ROBERT GORDON UNIVERSITY•ABERDEEN

Situation Awareness Approach to Context-Aware Case-Based Decision Support

Nuka D. Nwiabu

A thesis submitted in partial fulfilment
of the requirements of
Robert Gordon University
for the degree of Doctor of Philosophy

August 2012

Supervised by: Prof. Ian Allison, Prof. Patrik Holt, Peter Lowit, Prof. Babs Oyeneyin

Abstract

Context-aware case-based decision support systems (CACBDSS) use the context of users as one of the features for similarity assessment to provide solutions to problems. The combination of a context-aware case-based reasoning (CBR) with general domain knowledge has been shown to improve similarity assessment, solving domain specific problems and problems of uncertain knowledge. Whilst these CBR approaches in context awareness address problems of incomplete data and domain specific problems, future problems that are situation-dependent cannot be anticipated due to lack of data by the CACBDSS to make predictions. Future problems can be predicted through situation awareness (SA), a psychological concept of knowing what is happening around you in order to know the future.

The work conducted in this thesis explores the incorporation of SA to CACBDSS. It develops a framework to decouple the interface and underlying data model using an iterative research and design methodology. Two new approaches of using situation awareness to enhance CACBDSS are presented: (1) situation awareness as a problem identification component of CACBDSS (2) situation awareness for both problem identification and solving in CACBDSS. The first approach comprises of two distinct parts; SA, and CBR parts. The SA part understands the problem by using rules to interpret cues from the environment and users. The CBR part uses the knowledge from the SA part to provide solutions. The second approach is a fusion of the two technologies into a single case-based situation awareness (CBSA) model for situation awareness based on experience rather than rule, and problem solving predictions. The CBSA system perceives the users' context and the environment and uses them to understand the current situation by retrieving similar past situations. The futures of new situations are predicted through knowledge of the history of similar past situations.

Implementation of the two approaches in flow assurance control domain to predict the formation of hydrate shows improvements in both similarity assessment and problem solving predictions compared to CACBDSS without SA. Specifically, the second approach provides an improved decision support in scenarios where there are experienced situations. In the absence of experienced situations, the second approach offers more reliable solutions because of its rule-based capability. The adaptation of the user interface of the approaches to the current situation and the presentation of a reusable sequence of tasks in the situation reduces memory loads on operators.

The integrated research-design methodology used in realising these approaches links theory and practice, thinking and doing, achieving practical as well as research objectives. The action research with practitioners provided the understanding of the domain activities, the social settings, resources, and goals of users. The user-centered design process ensures an understanding of the users. The agile development model ensures an iterative work, enables faster development of a functional prototype, which are more easily communicated and tested, thus giving better input for the next iteration.

0.1 Acknowledgments

First, I want to express my gratitude to my supervisors; Prof. Ian Allison, Prof. Patrik Holt, Peter Lowit, and Prof. Babs Oyeneyin for the discussions, insights, support and patience throughout the period of my PhD research. I feel so privileged to be guided by such a formidable team and really learned a lot through the discussions.

Also I would like to thank Dr. Ibrahim Adeyanju who assisted me in the use of CBR technologies and was always ready to answer my questions.

I would like to extend my gratitude to my colleagues in Computer Technologies Centre (CTC) for their support and also for making CTC conducive for productive research namely Sadiq, Ben, Guofu, Richard, Noura, Leszek, Malcolm, Olivier, Peng, Amina, Abubakar, Michael, Lei, Kabiru, Alwawi, Yanghui, Konstantinos, Claire, David and Mehdi.

A special thanks to my wife, Lesi, for her love, encouragement and support.

The last and the biggest thanks goes to God who gave me everything that was required for the beginning and the completion of this project.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Decision support systems (DSS) have previously been identified as systems intended to support managerial decision making in semistructured decision situations [32]. Recent researches are focused on integrating knowledge into decision support systems [72]. Incorporating knowledge into decision support systems has been recognized as a means of gaining competitive advantage, formulating better problem-solving processes, improving decision quality and refining business operations [22]. The term knowledge-based decision support systems (KBDSS) is often used to describe the approaches to integrate decision support systems architecture with Artificial Intelligence (AI) or Expert System (ES) techniques.

In dynamic environments, decision making often involves the exploration of situations that do not yet exist (future situations) [32]. Analysing such situations requires a model or abstraction of reality rather than reality itself [190]. Models make the structure of the situation explicit. Models are used to portray the important aspects of reality in a particular situation while eliminating other aspects. The integration of KBDSS to a situation assessment model will provide KBDSS the capability to dynamically assess decision situations. The assessment will benefit operators in complex and dynamic environments with effective decision making strategies. Experience-based models such as case-based reasoning (CBR) models contain explicit knowledge which can be used to enrich the explanation of a KBDSS and thus making it intuitive [129]. CBR builds on an understanding of how humans assess situations [162], supporting recognition-primed decision (RPD) framework proposed by Klein et al. [162]. The RPD framework emphasises the role of experiences in human decision making processes during time critical situations. Klein pointed out that humans depend more on past experience rather than deliberate rational analysis of possible alternatives during time-critical decision making. Rita et al. [153] present a hypothesis for a framework including a case-based and knowledge-base for uncertainty handling in expert DSS. But the core problem of case-based reasoning or AI, learning, has not been considered in their basic framework configuration to aid future decision making or problem solving. In a case-based system, learning can occur in the process of memorizing new cases, classifying existing cases, and generalizing knowledge from cases [162]. As learning is a byproduct of problem solving for case-based reasoning [72], case-based decision support systems (CBDSS) not only is an alternative method for KBDSS, but also is a necessary beginning for integrating learning mechanism into DSS. Case-based reasoning methodology presents a foundation for a new technology of building intelligent DSS [162] that will use learning to understand different decision situations and solutions to problems in the situations.

A specific decision situation is recognised through situation awareness (SA). SA involves cognitive processes employed by operators in a complex and dynamic environment to understand the current state of the environment in order to anticipate its future state. SA consists of three layers: perception, comprehension, and projection [54]. The perception layer recognises all the necessary information or elements on the current situation. The comprehension layer interprets the perceived information in order to understand the current situation. The projection layer uses the understanding of the current situation to predict its future state. In the goal-directed perspective of SA advocated by Endsley et al [59] goal influences what the operator perceives, comprehends and anticipates [86].

A related concept to situation awareness is the notion of context awareness. Context, like goal, acts as a filter to SA. Context filters SA in relation to the specific need of individual operators. Context awareness allows systems to dynamically adapt to changes in a user's task domain, by updating relevant information and service provision, whereas situation awareness focuses on information about the state of the environment in which these tasks are carried out [64]. A human operator can use context to make sense of a confused, obscure, and conflicting situation. A system is said to be context aware if it uses context to provide relevant information and services to the user [148]. Context awareness was introduced by Schilit et al [167] to develop an application that adapts to the location of use, nearby people and objects, and the change of those objects over time. With technology advancement and the rapid growth of mobile computing in recent times, context awareness has attracted greater research attention [64].

The aim of the work presented here is to design case-based decision support systems that will achieve situation awareness for the specific needs of individual users. The system's user interface adapts to the user context in the current situation to display his/her needs. The scope covers any domain where the ability to be aware of the occurring situations and react experientially to them is required. The case study for the project is the flow assurance control domain. The system is implemented to predict the formation of hydrate in sub-sea oil and gas pipelines. Hydrate formation is one of the major challenges in deep-water oil and gas operation. Nigeria as an oil producing country is shifting its oil and gas activities from onshore and shallow water to deep-water operations. As part of capacity building to address flow assurance challenges from different perspectives this research is funded by the Nigerian government through its agency, Petroleum Technology Development Fund (PTDF) to explore computational approach to hydrate prevention.

1.1 Research Questions and Focus

The main research question and focus in the work presented here is how can case-based decision support systems form situation awareness through reasoning about perceivable context and elements from the environment?

The attempt to answer the main question raised four sub questions:

1. How can CBDSS use situation awareness for problem identification? Looking at problem solving and situation awareness as two separate tasks, an architecture where a case-based problem solving task is dependent on a rule-based situation awareness task will

be discussed. The task of situation awareness involves the use of historical data and rules to interpret cues from the environment with respect to an individual user context, and to anticipate future situations by performing statistical inference on these historical data. SA shall provide relevant information about the environment to be used in the case-based problem solving reasoning process.

2. Can the user interface of the system be modelled to reduce the cognitive load on the user? Existing user interface design does not solve the problem of drawing together the information required for SA systems in a way that minimises cognitive load. These approaches look at situations mainly from the perspective of situation assessment and classification and do not differentiate between a process and a state. Situation awareness is a state of knowledge while situation assessment and classification are the processes used to achieve that knowledge. This work will present a framework for user interface design of situation-aware systems that exploits inputs from users and the environment to provide information tailored to the user's tasks in specific situations based on SA as a state of knowledge. So how would the interface be designed and through what means will it reduce the cognitive load on operators?

3. How can CBDSS use situation awareness in both problem identification and solving? The idea of using SA for both problem identification and solving is based on the idea that the ability to reason about the world is closely linked with being knowledgeable about the world. Thus, a knowledge intensive perspective on reasoning methods is a potentially feasible approach. Case-based reasoning in general and knowledge intensive case-based reasoning in particular, appears to be a promising candidate method for identifying problems by situation awareness as well as problem solving. Case-based reasoning is an artificial intelligence method of assessing and classifying situations that builds on theories on how humans assess situations and react to them [3]. Knowledge intensive case-based reasoning closely couples episodic memory [189], in the form of cases, with the more general domain knowledge and then reasons about the current state of the world by maintaining a model of the causal relationships that exists.

4. Which research method will support collaborative work with domain experts? The

aim of the work presented here is to design systems that will help to solve organisational problems as well as provide answers to research questions. Modelling such systems requires the participation of domain practitioners in order to understand the activities of the domain, capture the practitioners requirements, refining the requirements for redesign at an early stage to save time and cost. A method that will link theory and practice, thinking and doing, reflecting on the process and the product, achieving practical as well as research objectives. The task of reflecting on the process and the product should adopt the INRECA methodology which consists of CBR development experiences, represented as software process models and stored in the experience base of an experience factory. The experience factory provides the framework for storing, accessing, and extending the guidelines for the CBR application development [16].

1.2 Research Aim and Objectives

The overall aim of this project was to develop appropriate modelling tools and methods to design and develop decision support systems in a process control decision environment that reflects the current situation. Most approaches do not enable the dynamic data situation to be modelled. Using case-based reasoning to be context aware allowed for a flexible identification of ongoing situations. The toolset enabled users to define the parameters for themselves.

The objectives for this research project were therefore, to:-

1. Investigate and review how current decision support systems anticipate and reflect the on-going situation in production control environments to support the decision makers.

2. Evaluate current approaches available for modelling the design of situation-aware interactive applications. Following a review of the literature, existing design notations were reviewed and assessed to critique the way they depict the changing underlying information to the user.

3. Investigate existing design patterns to consider how to decouple the interface and underlying data model.

4. Develop appropriate methods to be utilised by designers in a software development

environment.

5. Test the ideas in a problem domain as a proof of concepts.

6. Undertake an action-research-based evaluation of the developed tools through a series of pilot applications and evaluations to assess and improve the modelling notation through the use of error analysis.

1.3 Contributions

The work contributes in making situation awareness a useful tool in case-based decision support systems (CBDSS) for problem solving in complex environments. In theory, the project argues that the future of the current situation can be anticipated through experiential knowledge of the elements that defines the situation and the current state of the environment. Practically, the thesis contributes in the following ways:-

- Situation awareness as a problem identification component of case-based decision support systems (CBDSS). The approach comprises of two distinct parts; situation awareness (SA), and case-based reasoning (CBR) parts. The SA part keeps a finite history of the time space information of the domain and uses rule to interpret cues from the environment with respect to an individual user context, and to anticipate future situations by performing statistical inference on these historical data. The CBR part is the part that seeks to accomplish a particular task. Knowledge of the domain situation from the situation awareness part is used to recall situations that have happened in the past, in similar contexts, and for similar states of the environment. Enriching knowledge intensive case-based reasoning with additional knowledge (SA) makes retrieved solutions immediately ready to be used in the solving of new problems. SA improves both similarity assessment and problem solving prediction of the CBDSS.

- Situation awareness as a means of problem identification and problem solving in CBDSS. This second approach discusses the fusion of the CBR model and the SA model into a case-based situation awareness (CBSA) model for situation awareness based on experience rather than rules, similarity assessment and problem solving prediction. The CBSA system perceives the users' context and the environment and uses them to understand the current situation by retrieving similar past situations. Every past situation has a history. The future of a new situation (case) is predicted through the user expectation and knowledge of the history of a similar past situation. The approach provides decision support without knowledge of domain rules. This premise is used to develop an experience-based SA using case-based reasoning (CBR). Experience is a critical element for a human operator to have good situation awareness (SA) [88]. The approach provided a framework and an architecture for building efficient single-model case-based situation awareness systems with minimal time and cost. It also shows how the feature of expectations can be incorporated into users' context to enable the system meet the specific needs of individual operators.

- By developing a framework for user interface design of situation-aware systems that exploits inputs from users and the environment to provide information tailored to the user's tasks in specific situations. The user interface has the ability to execute reconfiguration after input variation so as to stay adapted to the current situation. The adaptation of the interface to the current situation and the presentation of reusable tasks in the situation with reduced number of commands, clicks, and options reduces cognitive loads on operators and thereby facilitates interactions. The approach also demonstrated a method of combining scenarios, Hierarchical Task Analysis (HTA), and requirements analysis in task modelling. The methods complement each other by using scenarios to stimulate and support reasoning in task analysis. Task analysis provides an integrated picture of tasks. Mapping real, complete and representative tasks of HTA to abstract and partial tasks of requirements analysis helps to ensure that all important users' tasks with their relationships and interactions are identified.

- By developing an integrated and iterative research/design methodology to realise the approaches. The technique involves the integration of action research (AR) method with user-centered design (UCD), and agile development (AD) to form a comprehensive research-design cycle. The integration of these different methods for the industry-based research results in a research-design process comprising three segments; scenarios, agile user-centered design, and business change. The systematic and iterative method links theory and practice, thinking and doing, achieving practical as well as research objectives. The action research (AR) with practitioners provided the understanding of the domain activities, the social settings, resources, and goals of users. The user-centered design (UCD) process ensures an understanding of the users. The agile development (AD) model ensures iterative development of the work, enabling faster development of a functional prototype, which is more easily communicated and tested.

- Carrying out a case study of the approaches in flow assurance control domain in the oil and gas industry for hydrate formation prediction in sub sea gas pipelines. Situation awareness in hydrate formation is a hydrate monitoring investigation which focuses on preventative prediction rather than solutions to hydrate formed situations. Hydrates are considered as nuisance because they block transmission lines, plug blowout preventers, jeopardise the foundation of deep-water platforms and pipelines. Hydrate prevention measures are more effective than remedial removal of hydrate plug. The state-of-the-art means of preventing hydrate in oil and gas operation is through the injection of inhibitors (methanol/glycol) to dehydrate the gas. The effectiveness of inhibitors reduces with increase in water depth. This system utilises sensor data deployed on sub-sea pipelines, elements from the ocean floor, general knowledge of hydrate control and past experiences to produce a method of predicting hydrate formation.

- From the action research this work has had output in both academic and practice-based publications. A paper [128] for the 2011 IEEE conference on situation awareness and decision support was developed. The research work won the Best Paper award. For practice, Roustabout Energy International magazine in March 2011 reported the product in relation to solving practical oil and gas hydrate problems.

1.4 Scope of study

The focus of this project work is on individual situation awareness. Special interest is placed on individual user's context and their customised SA based on their individual context. The approach highlights the relationship between situation awareness and context awareness, and how the two concepts together with domain knowledge improves the similarity assessment of a CBR system. To complete this project within the three years time frame, the study of situation awareness is not extended from individual SA to team SA. Team SA is multi-dimensional and comprises individual team member SA, shared SA between team members and also the combined SA of the whole team, the so-called "common picture". Also, the work does not cover the recent concept of distributive situation awareness (DSA), which treats team SA as a systems level phenomenon and focuses on the overall system itself as the unit of analysis rather than the individuals within it. DSA approaches are based on the notion that in order to understand behaviour in complex systems it is more useful to study the interactions between the parts of the system and the resultant emerging behaviour rather than study the parts themselves.

