2 Decision Making Support from Human Factor Aspects

In this chapter, we mainly have discussed from a system perspective: applying programming by demonstrations to run the ecosystem and assisting end-users' information search. In addition to that, however, we should consider human factor aspects in the system design. For instance, public displays should be enough attractive for users to try the service, otherwise it disappears into background and ecosystem does not work [76]. Moreover, incentives should be given to users in order to keep the quality of contents and to motivate skillful users to create useful scripts. For example, reputation mechanism with economic incentives (e.g., virtual currency) is well used in human-power-based search. A questioner gives points to the user who gave a good quality answer, and it motivates users to actively contribute to the system. Moreover, other users can check the reputation so that they can decide whether the proposed answer is trustworthily or not.

It is also considerable that end-users still cannot find the scripts that exactly fit to their interests. Moreover, other users' history can be boring, even though the trends mined from history data is an effective approach to affect users' behavior. Serendipity is one important factor to engage end-users with the system [80]. Since our target users share niche interests related to location, there could be a lot of chances to direct serendipities around the public display. For example, Twitter is the social communication media that enables users to distribute their short tweets over the Internet[6]. There are also several mash-up services that show the message on a geographic map with reflecting the location where a user tweets. This geographically bound information could bring the serendipities, since users happen to be aware of new interests even if their prior interests were different. Particularly, it is important to design the system to increase the chance to find new interests, since it provides a good user experience and will keep the user to be involved in the ecosystem.

[6]Twitter: `http://twitter.com/`

7.7 Conclusion of This Chapter

It is expected that public displays compensate the shortcomings of mobile web search. Public displays have a big advantage in its large display size and touch-based interaction for multi-player's use, but sometimes end-users such as elderly people do not have enough skill to make sufficient queries that derive interested information from the Internet. In this paper, we point out an important aspect of public displays: an interaction medium in social communication. We also performed preliminary experiments and elaborately analyzed subjects' interaction process and figured out design issues for further improvements, such as search history presentation.

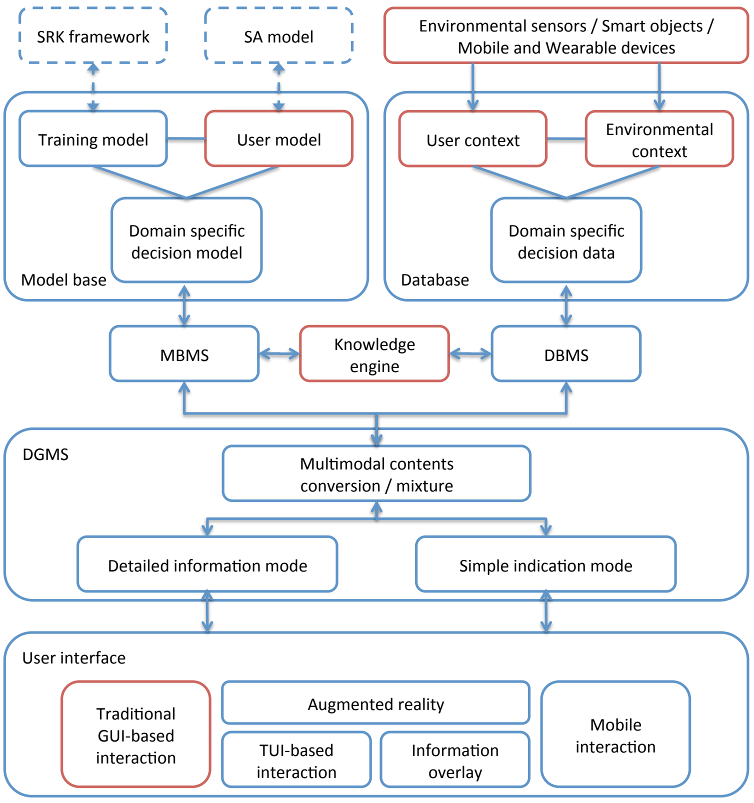


Figure 7.8: Focused components in this chapter

Figure 7.8 shows the focused ADSS framework components in this chapter. In the prototype, we assumed traditional GUI-based interaction even though public displays are extended with multi-touch technology nowadays. The map navigation service uses the environmental context information including its location to suggest possible POIs to the user. The user's context information is also considered to tailer the suggestion for them. More importantly, the system utilizes the history of search results to improve the quality of suggestion. Currently the knowledge engine is simply rank the decision support information based on the number of times the suggestion is used, but it could be replaced with smarter algorithms.

Chapter 8

A Context Acquisition Framework for Mobile Sensor Devices

8.1 Background

Within the past decade, mobile devices (e.g., PDA, mobile phone) have evolved with significant improvements in system performance, battery life, user interface design and wireless communication capability. A variety of feature-rich devices (e.g., smart phone) have been released into market and people can enjoy attractive mobile services (e.g., internet browsing, route navigation) anywhere they want. Mobile devices support our social activities today and the mobile computing technology has already been pervasive in our daily lives.

One important technology trend in the mobile computing evolution is sensors. Various sensors such as GPS receivers, accelerometers and capacitive touch sliders are incorporated to a mobile device, in order to enhance its user interface and functionality. Sensors have been originally used for monitoring the internal states of device (e.g., temperature, battery condition). In these recent years, however, the progress of miniaturization and low cost production of sensors have diversified the purpose; built-in sensors also cover user-oriented usages and applications. In other words, mobile devices have become able to perceive the external world, and they have been reaching the vision of context-awareness [101].

Since mobile devices are very personal and close to users, they are expected to play an important role in ubiquitous computing environments [18, 35]. Services can recognize context of a user and her surrounding environments with analyzing collected sensor data [132]. However, to date, sensors have been mainly used for detecting primitive context that relates to the mobile device itself (e.g., posture, motion). To implement richer context awareness, useful sensors have to be identified in empirical studies. Also it is important to discuss effective sensor deployment strategy upon the experiments.

In this chapter, we introduce a multi-sensory mobile device named Muffin. Muffin equips fifteen kinds of sensors to sense several types of contextual quantity such as acceleration, orientation, air temperature, a user's heart rate and so on. Muffin is a very unique device in the world, since it has been developed under a grand concept: implement as many different kinds of sensors as possible into a PDA sized box. Thus Muffin is a good prototype to perform empirical studies that aims to investigate sensors' characteristics and possibilities in context extraction process.

Empirical studies with Muffin showed practical issues in the mobile context extraction; the validity of sensor data and its analysis algorithms is not stable due to mobility. For example, biological sensor data are available under some limited conditions (e.g., "at a user's hand"). The mobility diversifies use cases of mobile device and we have to select an available sensor set according to the situation. On the other hand, advantages of the multi-sensory device were also identified; context can be detected in multiple ways with changing combination of sensors and analysis algorithms.

We have developed a software framework named Citron to utilize the advantages of multi-sensory mobile device. Citron supports parallel context analysis by employing the blackboard architecture [130]. Software APIs are offered to develop context analysis modules so that developers can easily monitor interesting context. Also we have developed a sample application on top of Citron and evaluated its feasibility. Based on Muffin and Citron development experiments, we discuss possibilities and limitations of context extraction in mobile devices.

In Section 8.2, we identify characteristics and requirements for realizing context-awareness on mobile devices. In Section 8.3, we introduce Muffin and point out some difficulties in mobile context extraction. In Section 8.4, Citron framework is introduced and a sample application is shown in Section 8.5. In Section 8.6, feasibility of our approach is evaluated.

8.2 Context Extraction with Mobile Devices

Mobile devices play a particularly important role in ubiquitous computing environments [125]. Like a partner, people carry them most of the day and use them very frequently in daily scenes. From the interaction point of view, they are medium between a user and context-aware services. Through the device, the user can access several services running in the background. Also the services can display information with or without interactive actuation (e.g., vibration, make sound and light) through the device. Considering from a context extraction aspect, mobile devices are expected to act as a monitor of a user. With logging sensor data and/or interaction history, context of the user can be inferred. It contributes to make the services context-awareness. Thus context extraction with mobile devices is one of key research theme in ubiquitous computing research.

At the early stage of context extraction from mobile devices, location and time information are the main resource of context [1]. To identify the location, GPS receiver is used in outdoor environments and other location systems are used to support indoor positioning [95]. Device status (e.g., network connection) and static personal information (e.g., name) are also processed as additional context resource. Since sensitivity is limited, however, additional sensors are required to extract more complex context.

Furthermore mobile devices' mobility causes two critical issues in the context extraction process. One is physically limited space where sensors can be incorporated. Even though detectable context heavily depends on equipped sensors [12], available sensors on a mobile device are limited due to its portable small package. The other is the dynamics of mobile environments. Since mobile context (e.g., a user's activity, location) changes continuously, it is difficult to track the transition with poor processing power. Thus some limited sensors have been attached for limited context extraction in traditional researches.

The most frequently used sensor on mobile context extraction is accelerometers. An accelerometer is cost efficient and sufficiently small to be incorporated into mobile devices. Furthermore acceleration data is effective to recognize a user's activities (e.g., walking, running, walking up/down stairs) and motion (e.g., shaking, rotating, knocking on) [61, 109]. Microphones are second-popular sensor after accelerometers, since ambient noise is useful to infer the place where a user exists (e.g., meeting room, restaurant, on the street) [35, 69]. Also microphones are used to recognize speaking or talking [109]. Lastly environmental sensors are commonly used on a mobile device; light and temperature sensors are used to extract environmental and a user's context [30].

Due to the issues, however, it has not been sufficiently discussed what types of sensors are useful for mobile context extraction. Also it has not been clarified what kind of context can be efficiently extracted from sensors on mobile devices. In the next section, we introduce Muffin that is a unique mobile device in terms of the unparalleled sensitivity. Based on preliminary experiments on Muffin, we point out practical issues in the context extraction process, and identify requirements to utilize the advantages of multi-sensory mobile devices.

8.2.1 Related Work

Hinckley *et al.* have added multiple sensors to a handheld PDA for enabling interactive user interface [40]. Developed device has an IR the proximity range sensor for measuring the proximity to a user, a touch sensor for detecting whether a user is holding it, and two axis accelerometers for detecting its tilt. Extracted context is the basic state of device and used only for user interface extension. A static set of context was analyzed, and context extraction over multiple sensor data was not discussed.

Siewiorek *et al.* have developed SenSay, which is a context-aware mobile phone for recognizing interruptible states [109]. Five kinds of sensors (i.e., voice microphone, ambient noise microphone, accelerometer, temperature sensor and visible light sensor) are mounted on a sensor box. SenSay extracts sensor data from the box and it decides the activity of a user (i.e., uninterruptible state, active state, idle state and normal state). It is similar to our work at the point they inferred a user's state from multiple extracted context. However flexible context extraction on different context was not discussed.

Gellersen *et al.* have proposed TEA (Technology Enabling Awareness) approach and they incorporated sensors into a mobile phone [35]. The TEA sensor board equips various kinds of sensors, such as a light sensor, microphone, CO-sensor, IR sensor, accelerometer and so on. They proposed preprocessing architecture called cue to reduce the analysis cost on context extraction. The output of each cue is classified into clusters by self-organizing map. Their approach is similar to ours, since abstract context is retrieved by stepwise approach and sensor data is processed in multiple ways. However non-exclusive context representation and parallel context analysis were not discussed.

Laerhoven *et al.* have shown a practical example of multiple interpretations of the same sensor data [61]. They created a wooden cube with a sensor module as a tangible input device and put an accelerometer into it. Then they recognized *gesture*, *orientation* and *top side* information from acceleration data with different analysis algorithms. Expanding analysis algorithms is more efficient for embedded devices rather than adding more sensors in terms of power consumption. Thus we take the approach to extract some of the basic states of Muffin.

8.3 Preliminary Experiments with Muffin

8.3.1 Muffin: a Multi-sensory Personal Device

Muffin is the prototype of mobile device for studying context awareness. It was developed in the collaboration work with Nokia Research Center. The significant characteristic of Muffin is its sensing capability: thirteen kinds of built-in sensors in the PDA sized box and two kinds of externally attached sensors are available (Figure 8.1).