The work is focused on problems in situations of uncertainties. For instance, situations where the general domain knowledge is difficult to extract and instead reasoning is based on experience or where it is difficult to formulate rules describing the situations.

1.5 Thesis Overview

The remainder of the thesis is as follows. Chapter 2 presents a review of situation awareness (SA) and how decision support systems are created to realise situation awareness. It then describes the different methods in measuring situation awareness. Chapter 3 presents the methodology used in achieving the research process and in modelling the design of the proposed systems. Chapter 4 presents the first proposed approach which models situation awareness as a problem identification component of case-based decision support systems (CBDSS). Chapter 5 discusses a situation-aware user interface (SAUI), a framework to decouple the interface and underlying data model of the first approach. Chapter 6 presents the second approach which extends situation awareness from problem identification to problem solving. The approach discusses SA as both problem identification and problem component of CBDSS. Chapter 7 is a critical analysis of the approaches. The thesis is summarised and interesting areas of future work are proposed in Chapter 8.

Chapter 2

Background

This chapter presents a review of situation awareness (SA) starting with its definition after a survey on some prior work. The question remains as to whether a definition of situation awareness should be limited to a content or should include the processes or functions linked to the awareness of the situation. Many definitions of situation awareness have been expanded by developing SA models in an attempt to answer the question.

A related concept to situation awareness is the notion of context awareness. These terms are often considered synonymous. The chapter draws a distinction between the use of these two concepts and shows how context aware decision support systems are created to realise situation awareness. The methods for measuring SA are discussed. Measuring SA is not a straightforward task and this has led to the emergence of several taxonomies of SA measurements. Although minor differences exist in these taxonomies, all their emphasis is on the measurement methods.

2.1 Situation Awareness

Situation awareness (SA) was first recognized in solving problems for crews in military aircraft during the World War I [142]. In the mid-1970s the US military ergonomists started investigating the factors affecting aircrew, and from then onwards, SA became an established concept [179]. The concept was later adopted by human factors researchers for studies of complex environments [54]. The use of predictive displays which enhance situation awareness of air traffic controllers made a significant improvement in air traffic control performance [63]. For the military, the focus for situation awareness is not so much on reducing errors but on obtaining the strategic advantage in the battle field. Although earlier studies on situation awareness were linked to aviation [61], the interest in situation awareness rapidly expanded into other complex and constantly changing domains.

Recently, related research activities look beyond the aviation and military domains to investigate situation awareness in a large variety of domains. The generalization of situation awareness is found in the work of Gaba et al. [68]. They described an analogy between the situation awareness requirements in aviation and anaesthesiology. Both domains imply complexity, dynamism, variable workload, high information load and risk. The availability of realistic simulators where surgical problems were replicated showed that situation awareness could be formed with the same method in both domains. Similarly, Pott et al. [140] built a decision support system that will use SA to improve the performance of anaesthetists in operating theatres in two different contexts: familiar and diagnosing. In the diagnosing context, the system generates a list of reasonable diagnoses and their history. The anaesthetist selects from the recommendation a diagnosis and its history to facilitate decision making. Decision making in a familiar context induces low cognitive workload of the anaesthetist and anaesthetists as experts might perform well without any decision support. But in familiar situations, errors in decision making can occur due to low vigilance or exhaustion from high workload.

Hoogendoorn et al. [80] modelled the effect of exhaustion on situation awareness. The approach used a fighter pilot training exercise to form two levels of beliefs, simple and complex beliefs [81]. A simple belief is a basic understanding about the current situation. Complex beliefs are the aggregation of multiple simple beliefs which are used to anticipate future situations. The higher the attention on a specific task, the higher the value of the beliefs that are important for the task. In contrast, if there is less attention to a specific belief as a result of exhaustion, that belief gradually becomes less active. The work also gave an insight into the effect of exhaustion due to work load on attention.

The problem of attention can be solved in situation awareness by decomposing tasks into subtasks. Attention can then be focused on important subtasks. Drury et al. [51] developed a method of decomposing situation awareness in order to provide an insight into the types of awareness that needs greater attention. The approach proposed the use of awareness decomposition as both a means of understanding specific needs of operators and as a tool to help evaluate whether those needs received attention. These needs include the location of an entity at every point in time. The certainty of a location in situation awareness could be a great challenge which Drury et al do not elaborate on. Ma et al. [107] proposed an approach to probabilistically model and represent potentially uncertain event locations described by human reporters in the form of free text. The approach modelled uncertain event locations as random variables that have certain probability density functions associated with them. The technique mapped free text onto the corresponding function defined over the domain with the assumption that people report event locations based on landmarks. The framework can be applied to spatial uncertainty but cannot handle temporal uncertainty.

Earlier findings in the field of nuclear power plant process control shows that situation awareness was less fruitful [198]. The approach was at odds with the key research that is now considered as the basis of the contemporary conceptualization of situation awareness. Woods et al. [198] insisted that situation awareness could be applied with benefit to this specific domain. Another attempt to study team situation awareness during the normal operations of a nuclear power plant also failed because all operations were perfectly executed [9]. The investigated situation offered no opportunity to make mistakes, and then few opportunities to observe fluctuations in situation awareness.

Gugerty and Tirre [71] studied situation awareness in car driving with almost the same techniques as those used in aviation. The assumption was that driving a car, with exception of speed, might not be fundamentally different from piloting an airplane. Their results showed that situation awareness methodology could be applied, in the same way, to tasks of every day life as it has been to tasks of human maximal performance. Jenner et al. [83] linked a lack of situation awareness to a variety of accidents investigated by the National Transportation Safety Board in the railroad, the marine, the pipeline industry, and the aviation. According to Molloy [119], surface transportation has the same automation pitfall aviation had to overcome several years ago. Overconfidence in the technology and a consequent loss of situation awareness is observed in the investigated accidents. Klein [92] introduced methods of investigating situation awareness that are similarly applicable to the analysis of errors in both aviation and the neonatal diagnosis of an extremely dangerous systemic infection in newborn babies. Thus, situation awareness appears to be a concept that can be applied in a very large number of domains.

2.2 Defining Situation Awareness

When experts speak about a general phenomenon called situation awareness, most discussions are reasonably consensual. In practice, there is a long list of concrete examples to persuade someone that situation awareness has its own reality and importance. When the same experts attempt to define this expression in words, the results shows contrasting views. To some, situation awareness is a "process" of integrating and interpreting environmental elements while to some others, SA is a "state" of knowledge of the environment [62]. At times, the definition is considered as imprecise, impossible to measure, circular, or too much bound to the characteristics of a particular situations. In other cases, it is too general and cannot be differentiated from other related concepts.

As Pew [139] pointed out, SA shares a common history with several psychological concepts such as intelligence, fatigue, vigilance attention, compatibility, stress, or workload. During decades, all these terms were poorly defined. However, each one became important because it brought attention on the critical processes or mental states that were previously unknown. Ultimately, they changed the ways to study human factor problems, and brought new benefits.

In conclusion to the persistent efforts to define situation awareness, Sarter and Woods [165](p.16) proposed that: "the term situation awareness should be viewed just as a label for a variety of cognitive processing activities that are critical to dynamic, event-driven, and multitask fields of practice". Such a point of view enables applied work on situation awareness to proceed, but in the long run, can be detrimental to the field and foremost to the development of general situation awareness measurement tools. It is important to address the issue of the definition of the object, Situation Awareness. The acceptance of a precise and universal definition of situation awareness would bring considerable advantages to the field. The various attempts at defining situation awareness in the existing literature will be closely looked at.

2.3 Situation Awareness Definitions

The three-level definition of situation awareness proposed by Endsley [54] has been adopted by many researchers. Given the importance of the objective, however, a more systematic analysis of this and other definitions is of interest. In order to extract the basis for a general definition of situation awareness, a useful approach is to start from the essential elements involved in situation awareness by considering two objects, the "Situation" and the "Person". The situation can be defined in term of events, objects, system, other persons and their mutual interactions. The person can be defined according to the cognitive processes involved in situation awareness or simply by a mental or internal state representing the situation.

2.3.1 Defining Situation Awareness from "the person" Perspective

From the person perspective, a given definition may be process-oriented, focusing on the link between the situation and the cognitive processes generating situation awareness. For example, Dominguez defined situation awareness as the "continuous extraction of environmental information, integration of this information with previous knowledge to form a coherent mental picture, and the use of that picture in directing future perception and anticipating future events" [50]. Her definition presents a set of four processes or functions on which SA depends: Information Extraction, Information Integration, Mental Picture Formation, Projection and Anticipation. Other definitions are state-oriented, focusing on the link between the situation and an internal representation of elements present in the situation. Adam [4] provides a clear example of a state-oriented definition that defines situation awareness as "knowing what is going on so I can figure out what to do". State-oriented definitions limit the description of processes involved in situation awareness. In fact, that distinction between the term "process" and "state" in situation awareness is similar to the opposition between the notion of direct and indirect perception as a basic mental process. Derived from the work of Gibson [69] direct perception is based on the principle that: 1. All the information necessary for perception is contained in the environment. 2. Perception is immediate and spontaneous.

It means that, in order to understand the process of perception, the understanding of the environment must take top priority. It also means that, there is no need to develop theories of perception based on inferred mental mechanisms of information processing from which perception would result. On the contrary, an information processing approach recognises that a mental representation of the world is based on processing with specific functions [46]. That approach requires an explicit description of the processes involved in providing humans with cognition.

It is important to distinguish between "Process" and "State". One of the difficulties in working with situation awareness is to avoid confusion between SA knowledge and the underlying cognitive processes such as perception, memory, attention, categorization, or decision-making. This difficulty becomes an issue to resolve particularly when situation awareness has to be measured. Tenney et al. [184] and Endsley [54] limited the term "situation awareness" to the achieved knowledge (state) about a situation. Endsley proposed the expression "situation assessment" to represent the cognitive processes that produce the knowledge (state).

Defining situation awareness as a state of relevant knowledge of which a person is aware is not without problems. Smith and Hancock [179] stated that situation awareness definition will encounter similar arguments to those going on in the study of "introspection" if it is not defined with regard to an external goal. Introspection describes operators' process of being attentively conscious of mental states that they are currently in. This consciousness of the operator's concurrent mental states is different from the relatively casual, disuse, and fleeting way humans are ordinarily conscious in many of our mental states [155]. Approaches based on introspection rely on a verbal report of mental states and so, it is accepted that introspection is the result of mental states and not a mere reflection of their current status [24]. It is well established that introspection is the result of mental states that operate on a more automatic cognitive mode like implicit memory or skilled performance on which expert performance is often based. Situation awareness cannot simply be equated to any verbal report of the content of consciousness about a situation. According to these authors, to equate situation awareness with momentary knowledge and mental models is to run the risk of allowing situation awareness to degenerate rapidly into "whatever is inside your (skilled) head" [155]. These authors also stated "...to comprehend SA without a viable understanding of the interaction between agents and their task environment would be virtually impossible" [179](p.140). Such comments stress the importance of considering situation awareness as a specific mental representation.

On the issue of situation awareness definition, one is faced with a double problem. On the one hand, if situation awareness is a state, it is essential to give a precise definition of the knowledge that defines the state. There should be some mapping between a situation schema and a knowledge schema. If one is to improve situation awareness, the situation awareness content definition should follow from these elements. On the other hand, if situation awareness depends on a set of processes that are not an intrinsic part of situation awareness as a state but on which situation awareness depends, it becomes important to specify which processes are essential to SA [24]. Situation awareness improvement, for instance, will depend upon changes in the operation of these processes.

2.3.2 Defining Situation Awareness from "the situation" Perspective

From the situation perspective, the definition can be classified as being "general" or "specific". A specific definition describes the situation in detail and precisely in terms of the objects, actions and variables related to the task performed. Prince et al. [144] gave an example of a specific definition to situation awareness as: "the ability to maintain an accurate perception of the surrounding environment, both internal and external to the aircraft as well as to identify problems and/or potential problems, recognize a need for action, note deviations in the mission, and maintain awareness of tasks performed". On the other hand, a general definition will refer to the situation in abstract, non-specific terms. A widely accepted general definition is given by Endsley [54] who defined situation awareness as “the perception of the 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.”

It is important to separate the general from the specific definitions. Perhaps a major part of the definition problem would be solved if the definitions were tightly bound to the sole situation or domains in which the studied process or mental state has a direct impact. That is to say, situation awareness would only be considered in a particular set of situation conditions and specific definitions could be proposed.

Adams et al.’s [5] opinion is that situation awareness is not always important and that SA is often needed in crisis situations. Going by Adams’ proposition, one may ask the question: what makes a situation a crisis if not the characteristics of a situation itself? On the one hand, restricting situation awareness to crisis situations would leave SA with an efficient specific operational definition but, on the other hand, this definition would be inadequate in terms of general properties [24].

Of course, if a new specific definition was required for every situation, effort to provide a general definition of situation awareness would be senseless. Gaba et al. [68] argued that situation awareness is as critical in anaesthesiology as it is in aviation since both domains include dynamism, complexity, high information load, variable workload, and risk. Common domain characteristics should be looked for. One way to solve the problem is to consider the situation part of the definition as being based on the generic properties of a situation within a class of situations. The situation elements, while lacking in some detail, would remain the same for all situations belonging to a given class [24]. A situation, according to Pew [139](p.34) is “a set of environmental conditions and system states with which the participant is interacting that can be characterised uniquely by a set of information, knowledge, and response options.”

Looking at the general and specific definitions, one can say that the general definition proposes constraints to what can be included in a specific definition of SA. That means a specific domain definition of SA must reflect on key elements of the general definition such as, perception of cues from the environment, comprehension of the meaning of the cues, projection of the status of the environment in the near future. The definition must identify which things (cues) the operator needs to perceive and understand. These are specific to individual systems and contexts, and as such are the one part of situation awareness that cannot be described in any valid way across domains. Also, spatial information cannot be described across domains. Space is highly useful for determining exactly which aspects of the environment are important for situation awareness. A person's situation awareness needs to incorporate information on that subset of the domain that is relevant to tasks and goals. Within this boundary, the cues or elements may be further subdivided into levels of importance for situation awareness or may assume a relevance continuum, depending on the problem context [54]. A specific definition should also recognise the fact that a person's situation awareness is highly temporal in nature, that is, it is not necessarily acquired instantaneously but is built up over time. Thus it takes into account the dynamics of the situation that are acquirable only over time. In that view, “situation” represents a bigger picture. It includes task and mission features, as well as the other human agents in the domain.

The classification of the situation perspective is seen to refer to what one is or should be aware of. A definition is classified as “specific” if it refers to the situation in terms of the actual operating domain or factors specific to that domain. It is classified as “general” if the definition does not include such particular factors and simply refers to the concept of situation [24].

The question remains as to whether a definition of situation awareness should be limited to a content or should include the processes or functions linked to the awareness of the situation [155]. Should situation awareness include or not what some authors refer to as situation assessment? It is not possible to provide a clear answer to that question from the strict analysis of situation awareness definitions but a point to note is that SA is not only produced by the processes of situation assessment, it also drives those same processes in a recurrent fashion. It is argued here that situation awareness should include situation assessment, since one's current awareness can determine what one pays attention to next and how one interprets the information perceived [62]. However, many authors have expanded their definition of situation awareness by developing SA models. In fact, the complexity of defining the cognitive side of situation awareness has led a number of authors to develop models of SA that are complex enough to make possible an explicit presentation and definition of the cognitive side of situation awareness.

2.4 Endsley's Model of Situation Awareness

Models of situation awareness are presented as models of how people achieve situation awareness in complex domains. Some of the models are not necessarily formal situation awareness models but, most often, they are descriptions of the status of situation awareness in a general model of cognitive processing, taking into account non-cognitive factors affecting the development of SA. Amongst these models, Endsley's model [54] clearly stands as the reference for most work performed on situation awareness. A number of other models focus on a specific aspect of situation awareness but remain within the constraints of Endsley's model. For instance, Maggart et al. [108] describe situation awareness in the context of infantry operations. They explicitly rely on Endsley's model [54] as a basis for describing the specific elements of situation awareness in that context while focusing on the domain in which infantrymen operate. Similarly, Bolstad et al. [21] and Salas et al. [161] address the important issue of team or shared situation awareness. They do so from the point of view of Endsley's model. Likewise, McGuiness et al. [115] base the development of a situation awareness measure on Endsley's model, while proposing some modifications to the original model [54].

Endsley's SA model (Figure 2.1) has two main parts: the core situation awareness model and the various sets of factors affecting situation awareness. The first part is called core SA model since it represents the processes directly responsible for situation awareness. Endsley's core SA model presents SA as a three-level mental representation: perception, comprehension, and projection. The second and much more elaborate part describes in detail the various factors affecting SA grouped into four broad classed: external world, individual factors, task and environmental factors, and a set of domain factors. These factors include contributions of all components of current human information processing models like goals, active schemas, past experience, attentional processes and working memory.

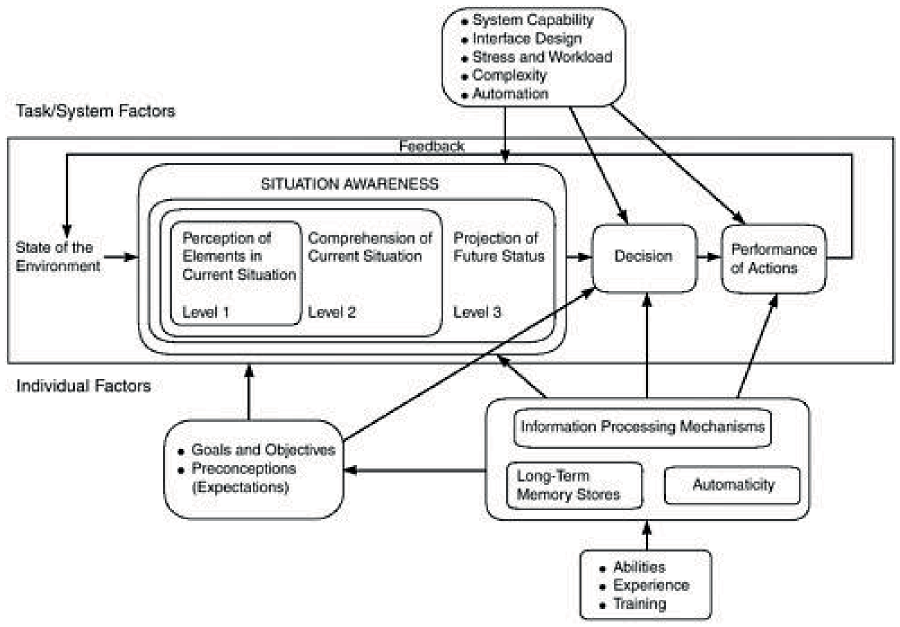


Figure 2.1: The Endsley (1995) Situation awareness model

Endsley states that first, an individual's ability to acquire and maintain situation awareness is a function of his/her cognitive abilities, which in turn is influenced by his/her innate abilities, experience and training. In addition, the individual may have some preconceptions and objectives that may influence his/her perception and interpretation of the situation. Endsley further states that situation awareness is also a function of the design of the system, both in terms of the degree to which the system provides the requisite information, and the format in which this information is provided. Finally, other features of the task domain, such as stress, workload, and system complexity may also affect situation awareness. Endsley's SA model also utilises a feedback method to direct behaviour in order to attain a desired situation awareness. However, Endsley makes it clear that actions and behaviour are separate stages that proceed from SA and that SA is recognised as a concept separate from decision making and performance.

Since the model of SA presented in Endsley [54] is the basis for much of the current modeling of core SA, the three levels of SA from that model will be briefly described.

2.4.1 Level 1: Perception of Elements of the Current Situation

To achieve situation awareness (SA), the first step is to perceive the status, attributes, and dynamics of relevant elements in the domain. It provides information about the status of the relevant elements in the environment. A military commander for instance, needs knowledge on the identity, location, capabilities, number and dynamics of the enemy in a particular area and their relationship to other reference points. A car driver needs knowledge of where other cars and traffic or obstacles are, their dynamics, the status and dynamics of his own car. A sub-sea engineer in a oil and gas industry needs to know the sea bed temperature, the wellhead temperature, the wellhead pressure, the gas composition, and the dynamics in the sub-sea flow lines to know if and when hydrate will be formed. The identification and definition of the elements to be perceived is the prerequisite for understanding of SA in a given domain. The things to be perceived and understood are specific to individual domain and contexts, and as such cannot be generally described across systems [54]. The choice of things to perceive and the way to perceive is directed by an individual's working and long-term memories. Advanced knowledge of the gas composition, identity, distance, and location of information, for instance, can significantly facilitate the perception of information in a hydrate monitoring domain. It therefore follows that, an operator's expectations about information will affect the accuracy and speed of his perception [59]. The development of expectations about situations are based on prior experiences in the domain.

Perception includes classification of information into understood representations. Long-term-memory stores contain knowledge that enables mental representations of the elements. Perceived elements are a subset of elements present in the environment. The subset is under attentional selection based on task requirements. The elements are structured into meaningful events located in time and space. The events are classified into known categories or mental representations through an individual's long-term memory [?]. The classification is a function of the individual's knowledge of the domain and will produce the elements of Level l of situation awareness. Classification provides the basis for the other two Levels (2 and 3) of Situation awareness. A good and well-developed memory will result in good classifications. For instance, an experienced sub-sea engineer will be able to use his context and knowledge of the sub sea environment to classify observed state of the flow lines e.g. Normal, Warning, and Danger [128].

2.4.2 Level 2: Comprehension of the Current Situation

Comprehension of the situation is the second Level (Level 2) and is based on the integration of the perceived attributes of Level 1. It is the synthesis of Level 1 disjointed elements. Comprehension is not simply being aware of the elements that are present, but includes an understanding of the meanings of those elements with respect to an individual's goals [54]. It provides an organized picture of the elements with a comprehension of the significance of objects and events. Schemata or mental models stored in long-term memory are the basis for Level 2 situation awareness. Mental models are complex schemata representing a given system. Level 2 situation awareness is then defined as a situational model depicting the current state of the mental model. The individual forms a holistic picture of the domain, comprehending the significance of objects by integrating knowledge of Level 1 elements with the other elements. Information processing is carried out in the working memory by combining new information with existing knowledge to form a composite picture of the situation. To reduce the load on the working memory humans naturally find understanding of situations by recalling similar past situations using long-term memory [94].

2.4.3 Level 3: Projection of Future Status

The ability to anticipate the future state of the domain constitutes the third and highest level of situation awareness. Projection is achieved through knowledge of the status and dynamics of the elements (Level 1) and comprehension of the situation (Level 2). The mental model provides means to go from an understood situation to the generation of probable scenarios as to the possible future states of the system. For example, knowing that the wellhead pressure of a certain flow line is high and the temperature has dropped allows a sub-sea engineer to project that hydrate is likely to be formed. Level 3 SA provides the knowledge necessary to decide on the most favourable course of action to meet one's objectives.

2.5 Functional Analysis of Situation Awareness

Endsley's model [54] describes the three Levels of situation awareness as a sequence of processes enabling situation awareness to operate in a linear hierarchical fashion. In that sequence, Level 1 is defined as perception of the environment that provides mental representation of the elements available in the environment. Perception feeds into Level 2, comprehension, where it is given a meaning. The linearity of situation awareness development from one level to the other can be questioned on theoretical grounds from current scientific knowledge about both the perception and comprehension processes. McGuiness et al. [115] argued that these levels should not be interpreted as a linear or hierarchical sequence but rather a network of parallel functions serving a common purpose.

In experimental psychology, there is a huge amount of concepts and phenomena contained in the field of perception. Useful scientific basis to address what is meant by perception in Endsley's model can be provided by the work on high-level vision. We can use high-level vision as a model of what Level 1 is supposed to generate. High-level vision maps visual representations to meaning and includes the understanding of processes and representations related to the interaction of cognition and perception, including the acquisition of information, the identification of objects and scenes, and short-term memory for visual information. Henderson et al. [75] review the current expectations from scene knowledge on scene identification and identification of objects in scenes. The perceptual schema model [18] is identified as a prevalent view about scene identification. It proposes that expectations derived from knowledge about the composition of a scene interact with the perceptual analysis of the elements present in the scene. The schema of a scene, typical of Level 2 situation awareness, would contain information about the elements and their spatial relations. The early activation of a schema, in processing information from the environment, would facilitate the processing of schema-consistent elements. Thus, the cognitive treatment typical of Level 2 situation awareness would affect retroactively the perceptual treatment in Level 1 SA [24].

Klein [92] holds a similar view when he argues that, given the large number of elements present in the environment, Level 1 situation awareness will rely on a subset of relevant elements. Relevance, according to Klein, stems from Level 2 situation awareness, and is the result of an understanding of a situation. Level 2 situation awareness is then assumed to play an active role in defining Level 1 situation awareness through the control of a selection process operating on the incoming input from the environment. Smith et al. [179] had made a similar proposition by stating that situation awareness (Level 2 SA) is not merely a snapshot of the domain and that it guides the building of a mental representation of external objects and events. These remarks explains why the graphic representation, showing Level 1 output feeding into Level 2 in the Endsley's model (Figure 2.1), should not be interpreted as describing a linear hierarchical system.

Adams et al. [5] propose an approach to situation awareness modeling which is different from Endsley's model. They base their model on the proposition that situation awareness is a state and a process. From that point of view, a model of situation awareness must include a description of the processes required for producing and updating the representation and understanding of the current state of the environment. They develop their model from Neisser's perception cycle [125], which is a very simple system developed for very general considerations about active cognition. Neisser's model [92] includes three basic processes: perception, memory schema, and active exploration (action). An essential element of that approach is that perception is controlled, in part, by the action process. Adams et al. [5] insist on the influence of the cognitive schema on the environment. However, for Adams, situation awareness is limited to the state of the active memory schema, the other two processes being the active processes that determine situation awareness at a given point in time. That approach makes it possible to define a linear but less hierarchical model since the perception cycle is a closed network in which each process can be seen as the beginning of the cycle. Given that constraint, the model is compatible with Endsley's SA model since it associates situation awareness proper with the memory schema process (Level 2).

In a similar view, Klein et al. [92], as presented earlier, proposes that situation awareness development is an active process of guided information seeking, rather that a passive receipt and storage of information. He points out that situation awareness is not only determined by the situation but also by what the operator is doing. Klein suggests that efforts should be deployed more to understand how situation awareness affects decision-making rather than to identify what is the content of situation awareness. These remarks from Klein et al. [92] are linked to the application of situation awareness in the context of his Recognition-Primed Decision (RPD) model. Following the RPD model, Klein attracts attention to the fact that individuals may function at different cognitive levels where situation awareness may have a more or less critical role to play. In fact his RPD model stresses the question of situation awareness being often considered as a conscious mental state. The recognition heuristic, central to Klein's model, enables efficient performance in a decision-making task without complete awareness of all the cognitive factors on which it is based. That possibility is interesting since it will define constraints to situation awareness measurement methods depending on the level of consciousness of the mental processes and knowledge controlling behavior in a given part of a task.

2.6 Situation Awareness Content

The two basic content architectures for SA, situation models and mental models (Figure 2.2), are considered in this section.

2.6.1 SA and Situation Models

Endsley's model is an approach to address the most basic issue of modeling situation awareness as a representational state. Another interesting approach comes from Shebilske et al. [173]. They stress the importance of situation awareness modeling to considering other models outside the concept of situation awareness. These models provide an enrichment of current situation awareness models. According to Shebilske et al. [173], Endsley's model [58] referring to situation awareness as a situation model has been influenced by Dijk et al.'s 1983 model of discourse comprehension [49]. Dijk et al. [49] presented a distinction between text representation and situational representation. Text representations are in terms of text structure and components of text information. Situation models produce text comprehension by connecting text representation to experience knowledge stored in episodic memory. Episodic memory is a component of human memory in which prior personal experience, located in a time-space frame, are stored. Accordingly, Dijk et al. [49] describes situation models with four components: reference, situation parameters, coherence, and perspective. Using such a model, that is already fully developed in other domains as a basis for situation awareness, provides a useful basis for SA modeling [24].

Another interesting development in discourse comprehension further contributes to situation awareness modeling. Dijk et al. [49] proposes that context models have to be added to situation models. He renamed situation models as event models and presents a structure schema of context models including domain information and situation information. The situation information contains time, location, participants' roles, action and cognition parameters. Context models are a form of experienced mental models and are in fact memories of past occurrences, whereas events or situation models include personal knowledge. Looking at these models, Level 2 and 3 SA includes the personal and experience information stored in episodic memory. This does not appear to have the same meaning as Endsley's interpretation who stated that situation awareness is a situation model [58]. Endsley defined situation model as "a schema depicting the current state of the mental model of the system " [54] (p. 43). Hendy [76] made a similar proposition without referring to situation models, based on a definition of mental models as the set of perceptual and goal representation, which induces behaviours such that the distance between the environment and the mental model is reduced. He states that:

"The term Situation Awareness (SA) particularly relates to that dynamic and transient state of mental model, which is produced by an ongoing process of information of some work. While the concept can be generalized to all tasks, no matter what their complexity, the term SA is usually used when considering tasks that have strategic and tactical components such as flying an aircraft, controlling or monitoring a plant, or tactical decision-making".

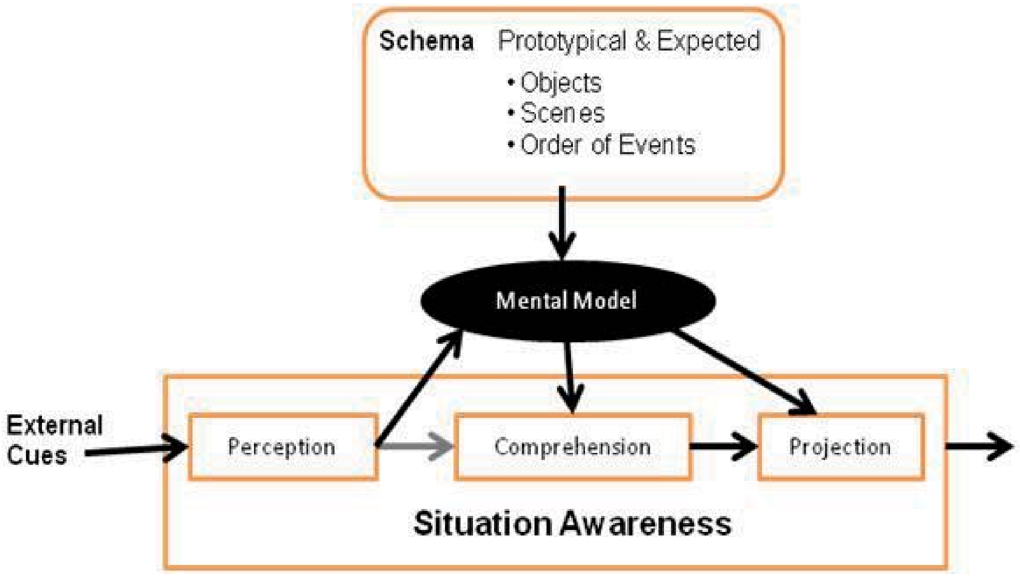


Figure 2.2: Relationship between situation and mental models(Endsley, 2000)

Thus, it could be appropriate to consider situation awareness as being the current state of the mental model applied to a task. However, mental models and situation models are not identical. While Endsley's and Hendy's positions could be considered compatible, for that to be the case, it would require situation models to correspond formally to mental models. According to Breton [24], if SA is considered a form of situation model, important benefits could be gained by applying to SA the developments of situation models made in other fields. For instance, SA requirements would have to include personal elements like past memories, role and experience.

2.6.2 SA and Mental Models

Situation awareness can be viewed as a mental representation of the world enabling a comprehension and projection of future states in the terms of Endsley's model. But the concept of mental models has a very long tradition in applied cognition. It has often been used in studies trying to model, amongst others, human control of various processes. In Cognitive Psychology and Cognitive Science mental models have relatively clear meanings. However, the HCI community tends to use the concept very loosely and this is not helpful. Markman et al. [111] define mental models as a representation of some domain or situation that supports understanding, reasoning, and prediction. Rouse and Morris [159] (p.7) defined mental models as “mechanisms whereby humans are able to generate descriptions of system purpose and form, explanations of system functioning and observed system states, and predictions of future states”.