The sensors can be roughly divided into four groups. First group is environmental sensors: an air temperature sensor, a relative humidity sensor and a barometer. Second group is physiological

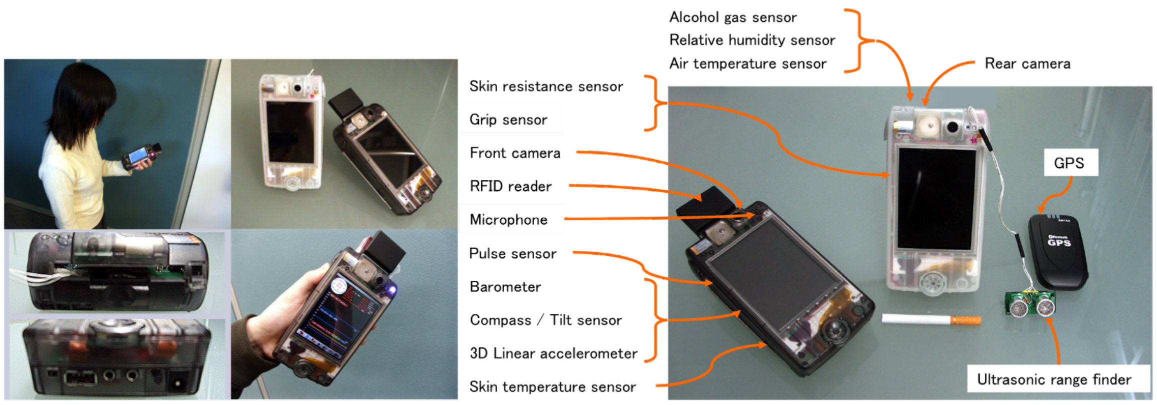


FIGURE 8.1: Muffin terminal and available sensors

sensors: an alcohol sensor, a pulse sensor, a skin temperature sensor and a skin resistance sensor. Third group is motion/location sensors: a compass/tilt sensor, a 3D linear accelerometer, a grip sensor, an ultrasonic range finder and a GPS receiver. The ultrasonic range finder and the GPS receiver are externally attached as optional sensors. Last group is remaining sensors: an RFID reader, front/rear cameras and a microphone. Linux operating system runs on Muffin, so each sensor can be accessed as a device file (e.g., /dev/AccelX). Also Muffin equips ordinary user interfaces (e.g., touch screen, micro joy stick, microphone, vibration motor) and connection interfaces (e.g., IrDA, Bluetooth, wireless LAN, USB port).

8.3.2 Context-Awareness on Muffin

Muffin has so many kinds of sensors that it can quantify several kinds of physical phenomena. Table 8.1 shows examples of context that could be extracted from Muffin's sensors. In the table, context is divided into three categories based on its subject: Muffin terminal, a user, and environments.

Based on this classification, we performed preliminary experiments in order to confirm feasible cases. As a result, we found important points and design issues in the context extraction with Muffin as follows.

Muffin: Muffin terminal's context could be extracted accurately, because used sensors output valid data. Also the sensors are so responsive that context can be analyzed in real time. Furthermore Muffin's state can be clearly classified into exclusive classes, so that simple algorithms such as threshold analysis could be applied. For example, Muffin takes either "at user's hand" state or "not at a user's hand" state at a certain moment, and it can be analyzed by simply comparing grip sensor data with threshold value.

User: There are three important issues in the process of user context extraction. The first issue is the availability of sensors and context analysis algorithm. User context frequently changes in

Table 8.1: Example of context that could be recognized by Muffin's sensors

Class	Description	Sensor
Muffin terminal's context		
Motion	Moving speed	3D accelerometer, ultrasonic range finder
	Trajectory	
Posture	Top side	3D accelerometer, digital compass
	Tilt	
	Orientation	
Placement	At a user's hand	Skin resistance sensor, grip sensor
User context		
Activity	Stationary state (e.g., standing, sitting)	3D accelerometer, ultrasonic range finder, digital compass
	Moving state (e.g., walking, running, going up/down stairs)	
Geographical information	Location	GPS receiver
	Orientation	Digital compass
Physical condition	Stress level	Skin resistance/temperature sensor, pulse sensor, grip sensor
	Alcoholic level	Alcohol sensor
Emotion	Exciting	Skin resistance/temperature sensor, 3D accelerometer, grip sensor, pulse sensor
	Surprising	
	Fearing	
Environmental context		
Air condition	Air temperature	Air temperature sensor
	Air humidity	Relative humidity sensor
	Air pressure	Barometer
Sound	Ambient noise	Microphone
	Talking voice	

mobile computing environments, so available sensors and algorithms also change accordingly. In order to extract user context, a user has to carry Muffin in some ways. However, available sensors and algorithms change according to the position or situation in which Muffin is used. For example useful analysis algorithm for detecting whether standing or sitting from acceleration data changes according to Muffin's context: whether Muffin is held or waist-mounted. Also if it uses an ultra range finder to correct analysis results by measuring the distance from floor, the validity of the sensors also changes.

The second issue is the delay caused from time-consuming context extraction process. In the previous example, detecting sitting state is easy at five minutes after the user actually sits. To detect the event instantaneously, we need to analyze the wave pattern of sensor data just in a few hundred milliseconds. This issue should be discussed also about other sensing domains, such as emotion awareness. To retrieve valid physiological sensor data, Muffin has to be grabbed by a user's hand. However, short-term sensor data is not sufficient to recognize physiological context. For example sensor data collected from a skin temperature sensor changes very slowly, and sensor data from a pulse sensor changes too rapidly. In other words it is necessary to log the average of extracted sensor data and compare them in the enough span of time. The last issue is the complexity and ambiguity of context. The definition of complex context, such as angriness or feeling of hunger, changes according to situation and application. For example the meaning of loud voice differs in different situations (e.g., meeting room and park). To appropriately recognize context, further information about a user (e.g., location, activity) is required in addition to sensor data (e.g., microphone).

Environment: Environmental context such as air temperature directly relates to raw sensor data, so it is not difficult to calculate them. However, Muffin gets hot internally as time goes on, and the heat affects environmental sensors. As a result, sometimes measurement does not work and collected sensor data become not useful. This problem arises from mechanical design, so we should refine the placement of sensors in Muffin and protect them from the heat affect.

8.3.3 Design Issues in Context Extraction Process

We have pointed out practical difficulties in mobile context extraction. Even about simple states such as *standing* or *sitting*, mobility affects the availability of sensors and analysis algorithms. In physical condition or emotion sensing cases, such difficulties become increasingly prominent, because the complexity of context increases. Therefore we should go back to consider simple context extraction cases, and then discuss about design issues for effective and robust context extraction with Muffin.

At first, complex context should be represented as a combination of other simple context. If the complex context can be decomposed into primitive one like Muffin's activity, it becomes easy to reconstruct them and represent higher abstract context. This approach is also important in terms of decreasing the ambiguity of context. Next, we should observe physical phenomena from multiple aspects of view. In order to increase the quality of information, additional sensor data or context are required. In addition, alternative context analysis algorithms should be prepared to ensure robust context extraction, because required sensors could suddenly become not useful as described in Section 8.3.2.

This approach also contributes to avoid the time-consuming process issue. In some cases, optional sensors or alternative analysis algorithms are more effective to recognize context than a default approach. For example an ultrasonic range finder is useful to detect whether a user sit down, in the case that she is holding Muffin and the sensor can measure the distance from floor. While one sensor data offers multiple meanings, one context can be recognized in multiple ways.

At last, it is required to specify relationship and dependency among decomposed context. Our experiments show dependency relations among context. For example, available analysis algorithms or sensors change according to the situation, because there are several styles to use

Muffin. In Figure 8.2 possible three cases of context dependency are identified. These relationships are classified based on a hierarchical context modeling. The resources are inputs (i.e., context or sensor data) to context analysis modules. The context analysis modules work for the extraction of predetermined context with analysis algorithms.

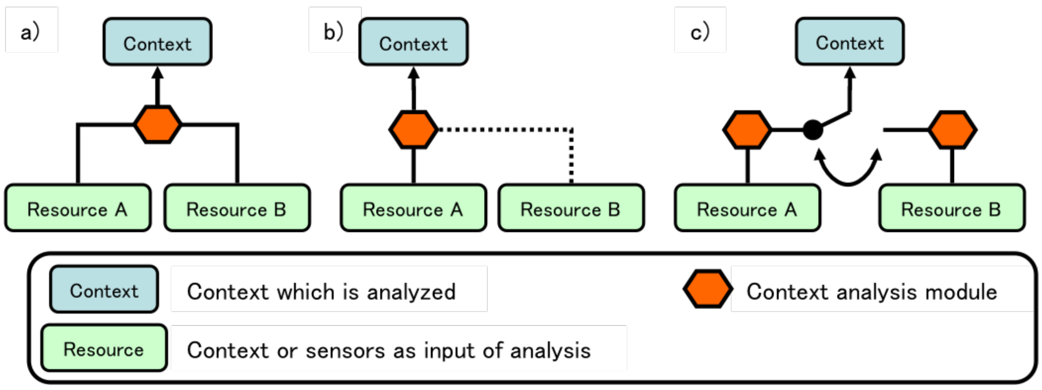


FIGURE 8.2: Relationship and dependencies among context

The case a) is a basic hierarchical context abstraction with the combination of Resource A and Resource B. The case b) and the case c) are its variations. In case b), context is mainly represented as a result of Resource A processing. Resource B is supplementary used to configure the analysis module's algorithm. In the case c), context is represented as a processing result of Resource A or Resource B. When the availability of one analysis module is not sufficient, other alternative module works instead of the original one.

As a result of the discussion, design issues in context extraction with Muffin have been pointed. They are not separated issues, but related to each other as shown in Figure 8.3. In the figure, there are four basic context that represent Muffin's context: *held or not*, *top side*, *moving or stop*, and *activity*. Every subject takes an exclusive states and the state of Muffin can be clearly classified into one of them.

However the relationship among context is non-exclusive, so more complex context can be represented with a context relationship as described above. For example, if the display of Muffin turns up (*top side*) and a user holds it (*held or not*), the user might look into Muffin's display (*under observation or not*). This is one of the context relationship represented as the case a).

In order to extract a wide variety of context, one sensor should be analyzed from multiple aspects of view. In Figure 8.3, three different analysis modules analyze acceleration data, and it is processed into different context and meanings. Furthermore context is also observed from multiple viewpoints. In the figure, *activity* is recognized from two different sensors: accelerometer and ultra range finder. One of them is selected according to the availability of sensors. This example shows the case c) that dynamically changes its analysis module and required resources.

Also we can find the case b): the analysis module of *walking or running or not* changes its threshold according to the posture of Muffin terminal.

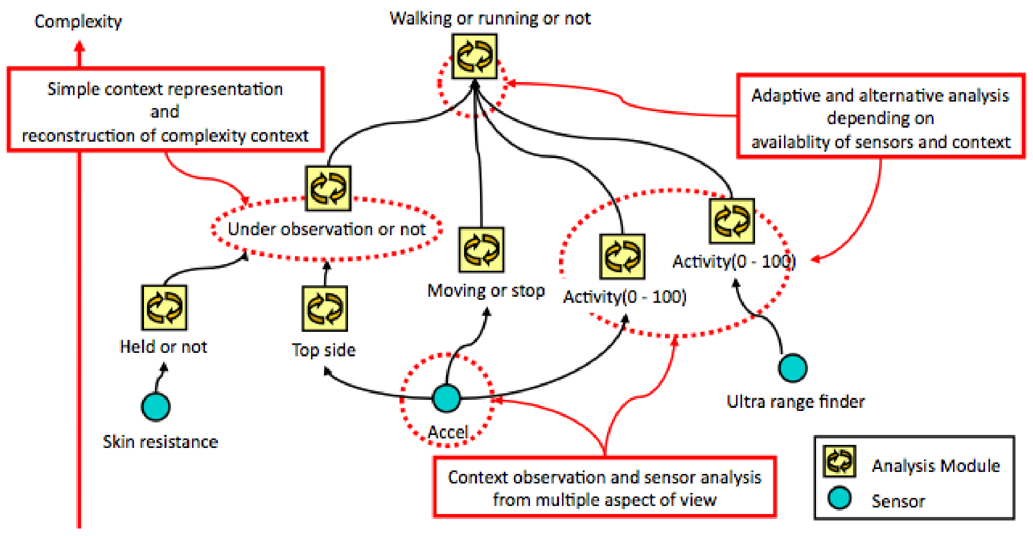

Complexity
Walking or running or not
Simple context representation and reconstruction of complexity context
Adaptive and alternative analysis depending on availablity of sensors and context
Under observation or not
Moving or stop
Activity(0 - 100)
Activity(0 - 100)
Held or not
Top side
Skin resistance
Accel
Ultra range finder
Context observation and sensor analysis from multiple aspect of view
Analysis Module
Sensor

FIGURE 8.3: Context-processing diagram with the design issues

We have refined this context-processing diagram into a software framework named Citron, which offers a software API for context analysis module development. In Section 8.4, we introduce Citron architecture and its features.

8.4 Citron: Context Information Acquisition Framework for Muffin

8.4.1 System Architecture

To implement the design issues discussed in Section 8.3.3, we implemented:

1. A context analysis module framework that allows developers to specify relationship among modules.
2. A shared space for module communication that stores extracted context.

In this implementation, we employed the blackboard architecture to coordinate context analysis modules. Furthermore we defined each context analysis module as a worker for exclusive context extraction.

The blackboard architecture is a data centric processing architecture that has originally been developed for speech understanding and artificial intelligence. There are one shared message board and multiple worker modules for collaborative data analysis. Each module reads information from the message board as a resource, and after processing the data it writes the result to the board. Thus module communication is established without the knowledge about other modules. Furthermore extracted data are analyzed and complimented in the communication process.

To implement the blackboard architecture, we adopt tuple space based programming model [16]. The tuple space is one of implementations for inter-process communication among independent processes. The tuple space refines a shared space with a very flexible data type called tuple, and it enables applications to search tuples with a template-based query. The important characteristics of the tuple space based system are 1) loose coupling of worker modules, 2) information sharing among worker modules and 3) flexible data representation.

Especially in ubiquitous computing environments, devices and services should work without specific knowledge (i.e., IP, port) about others, since the environment dynamically changes. Moreover context for several applications could be represented in a unified format because of tuple's flexible data type. Winograd discussed the characteristics and trade off between the blackboard architecture and other frameworks on the purpose of multi-process coordination [130]. We have also employed the blackboard architecture for designing Citron, because it is an adequate architecture to implement the design issues. Figure 8.4 shows the overview of Citron architecture.

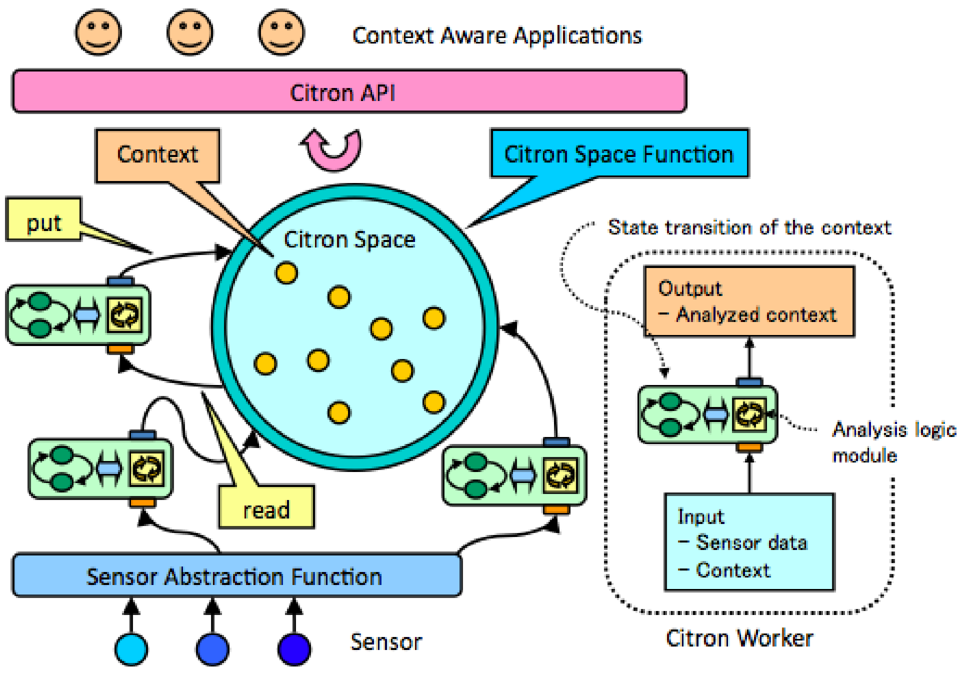


FIGURE 8.4: Citron architecture overview

Citron consists of two software components: Citron Worker and Citron Space. Citron Worker is a sensor data analysis module. Each worker collects sensor data from Muffin's sensors and it retrieves context from Citron Space. Also each worker takes responsibility on single context extraction. For example, a worker that recognizes *held or not* observes the value of a skin resistance sensor in order to detect *held* or *free* with threshold analysis.

Citron Space is a tuple space and it stores tuples that represents context analyzed by Citron Workers. Citron Space handles data management requests from both Citron Workers and the applications running on top of Citron. Context is represented as a set of meta-information, such as subject, state and time. Detailed explanation will be given in Section 8.4.2.

There are two internal functions (i.e., Sensor abstraction function and Citron Space function) in the system, and a software API is offered to develop external applications. The sensor abstraction function is just a simple wrapper function for accessing device files of sensors, thus Citron Workers extract sensor data with this function. Citron Space functions allow accessing Citron Space with three types of methods as shown in Table 8.2

Table 8.2: Briefly description of Citron Space functions

Method	Description
put	Insert context into Citron Space
read	Read context from Citron Space by template matching
get	Read and remove context from Citron Space by template matching

Lastly Citron API is published to enable applications to retrieve context from Citron Space. More details are described in Section 8.4.3.

8.4.2 Context Representation

In Citron architecture, context is represented as a set of meta-information:

$Context := \{ID, Subject, State, Time, Lag, Interval\}$

Also Table 8.3 explains each meta-information that makes up tuples.

Table 8.3: Explanation of Tuple fields and corresponding meta-information

Field	Description
ID	Citron Worker's unique identifier
Subject	Subject of the context (i.e. *orientation*, *walking*, *under_watch*)
State	Verb of the context (i.e. *NW*, *at_rest*, *free*)
Time	Time when the context is analyzed
Lag	Time lag in the analysis
Interval	Update interval of the context

For example context that is written as *a user is not walking (i.e., just standing)* is represented as follows:

$\{``\_walk", ``walking", ``resting", 1107005245, 0, 100\}$

In this case, the state could be either *walking* or *at_rest*. On the other hand the subject field is fixed, so Citron Worker extracts only one specified context. The lag and interval field are the meta-information of context, which are inherent in its analysis algorithm. Some analysis algorithms such as FFT analysis require a certain amount of data and time for buffering. Thus the lag field shows the time lag to applications so that it can be handled in appropriate manner. The interval field shows the freshness of context. Also the ID field specifies the identifier of Citron Worker that created the tuple. Tuples in Citron Space are updated every its polling interval by the Citron Worker that is associated with same ID. Thus applications can examine the freshness of context by looking up the interval field and the time field.

8.4.3 Implementation

Citron is written in C language, and it offers 1) a framework for Citron Worker development and 2) software API for application development.

Citron Worker Framework: This framework offers the abstraction of context analysis module so that developers can easily implement Citron Workers. The framework provides common functions of context analysis modules, such as connection management to Citron Space and sensor data retrieval from Muffin. Thus developers can concentrate on implementing analysis algorithms. When Citron Worker is initialized, the specification of the analysis, such as the type of sensor and the subject of context, has to be declared. First, a developer has to set value to the tuple fields. Then Citron Worker starts to run and invokes an analysis function at every specified interval. This analysis function executes the analysis algorithm with retrieving sensor data and context. The analysis result is returned as context and the Citron Worker shares it in Citron Space. In the current implementation, Citron Worker only puts the result of analysis when the state of context is changed, in order to reduce the load of Citron Space.

Citron API: Applications running on top of Citron access to Citron Space with invoking below functions. Context_t is the data structure implementing the context representation shown in Section 8.4.2.

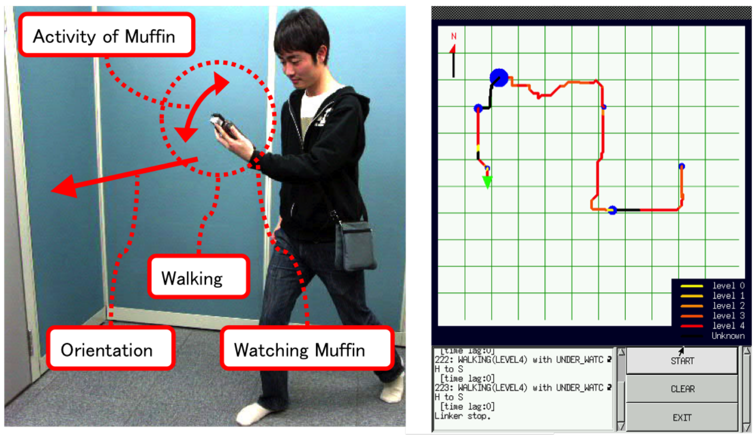
- char* libcitron_get_state(char* subject);
- context_t* libcitron_get_context(char* subject);
- int libcitron_add_event_handler(const char* subject, void (*handler)(char*));
- void libcitron_remove_event_handler(void);

Citron Space is implemented based on LinuxTuples, which is a tuple space implementation on Linux operating system[1]. Originally tuple space allows flexible wildcard-based query for tuple search, but in Citron it is wrapped as a simple function: only the subject field is used as a key for search. This design is sufficiently useful for many applications and it decreases the workload of Citron Space. Furthermore Citron API also provides callback function management interface. Developers can register/remove callback functions to handle state change events in Citron Space.

We have developed several Citron Workers and context-aware applications to evaluate the Citron framework. In the next section, we introduce one sample application and Citron Workers developed for the application.

8.5 Sample Application

We developed a sample context-aware application named RouteTracer, and it uses a user's context extracted by Muffin. This application displays the track of walking route in real time with a user's state. RouteTracer also shows walking speed and the duration that a user stayed at the same point. The walking speed is divided into five levels and it can be distinguished with different colors. The point where the user stopped is represented as a circle and its radius becomes larger as time advances. Figure 8.5 shows a sample map image generated by RouteTracer (right) and the user's states that are used to draw the map (left).



```
 [time lag:0]
222: WALKING(LEVEL4) with UNDER_WATC
H to S
 [time lag:0]
223: WALKING(LEVEL4) with UNDER_WATC
H to S
 [time lag:0]
Linker stop.
```

FIGURE 8.5: RouteTracer

In this application, three kinds of context are required: *walking state*, *walking direction* and *walking speed*. To draw the track of route, at least *walking direction* and *walking state* are necessary. The *walking speed* context is optional, however, it is necessary to speculate the distance for creating more accurate map. The *walking speed* context is inferred with the activity

[1] Linux Tuples: `http://www.c2.com/cgi/wiki?LinuxTuples`

level of Muffin. In preliminary experiments we found that the activity correlates with the walking speed of a user, while she is walking with looking Muffin's display. Thus *watching* context is required to determine walking speed. Six Citron Workers run to extract corresponding context in total: *orientation*, *walking activity*, *watching*, *holding* and *top side*. Figure 8.6 shows the relationship among context required by RouteTracer application.

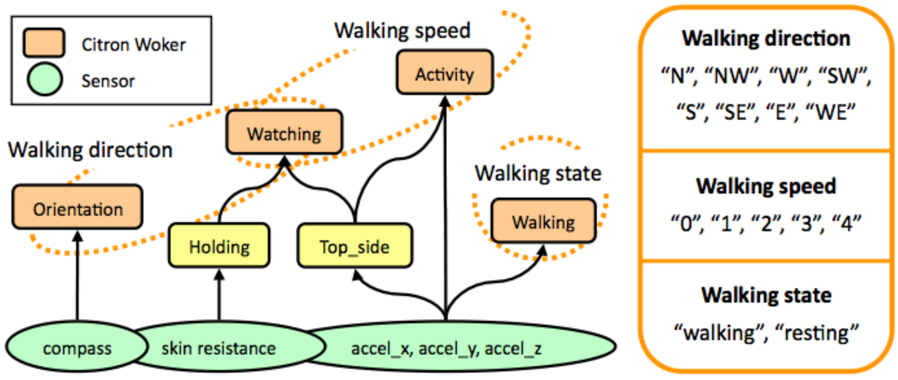


FIGURE 8.6: Required context and Citron Worker in RouteTracer

The *orientation* worker retrieves sensor data derived from a compass, and then it analyzes which orientation Muffin is heading. This orientation can be regarded as the direction of walking, when a user is watching Muffin. The *watching* worker recognizes whether the user is watching Muffin or not, based on a result of *holding* and *top side* worker analysis. If the user holds Muffin and the display of Muffin looks towards the user's face, the *watching* worker recognizes the state as *the user is watching Muffin*. The *holding* worker analyzes sensor data derived from a skin resistance sensor, and the *top side* worker analyzes gravity acceleration. The *activity* worker also retrieves acceleration data and it recognizes the activity level of Muffin with FFT analysis. This worker requires context generated by the *top side* worker, since the axis for motion detection changes according to the topside of Muffin. As described above, activity status is divided into five levels and treated as the walking speed. The *walking* worker decides whether the user is moving or resting with acceleration data: it is used as the *walking state* in RouteTracer.

It is also possible for RouteTracer to recognize the walking state only with *walking speed*. Performance in the analysis is not good, however, because the *activity* worker has to take about 6.4 seconds, in order to collect 128 samples into buffer and analyze them at every 50 msec. On the other hand, the *walking* worker analyzes the user's walking status with simple zero cross detection in real time. Thus it is expected that the parallel analysis using these two Citron Workers enable RouteTracer to be more responsive to recognize the walking state. In the next section, we evaluate Citron with measuring the overhead in invoking API. Also we compare the accuracy of map drawn by RouteTracer with changing Citron Workers.

8.6 Evaluation

8.6.1 Performance

We have measured the overhead in accessing Citron Space with invoking Citron Space functions described in Section 8.4.1. The dependency relation between the execution overhead and the number of running Citron Workers is also evaluated. Each worker invokes a Citron Space function ten times, and the execution time is measured as the mean time of all workers that run in parallel. Figure 8.7 shows the result of experiment.