For these authors, mental models are multi-purpose mental devices. The three basic functions: (1) Description of system and form (2) Explanation of system functioning and observed system states and (3) Predictions of future system state, are all compatible with the three-level situation awareness model proposed by Endsley [54]. However, for Rouse and Morris [159], mental models are not a state but are sets of processes. It appears to be difficult to distinguish state from process in situation awareness, when it is viewed as a form of mental model. Given that point of view, Rouse and Morris [159] argue that the forms of mental models are dependent on the type of task and the input-output relations in which operators are involved. Different tasks were distributed in a two-dimensional domain space with one axis as the level of behavoural discretion and the other axis as the implicit/explicit nature of model manipulation. The implicit/explicit dimension represents whether or not a person is aware of his manipulation of a mental model. Level of behavioural discretion represents the extent that a person’s behaviour is influenced by his choice as opposed to dictated by the task. Rouse and Morris [159] finally recommended inferential methods like formal modeling when there is less discretion, and verbalization methods, like interviews or verbal protocols, when there is explicit manipulation. McGuiness et al. [115] and Smith et al. [179] also stress the importance of considering the distinction between explicit and implicit information or knowledge. McGuiness et al. [115] propose that comprehension (Level 2 situation awareness) is based on implicit information.

Rasmussen et al. [147] describe the model of cognitive control of human activities in terms of a three-level cognitive control system: Skill-based, Rule-based and Knowledge-based. Skilled-based control depends on a tacit dynamic world model structured on the operational space and activated by direct perception of relevant spatio-temporal features of objects in the environment. At the Rule-based level, control is linked to conscious preparation of plans for action and organization and sequence of actions based on rules governing the situation or imagined future encounters. Finally, the Knowledge-based level relies on a conscious symbolic mental model, representing the deep structure of the work environment, forming the reference on which information is interpreted. According to Rasmussen et al. [147], mental models refer, in general, to the features of the cognitive representation that represents the properties of the task environment to support and control of action. Given such a definition, mental models could develop at all levels of information processing. In the context of the skill, rules and knowledge (SRK) model, they propose to limit the definition of mental models as being declarative representations of the relational network that conceptualizes the invariant structure of the environment and the constraints governing the regularity of its behaviour. They argue that mental models operate at the knowledge level and that, if situation awareness were a mental model it would be restricted to a declarative and explicit knowledge of a system. Endsley [54] referred to implicit representations of these constraints in terms of procedural, episodic, and tacit knowledge. These representations provide a context for some form of judgment and contributes to situation awareness in the form of references to prior experience. The approach presented mental models as "default information that helps form higher levels of SA even when needed data is missing or incomplete" [59]. For example, an engineer may perceive several dynamics in the flow lines (considered to be important elements per the mental model) recognized as hydrate forming conditions based on critical cues (perception). By pattern-matching to prototypes in memory, these separate pieces of information may be classified as a particular recognized hydrate formation (comprehension). According to an internally held mental model, the engineer is able to generate probable scenarios for this type of condition (projection). Based on this high-level SA, the engineer is then able to select suitable actions that will prevent the formation. To use mental model to achieve SA is based on the individual's ability to recognize relevant features in the domain cues that will map to relevant features in the model [28]. The advantage of the approach is that the current situation does not necessarily have to be exactly like the previous one [84].

From a different perspective, Moray [120] considers that mental models operate at all levels of information processing and not only at the knowledge level. By their argument, situation awareness is a more varied concept and its measurement should have to take into account the level at which an operator is operating in a given task. Moray [120] provides an interesting taxonomy of mental models that sheds some light on the contribution of the concept to situation awareness modeling. His main point is that the structure of mental models varies according to the dynamics and complexity of the system being modelled. A highly complex system, like a oil and gas processing plant, will be represented by multiple and often embedded mental models. The nature of mental models is influenced by the strength of the physical coupling between the operator and the task. Moray proposes that a model of the environment might simply be related to perception but that in a more proper definition, it will refer to a content in long-term memory. Furthermore he claims that there are many mental models of a system going from a mental model of the physical function of the system to models of general functions and goal, mean, ends of a system [24].

2.7 Goals in Cognition

Situation awareness is fundamentally linked with an individual or team goals which are integrally linked with both the context and the decisions for which the situation awareness is being sought. Information is considered to be part of situation awareness if it is required to achieve a goal [24]. An operator seeks data in light of his goal and may have more than one goal at a time. In what Jones et al. [86] termed a top-down decision process, an operator's goals and plans direct which aspects of the domain are attended to in the formation of situation awareness. The information is then integrated and interpreted in light of these goals to form the second level (comprehension) of situation awareness. The information becomes meaningful when integrated in view of what they indicate about the goal of operating the system.

Along with the top-down goal processing there is bottom-up environmental cues processing. Operators have goals that are considered to be ideal states of the system that they wish to achieve. The same set of goals may exist frequently or may change often for a given system. In the course of integrated processing of goal and environmental elements, features in the domain that will require new plans to be activated to meet active goals may be recognized [86]. A response to recognised features may lead to a change in current goals and plans. Similar to goal is the notion of context.

2.8 Context

A context is a descriptor or a set of descriptors that defines a situation or a scenario [96]. A scenario is defined as "the world state", a situation that is a snapshot or an instance of the world at some given time, namely, all attributes of the world, including all objects, their properties and internal states, and the relationships between them [191]. Context, like goal, acts as a filter to situation awareness. Context filters situation awareness in relation to the specific need of individual operators. Context contains a user's current goal, selected as the most important among competing goals, which acts to direct the selection of a mental model. A context can convey a different facet, a different point of view, or a different understanding of a situation. Situations are characterized by several contexts [172] and it is common knowledge that contexts are of immense importance in human psychological reasoning in different situations.

2.8.1 Context Awareness

A related concept to situation awareness is the notion of context awareness. Dey [47] defines context as "any information that can be used to characterize the situation of an entity". A system is said to be context aware if it uses context to provide relevant information and services to the user [148]. Context awareness was introduced by Schilit [167] to develop an application that adapts to the location of use, nearby people and objects, and the change of those objects over time. With technology advancement and the rapid growth of mobile computing in recent times, context awareness has attracted greater research attention [64].

A context-aware application is characterised by its presentation of information to a user [93], automatic execution of a service, and tagging of context to information for later retrieval [148]. Context can be useful in both static and dynamic domains. From a static representational point of view, Raz et al. [148] used context to provide access control that grants permission to authorise users in a team environment where information was shared and accessed by multiple users. The context-aware system from Raz's model senses and reacts to the user based on his context. Similarly, Choi [37] used context to produce some access control algorithms to design an adaptive access control system to grant access to roles, carry out role delegation, role revocation, permission modification, and permission restoration. The role-based access control (RBAC) policy which provides access permissions to roles rather than users was favoured in their design. Teo [185] presented context as a dynamic construct arising from interactions. Teo proposed a scheme where contexts are determined by the activities of users dynamically by varying attributes in different locations and on different activities. The activity-driven model, provided clarity to dynamically identify context. The aim of context identification in the approach is to extract relevant information associated with the context. Gupta and Mukherjee [72] combined clustering and case-based reasoning (CBR) in extracting relevant knowledge in a context-aware application. Data from multiple sensors are clustered based on attributes such as type, size, and purpose. The cluster attributes serves as the problem part for situation awareness using CBR to extract similar past cases that matches the size, type, and purpose of the cluster.

The term context awareness and situation awareness are used interchangeably by some authors as if the two concepts mean the same thing. For example, Kofod-Petersen et al. [95] and [96] represented all the parameters for the problem description, both static and dynamic, as context at the perception (first) layer in their three layered architecture comparable to the Endsley SA model. With this single architecture, context awareness (CA) and situation awareness (SA) were discussed synonymously.

To avoid the seeming confusion over the use of these two concepts, it is important that we stick strictly to the generally accepted definition [54] of situation awareness, which focuses on the modeling on a user's environment. Context awareness allows systems to dynamically adapt to changes in a user's task domain, by updating relevant information and service provision, whereas situation awareness focuses on information about the state of the environment in which these tasks are carried out [64]. Feng et al work in the domain of military command and control provided a clearer distinction between these two concepts in a system that used context-aware domain knowledge to provide customized decision support to users through agents which extract information from the situation awareness model to the users in accordance with user's contexts. However, the scope of the decision support of Feng's approach was not extended to situations of uncertainties and partial knowledge due to its purely rule-based nature. Machine learning techniques have been shown to be useful for dealing with uncertain knowledge [195]. For instance, case-based reasoning (CBR) is effective where the general domain knowledge is difficult to extract and instead requires reasoning based on local knowledge or where it is difficult to formulate rules describing the situations [72].

2.8.2 Context-Aware Case-Based Reasoning

Case-based reasoning (CBR) is one of the most effective paradigms of knowledge-based systems [103]. CBR helps in situations of incomplete domain data [151]. CBR draws from experiences of past cases in order to solve new problems. The CBR paradigm is consistent with human natural problem solving methods of using a previous solution that was successful for a problem in the past to solve a similar new problem [162].

The user in case-based reasoning queries the database when trying to solve a new problem (Figure 2.3). The system searches for similar past solutions by matching and comparing the current problem to old problems. Previous solutions are retrieved based on a correspondence of the new problem to some past problems. The system retrieves a set of similar cases and then evaluates the similarity between each case and the query. The most similar case(s) retrieved are presented to the user as possible scenarios for the current problem. If the solution retrieved is applicable to the problem, the user reuses the solution, and if it cannot be reused, the solution is adapted manually or automatically. When the validity of the solution has been determined, the user retains it with the new problem as a new case in the database for future use. At this point, the case is considered to have been learnt [73]. The focus the process is to arrive at a conclusion that fulfills to the goal or context of a new problem. Contexts are not cases per se but are transformed into cases

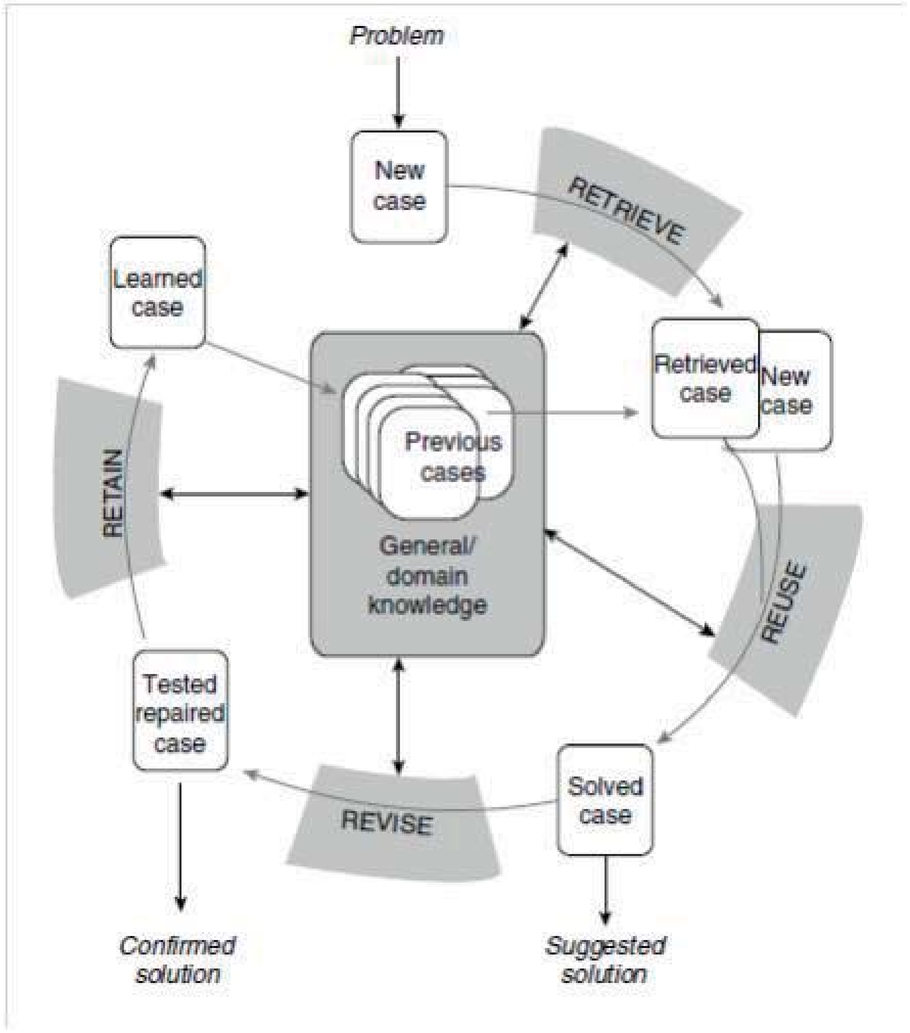


Figure 2.3: CBR cycle (Aamodt and Plaza, 1994)

or can be used to identify cases. Zimmermann [200] used contexts just like cases in a case-based reasoning system in a mobile scenario. The user context was enclosed in cases to facilitate comparison of contexts, and provide solutions based on context-similarities.

Vacek et al. [192] used case-based reasoning in a computational model of situation awareness for autonomous driving. CBR was used to interpret the current situation and selecting the appropriate behaviour. Future situation behaviours were known by their projected consequences using the expectation value. The expectation value is calculated for each applicable behaviour and the behaviour with the highest overall value is selected as the behaviour of the future situation. Ting et al. [186] also applied features of expectations during the projection stage in work on using CBR to build a computational SA model for military operation in urban terrain (MOUT) simulations. The approach uses violation of expectations to determine behaviours or actions. Violation of expectation in the approach is categorised as invariant and variant. Invariant expectations must be fulfilled or else there is danger whereas the violation of variant expectations is merely an alert of possible threats. The system of both Vacek et al. [192] and Ting et al. [186] rely only on cues from the environment without considering the user's context.

Unlike CBR systems, rule-based systems require a careful procedure in order to ensure the consistency of the system. A set of rules is worked out in order to understand the situation. Background knowledge is given implicitly in the rules and the order of the rules. Rule-based systems are not able to work with experiences [72] and rules are typically created by a limited number of experts. Their knowledge and ignorance are implicitly reflected in the rules [86]. Unlike experience-based systems, the best way to explain a decision in rule-based systems is to report the chain of inferences. Case-based systems have several advantages compared to classical rule-based systems. They facilitate better maintainability and expandability than rule based systems [162] since new knowledge is added by integrating new cases automatically to the case-base. Partial matching is another advantage of case-based systems. Even if a case does not match exactly, it can still be considered for problem solving [151]. Cases are represented in CBR systems either as data, information, or knowledge. Representing cases as knowledge in a CBR system implies that the case base is not merely a source of information for the operator but a knowledge base that is actively used for the system's reasoning process [2]. Cases may be the only type of knowledge in such a system or they may be combined with other knowledge types e.g. in CREEK. These systems incorporate new cases in a way that makes them immediately ready to be used in the solving of new problems. A CBR system can facilitate a gradual transformation from a pure database or information system, to a knowledge-based system. In this way a system will always have its data available in a non-generalized form, and their active use can be incrementally put into effect by adding interpretation and reasoning capabilities to the system as the use of the system identifies what active decision support operators really want.