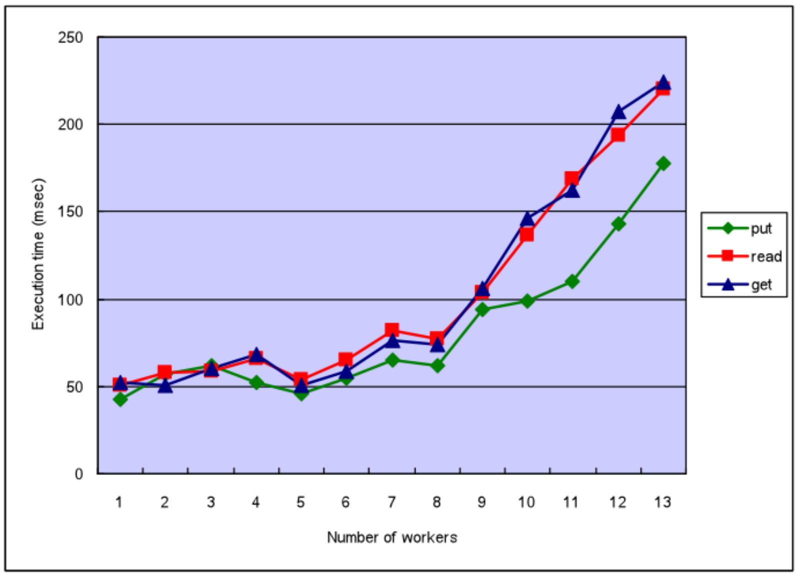


FIGURE 8.7: Relationship between execution time and the number of running workers

Figure 8.7 clarified that the overhead in Citron Space access increases as the number of Citron Worker increases. Especially the execution time remarkably increases when the number of Citron Worker goes over eight. Also due to the template-matching search, *read* and *get* functions recorded longer execution time than *put* function.

8.6.2 Experiences with the Sample Application

In this section, we evaluate the effect of Citron Worker coordination using RouteTracer. We compared drawn maps in three cases: using only *walking state* (case 1), using only *walking speed* (case 2), and using both for hybrid analysis (case 3). Figure 8.8 shows the walking route in the left side. A participant walks the route with holding Muffin in her hand. The participant was instructed not to pay attention to the display in order to remove the effects of intentional map creation. To clarify Citron Workers effect, the participant intentionally changes walking speed. The maximum walking speed in this examination is about 5 km/h and normally it is 3 km/h.

Also two stop points were instructed on the route, so the participant had to stop for 10 seconds at the points.

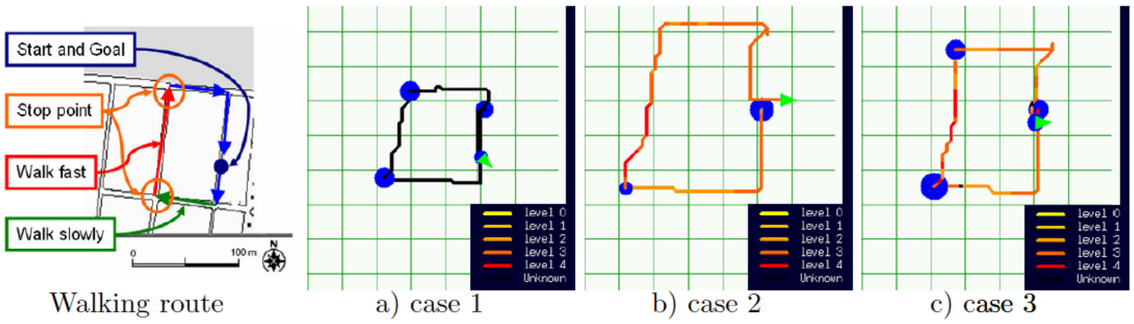


FIGURE 8.8: Walking route and drawn maps by RouteTracer in each experiments

Figure 8.8 also shows the result of each examination case. Figure a) is drawn with the time of walking and its orientation. Turning point and time that the participant has stopped can be clearly confirmed. This is because that the *walking* worker responds quickly when it recognizes the walking status change. However the walking speed is not reflected, so the length of each edge is inferred based on walking time. As contrasted with Figure a), Figure b) speculates the walking distance based on the walking speed. As a result, the drawn map becomes more similar to the actual map than Figure a). Figure b) also shows that the *activity* worker could not detect the stop points, because 10 seconds are not sufficient time for the FFT analysis to recognize the state change. Figure c) shows a more accurate map than Figure a) and Figure b). This case exploits the advantage of each analysis method and it also shows the effectiveness of hybrid context extraction with multiple types of analysis methods. The stop points were detected clearly, and the shape of map is the most accurate.

8.7 Conclusion of This Chapter

In this chapter, we addressed possibility and importance of mobile devices in ubiquitous computing environments. In order to clarify possibilities and limitations in context extraction with a mobile device, we introduced Muffin that is a prototype of multi-sensory personal device. Based on preliminary experiments, we pointed out design issues and proposed a software framework named Citron. A sample application was developed and evaluated feasibility of our approach.

We identified two further issues in the experiments. One is Citron's performance issue caused from the blackboard architecture. Parallel context analysis with multiple sensors heavily burdens mobile devices. Thus we should optimize the performance and reduce the load with redesigning Citron. Moreover it is assumed that the limitation of workable Citron Worker heavily depends on the implementation of Citron Space. As described in Section 8.4.3, Citron Space is implemented as a wrapper function of LinuxTuples. The performance can be improved if the implementation is optimized for context processing.

The other one is limitation of context extraction on Muffin. Most of physiological sensors on Muffin require some constraints to be used, such as the position of a finger and the style of holding, to measure valid data. Thus accuracy of such sensors changes so frequently according to the situation. It follows that other sensor devices (e.g., wearable sensors) are required as the alternative resource of context analysis. Citron can coordinate such remote devices easily, since the blackboard architecture is suited to dynamically add/remove knowledge resources and corresponding analysis algorithms.

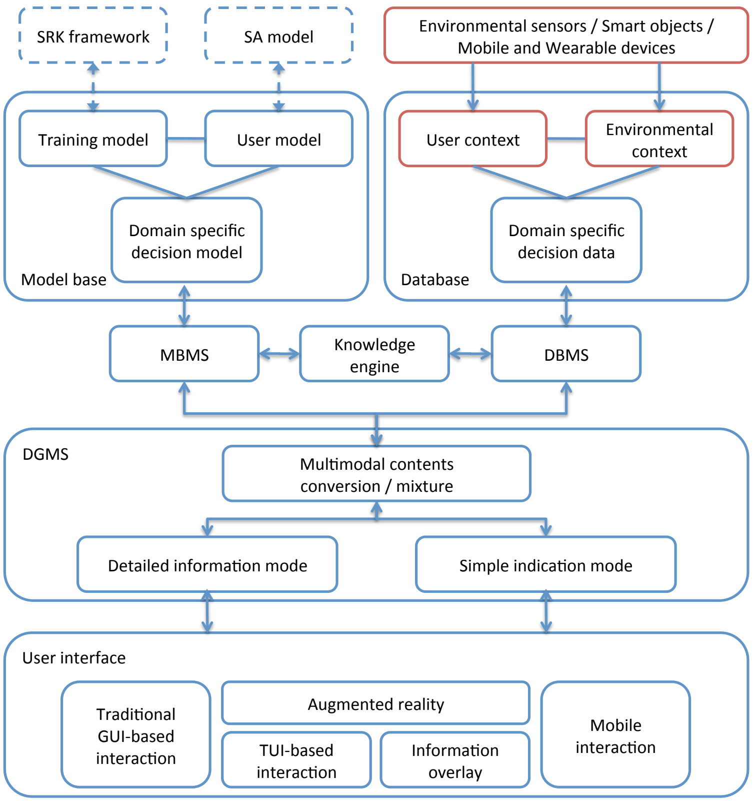


FIGURE 8.9: Focused components in this chapter

Figure 8.9 shows the focused ADSS framework components in this chapter. Citron itself is not a ADSS, but it is an enabling technology that is a part of the framework. A variety of context information can be integrated and processed into higher abstraction on Citron.

Chapter 9

Discussion

In the previous chapters, we introduced four ADSS case studies upon the framework. They brought us experiences and practical knowledge on decision support in the AmI environment. Based on the findings, we identified three design issues for further discussion and improvement of the framework: emotion, intention, and attention. These human factors affect each other in a decision making process, and ADSS developers should specially consider them into the system design. We illustrate the design issues in Figure 9.1 based on the Endsley's situation awareness model, since it well describes the relationships between a decision process and individual factors. As illustrated with different colors, each factors corresponds to cognitive factors at the bottom of the figure. For example, the experience should involve both aspects of skill development and emotional user experience. As it affects cognitive functions colored by blue, the emotional experience promotes skill acquisition e.g., an occurrence linked with a positive emotional response would remain longer in a memory. Moreover, people would allocate more cognitive resources as the expected outcome satisfies them with positive feelings. Below sections explain more detail of each design issues.

9.1 Emotion

As mentioned above, emotion can be regarded as both an origin and a goal of decision making. People often explore the better way that makes them happier, or even in the decision making process continuously evokes emotional responses as we experimented in the EmoPoker system (Chapter 5). Once the positive emotion is anticipated as an outcome, the intention towards particular decision (or decision problem domain) is reinforced by the expectation. In contrast, negative emotion or experience would weaken the intention and prevent from allocating cognitive resources on the decision problem.

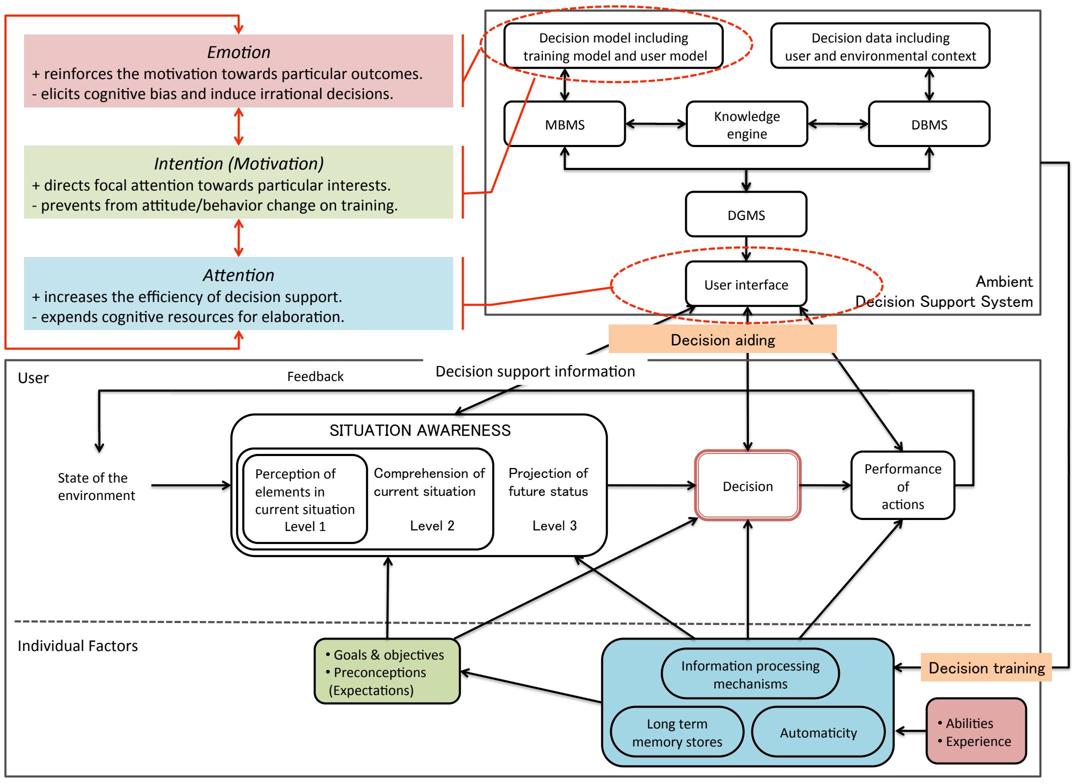

Emotion
+ reinforces the motivation towards particular outcomes.
- elicits cognitive bias and induce irrational decisions.
Intention (Motivation)
+ directs focal attention towards particular interests.
- prevents from attitude/behavior change on training.
Attention
+ increases the efficiency of decision support.
- expends cognitive resources for elaboration.
Decision model including training model and user model
Decision data including user and environmental context
MBMS
Knowledge engine
DBMS
DGMS
User interface
Ambient Decision Support System
Decision aiding
User
Feedback
Decision support information
SITUATION AWARENESS
Perception of elements in current situation Level 1
Comprehension of current situation Level 2
Projection of future status Level 3
State of the environment
Decision
Performance of actions
Individual Factors
• Goals & objectives
• Preconceptions (Expectations)
Information processing mechanisms
Long term memory stores
Automaticity
Decision training
• Abilities
• Experience

Figure 9.1: Three design issues illustrated in the ADSS framework and situation awareness model

Moreover, as a negative aspect of emotion, it is well recognized that emotional status influences cognitive capability and the rationality of a decision [87, 102]. For example, high levels of autonomic arousal on anxiety impair working memory capacity. Emotional states are not necessarily intensive, but mild ones, namely, mood also affect decision making process. Schwarz proposed the *feelings-as-information theory* to conceptualize the role of subjective experiences including moods, emotions, metacognitive experiences, and badly sensations at a decision moment [103–105]. The theory postulates that people use their feelings as a source of information, as well as declarative information come up to mind. For example, moods convey valence information that induces positive judgements when people are in happy rather than sad, and vice versa. Different feelings provide different types of information, and even valid information is conveyed to a decision maker, an unrelated influence can lead them astray.

The ADSS system developers should consider emotional states including both mood and intense emotion. In AmI environments, a decision maker's emotional state could be affected by a variety of stimuli including social communication, and possibly become too unstable to keep rational decision making. For example, on the way to an important meeting, train delay would evoke strong mental stress. The pressure of circumstances temporally shrink the volume of available cognitive resources, and the decision maker cannot afford to perform interaction with an ADSS as usual. The perception stage of situation awareness is also affected by cognitive bias, and as a result only limited information is conveyed for decision making.

As we emphasized in the EmoPoker system development, a primary cause of irrational decisions is the uncertainty. The incompleteness does not always occur due to the characteristic of decision problems, but also from the lack of situation awareness. A decision maker often misses the information in the cognitively peripheral area where the attention is not directed. The cognitive blindness could lead to develop an incomplete mental model, and prevent them from handling events coming from outside of the sense. For example, in Chapter 6 we introduced that the automatic payment increased concern and reluctance to the service. In addition to the cognitive bias in payment transaction, which is explained in the prospect theory, the automaticity brought by an ADSS amplifies the anxiety. If the system automaticity does not well align with the user's expectation, it will break the consistency of future vision constructed in the projection phase of situation awareness. The automatic payment could happen frequently with short term interval, thus periodical feedback with recognizable feedforward information will decrease cognitive effort to keep updated.

9.2 Intention

Intention involves the motivation and the attitude that represents attitude towards particular goals and objectives. Thus naturally the intention induces the user to select decision problems and make decisions in order to achieve the goal. Intention to the goal determines how a decision maker actively behaves to solve a problem. As strongly desires to accomplish the task, more aggressive strategies would be used with higher priority than other tasks. In other words, it directs attention to the problem and leads to allocate more cognitive resources for decision making. For example, Ajzen proposed the *theory of planned behavior* as an extension of the *theory of reasoned action* [3–5]. The theory identifies three key components that determine behavior and intention: attitude, subjective norm, and perceived behavioral control. Attitude towards behavior is determined by the total set of accessible behavioral beliefs and represents the degree to which performance of the behavior is positively or negatively valued. Subjective norm is the perceived social pressure to engage or not to engage in behavior, and is determined by the total set of accessible normative beliefs. Lastly, perceived behavioral control refers to perception of ability to perform specific behavior, and the total set of accessible control beliefs determines it. Consequently intention integrates these components and leads particular behavior.

Moreover, it is important to keep intention (motivation) through a decision training process. As we reviewed in the Rasmussen's SRK model, every first decision is taken by a novice decision maker. Experiences gradually mature the decision process and eventually the decision making behavior goes beyond the consciousness, namely, skill-based modes. Higher elaboration promotes to construct schema for further decision making. Thus ADSSs should provide the user with incentives along with positive emotional experiences in order to keep the intention during the training process.

As for incentive design, persuasion could be regarded as one aspect of decision training, and conveying persuasive messages is also important as well as cognitive skill development. For example, behavior change towards better quality of life is one important application domain for ADSSs [79]. As we learned in Chapter 6 and Chapter 7, incentives possibly shape behavior into more desirable patterns. Petty and Cacioppo proposed two routes to process persuasive messages in the *elaboration likelihood model*: central route and peripheral route [14, 85]. Messages processed through the central route with higher elaboration perform attitude change that results in sticky behavior alteration. Thus, incentives need to be provided with careful consideration on presentation, which draws sufficient attention and aligns with their intention.

The social context is also important for designing decision training. In Chapter 6, we argued that a variety of incentives should be provided in order to realize sustainable behavior change. As we studied, economic incentives are a powerful factor that affects a decision making process, but it tends to elicit instant behavior change without any comprehension. As the elaboration likelihood model postulates, the choice of route affects people's attitude. The central route that provides high elaboration leads to stronger attitude formulation than the peripheral route. Economic incentives possibly lead people to use peripheral route, since the economic value is intuitive and attractive enough to change their behavior without consciousness. Economic activities are essential for everyday life, and thus we tend to make skill-based response especially for small amount transactions. In Chapter 7, we implemented a social incentive as suggestions from other users' search history. People often compare other users' behavior with own to make decisions in a social life. As many web services provide system platforms to share the opinions and reputations, our decision is strongly motivated by virtual experiences brought by the predecessors since the information enhances the projection phase in situation awareness.

9.3 Attention

The efficiency of decision support is largely affected by the volume of attention, as it determines how much cognitive resources are allocated for a decision making process. As mentioned above, people tend to direct attention to their interests, thus attention control is an important aspect to design decision support in the AmI environment. For example, in Chapter 7, we found that the focal area of public display is smaller than expected, and the subjects needed to intentionally switch their attention to the pane where decision support information appears. On the other hand, in Chapter 4, the simplified interaction mode was not attractive enough and the users started to direct their attention to the scenery, rather than notifications. In contrast to traditional DSSs, a decision maker is exposed into dynamically changing environments, and needs to handle multiple cognitive cues that appear on a variety of devices and contexts. Thus a variety of contexts including both external world and internal mental status need to be considered into an ADSS design. Limited volume of cognitive resource is allocated for each task, and cognitive effort is required to handle interruptions by other tasks and perform task switching.

As addressed in Chapter 4, as well as incentive design, cognitive load is an important factor in the attention control. The components of human information processing are roughly categorized into two functions: processor and memory [15, 128]. The term cognitive load refers to the load on working memory during instruction [114]. There are three distinct types of cognitive load: intrinsic cognitive load, extraneous cognitive load, and germane cognitive load [116]. The intrinsic cognitive load points the inherent difficulty of instruction such as digital calculation, and the extraneous one is provided by instructional designers: it depends on how information is presented especially for learners, and the designer can reduce the load on purpose. The last germane cognitive load is devoted to the processing, construction and automation of schemas. For example, elaboration on a problem requires higher cognitive load, but if it results in successful scheme construction, it would reduce cognitive load in future problem solving processes. Sweller developed the *cognitive load theory* based on the cognitive processing model, in order to provide guidelines for the presentation of information and encourage learner activities that optimize intellectual performance [115, 116].

As the cognitive load theory emphasizes the importance to design instructions and higher level of elaboration flow, the presentation format of decision information should be designed according to the applicable cognitive load in each decision training stages. As for decision aiding, modality effect and the interactivity of devices also affect the cognitive load in a decision making process [47]. For example, Wicken's multiple resource theory implicates that cognitive cues in the same modality conflict as they are processed through the same cognitive resources [127, 129]. As we proposed in Chapter 4, multimodal indication is an approach to convey the decision information with balancing the cognitive load. After making a decision, people take the performance of actions into account as feedback for future decisions. Thus, as well as decision problems, the outcome that draws particular attention would elicit greater emotional response. Moreover, preconceptions relating to the motivation can be also affected, as available cognitive resources determine the granularity of projection capability.

In this chapter, we reviewed the findings in the case studies and identified the dependencies among the design issues. In the next chapter, we conclude the paper with implicating future work to revise the ADSS framework.

Chapter 10

Conclusion and Future Work

In this paper, we proposed a system framework for ADSS development. We introduced four case studies and identified three design issues from the development experiences: emotion, intention, and attention. We also summarized dependencies among these human factors with illustrating into the framework. As we firstly stated the importance of understanding decision making mechanism, an ADSS needs to be designed from both technical and cognitive perspectives. Otherwise, the service could not be truly useful and would remain in the foreground even though the user needs to handle other decision problems in such a multitasking environment. We believe that the framework supports developers to design the ADSS, which balances the cognitive effort required to make decisions by coordinating the human factors. In the future work, we iteratively revise and improve the framework by increasing practical case studies.

For example, except for the map service introduced in Chapter 7, we have mainly focused on single decision maker's use cases. However, as we emphasized the importance of social context, in the next step ADSSs should offer cooperative decision support functions for multiple decision makers. As the case study development progresses, we faced with the difficulties to develop standardized models and design the (semi-)optimal strategy for decision support. One of the reasons is simply lack of experts, and another is a challenge to model the "truth" of a decision problem. For example, in the Augmented Go case, there is a standard metric to evaluate the playing skill as grade. However, beyond a certain level, the importance of other factors such as heuristics and tricky tactics increases. Thus, to deal with the lack of experts, an ADSS should expose the model improvement process into cloud knowledge over the web. Ideally experts should be able to model their own expertise without specific programming knowledge to disclose their heuristics. Then a decision maker can choose and switch the decision model based on the efficacy and reputation. Even though it is challenging, communication among experts should be promoted on a discussion platform so that peer review process would improve the implemented strategy and models in an ADSS. As we discussed in Chapter 7, an ecosystem should be designed with incentive schema to encourage the developers to join and promote the knowledge aggregation process.

ADSSs also need promote to develop the sense of situation awareness upon predictable system behavior. Even though sensor technologies and activity recognition techniques have been matured, perfect context awareness is still challenging to realize especially in complicated environments. Moreover, as we pointed in Chapter 6, there could be a gap between the user's subjective understanding of own activity and detected context information. An ADSS should behave predictably to a decision maker, but it is often difficult to build semantic consensus due to cognitive bias and misunderstanding inherent in situation awareness development. Thus system behavior and rules need to be transparent to a decision maker as much as possible. Periodical feedforward information would enhance the projection capability and keep the mental model updated. For example, in the micro-pricing scenario, the user should preliminary understand basic rules of transactions, such as actions to be charged and how the system recognizes them. Furthermore, a decision maker should be able to identify incorrect context information, and report or repair the matter if necessary.

Appendix A

Illustrations of Referenced Theories

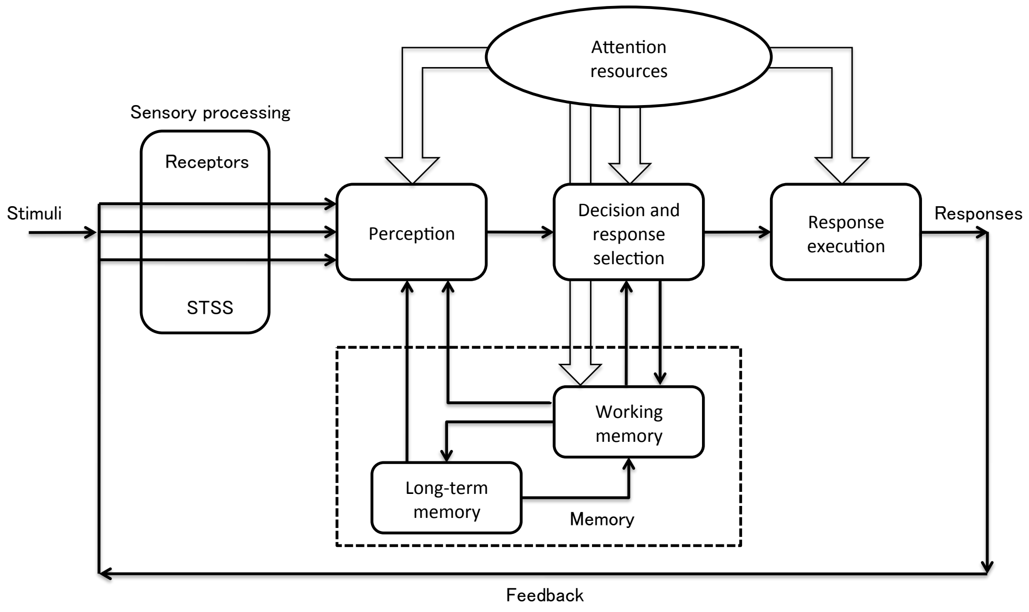


FIGURE A.1: Wicken's Information Processing Model, adapted from [128].

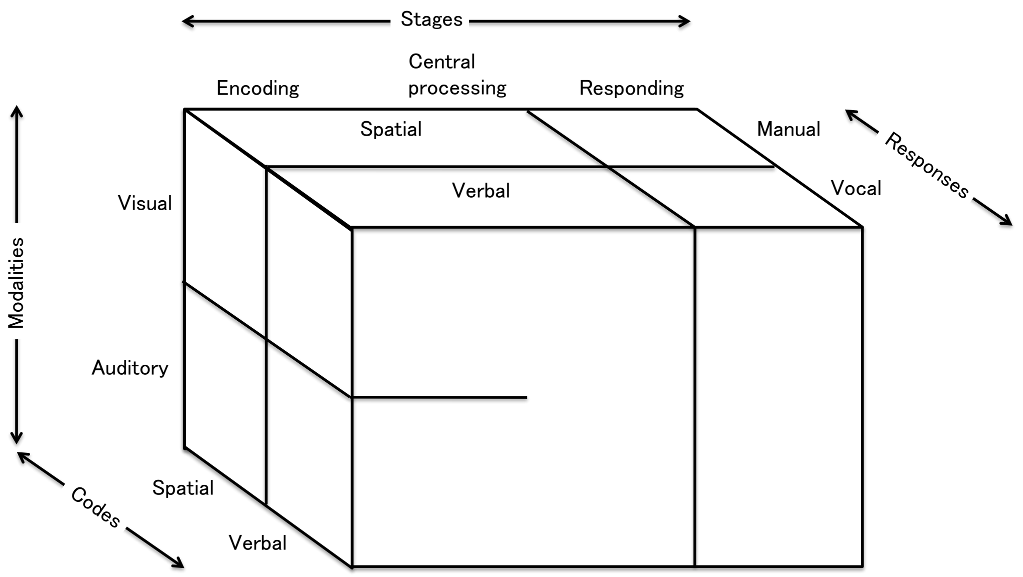


Figure A.2: Wicken's Multiple Resource Theory model, adapted from [127].