Knowledge Intensive Case-based Reasoning

Knowledge-intensive CBR assumes that cases are enriched with general domain knowledge. The meaning of the term "knowledge-intensive" may also vary, depending on what viewpoint to the concept of knowledge that a researcher has. What some people refer to as knowledge in the context of a case may be referred to as information by others or even as data [2]. It is commonly agreed that in the context of knowledge systems, the three concepts of data, information and knowledge can to a certain degree be viewed as a hierarchy [96]. Data are uninterpreted characters, unprocessed facts, signals, patterns, and signs that have no meaning. When data are being interpreted, organised or assigned meaning they become information. Once the data have been given an interpretation as information, it is elaborated upon in order to be better understood, and in order to derive new information. The elaboration process is a learning process. Knowledge is learned information, i.e. information that has been processed and incorporated into a system an agent's reasoning resources, and made ready for active use within a decision process. Knowledge is closely related to the widely shared view that learning is the process of integrating information into a body of knowledge [2]. When dealing with knowledge intensive systems, the knowledge base involved can be viewed as " a qualitative model of the part of the real world that the system is to reason about" [2], p. 5. The CREEK method [177] is a knowledge intensive case-based reasoning architecture where cases are submerged into the general domain knowledge ; particularly developed to tackle problem-solving and learning in open and weak-theory domains. This model is realised through a multi-relational semantic network, where an object-oriented, frame-based representation is used to capture both cases and domain knowledge. When modelling knowledge in CREEK, a combination of a top-down process for initial knowledge acquisition, and a bottom-up process of continuous learning through retaining cases, is used [96]. The top-down process is used to acquire and develop the conceptual model required to define the domain in question. Each of the CBR steps (retrieve, reuse, revise and retain) in CREEK utilises a general explanation engine [1], consisting of an activate explain-focus cycle. This cycle initially activates the relevant parts of the semantic network, it then generates and explains derived information within the activated knowledge structure, and finally, it focuses the selection of a conclusion that conforms to the goal in question.

Different design concepts as already discussed are considered and designers will need to be able to analyse which are better and which may lead to problems in building decision support systems that will reflect the current situation. In the process of sorting out an appropriate concept, designers may be surprise to find that certain design features do not work as well as anticipated. Objective measurement of design features, therefore, forms another building block of SA-oriented designs. Each design is empirically tested to identify any unforeseen issues that can negatively impact situation awareness, and allow the relative benefits of different design options to be considered. The measurement approaches of SA-Oriented designs can either directly measure situation awareness, or infer it from observable processes, behaviours, or performance outcomes.

2.9 Measuring Situation Awareness

There is no single all-purpose method available for measuring SA but there is an array of instruments adapted for a given situation awareness measurement need. These instruments do not all measure the same aspect of situation awareness. A common strategy adopted for situation awareness measurement is to combine a set of instruments in order to get a more global perspective on situation awareness. Measuring situation awareness is not a straightforward task and this has led to the emergence of several taxonomies of SA measurements [139]. Although minor differences exist in these taxonomies, all their emphasis is on the measurement methods. A method-based classification fulfills the actual needs. The measures of situation awareness can be classified according to the following categories: (1) measurement based on the observation of on-going activities (process indices, performance measures, behavioural measures, and the observer rating measures) (2) direct measurement techniques (think aloud, freezing, and real-time probes) (3) retrospective measurement techniques (recall awareness, and recall situation) and (4) team situation awareness measurement.

2.9.1 The Process Indices

The process indices show that an operator is actually making a situation assessment, a process that should result in situation awareness. The process indices of situation awareness are mostly looked for in psycho-physiological data. These psycho-physiological measures are not situation awareness measurement tools but are mostly used in difference fields of research. They provide indices from which situation awareness contents can be inferred [55].

Wilson [196] discovered two advantages of the psycho-physiological measurement of situation awareness. First, these measures are continuous and they do not require the freezing of an ongoing action. Wilson rightly pointed out that these measures are essential since situation awareness is typically required in a constantly changing environment. Probes or queries are not always made at the right time. Second, since the psycho-physiological indices can be recorded, it is also possible to track back what happened when critical information should have triggered changes in situation awareness. Wilson concluded that the psycho-physiological data could indicate when information was detected but, it is impossible to infer how this information was used. This type of data is indicative that a cognitive activity is ongoing. In other words, it provides more information about a process than about a mental state [24]. At least, if the operator is in the process of forming situation awareness, it can be hypothesized that actual situation awareness is either poor or judged out-of-date. From the physiological measures proposed by Wilson, four different types can be identified: (1) Electro Encephalo Graphy (EEG) can determine if the person is asleep, fatigued or mentally overloaded (2) Transient heart rate (3) Event related potentials or desynchronization, and (4) Electrodermal activity can inform about the detection of critical events.

Vidulich et al. [194] carried out a study on EEG and eye-blink in tasks that allowed either a poor or a better situation awareness. Eye-blinks are recorded by a technological system that measures their frequencies. In the poor situation awareness condition, the EEG spectrum indicated higher theta frequencies at the expense of the alpha frequencies, and the subjects expressed a higher rate of eye-blinks that were of a shorter duration [24].

These data are inconclusive but are indicative of a difficulty with mental workload, or of the activities associated with the process of situation assessment.

Andersen et al. [9] used an eye-tracking device to investigate how nuclear control room operators were using different monitor devices to regulate team SA. Unfortunately, it was difficult to determine how situation awareness influenced the task they monitored. Marshall [112] studied the eye tracking of Navy personnel during interaction with displays in a tactical command and control training situations. The point-of-gaze and the pupil dilatation were used to measure SA. The eye point-of-gaze is a technological apparatus used to determine where the person is looking or where he should look. While the point-of-gaze provided information about what the personnel were looking at, the pupil dilatation provided information about the cognitive efforts elicited by these data.

2.9.2 Performance Measures

Performance measures are either a response-time or a rate of success used in different fields of research. The response time might be the less interfered measures of situation awareness. Performance measures are closely bound to the situation, for instance the success of a given investigation, the response time to a specific hazard, the frequency of an appropriate response to a unique problem. For this reason, no specific instrument can be applied to a variety of situations. Each variable depends on the nature of the task under examination. The advantage of the performance measures is in the fact that the situation is not disturbed by the measure [24]. They can be used either in simulators or in real-life situations but their sensitivity is debatable for two reasons:

1. An operator may have an excellent situation awareness but could respond to a situation without any need to use it. It must be established that situation awareness is required for an operator to give the right answer to problems, before making use of performance measure. This problem happened during an experiment with nuclear plant operators [9]. The task was so well structured that no participant achieved less than expected. Even though situation awareness may have varied during the task, the performance was constant.

2. Lack of expertise or competence can also interfere with the situation awareness measure. In spite of optimal situation awareness, an operator can select a wrong answer to the right question.

A requirement for performance measurement is that situation awareness is the main, if not the sole, factor that influences a dependent variable. Otherwise, the results would be confusing. Such a criterion can be tested by general agreement among all the stakeholders in a scenario [139]. Pritchett et al. [145] propose the use of performance measures only in situations where the directory of all possible responses is known and where it is clear that a subset indicates poor situation awareness while another subset reveals a richer situation awareness. In order to avoid interference from other human factors, performance measures are mostly limited to simple experimental design such as:

1. Two pilots approach a dual runway airport. The first pilot is committed to the first runway. The second pilot starts his approach to the second runway. Unpredictably, the second pilot changes his route to the first runway. The time required by the first pilot to take action to avoid the second aircraft is a direct performance measure [25]. In this case, the action is not interrupted. The first pilot only needs to detect the conflicting behaviour of the second pilot. This is a Level I measure.

2. An air traffic controller gives incorrect information to a pilot. The time needed to understand that the information cannot be true is a typical Level 2 SA measure [59].

3. A crew member is working in the usual normal conditions of a regular flight when all displays are blanked. During this time, the position offset of the aircraft is modified. Displays are then restored. The time to detect the modified offset and to recover, and the percentage of success are SA measures [25]. In this case, the crew had to detect a change while they were trying to understand what happened. Both detection and comprehension were implied. These measures were Level 1 and 2 measurements.

Performance measures are commonly used in experiments where probable manipulation of situation awareness is expected to influence performance. Vidulich et al. [194] tested if, in several of these methods, the performance measures reported more situation awareness changes than other situation awareness measures. Interestingly, the ratio of positive outcomes with performance-based measure (0,70) is similar to those of other types of measurement (0,71).

2.9.3 Behavioural Measures

Behavioural measures are different from performance measures. The focus in behavioural measures is on methods rather than on outcomes. Although an operator may reach an increased situation awareness level, it does not imply that this potential will be effectively used. Enhanced situation awareness may have changed the way an operator behaves without any noticeable change in performance. For instance, a person with poor SA may need to look more frequently at a monitor, while a person with a rich situation awareness knows when to look at. Both may have detected the right information when it was needed and then performed adequately. However, their methods were quite different due to differences in situation awareness.

There are some behaviours that can be quite difficult to identify, for instance monitoring a sound from a headspeaker [9]. The inability to identify a change in behaviour does not mean that no change in situation awareness happened. Payne et al. [138] designed an experimental device for behavioural measures in situation awareness. On a computer screen, participants used mouse clicks in order to get the information they needed to make appropriate decisions. This force-the-user-to-ask technique was used by Ricks et al. [152] as a situation awareness measure during a simulated aircraft flight. Trafton [187] used verbal protocols and videotapes of weather forecasters while they were at work. Finally, Andersen et al. [9](pp.267) suggested that "the degree of correspondence (i.e. in terms of percentage) between a pre-defined ideal behaviour and the observed behaviour may constitute a measure of situation awareness." Then, with the force-the-user-to-ask techniques, the inferences made from behaviour observations can lead to the identification of specific situation awareness content or to comparison with an achievable situation awareness content inferred from the observation of an achievable ideal behaviour.

2.9.4 Freezing Techniques

Freezing techniques are used in simulators where actions can be temporarily suspended and resumed. An example of the technique is the Situation Awareness Global Assessment Technique (SAGAT). SAGAT was created in 1987 by Endsley [54], and it is now a reference among query techniques. SAGAT, which offers a great level of control over the measurement process, was initially designed to work in a simulated environment where the action can be frozen at randomly selected times. During these pauses, displays are blanked, and participants are queried to report their perception and knowledge of all SA-related aspects of the task. When the situation is frozen, the pace of action can pass from an intense activity to rest. It is not evident that the participant will continue in the same mindset when the action is resumed. Sarter et al. [164] criticized the freezing technique as possibly flawed by intrusiveness. In response, Endsley [54] found no difference in performance between probe intervals and no-probe intervals. Another answer was later brought by Endsley [60]. In one condition, subjects were told that no 'stop' would happen; only performance was measured. In another condition, subjects were warned that a situation awareness probe might happen. In one half of the probe expected trials, stop really happened; none happened during the other half. Results indicate that an interruption and even the expectancy of an interruption had no impact on performance. Even if the probe-time can be anticipated, subject performance cannot be influenced by characteristic demands when all parameters are investigated. Nevertheless, Endsley recommends randomizing the time course of probes.

Typically, when the action is frozen, simulator screens are blanked. This blanking screen technique was criticized by Sarter et al. [164] because it restricts situation awareness measurement to its sole memory component. If a person formed a good Level 2 SA of situation, he might not remember an exact parameter, but he might know where to look for it. When screens are blanked, Endsley [54] observed only a slow decay of memory of between 20 seconds and 6 minutes. This observation answered concerns about a possible lack of short-term memory interference, but it does not answer the second criticism that such a blanking weakens the Level 2 SA (comprehension) assessment.

A compromise was by found Hilburn [78] using a partial blanking technique. During a flight simulation, an ATC experienced the failure of a secondary radar (a loss of all text) while all other radar functions remained. If the controller can answer immediately, then the data was inside the situation awareness content. If the right answer follows a reference to the second screen, then experiment is cued about a Level 2 situation awareness. With this partial blanking technique, the situation awareness content is inferred from the observation of specific behaviors related to situation awareness.

The sensitivity of the SAGAT was later reviewed by Vidulich [193]. Not only does SAGAT answer well to different methodological approaches, it is also sensitive in different application domains. The criterion validity and reliability of the SAGAT has been reasonably tested. The test-retest coefficients vary between 0.92 and 0.99 [57]. The attempts to combine a set of situation awareness queries into one global SA score failed [56]. The SAGAT probes cannot be summed together in order to build a global score [56].

2.9.5 Real-Time Probes

The SAGAT requires the freezing of an ongoing action, mostly during simulated tasks. Many situations cannot be investigated this way. Real-time probes are an alternative to the SAGAT. The on-line probe techniques are suited when the pace of a situation is relatively slow or when there is some period of inactivity [139]. With SAGAT, when the action is frozen, a full set of queries is obtained. With real-time probes, individual queries are randomly disseminated along the action. Contrary to SAGAT, the focus is on time to respond rather than on response accuracy. Jones et al. [85] tested the fidelity and the effectiveness of real-time probes, and compared this measurement with SAGAT, SART and workload measure. Each measurement was taken during two different scenarios (peacetime and wartime) that respectively favour good and bad situation awareness. If a measure is sensitive to situation awareness, then there would be significant results between scenarios. Individually, the probes were not sensitive but when aggregated, the response time was marginally significant. On the contrary, the SAGAT queries cannot be aggregated to build an overall score.

The advantage of real-time probes over SAGAT is their applicability in situations that cannot be interrupted. Real-time probes are less intrusive than the SAGAT queries. It offers good control over the measurement process without threatening the level of situation realism. However, fewer queries can be asked during a particular lap of time [52]. Thus, this measurement is less exhaustive. Also, even though the probes do not interrupt the on-going action, they can nevertheless interfere with cognitive processes. Up to now, the real-time probes are a promising measurement of situation awareness.

2.10 Key Issues

This chapter has shown that situation awareness can be understood from a number of perspectives. In particular, a distinction is drawn between SA as a "process" and SA as a "state" of knowledge. However, it is generally agreed that if SA depends on a set of processes that are not an intrinsic part of SA as a state but on which SA depends, it becomes important to specify which processes are essential to SA [24]. To address the issue of process specification in SA, Endsley [58] proposed the expression "situation assessment" to represent the cognitive processes that produce the knowledge (state).

Furthermore, the attempt to simplify the concept of situation awareness from the complexity of defining and measuring its cognitive side has led to the development of models, building on SA relationships with other known applied cognitive models such as situation and mental models. Endsley [58] referred to situation awareness as a situation model, stating that a situation model is a schema depicting the current state of an operator's mental model. Mental model is an operator's static knowledge that can be recalled to achieve a goal [58]. Goals influence what the operator perceives, comprehends and anticipates in situation awareness [86]. Top-down goal processing along with bottom-up environmental cues processing provides a focused SA of the domain. A similar top-down processing is the notion of context. Context, like goal, acts as a filter to SA. Context filters SA in relation to the specific need of individual operators. Feng et al. [64] incorporated user context in a situation model for exploiting goal-based contextual information to achieve user-specific situation awareness using agents. The agents, one for each individual operator, communicate with the situation model and extract information of relevance for presentation to their respective users in accordance to the user context. Defining necessary heuristics based on bounded definition of the domain and responding to each and every new development was difficult due to Feng's rule-based decision support engine [64]. Experience-based systems such as case-based systems facilitates better maintainability and expandability than rule-based systems [162] since new knowledge is added by integrating new cases automatically to the case-base. Kofod-Petersen et al. [96] used case-based reasoning in modelling SA in an ambient intelligent system. The "perception" and the "awareness" layers of the system are comparable to Endsley's perception and comprehension layers of situation awareness. The third (sensitivity) layer adapts the ambient system's behaviour to the current situation. The sensitivity layer does not anticipate future situations to make it a projection layer. The adaptation of the system to the current situation was possible by combining a user's context with environmental elements at the perception level.

Building on Endsley's work this project exploits the idea that top-down context processing along with bottom-up environmental cues processing provides a focused situation awareness of the domain. Context has been shown to improve similarity assessment and problem solving of case-based systems. However, the usefulness of situation awareness has not been fully exploited in such systems. Situation awareness can complement context to improve case-based systems in variety of ways including the design of an adaptive user interface that will reduce the cognitive load on operators.

Therefore the remainder of the thesis provides:

1. A method of collaborative work with operators for effective experience and requirements gathering (Chapter 3)

2. An architecture that will model how situation awareness can be used to identify problems in experience-based decision making (Chapter 4)

3. A user interface of the system that will adapt to the operator's context (Chapter 5).

4. Also, an experience-based SA architecture that will model how situation awareness can be used both for problem identification and solving (Chapter 6)

Chapter 3

Action Research for Agile User-Centred Design

The aim of this project is to answer some theoretical research questions and also to produce a software product that will solve an organisational problem. The product should meet the functional and usability needs of the user. To achieve this goal, collaboration with domain practitioners in order to understand the activities of the domain, capture the practitioners requirements, refine the requirements for redesign at an early stage to save time and cost is required. This chapter argues that an approach integrating action research, agile development concepts and user-centred design addresses the stated objectives. Each of these are iterative, action-based learning approaches. Drawing them together enables the building of a series of solutions based on agreed priorities, theoretical prior knowledge, user-related activities and constant evaluation. Out of this a synergy of practice-based solutions and theoretical developments are drawn. The chapter starts with survey of the background of action research (AR), agile development (AD), user-centred design (UCD), and scenario-based agile UCD. From this background an integrated approach to answer theoretical research questions and solve an organisational problem is developed.