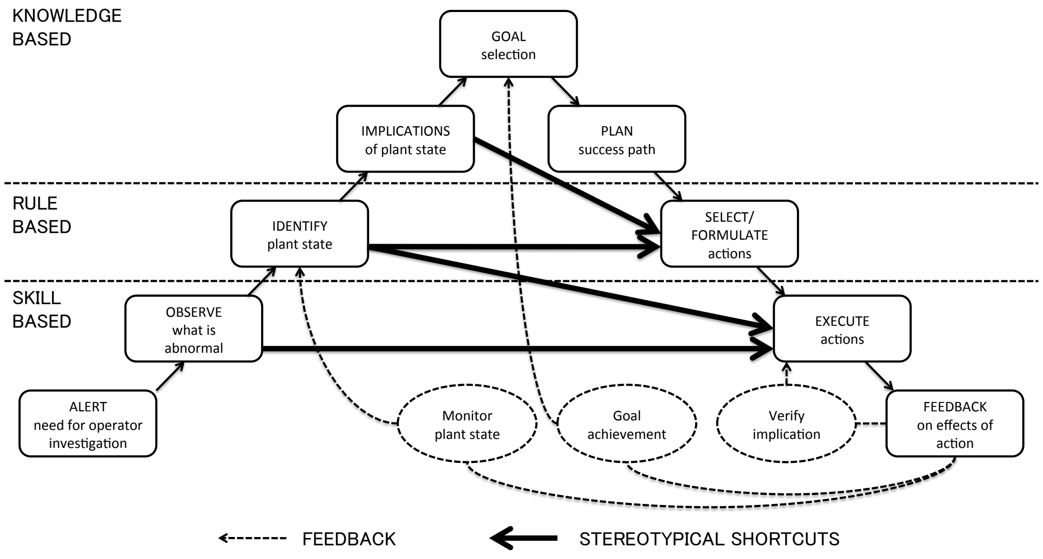


Figure A.3: Rasmussen's Step Ladder Model, adapted from [97]

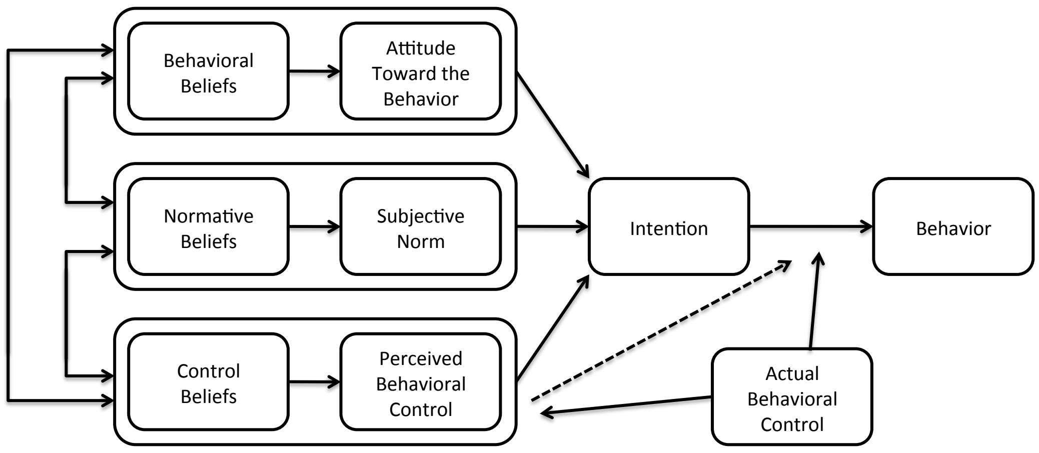


FIGURE A.4: Ajzen's Theory of Planned Behavior model, adapted from [3].

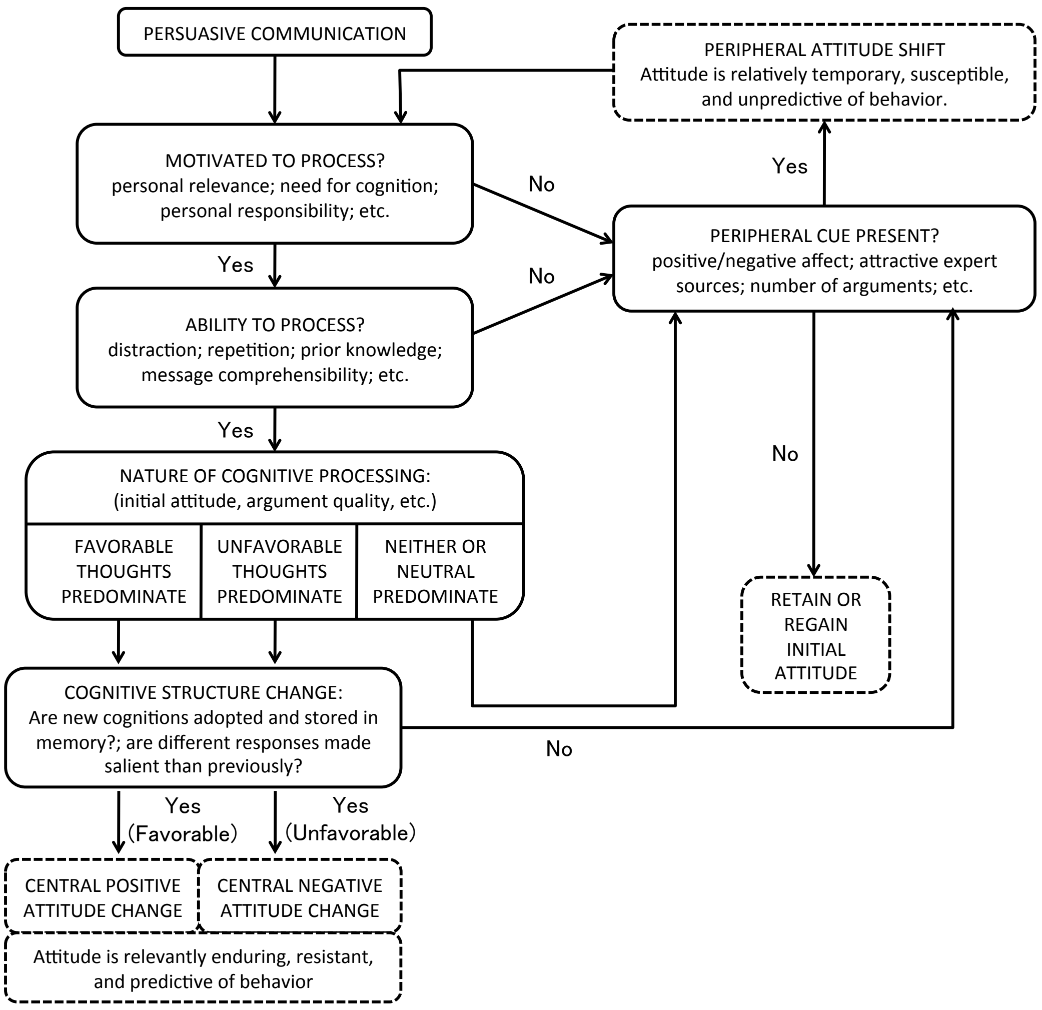

PERSUASIVE COMMUNICATION
PERIPHERAL ATTITUDE SHIFT
Attitude is relatively temporary, susceptible, and unpredictive of behavior.
MOTIVATED TO PROCESS?
personal relevance; need for cognition; personal responsibility; etc.
No
Yes
PERIPHERAL CUE PRESENT?
positive/negative affect; attractive expert sources; number of arguments; etc.
Yes
No
ABILITY TO PROCESS?
distraction; repetition; prior knowledge; message comprehensibility; etc.
Yes
No
NATURE OF COGNITIVE PROCESSING:
(initial attitude, argument quality, etc.)
FAVORABLE THOUGHTS PREDOMINATE
UNFAVORABLE THOUGHTS PREDOMINATE
NEITHER OR NEUTRAL PREDOMINATE
RETAIN OR REGAIN INITIAL ATTITUDE
COGNITIVE STRUCTURE CHANGE:
Are new cognitions adopted and stored in memory?; are different responses made salient than previously?
No
Yes (Favorable)
Yes (Unfavorable)
CENTRAL POSITIVE ATTITUDE CHANGE
CENTRAL NEGATIVE ATTITUDE CHANGE
Attitude is relevantly enduring, resistant, and predictive of behavior

FIGURE A.5: Petty and Cacioppo's Elaboration Likelihood Model, adapted from [86].

Bibliography

[1] G. Abowd, C. Atkeson, J. Hong, S. Long, R. Kooper, and M. Pinkerton. Cyberguide: a mobile context-aware tour guide. *Wireless Networks*, 3(5), Oct 1997.

[2] L. Adelman. Evaluating decision support and expert systems. *Evaluating decision support and expert systems*, Jan 1992.

[3] I. Ajzen. From intentions to actions: a theory of planned behavior. *Springer Series in Social Psychology*, pages 11–39, 1985.

[4] I. Ajzen. The theory of planned bahavior. *Organizational Behavior and Human Decision Processes*, 50:179–211, May 1991.

[5] I. Ajzen and M. Fishbein. Understanding attitudes and predicting social behavior. page 278, Jan 1980.

[6] S. Alter. Decision support systems: Current practice and continuing challenges. page 316, 1980.

[7] A. Angus, D. Papadogkonas, G. Papamarkos, G. Roussos, G. Lane, K. Martin, N. West, S. Thelwall, Z. Sujon, and R. Silverstone. Urban social tapestries. *IEEE Pervasive Computing*, 7(4), Oct 2008.

[8] R. N. Anthony. Planning and control systems: A framework for analysis. page 180, 1965.

[9] G. Ariav and M. Ginzberg. Dss design: a systemic view of decision support. *Communications of the ACM*, 28(10), Oct 1985.

[10] D. Ariely. Predictably irrational: The hidden forces that shape our decisions. page 384, Jan 2008.

[11] D. Ariely and G. Loewenstein. The heat of the moment: the effect of sexual arousal on sexual decision making. *Journal of Behavioral Decision Making*, 19(2):87–98, Apr 2006.

[12] M. Beigl, A. Krohn, T. Zimmer, and C. Decker. Typical sensors needed in ubiquitous and pervasive computing. *INSS'04: Proceedings of the 1st International Workshop on Networked Sensing Systems*, pages 153–158, 2004.

[13] S. Burigat, L. Chittaro, and S. Gabrielli. Navigation techniques for small-screen devices: An evaluation on maps and web pages. *International Journal of Human-Computer Studies*, 66(2), Feb 2008.

[14] J. T. Cacioppo, R. E. Petty, C. F. Kao, and R. Rodriguez. Central and peripheral routes to persuasion: An individual difference perspective. *Journal of Personality and Social Psychology*, 51(5):1032–1043, Sep 1986.

[15] S. K. Card, T. P. Moran, and A. Newell. The psychology of human-computer interaction. page 488, Jan 1983.

[16] N. Carriero and D. Gelernter. Linda in context. *Communications of the ACM*, 32(4), Apr 1989.

[17] S. Carter, E. Churchill, L. Denoue, J. Helfman, and L. Nelson. Digital graffiti: public annotation of multimedia content. *CHI '04: CHI '04 extended abstracts on Human factors in computing systems*, Apr 2004.

[18] G. Chen and D. Kotz. A survey of context-aware mobile computing research. *Dartmouth Computer Science Technical Report TR2000-381*, pages 1–16, Aug 2000.

[19] N. Cooper, A. Keatley, M. Dahlquist, S. Mann, H. Slay, J. Zucco, R. Smith, and B. Thomas. Augmented reality chinese checkers. *ACE '04: Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology*, Sep 2004.

[20] J. F. Courtney. Decision making and knowledge management in inquiring organizations: toward a new decision-making paradigm for dss. *Decision Support Systems*, 31(1):17–38, May 2001.

[21] J. Dankelman, M. Wentink, C. Grimbergen, H. Stassen, and J. Reekers. Does virtual reality training make sense in interventional radiology? training skill-, rule- and knowledge-based behavior. *CardioVascular and Interventional Radiology*, 27(5):417–421, Sep 2004.

[22] M. J. Druzdzel and R. R. Flynn. Decision support systems. *Encyclopedia of Library and Information Science: Second Edition*, pages 794–802, 2003.

[23] K. Ducatel, M. Bogdanowicz, F. Scapolo, J. Leijten, and J. C. Burgelman. Scenarios for ambient intelligence in 2010. 2001.

[24] D. Embrey. Understanding human behaviour and error. *zonecours.hec.ca*, Jan 2005.

[25] M. R. Endsley. Design and evaluation for situation awareness enhancement. *Proceedings of the Human Factors Society 32nd Annual Meeting*, pages 97–101, 1988.

[26] M. R. Endsley. Measurement of situation awareness in dynamic systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 37(1):65–84, 1995.

[27] M. R. Endsley. Toward a theory of situation awareness in dynamic systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 37(1):32–64, 1995.

[28] M. R. Endsley and D. J. Garland. Situation awareness: analysis and measurement. page 383, Jan 2000.

[29] J. Erp, H. Veen, C. Jansen, and T. Dobbins. Waypoint navigation with a vibrotactile waist belt. *Transactions on Applied Perception*, 2(2), Apr 2005.

[30] P. Fahy and S. Clarke. Cass - middleware for mobile context-aware applications. *The MobiSys 2004 Context-Awareness Workshop*, pages 1–6, Jan 2004.

[31] R. L. Ferguson and C. H. Jones. A computer aided decision system. *MANAGEMENT SCIENCE*, 15(10):B–550–B–561, 1969.

[32] D. Fitton, V. Sundramoorthy, G. Kortuem, J. Brown, C. Efstratiou, J. Finney, and N. Davies. Exploring the design of pay-per-use objects in the construction domain. *EuroSSC '08: Proceedings of the 3rd European Conference on Smart Sensing and Context*, Oct 2008.

[33] C. Floerkemeier and F. Mattern. Smart playing cards – enhancing the gaming experience with rfid. *Proceedings of the Third International Workshop on Pervasive Gaming Applications - PerGames 2006 at PERVASIVE 2006*, pages 27–36, Mar 2006.

[34] H.-W. Gellersen. Modality abstraction: Capturing logical interaction design as abstraction from "user interfaces for all". *1st ERCIM Workshop on "User Interfaces for All"*, Jan 1995.

[35] H.-W. Gellersen, A. Schmidt, and M. Beigl. Multi-sensor context-awareness in mobile devices and smart artifacts. *Mobile Networks and Applications*, 7(5), Oct 2002.

[36] C. Gonzalez and J. Wimisberg. Situation awareness in dynamic decision making: Effects of practice and working memory. *Journal of Cognitive Engineering and Decision Making*, 1(1):56–74, 2007.

[37] G. A. Gorry and M. S. S. Morton. A framework for management information systems. *Sloan Management Review*, 13(1):50–70, 1971.

[38] P. Haettenschwiler. Neues anwender freundliches konzept der entscheidungsunterstützung. *Gutes Entscheiden in Wirtschaft, Politik und Gesellschaft*, pages 189–208, 1999.

[39] G. Hardin. The tragedy of the commons. *Science*, 162:1243–1248, 1968.

[40] K. Hinckley, J. Pierce, M. Sinclair, and E. Horvitz. Sensing techniques for mobile interaction. *UIST'00: Proceedings of the 13th annual ACM symposium on User interface software and technology*, pages 1–10, Nov 2000.

[41] S. Hinske, M. Lampe, C. Magerkurth, and C. Rocker. Classifying pervasive games: On pervasive computing and mixed reality. *Concepts and technologies for Pervasive Games - A Reader for Pervasive Gaming Research*, 1, Jun 2007.

[42] S. Hinske and M. Langheinrich. W41k: digitally augmenting traditional game environments. *TEI '09: Proceedings of the 3rd International Conference on Tangible and Embedded Interaction*, Feb 2009.

[43] C. W. Holsapple. Dss architecture and types. *Handbook on Decision Support Systems 1*, pages 163–189, 2008.

[44] T. Hurtig and K. Jokinen. Modality fusion in a route navigation system. *Proceedings of the Workshop on Effective Multimodal Dialogue Interfaces EMMDI-2006*, pages 1–6, Jan 2006.

[45] H. Ishii, C. Wisneski, J. Orbanes, B. Chun, and J. Paradiso. Pingpongplus: Design of an athletic-tangible interface for computer-supported cooperative play. *CHI'99: Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*, pages 394–401, Jan 1999.

[46] A. Jalote-Parmar and P. Badke-Schaub. Workflow integration matrix: a framework to support the development of surgical information systems. *Design Studies*, 29(4):338–368, 2008.

[47] A. Jalote-Parmar, P. Badke-Schaub, W. Ali, and E. Samset. Cognitive processes as integrative component for developing expert decision-making systems: A workflow centered framework. *Journal of Biomedical Informatics*, 43(1), Feb 2010.

[48] K. Jegers. Pervasive game flow: understanding player enjoyment in pervasive gaming. *Computers in Entertainment*, 5(1), Jan 2007.

[49] D. Johnson and J. Wiles. Effective affective user interface design in games. *Ergonomics*, 46(13 & 14):1332–1345, Oct 2003.

[50] P. Johnson. Usability and mobility; interactions on the move. *Proceedings of the First Workshop on Human Computer Interaction with Mobile Devices*, May 1998.

[51] M. Jones, G. Bradley, S. Jones, and G. Holmes. Navigation-by-music for pedestrians: an initial prototype and evaluation. *Proceedings of the International Symposium on Intelligent Environments*, pages 95–101, Apr 2006.

[52] D. Kahneman and A. Tversky. Prospect theory: An analysis of decision under risk. *Econometrica*, 47(2):263–292, Mar 1979.

[53] K. Kallinen, M. Salminen, N. Ravaja, and K. Yanev. Psychophysiological responses to online poker game. *IADIS'09: Proceedings of Game and Entertainment Technologies*, pages 35–43, 2009.

[54] A. Kankainen and A. Oulasvirta. Design ideas for everyday mobile and ubiquitous computing based on qualitative user data. *ERCIM'02: Proceedings of the User interfaces for all 7th international conference on Universal access: theoretical perspectives, practice, and experience*, Oct 2002.

[55] K. Kappel and T. Grechenig. "show-me": water consumption at a glance to promote water conservation in the shower. *Persuasive '09: Proceedings of the 4th International Conference on Persuasive Technology*, Apr 2009.

[56] S. Karnouskos. Mobile payment: A journey through existing procedures and standardization initiatives. *IEEE Communications Surveys & Tutorials*, 6(4):44 – 66, 2004.

[57] F. Kawsar, T. Nakajima, and K. Fujinami. Deploy spontaneously: supporting end-users in building and enhancing a smart home. *UbiComp '08: Proceedings of the 10th international conference on Ubiquitous computing*, Sep 2008.

[58] P. G. W. Keen. Decision support systems : a research perspective. *Decision support systems : issues and challenges*, 1980.

[59] P. G. W. Keen and M. S. S. Morton. Decision support systems: An organizational perspective. 1978.

[60] S. Kristoffersen and F. Ljungberg. "making place" to make it work: empirical explorations of hci for mobile cscw. *Proceedings of the international ACM SIGGROUP conference on Supporting group work*, pages 276–285, Jan 1999.

[61] K. V. Laerhoven, N. Villar, and H.-W. Gellersen. Multi-level sensory interpretation and adaptation in a mobile cube. *AIMS'03: Proceedings of the 3rd workshop on Artificial Intelligence in Mobile Systems*, pages 111–117, Oct 2003.

[62] V. Lehdonvirta, H. Soma, H. Ito, H. Kimura, and T. Nakajima. Ubipay: conducting everyday payments with minimum user involvement. *CHI '08: CHI '08 extended abstracts on Human factors in computing systems*, Apr 2008.

[63] Q. Limbourg and J. Vanderdonckt. Multimodality and context-aware adaptation. *Building the Information Society*, 156:427–432, Jan 2004.

[64] J. Lin, J. Wong, J. Nichols, A. Cypher, and T. Lau. End-user programming of mashups with vegemite. *IUI '09: Proceedingsc of the 13th international conference on Intelligent user interfaces*, Feb 2009.

[65] C. Magerkurth, A. Cheok, R. Mandryk, and T. Nilsen. Pervasive games: bringing computer entertainment back to the real world. *Computers in Entertainment*, 3(3), Jul 2005.

[66] P. Maglio, T. Matlock, C. Campbell, S. Zhai, and B. Smith. Gaze and speech in attentive user interfaces. *ICMI '00: Proceedings of the 3rd International Conference on Advances in Multimodal Interfaces*, Oct 2000.

[67] S. Mainwaring, W. March, and B. Maurer. From meiwaku to tokushita!: lessons for digital money design from japan. *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, Apr 2008.

[68] N. Mallat, M. Rossi, V. Tuunainen, and A. Öörni. An empirical investigation of mobile ticketing service adoption in public transportation. *Personal and Ubiquitous Computing*, 12(1), Jan 2008.

[69] J. Mäntyjärvi, J. Himberg, P. Kangas, U. Tuomela, and P. Huuskonen. Sensor signal data set for exploring context recognition of mobile devices. *Workshop "Benchmarks and a database for context recognition" in conjuction with the 2nd Int. Conf. on Pervasive Computing (PERVASIVE 2004)*, Apr 2004.

[70] G. Marakas. Decision support systems in the twenty-first century. *Decision support systems in the twenty-first century*, Dec 1998.

[71] J. Marila and S. Ronkainen. Time-out in user interface: the case of mobile text input. *Personal and Ubiquitous Computing*, 8(2), May 2004.

[72] A. M. McCosh. Comments on 'a brief history of dss'. *email to D. Power*, Oct 2002.

[73] S. Mizobuchi, M. Chignell, and D. Newton. Mobile text entry: relationship between walking speed and text input task difficulty. *MobileHCI '05: Proceedings of the 7th international conference on Human computer interaction with mobile devices & services*, Sep 2005.

[74] M. Motterlini and C. Somajni. Economia emocional/ emotional economy: En que nos gastamos el dinero y por que/ in what do we waste money in and why. page 306, Jan 2008.

[75] J. Müller, M. Jentsch, C. Kray, and A. Krüger. Exploring factors that influence the combined use of mobile devices and public displays for pedestrian navigation. *NordiCHI'08: Proceedings of the 5th Nordic conference on Human-computer interaction: building bridges*, pages 308–317, 2008.

[76] J. Müller, D. Wilmsmann, J. Exeler, M. Buzeck, A. Schmidt, T. Jay, and A. Krüger. Display blindness: The effect of expectations on attention towards digital signage. *Pervasive '09: Proceedings of the 7th International Conference on Pervasive Computing*, May 2009.

[77] T. Mustonen, M. Olkkonen, and J. Hakkinen. Examining mobile phone text legibility while walking. *CHI '04: CHI '04 extended abstracts on Human factors in computing systems*, Apr 2004.

[78] L. Nacke, C. Lindley, and S. Stellmach. Log who's playing: Psychophysiological game analysis made easy through event logging. *Proceedings of the 2nd International Conference on Fun and Games*, Oct 2008.

[79] T. Nakajima, V. Lehdonvirta, E. Tokunaga, and H. Kimura. Reflecting human behavior to motivate desirable lifestyle. *DIS '08: Proceedings of the 7th ACM conference on Designing interactive systems*, Feb 2008.

[80] M. Newman, J. Sedivy, C. Neuwirth, W. Edwards, J. Hong, S. Izadi, K. Marcelo, and T. Smith. Designing for serendipity: supporting end-user configuration of ubiquitous computing environments. *DIS '02: Proceedings of the 4th conference on Designing interactive systems: processes, practices, methods, and techniques*, Jun 2002.

[81] J. Niehans. Transaction costs. *The New Palgrave: A Dictionary of Economics. First Edition*, 4:667–680, 1987.

[82] J. Pascoe, N. Ryan, and D. Morse. Using while moving: Hci issues in fieldwork environments. *Transactions on Computer-Human Interaction*, 7(3), Sep 2000.

[83] J. M. Pearson and J. Shim. An empirical investigation into dss structures and environments. *Decision Support Systems*, 13(2):141–158, 1995.

[84] P. Peltonen, A. Salovaara, G. Jacucci, T. Ilmonen, C. Ardito, P. Saarikko, and V. Batra. Extending large-scale event participation with user-created mobile media on a public display. *MUM '07: Proceedings of the 6th international conference on Mobile and ubiquitous multimedia*, Dec 2007.

[85] R. E. Petty and J. T. Cacioppo. Communication and persuasion: Central and peripheral routes to attitude change. *Springer Series in Social Psychology*, page 262, Sep 1986.

[86] R. E. Petty and J. T. Cacioppo. The elaboration likelihood model of persuasion. *Advances in experimental social psychology*, 19:123–205, 1986.

[87] M. Pham. Emotion and rationality: A critical review and interpretation of empirical evidence. *Review of General Psychology*, Jan 2007.

[88] M. Poch, J. Comas, I. R. guez Roda, M. Sanchez-Marre, and U. C. s. Designing and building real environmental decision support systems. *Environmental Modelling & Software*, 19(9):857–873, Nov 2004.

[89] D. J. Power. What is a dss? *The On-Line Executive Journal for Data-Intensive Decision Support*, 1(3), 1997.

[90] D. J. Power. Web-based and model-driven decision support systems: Concepts and issues. *AMCIS 2000: Proceedings of the 16th Americas Conference on Information Systems*, 2000.

[91] D. J. Power. Categorizing decision support systems: a multidimensional approach. *Decision making support systems*, Jan 2002.

[92] D. J. Power. Decision support systems: concepts and resources for managers. page 251, Jan 2002.

[93] D. J. Power. Specifying an expanded framework for classifying and describing decision support systems. *Communications of the Association for Information Systems*, 13(1):158–166, Feb 2004.