3.1 Action Research Method

Action research (AR) is a collaborative process of progressive problem solving between researchers and practitioners to solve organisational problems [13]. The dual commitment in action research is to study a system and concurrently to collaborate with operators of the system in changing it in what is together regarded as a desirable direction. The active collaboration of researcher and practitioners to achieve this twin goal requires co-learning as a primary aspect of the research process. What separates action research from general professional practices, consulting, or daily problem-solving is the emphasis on systematic study of problems and to ensure the intervention is informed by theoretical considerations. The researcher spends much of his time on refining the methodological tools to suit the exigencies of the situation, and on collecting, analyzing, and presenting data on an ongoing, cyclical basis.

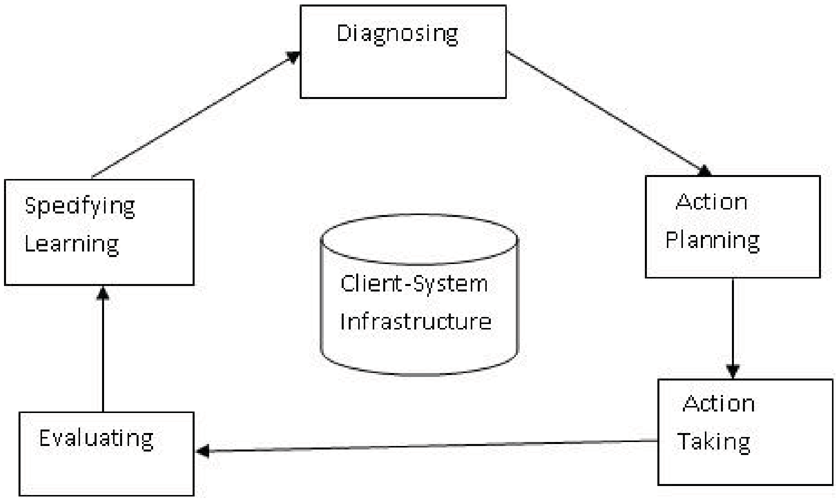


Figure 3.1: The Action Research Cycle (Susman, 1983)

Susman and Evered [182] distinguishes five phases to be conducted within each research cycle (Figure 3.1). Firstly, a problem is identified and data is collected for a more detailed diagnosis. This is followed by collective postulation of several possible solutions, from which a single plan of action emerges and is implemented. Data on the results of the intervention are collected and analyzed, and the findings are interpreted in light of how successful the action has been. At this point, the problem is re-assessed and the process begins another cycle. This process continues until the problem is solved.

The initiation of the problem to be solved can be research-driven, a situation where the researcher in possession of a general theoretical approach to solving a particular problem looks for an organisation that has such a problem situation. It could also be problem-driven, a situation where practitioners confronted with problems seek the help from theoretical specialists. Sometimes, the initiation can be a combination of the two ways probably arising from a meeting between the researcher and the practitioners [10]. However, the fundamental area of consideration before setting the cycle to solve the problem in motion is the issue of the number of cycles to engage in and the timescale to achieve results. Whatever the decision may be, there has to be an appropriate balance between action and reflection [149].

Action research provides the platform for researchers, through actions in practical situations, to gain access to study complex social phenomena [13]. It is a way for the researcher to step out of the academic environment and to support practical needs. Action research seeks to bridge the gap between theory and practice, so evaluation should include examinations of the practical accomplishments, which might take various forms, such as new designs, technical reports, networks, or improved software processes [23] [114]. In accordance with its principles, it is chosen when circumstances require flexibility, the involvement of the people in the research, or change must take place quickly or holistically [122].

3.1.1 History and Application of AR

Modern action research originated in the 1940s in two independent research programs with the development of action-based social psychology [13]. Lewin [100] [101] developed a field-theory version of action research for Group Dynamics in order to study social psychology at the Michigan University Research Centre. Tavistock Clinic which later is known as the Tavistock Institute, independently developed an operational research version of action research [188]. The Tavistock Institute used AR to study psychological and social disorders among veterans of battlefields and prisoner-of-war camps. The two developments converged when Tavistock joined Lewin [13].

Tavistock and Lewin inspired a vast stream of studies in action research, although adherents developed slowly due to funding. The post-war funding structure of social science research did not encourage AR because it was largely sponsored by public money [13]. Researchers tended to seek projects that relied on quantitative data and the computer analysis that satisfied the governments' vision of science [34]. This post-war emphasis on professionalism and precise data collection methods led to a general decline in qualitative research skills [38]. As a result, AR methods were seldom applied, and often of marginal scientific quality [163].

The marginalization of AR helped mature the recognition that AR operated with a different epistemology than traditional science [182]. Early work by Mumford et al. [123], brought her Tavistock experience into the information systems field as a systems development technique called ETHICS. Checkland and Holwell's use of AR in connection with systems analysis is another landmark for the technique in the information systems community [34]. Checkland and Holwell used AR to develop soft systems methodology, and as a result, AR concepts for gaining professional knowledge permeate the soft systems approach itself. Checkland and Holwell also explicitly linked AR to the philosophy of science and systems science [34].

Action research is incorporated and modified into many different research disciplines [20]. In recent years, action research was explicitly introduced to information systems (IS) and as a purely research methodology by Wood-Harper et al. [12]. Like Mumford and Checkland et al, Wood-Harper et al also incorporated AR concepts into an action-based systems development methodology called Multiview [197]. Scholl [168] combined AR and system dynamics (SD) to solve ill-defined problems. The integration of action research to system dynamics is a participative modelling intervention or group model building [168]. Oyo et al. [134] added some details to participative system dynamics modelling process by extending it into a process design of six iterative phases, each with explicit outcomes. The approach demonstrates that integrating action research into system dynamics yields comprehensive process modelling phases in which the purpose and benefits of modelling are entailed within the phases and not only through the outcome of modelling [8]. The integration of action research and system dynamics by Scholl [168] outlines several potential areas for the integrated research design. While Scholl acknowledges that these methods belong to separate continuums of research paradigm, i.e., system dynamics on the quantitative side while action research on the qualitative side, he maintains that they both address ill-defined or messy problems and hence can complement each other. Narrowing down to action research, Checkland and Holwell [34] points out that the most unique aspect of AR is in its iterative process of problem diagnosis, action intervention, and reflective learning by the researcher and participants. On the other hand, a system dynamics project starts from a problem to be solved or an undesirable behaviour that is to be corrected or avoided [66]. This primary step uses the wealth of information that people possess in their heads. Mental knowledge is a rich source of information about a system because people know the structure of a system, and the policies that guide decisions in their minds. Based on these and other concepts about action research and system dynamics, Scholl [168] adds that group model building (GMB) - also regarded as "client-centred system dynamics modelling" [7] brings system dynamics in close vicinity of action research through joint participation.

Joint participation is key to action research in the way it is to agile software development and user-centred design. Effective stakeholders' participation in action research can benefit agile development and user-centred design.

3.2 Agile Development

Agile development (AD) is defined as "an iterative and incremental (evolutionary) approach to software development which is performed in a highly collaborative manner by self-organizing teams within an effective governance framework with just enough ceremony that produces high quality software in a cost effective and timely manner which meets the changing needs of its stakeholders" [170]. Generally, development requires some sort of plan, but plans always change. Therefore, agile methods do the minimum of up-front planning so that as little time as possible is spent on a plan that will soon be obsolete [17]. Any project will miss the mark to some degree, and the only way to know if something works is to test it [169]. The sooner project results are tested, the sooner errors and discrepancies are discovered. So agile methods focus on short release cycles and short iterations. Engineers receive user feedback consistently throughout the development cycle, rather than near the end at formally defined stages. Each iteration is a coherent subset of functionality that can be tested directly by the users. The main distinction between agile requirements engineering and traditional requirements engineering is that the former welcomes rapidly changing requirements even late in the software development process and the latter gathers and specifies requirements up front prior to software development [6]. The dynamic nature of complex organizations makes continuously changing requirements normal, hence it is difficult to gather and specify complete, stable and accurate requirements up front. Rapid changes in competitive threats, stakeholder preferences, development technology, and time-to-market pressures make pre-specified requirements inappropriate [117].

The Agile Movement was formed as it became obvious that real-world projects required software developers to react to changes in requirements for organizational software [180] with high flexibility without causing any harm to the quality of the software. A critical element of AD is the light weight planning [42], and the focus on the participating stakeholders and the software itself in favour of the formal definition of requirements and a strict application of defined processes [99].

However, the role stakeholders should play in developing a system, how they should be involved, and how much they should be involved has been a matter of dispute [74]. User involvement is a central concern to the field of Human Computer Interaction (HCI), and the importance of integrating software engineering and HCI methods has been recognised for many years now [201]. This integrated approach of user-centric perspective into software engineering practice makes usability awareness widely known and software products more user-centred and usable [171]. The Agile Manifesto and action research emphasises the importance of involving customers and practitioners respectively in development projects, but this practice is proving to be problematic [113], and it is rare for a real end-user to take the role of customer or a practitioner. User Centred Design (UCD) is an approach which aims to involve the users in a meaningful and appropriate way throughout a systems development [141] [70]. Gould et al. [70] first proposed three principles of UCD in the mid-80s, and over 20 years since then, various techniques for involving users successfully have been developed. Integrating UCD and agile development therefore has the potential to help an action researcher with the difficult practice of involving practitioners, and the wider concern of how to integrate HCI concerns with software engineering.

3.3 User-Centred Design

User-centred design (UCD) is a design philosophy that puts the intended users of a system at the centre of its design and development by involving the users at key points in the project to ensure the system meets their requirements. Norman [127] defined user-centred design as “a philosophy based on the needs and interests of the user, with an emphasis on making products usable and understandable” [127]. User-centred design most commonly comprises four stages: requirements tasks, design, prototyping, and evaluation (Figure 3.2). The process of iterative design is controlled through the processes of evaluation.

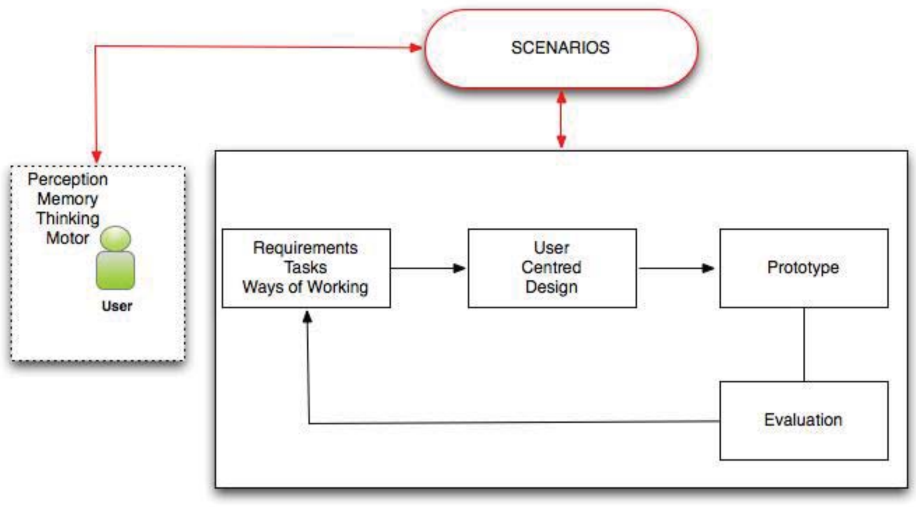


Figure 3.2: User-centred design (Holt, 2009 [79])

These stages are carried out in an iterative fashion, with the cycle being repeated until the system usability objectives are met. The ISO-standard 13407 user-centred design processes for interactive systems (ISO 1999) describes UCD as an iterative process that should follow the following key principles:

- Involvement of users and very good understanding of user and task requirements.
- Appropriate allocation of function between user and system.
- Multi-disciplinary design teams.
- Iteration of design solutions.

From these principles, users must be involved in UCD to have a clear understanding of their tasks and requirements. Secondly, it is important to determine which part of the task should be carried out by the user and which should be carried out by the system [109]. For example, focusing users on tasks that require their expertise and then the tedious time-consuming routine work carried out by the system. Also, user-centred design should be collaborative in its nature involving the exchange of ideas from different competences [109]. And then finally, the design should be iterative. Both agile and user-centred design recognise initial attempts will be flawed and need early testing rather than late user acceptance to uncover and correct bugs/usability defects.

User-centred design (UCD) methods emphasise the importance of working with stakeholders using participatory design, ethnographic methods and usability testing [43] [158]. Agile methods also value client collaboration over contract negotiation. From a usability standpoint, the problem with agile methods is the absence of any mention of the importance of end user collaboration. The agile values of working software over comprehensive documentation and responding to change over following a plan are also somewhat at odds with usability approaches such as scenario-based design (SBD). SBD follows a layered approach where the entire system is designed in phases with increasing fidelity [158]. However, the user-centred design approach assumes end to end involvement of end user in the development process from requirements analysis to usability testing of the implemented system [158]. Cooper et al. [43] proposed an agile user-centred design approach where the user interface is first designed using traditional usability engineering methods before passing it on to the developer for implementation. The problem with this approach is that it prevents implementation from starting right away and conflicts with the principle of frequent delivery of working software to clients [99]. Patton [137] introduced approaches to agile development where developers are given some training in user-centred design methods so they can both design and implement the system. A potential challenge with this technique is that if developers are both designing the user interfaces and implementing them, they may tend to sacrifice usability for ease of development [43]. Miller et al. [118] highlighted the importance for having a separate usability engineer working within the team for user interface intensive systems. One problem with the Miller approach is how to balance the different needs of the usability engineer and software developers throughout development as issues such as new requirements, tight deadlines and limited resources emerge. Integrating user-centred design into agile methods can introduce competing goals between development and usability, particularly when there are distinct principles that are working in parallel [65]. The agile principle is: "simplicity–the art of maximizing the amount of work not done–is essential" [99]. Simplicity in the UI often does not necessarily align with simplicity in the implementation [126]. However, power imbalances can arise in an agile user-centred design team with usability engineers not having an adequate say in how the design and implementation of the system proceeds [33]. Addressing this problem requires committed, motivated team members. It also requires an understanding of the different goals of a project and the importance of usability in terms of importance to delivering value to the client [14]. This also relates back to the issue of communication and collaboration between usability and development. Scenarios are regarded as a communications mechanism in agile user-centred design that attempt to bridge the gap between the agile process and usability [31].

3.3.1 Scenario-Based Techniques in Agile UCD

Scenarios had a long tradition in user-centred design even before that term was coined. Scenarios can represent entire activities, describing the social settings, resources, and goals of users, looking at the "big picture" of how work is done and not a narrow description of the task [124] (Figure 3.3).