[94] D. J. Power. A brief history of decision support systems. *Handbook on Decision Support Systems 1*, pages 121–140, 2008.

[95] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan. The cricket location-support system. *ACM MOBICOMM'00: Proceedings of the 6th ACM International Conference on Mobile Computing and Networking*, pages 32–43, Mar 2000.

[96] J. Rasmussen. Skills, rules, and knowledge; signals, signs, and symbols, and other distinctions in human performance models. *IEEE Transactions on Systems, Man and Cybernetics*, 13(3):257–266, Jan 1983.

[97] J. Rasmussen. Information processing and human-machine interaction: An approach to cognitive engineering. *Information Processing and Human-Machine Interaction: An Approach to Cognitive Engineering*, Sep 1986.

[98] J. Reason. Human error. page 302, Jan 1990.

[99] A. Sage. Decision support systems engineering. *Decision support systems engineering*, Aug 1991.

[100] N. Sawhney and C. Schmandt. Nomadic radio: speech and audio interaction for contextual messaging in nomadic environments. *Transactions on Computer-Human Interaction*, 7(3), Sep 2000.

[101] B. N. Schilit, N. Adams, and R. Want. Context-aware computing applications. *Proceedings of the Workshop on Mobile Computing Systems and Applications*, pages 85–90, Dec 1994.

[102] N. Schwarz. Emotion, cognition, and decision making. *Cognition & Emotion*, 14(4):433–440, Jul 2000.

[103] N. Schwarz. Feelings-as-information theory. *Handbook of theories of social psychology*, page (in press), Jan 2010.

[104] N. Schwarz and G. L. Clore. Mood, misattribution, and judgments of well-being: Informative and directive functions of affective states. *Journal of Personality and Social Psychology*, 45(3):513–523, 1983.

[105] N. Schwarz and G. L. Clore. Mood as information: 20 years later. *Psychological Inquiry*, 14(3&4):296–303, 2003.

[106] J. P. Shim, M. Warkentin, J. F. Courtney, D. J. Power, R. Sharda, and C. Carlsson. Past, present, and future of decision support technology. *Decision Support Systems*, 33(2), Jun 2002.

[107] M. Shiraishi, Y. Washio, C. Takayama, V. Lehdonvirta, H. Kimura, and T. Nakajima. Using individual, social and economic persuasion techniques to reduce co2 emissions in a family setting. *Persuasive '09: Proceedings of the 4th International Conference on Persuasive Technology*, Apr 2009.

[108] A. Shirazi, T. Döring, P. Parvahan, B. Ahrens, and A. Schmidt. Poker surface: combining a multi-touch table and mobile phones in interactive card games. *MobileHCI '09: Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services*, Sep 2009.

[109] D. Siewiorek, A. Smailagic, J. Furukawa, A. Krause, N. Moraveji, K. Reiger, J. Shaffer, and F. Wong. Sensay: A context-aware mobile phone. *ISWC '03: Proceedings of the 7th IEEE International Symposium on Wearable Computers*, Oct 2003.

[110] H. A. Simon. The new science of management decision. *The New Science of Management Decision*, 1977.

[111] C. L. Smith. Computer-supported decision making: Meeting the decision demands of modern organizations. page 172, Jan 1998.

[112] R. H. Sprague. A framework for the development of decision support systems. *MIS Quarterly*, 4(4):1–26, 1980.

[113] R. H. Sprague and E. D. Carlson. Building effective decision support systems. 1982.

[114] J. Sweller. Cognitive load during problem solving: Effects on learning. *Cognitive Science*, 12(2):257–285, 1988.

[115] J. Sweller. Cognitive load theory, learning difficulty, and instructional design. *Learning and Instruction*, 4(4):295–312, 1994.

[116] J. Sweller, J. J. G. van Merrienboer, and F. G. W. C. Paas. Cognitive architecture and instructional design. *EDUCATIONAL PSYCHOLOGY REVIEW*, 10(3):251–296, 1998.

[117] S. Tamminen, A. Oulasvirta, K. Toiskallio, and A. Kankainen. Understanding mobile contexts. *Personal and Ubiquitous Computing*, 8(2), May 2004.

[118] K. Tsukada and M. Yasumura. Activebelt: Belt-type wearable tactile display for directional navigation. *Lecture Notes in Computer Science. Springer-Verlag GmbH, Oct 2004*, pages 384–399, Jan 2004.

[119] M. Turunen, J. Hakulinen, A. Kainulainen, A. Melto, and T. Hurtig. Design of a rich multimodal interface for mobile spoken route guidance. *Proceedings of Interspeech 2007 - Eurospeech*, pages 2193–2196, Mar 2007.

[120] A. Tversky and D. Kahneman. The framing of decisions and the psychology of choice. *Science*, 211(4481):453–458, Jan 1981.

[121] R. Vertegaal. Attentive user interfaces. *Communications of the ACM*, 46(3):26, Mar 2003.

[122] W. E. Walker. Organizational decision support systems: Centralized support for decentralized organizations. *ANNALS OF OPERATIONS RESEARCH*, 51(6):283–298, 1994.

[123] G. Wang, S. Yang, and Y. Han. Mashroom: end-user mashup programming using nested tables. *WWW '09: Proceedings of the 18th international conference on World wide web*, Apr 2009.

[124] W. Weber, J. M. Rabaey, and E. Aarts. Ambient intelligence. page 373, Jan 2005.

[125] M. Weiser. The computer for the 21st century. *Scientific American*, 265(3):66–75, Sep 1991.

[126] M. Weiser and J. S. Brown. The coming age of calm technology. 1996.

[127] C. D. Wickens. Processing resources in attention. *Varieties of Attention*, pages 63–101, 1984.

[128] C. D. Wickens. Engineering psychology and human performance. page 560, Jan 1992.

[129] C. D. Wickens. Multiple resources and performance prediction. *Theoretical Issues in Ergonomics Science*, 3(2):159–177, Jul 2002.

[130] T. Winograd. Architectures for context. *Human-Computer Interaction*, 16(2):401–419, May 2001.

[131] E. Wulfert, C. Franco, K. Williams, B. Roland, and J. H. Maxson. The role of money in the excitement of gambling. *Psychology of Addictive Behaviors*, 22(3):380–90, Sep 2008.

[132] T. Yamabe, K. Fujinami, and T. Nakajima. Experiences with building sentient materials using various sensors. *ICDCSW '04: Proceedings of the 24th IEEE International Conference on Distributed Computing Systems Workshops*, 7:445–450, 2004.

[133] T. Yamabe, A. Takagi, and T. Nakajima. Citron: A context information acquisition framework for personal devices. *RTCSA '05: Proceedings of the 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, 2005.

[134] T. Yamabe and K. Takahashi. Experiments in mobile user interface adaptation for walking users. *IPC '07: Proceedings of the 2007 International Conference on Intelligent Pervasive Computing*, pages 280–284, 2007.

[135] T. Yamabe, K. Takahashi, and T. Nakajima. Towards mobility oriented interaction design: experiments in pedestrian navigation on mobile devices. *MobiQuitous '08: Proceedings of the 5th Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services*, pages 1–10, 2008.

[136] W. W. Zachary and J. M. Ryder. Decision support systems: Integrating decision aiding and decision training. *Handbook of Human-Computer Interaction (Second Edition)*, pages 1235 – 1258, 1997.

Publications

[1] Takahiro Shichinohe, Tetsuo Yamabe, Takahiro Iwata, and Tatsuo Nakajima. Augmented calligraphy: Experimental feedback design for writing skill development. *TEI '11: Proceedings of the fifth international conference on Tangible, embedded, and embodied interaction*, 2011.

[2] Takahiro Shichinohe, Tetsuo Yamabe, Takahiro Iwata, and Tatsuo Nakajima. Demonstration of augmented calligraphy system. *IoT '10: Adjunct Proceedings of the Internet of Things Conference 2010*, 2010.

[3] Tetsuo Yamabe, Ilkka Kosunen, Inger Ekman, Lassi A Liikkanen, Kai Kuikkaniemi, and Tatsuo Nakajima. Biofeedback training with emopoker: Controlling emotional arousalfor better poker play. *FNG '10: Fun and Games Conference 2010*, 2010.

[4] Takahiro Iwata, Tetsuo Yamabe, and Tatsuo Nakajima. Towards a mobility enhanced user interface design for multi-task environments: An experimental study on cognitive workload measurement. *IE '10 Proceedings of the 2010 Sixth International Conference on Intelligent Environmen*, pages 106–111, 2010.

[5] Tetsuo Yamabe, Yasuyuki Washio, Sota Matsuzawa, and Tatsuo Nakajima. Empowering end-users to find point-of-interests with a public display. *ICPS '10: Proceedings of the 2010 International Conference on Pervasive Services*, 2010.

[6] Tetsuo Yamabe, Vili Lehdonvirta, Hitoshi Ito, Hayuru Soma, Hiroaki Kimura, and Tatsuo Nakajima. Activity-based micro-pricing: Realizing sustainable behavior changes through economic incentives. *Persuasive '10: Proceedings of The Fifth International Conference on Persuasive Technology*, 2010.

[7] Takahiro Iwata, Tetsuo Yamabe, Mikko Polojärvi, and Tatsuo Nakajima. Traditional games meet ict: a case study on go game augmentation. *TEI '10: Proceedings of the 4th international conference on Tangible, embedded, and embodied interaction*, 2010.

[8] Tetsuo Yamabe, Vili Lehdonvirta, Hitoshi Ito, Hayuru Soma, Hiroaki Kimura, and Tatsuo Nakajima. Applying pervasive technologies to create economic incentives that alter consumer behavior. *Ubicomp '09: Proceedings of the 11th international conference on Ubiquitous computing*, 2009.

[9] Hitoshi Ito, Tetsuo Yamabe, and Tatsuo Nakajima. Design issues of smart objects in activity-based micro pricing systems. *DIPSO '09: The Third International Workshop on Design and*

Integration Principles for Smart Objects, 2009.

[10] Tetsuo Yamabe and Tatsuo Nakajima. Possibilities and limitations of context extraction in mobile devices: Experiments with a multi-sensory personal device. *International Journal of Multimedia and Ubiquitous Engineering*, 4(4), 2009.

[11] Vili Lehdonvirta, Hayuru Soma, Hitoshi Ito, Tetsuo Yamabe, Hiroaki Kimura, and Tatsuo Nakajima. Ubipay: minimizing transaction costs with smart mobile payments. *Mobility '09: Proceedings of the 6th International Conference on Mobile Technology, Application & Systems*, 2009.

[12] Tetsuo Yamabe, Kiyotaka Takahashi, and Tatsuo Nakajima. Design issues and an empirical study in mobility oriented service development. *MobMid '08: Proceedings of the 1st workshop on Mobile middleware: embracing the personal communication device*, 2008.

[13] Tetsuo Yamabe, Kiyotaka Takahashi, and Tatsuo Nakajima. Towards mobility oriented interaction design: experiments in pedestrian navigation on mobile devices. *MobiQuitous '08: Proceedings of the 5th Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services*, pages 1–10, 2008.

[14] Tetsuo Yamabe, Kiyotaka Takahashi, and Tatsuo Nakajima. Demonstration of a mobility-enhanced pedestrian navigation on mobile devices. *Mobiquitous '08 Proceedings of the 5th Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services*, page 29, 2008.

[15] Tatsuo Nakajima, Hiroaki Kimura, Tetsuo Yamabe, Vili Lehdonvirta, Chihiro Takayama, Miyuki Shiraishi, and Yasuyuki Washio. Using aesthetic and empathetic expressions to motivate desirable lifestyle. *EuroSSC '08 Proceedings of the 3rd European Conference on Smart Sensing and Context*, pages 220–234, 2008.

[16] Tetsuo Yamabe and Kiyotaka Takahashi. Experiments in mobile user interface adaptation for walking users. *IPC '07: Proceedings of the 2007 International Conference on Intelligent Pervasive Computing*, pages 280–284, 2007.

[17] Kiyotaka Takahashi and Tetsuo Yamabe. A proposal on adaptive service migration framework for device modality using media type conversion. *IPC '07: Proceedings of the The 2007 International Conference on Intelligent Pervasive Computing*, pages 249–253, 2007.

[18] Tetsuo Yamabe, Ayako Takagi, and Tatsuo Nakajima. Citron: A context information acquisition framework for personal devices. *RTCSA '05: Proceedings of the 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, 2005.

[19] Kaori Fujinami, Tetsuo Yamabe, and Tatsuo Nakajima. "take me with you!": a case study of context-aware application integrating cyber and physical spaces. *SAC '04: Proceedings of the 2004 ACM symposium on Applied computing*, 2004.

[20] Kaori Fujinami, Tetsuo Yamabe, and Tatsuo Nakajima. Bazaar: A conceptual framework for physical space applications. *UCS '04: Proceedings of the 2nd International Symposium on Ubiquitous Computing Systems*, pages 174–191, 2004.

[21] Tetsuo Yamabe, Kaori Fujinami, and Tatsuo Nakajima. Experiences with building sentient materials using various sensors. *ICDCSW '04: Proceedings of the 24th IEEE International Conference on Distributed Computing Systems Workshops*, 7:445–450, 2004.