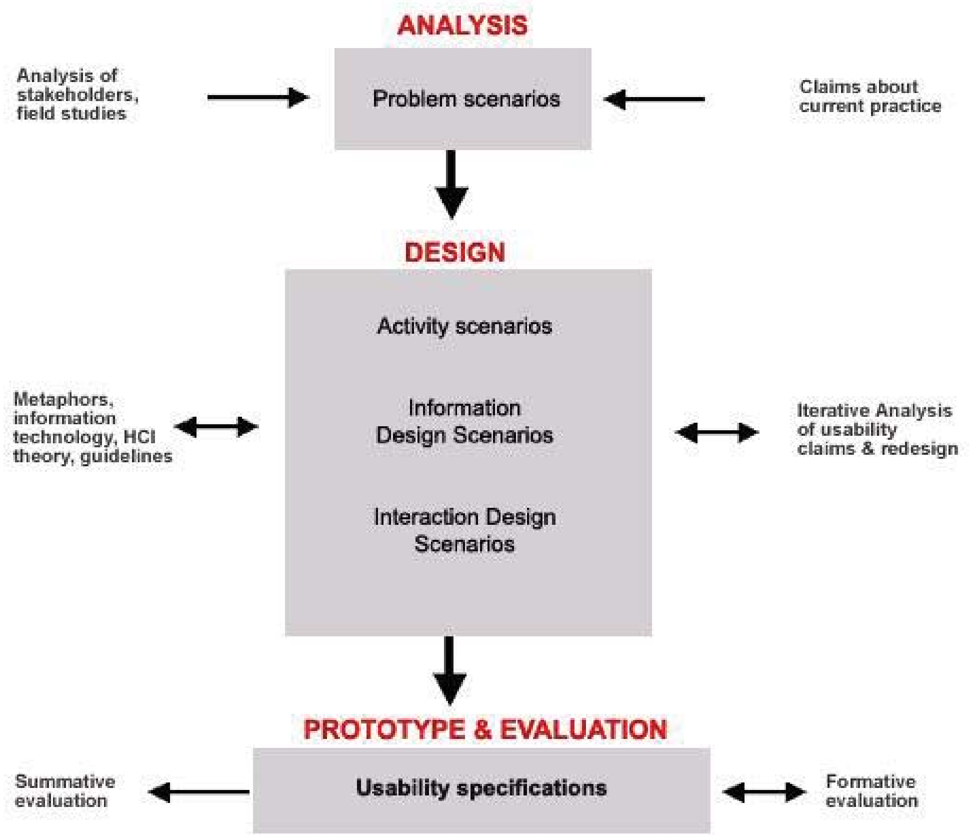

ANALYSIS
Analysis of stakeholders, field studies
Problem scenarios
Claims about current practice
DESIGN
Activity scenarios
Metaphors, information technology, HCI theory, guidelines
Information Design Scenarios
Iterative Analysis of usability claims & redesign
Interaction Design Scenarios
PROTOTYPE & EVALUATION
Summative evaluation
Usability specifications
Formative evaluation

Figure 3.3: Scenario-based design model (Rosson and Carroll, 2002 [158])

Kyng [98] employed scenarios as a communications medium for software development. Carroll et al. [31] proposed scenarios enriched with claims as a method to speed up the task-artefact cycle and developed their approach into a full process called Scenario-Based Design [132]. Scenarios are widely used in organisations as a guide for development in a less elaborate form and not relying on textual representation for refinement of the interaction design, but however, with rather promising results [77]. Scenarios close the gap that exists in agile methodology when it comes to work context analysis and defining a consistent design vision. Scenarios describing the situation focus on problems in their context [110] and provide requirements documentation that is both complete and at sufficient level of detail to remain flexible regarding details in the design [131]. This enables the formation of an almost holistic vision of a future system at an early point in development, which provides the basis for the reflected negotiation of core requirements. Scenarios that describe the future system development include concrete activities of its users; this identifies the core functionality of the design and allows extending the description simply through writing further scenarios [31]. These scenarios are written with participation of the developers, and evaluated with participation of the end users [131]. Developers and users are part of the project stakeholders. A stakeholder is any person or entity with a declared or conceivable interest in a project [44]. The identification of stakeholders is done during the "information gathering" phase of a project [44]. There is the direct, and the indirect stakeholders. The direct stakeholders are the users of the system, the people directly affected by the system who have the most impact on and interest in the project. Direct stakeholders carry out the processes, serve the processes, and are served by processes. The indirect stakeholders consist of two groups; decision-makers and external agencies. The decision-makers are the management of the organisation that approves the implementation of the change. The external agencies are outside the organisation, and though not directly involved in the change, their opinions could affect the change [44].

Scenarios are open-ended, as they are used in design; new questions (propositions) (Figure 3.4) emerge, which can be answered only by returning to users [110].

This makes scenarios a helpful technique for user-centred design, but on the other hand system's features in scenarios are embedded in complex narratives that are intended to illustrate how each of the features affect a user's specific task experience, and how the features interact within and across tasks. These specific system functions or features may appear in several scenarios which require a great deal of care to understand the different contexts for those features [157].

3.4 Integrated Approach for the Hydrate Case Study

In this study, an integrated approach based on an action research model is adopted. In action research, the researcher and people both develop knowledge to improve practice. Knowledge is developed through intervention in specific problem domains. Practice is improved through changing current practices. These two purposes are intertwined as theory is drawn on, developed, applied and evaluated. Action research is a collaborative activ-

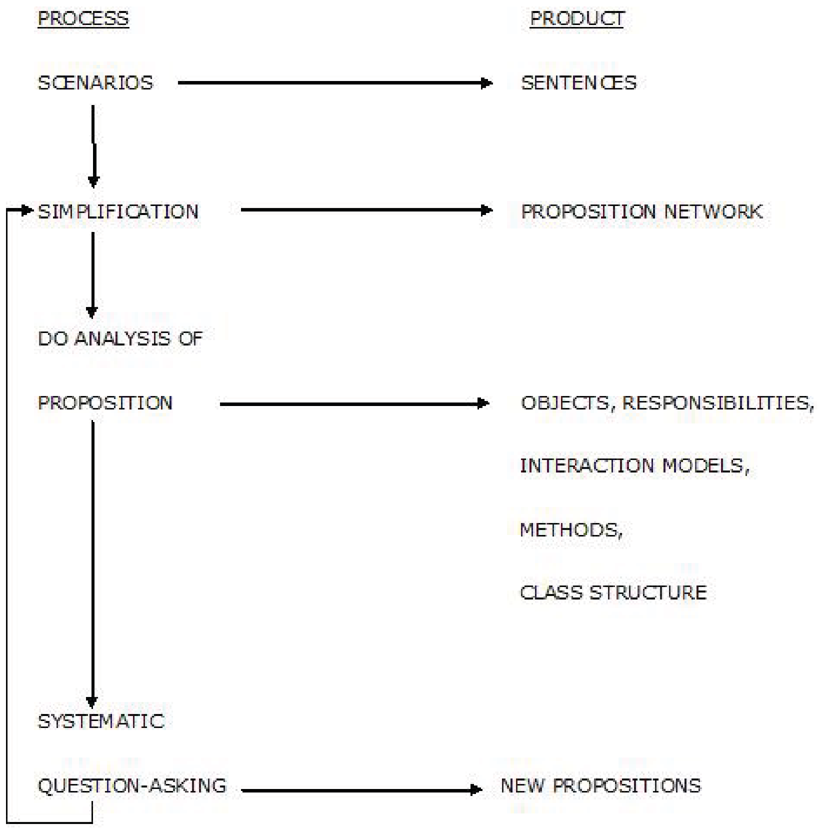

PROCESS
PRODUCT
SCENARIOS
SENTENCES
SIMPLIFICATION
PROPOSITION NETWORK
DO ANALYSIS OF PROPOSITION
OBJECTS, RESPONSIBILITIES, INTERACTION MODELS, METHODS, CLASS STRUCTURE
SYSTEMATIC QUESTION-ASKING
NEW PROPOSITIONS

Figure 3.4: Scenario-based systematic question-asking (Robertson, 1996)

ity involving researchers and participants working together to address a current issue in the domain, thereby informing the wider body of knowledge through a cycle of reflection on theory and practice [10]. The result of the work is the refinement of the theoretical knowledge-base. Data was sourced from three industrial organisations but due to confidentiality and sensitivities, direct intervention within these companies proved not to be an option.

A multidisciplinary supervision team from well engineering and computing supported the researcher given the dual nature of the project. The research involved a group of academic experts and industry participants within the IDEAS Research Institute from a North Sea flow assurance project. The group of experts acted as a focus group for the research project. The group varied slightly between meetings but consists of 8-10 participants at each meeting. The industry participants were practitioners in flow assurance from a variety of companies. The group provided insights to the current practices and issues, participated in the design, development and evaluation of the system, and reflected on the outcomes to shape the lessons from the work.

The purpose of this project was to model decision support systems through an action research participatory project [13]. Within an action research methodology, different methods are integrated resulting in a research-design process comprising three building blocks; scenarios, agile user-centered design, and business change. Each of the blocks contained internal iterative processes (Figure 3.5).

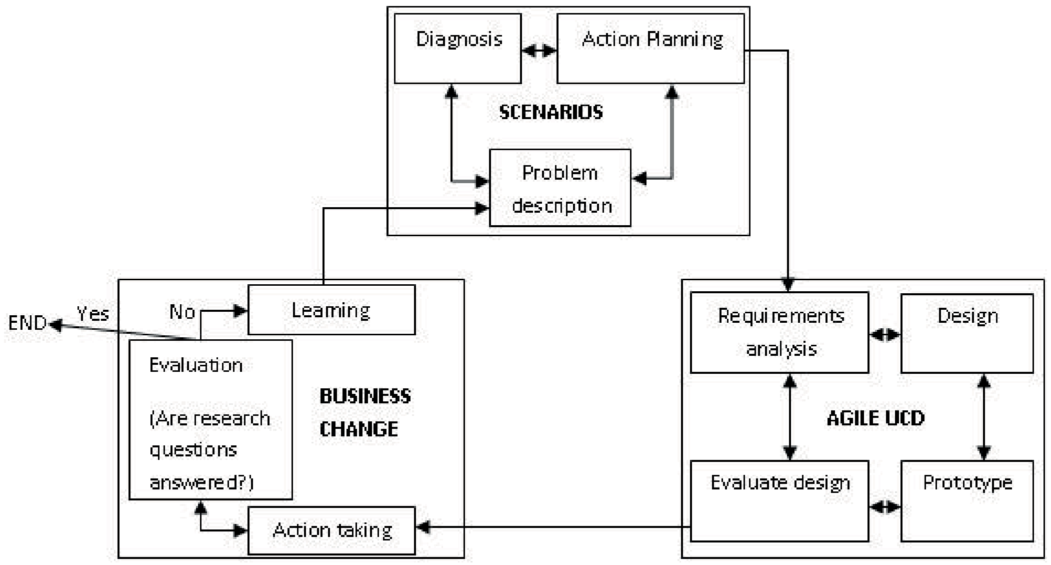


Figure 3.5: Action research-design model

Scenario development followed an internal process of stating the current situation (problem description), reviewing the situation (diagnosis), and identifying actions (action planning). The second block, agile user-centred design, is an “action tool development” process situated between two action research stages, action planning and action taking. The argument here is that in information systems AR, after actions have been planned (identified), there must be tools to be used in action taking before moving to action taking. The tool design and development process followed four iterative UCD processes, requirements analysis, design, prototype, and design evaluation. Similarly, the last block, business change, followed the process of reflecting on the implementation (action taking), evaluating the impact of the tool in action performance (evaluating), and reflecting on the overall research process (learning).

3.4.1 Developing Scenarios in the Hydrate Case Study

The purpose of the action to be investigated must be established before the project. In this project, the hypothesis is that the modelling of situation awareness as part of context-aware case-based decision support systems will improve decision making process of operators in dynamic environments. This theoretical assumption is first of all defined using scenarios. Scenarios in this project comprised problem descriptions, diagnoses, and action planning.

Problem descriptions are provided as text descriptions. To diagnose the problem, natural language scenarios are simplified to produce a set of propositions that are subject to objected-oriented analysis which in turn generate objects (Figure 3.4). The objects are identified, and the methods and interactions that would produce the behaviour are defined [154]. These processes are achieved by using domain knowledge and object-oriented system design skills to elaborate on the explicit knowledge provided by the scenarios. The task scenario was simplified by partitioning the scenario into propositions to identify candidate design objects. The propositions, apart from helping to identify candidate objects, also served as guides to object interactions. It shows the objects that are active and the ones that have been acted upon. The propositions also show the interrelations among the objects and some basics about the properties of the objects [154].

At times, the propositional analysis provided useful information but it was not always sufficiently detailed for guiding design. Systematic questioning was therefore used to elaborate and refine the propositional list [154]. Questions were asked on each item of the propositional list. The why-questions were used to receive both intentional and causal information. The how-questions provided the procedural, causal, and enablement information. Answers to the why-and-how questions exposed some of the content of the problem space and generated materials for the research. These answers revealed important information that was not explicit in the scenarios but useful in identifying other candidate objects.

Further diagnoses were carried out using claim analysis. A claim in a scenario is the set of causal relations associated to a given feature within that scenario [158]. Claims analysis provided design reasoning by documenting the designer's hypotheses about how these features may affect usability, provided a balanced view of the scenario and, extend the problem scenarios or elaborate the initial scenario specification [157]. Claims analysis in scenario-based design focused on reasoning on the important design tradeoffs for a particular usability situation [158].

The last level of scenarios is action planning. Action planing draws from prior theory to develop a conceptual solution to the diagnosed problem. In the diagnosis, the focus is on the core tasks of the problem domain and what the system will support. In action planning, the emphasis is on how to apply and develop existing knowledge to address the users' problems. At this level of the project there is a better idea of the actions that are required by the users, and how the system responds to these actions. Here, a conceptual framework to solve the identified practice-based problem was developed.

3.4.2 Agile User-Centred Design for Hydrate Prediction

The concept developed in action planning is used for the design of a tool to solve the problem. Agile, user-centred design and development method (Agile UCD) is an iterative and evolutionary development comprising of requirement analysis, design, prototype, and design evaluation. Tools to be used in carrying out actions are developed at this level. Here the design and implementation of a situation-aware decision support system for hydrate control in the organisation is carried out. The hydrate prediction method, as instantiated in the organisation's flow assurance system, was developed within an agile development approach. The project was motivated by flow assurance challenges resulting from hydrate formations in deep water oil and gas operations. Throughout the project the solving of the practical issues remained core. At the beginning of the first of the two years of collaborative work with practitioners, the researcher acted as an observer. In the last year of the two years, the researcher evaluated the outcomes of the research-design on hydrate prevention.

Merging user-centred design within an agile approach is intuitively attractive given the user-focused nature of both activities. Others have begun to show how they have done this for agile developments [65]. Here there was no intention to strictly merge techniques from the agile and user-centred design methods but simply to emphasise the iterative, continuous nature of the engagement and learning. There were challenges in doing this because whilst UCD is user focused it is not naturally agile in nature.

First there was an upfront requirement gathering stage for user-centred design that occurred before the software prototype development began. The object-oriented method was used to explore and model the requirements and the functional specifications of user-system interactions to achieve a user-centred design. The elaborated scenario from action planning formed the basis for the requirements analysis using the Unified Modelling Language (UML) [15]. In requirements analysis, the functional and non-functional requirements were analysed. Diary studies and persona development were used to understand the functional requirements of the users. For a complete analysis which includes non-functional requirements, information from structured interviews, contextual inquiry and task analysis with domain practitioners were used. The researcher performed contextual inquiry to better understand who the users were and their needs in terms of what tasks they needed to perform. Harmonization of the functional and non-functional requirements in order to have the requirement specifications that served as a roadmap for the design. A persona was constructed and documented in scenarios of interaction, and functionalities detailed by use of use cases to describe the sequences of interactions between the user and the system to achieve a sub-goal in accordance to a user context of use.

The upfront requirement work for the user interface design is a key difference between the agile methods and user-centred design. UCD naturally aims for a nearly complete set of requirements before development begins [183] but in this agile and user-centred design project requirements analysis was time boxed for short periods of time. As a result, this shortened the upfront UCD requirements process and, hence, small sets of requirements were derived from scenario-based systematic question-asking and Use Case analysis for each design iteration.

After the requirement analysis was completed, the next stage was the design stage, the iterative stage. At this stage the researcher and the practitioners take decisions on which of the requirements will be implemented in a particular iteration and which requirements are moved to the next iteration. A conceptual design is produced, supported by UML to give form to the functional requirements, and features from the users' view. The design produced an information architecture, and interaction standards development representing each identified object with a class, and the classes have associations that correspond to the links between objects.

Once a decision was reached on which requirements would go into the current iteration, a proof of concept prototype development begins. In an ideal agile and user-centred design development, while requirements for the current iteration is implemented, more contextual inquiry and UI testing to be used for the next iteration continues [67]. That means that development and UCD are carried out concurrently. The UCD process sets the stage for the next iteration. The challenge here is that the researcher was the key actor in both development and UI design and hence, working concurrently on both development and further requirements for the next iteration was resource constrained. Further requirements work were carried out if necessary after the current development process was completed.

Providing extensive functionality in development for effective performance is not enough. Users must understand what the functions do and how to use them. Usability or performance evaluations follow prototyping. The prototype was evaluated jointly with practitioners to assess the efficacy of the design on the practical problems they faced. If a prototype passed usability or performance test, it was marked as a finished product and was used for intervention strategies. On the other hand, if a prototype failed usability or performance test, the UCD requirements will be refined for another agile development and the process begins all over again. The evaluation reviewed the design process on a cyclic basis to ensure that all of the specified goals or objectives in the project plan for a satisfactory result were met.

3.4.3 Business Change in Hydrate Formation

In the last phase, business change, theory was informed by action taking using the developed tool. Lessons related to how the designs improved decision making were learned. Also, how the participative user-centred design process fitted with the agile development methods to support the creation of a system that improved practice was reported. Reports by the researcher to the supervisory team provided a summary of the design success criteria that were met, success criteria that were not met and reasons for the problem, what we can learn from the design to improve research and practice for the next prototype. In particular, the report identified any techniques or practices used in this design that worked well, and which the project team felt would benefit current and future projects.

The research evaluation reports do not serve to improve the design that is being evaluated; they serve to improve the next prototype to be worked on. Lessons were learned for next time through the cycle. After each cycle a new set of scenarios and related user-centred designs were developed from the lessons learned in the previous prototype. The iterative research and design cycles continued until the research questions were answered.

Chapter 4

Situation Awareness for Problem Identification

4.1 Introduction

Problem solving with case-based decision support is carried out by using and modifying solutions of similar past problems. Similar past problems are identified by matching features of the new problem to features of past problems in the case-base. Similarity is based on features that describe the nature of past cases in the case-base [162]. In complex domains, problems are not described by their inherent nature only. Other factors influences and describes problems. The addition of context in problem description in complex domains has shown to improve similarity assessment [200].

This chapter argues that problems in complex and dynamic domains develop from an evolving state of the environment that cannot be described based on static features of past cases and the user context alone. The work suggests situation awareness modelling as a means to assess and identify situation-dependent problems based on the context of users. And that features for reasoning about a problem following a CBR process should include the identified situation. The approach enables case-based decision support systems to form situation awareness through reasoning about perceivable context and elements of the environment during problem solving.

This chapter combines the concept of situation awareness, context awareness, case-based reasoning, and general domain knowledge in decision support. The case-based reasoning component of the system is the part that seeks to accomplish a problem solving task. The situation awareness component uses the context of the user to provide relevant information about the situation to be used in the reasoning process. The general domain knowledge provides explanations to the outcome of the reasoning process. The approach is useful in solving and anticipating situation-dependent problems. This chapter focuses on the architecture of the approach. The approach is evaluated for the prediction of hydrate formation in subsea gas pipelines, a scenario in flow assurance control in the oil and gas industry. Following the research-design methodology, the work start with scenarios followed by an agile UCD. A design evaluation is then carried out and finally, a reflection for business change.

4.2 Scenarios

Scenarios, the first segment of the research-design process comprised problem description, diagnosis, and action planning.

4.2.1 Problem Description

In this sub-section the domain problem is described and, following the Robertson model in Figure 3.4, narrative texts are used to describe a scenario in hydrate formation prediction in sub sea oil and gas pipelines (Figure 4.1).

Natural gas hydrates are solid crystalline compounds that are formed by the chemical combination of natural gas and water under high pressure and low temperature. Wellhead temperatures are normally colder than that of the reservoir, which usually contain water, so that water condenses from the gas at the wellhead and enters the flow lines from the well.

If the pressure at the wellhead is high, the gas may remain saturated in the flow lines or become saturated due to further cooling of the gas as it flows through the lines. This situation exacerbates hydrates formation in oil and gas flow lines starting from entrainment

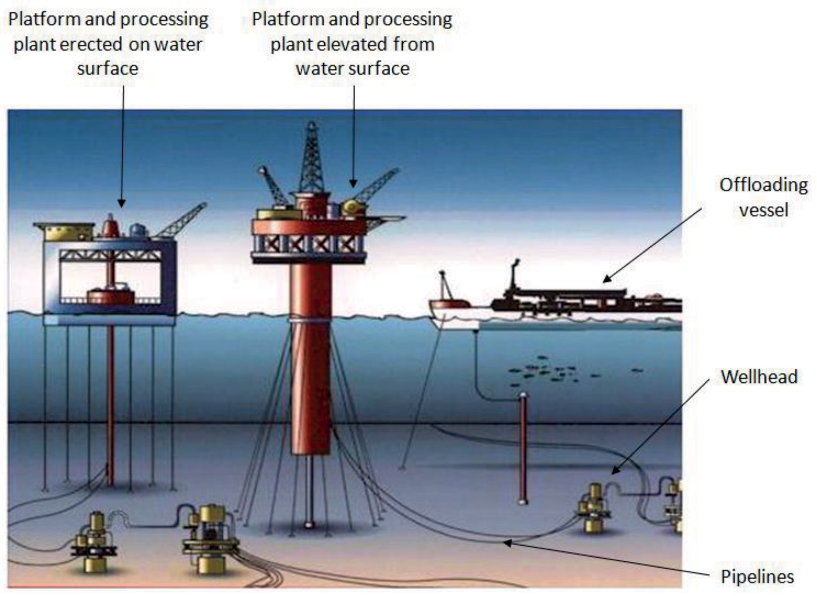

Platform and processing plant erected on water surface
Platform and processing plant elevated from water surface
Offloading vessel
Wellhead
Pipelines

Figure 4.1: Sub-sea oil and gas operation (Shah and Deniz, 2009)

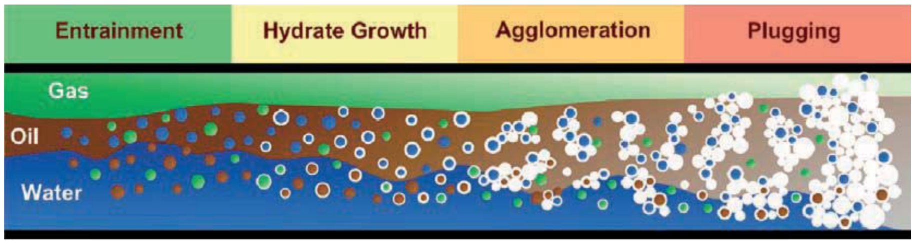


Figure 4.2: Conceptual model for hydrate formation in a multiphase flow systems consisting of water, oil, and gas (Zerpa, 2011)

to plugging (Figure 4.2). Hydrates plugging blocks flow lines (Figure 4.3) making oil and gas operators lose millions of dollars.

To prevent hydrate formation and maintain steady flow in fields, oil and gas operators carry out flow assurance analysis to determine the wellhead temperature and pressure that can transport a given volume of gas to the platform. They also carry out other preventive methods such dehydrating the gas, insulating the pipelines, injecting chemicals to break hydrogen bonds. These approaches do not address all the problems of hydrate control due to uncertainties in sea floor temperature characterised by wind direction, wind speed, and solar radiation of a geographical location [106]. Furthermore, the amount of water (a key element in hydrate formation) associated with the gas from the reservoir is always changing. These factors makes the task of hydrate formation control very complex and

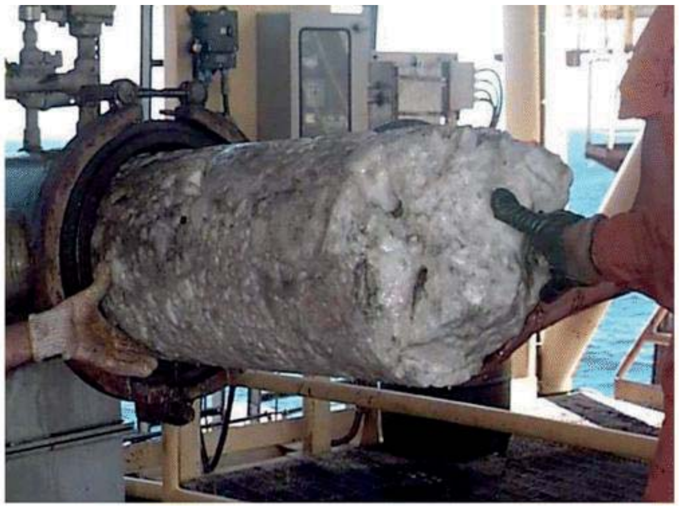


Figure 4.3: Hydrate block-Petrobas, Brazil (Koh et al, 2011)

dynamic and therefore an ideal case study for this research.

4.2.2 Scenario in Narrative Text

An engineer monitoring the formation of hydrate plans to use a system that used operational rules to understand situations in sub-sea oil and gas pipelines. The aim is to use knowledge of the situation to retrieve successful past solutions from an experience-based system. The systems should understand the current situation and extract similar past situations using the engineer's context and elements from the sub-sea pipelines. The situation presented is a warning situation. Tasks extracted using the current situation are, reduce water dew point task, and chemical injection task. He/She decides to use a chemical injection method to solve the problem in the absence of dehydrator. The available chemical for the engineer to use is methanol which is cheaper on a volume basis than glycol. Methanol is distributed in three phases; aqueous, vapour, and liquid. At the aqueous phase the engineer uses the Hammerschmidt equation to determine the methanol molecular weight and k-value before injection.

The problem scenario contains identified elements such as projects stakeholders, their activities, and tools or artefacts they use. It represents and illustrates current practice in the domain but changes based on circumstances and tasks of operators (see Section 6.3.1).

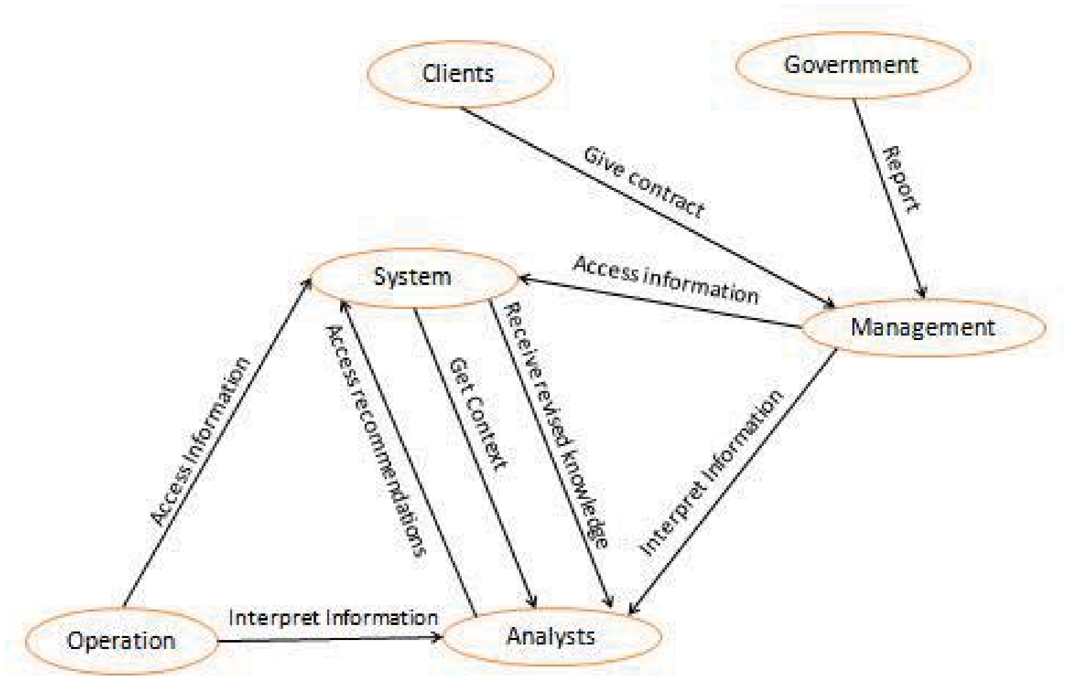


Figure 4.4: Stakeholders diagram

From the above (Figure 4.4), the Analysts (Engineers) as direct stakeholders, serve the System by providing contextual data, and revising the knowledge base. The Analysts are served by the System by receiving recommendations from the System. The other direct stakeholders are the operational division, and the management who are both served by the System and the Analysts. The operational division, and the management receives information from the System but these information are interpreted to them by the Analysts. The interest of management is mainly on the production figures. They (management) request this information (figures) from the Analysts after receiving information from the System. The company Management provide Government and Clients with information on organisatonal plans.

4.2.3 Diagnosis

In diagnosis, the above problem description is simplified in a network of propositions. The propositions are analyzed to generate objects, responsibilities, interaction models, methods, and class structure. With systematic question-asking method involving some why, how, and what questions, the propositions are further analyzed to generate new propositions. And lastly, in action planning, the new set of objects, interactions; methods etc from the new proposition are used to elaborate the scenario, making it more appropriate for functional requirements (use case) analysis.

Proposition Analysis

The task scenario was simplified by partitioning the scenario into propositions to identify candidate design objects as follows:

1. Engineer monitors the formation of hydrate
2. Engineer uses system
3. System understands situation
4. Knowledge is retrieved from experience-based system
5. System senses the sea floor environment
6. System integrate context and cues
7. Chemical injection is a task
8. Reduce water dew point is a task
9. Hammerschmidt equation determine molecular wt. and k-value
10. Methanol distributes in three phases, aqueous, vapour, and liquid.

Candidate Objects Identification

Candidate objects are identified as both the subjects and objects of the above propositions. The propositions apart from helping to identify candidate objects also served as guides to object interactions e.g. showing interactions between Engineer and the Systems. It shows the objects that are active, for example “engineer” and the ones that have been acted upon, for example “system”. The propositions also show the interrelations among the objects and some basics about the properties of the objects [154], e.g. the object "molecular weight" and "k-value" defined by the "Hammerschdt equation". Some objects identified are, engineer, hydrate formation, system, chemical injection, methanol, sea floor environment, use solution, retrieve, and experience-base.

Scenario Exploration by Systematic Question-Asking

The proposition analysis provided useful information but it was not sufficiently detailed for design. Each proposition has a setting that explicitly describes the starting state of the current situation and implicitly depicts the characters that take part in the situation in the scenario. Each proposition has actors who perform tasks to achieve goals in different situations in a scenario. Each task can be regarded as what needs to be done in the situation. The user tasks were analysed in terms of the answers to the questions, "Who should be responsible for the situation?", and "What should be known to act on the situation?". Systematic question-asking [154] was used to elaborate the propositional list. Questions were asked on each item of the propositional list. The why-questions were used to receive both intentional and causal information. The how-questions provided the procedural, causal, and enablement information. Answers to the why-and-how questions exposed some of the content of the problem space and generated materials for the work.

Some why-questions were asked, for example:

1. Why is knowledge of the environment required to understand the situation?
2. Why is chemical injection a method of preventing hydrate formation?
3. Why did the Engineer not use "reduce water dew point" method?
4. Why did the system integrate context and cues?
5. Why are past situations retrieved?

Some how-questions were also asked, for example:

1. How are context and cues integrated?
2. How does the system retrieve past situations?
3. How does the system "understand" situations?
4. How do engineers carry out chemical injection tasks?

5. How is a retrieved situation handled?

Answers to the why-questions revealed some important information that was not explicit in the scenarios. For example, an answer to a the question “Why is knowledge of the environment required to understand the situation?” reveals that the dynamic state of the environment affects the state of the gas and its flow in the pipelines. This information is not explicit in the scenario but helped in identifying other candidate objects. In addition, the question “why are past situations retrieved?” enabled the identification of a new candidate object, solution.

The answer to the question on “How are context and cues integrated?” gave the understanding that there must be perception before integration. This answer provided us a new candidate object, “perception”. Similarly, the answer to the question “How does systems retrieve past situations?” gave birth to a new candidate object, “assess similarity”. The answer to the how-question on “how does system understand situations?” gave the understanding that context and elements from the environment are fused and interpreted. This answer provided us a new candidate object, interpretation. Also, an answer to both “How does system understand situations?” and “How does systems retrieve past situations?” provided the understanding that engineers uses two sub-systems, SA system and KiCBR system. Similarly, the answer to the question “How is a retrieved situation handled?” gave birth to new candidate objects, reuse, modify, and preserve.

Claims Analysis

A claim in a scenario is the set of causal relations associated to a given feature (task) within that scenario [156]. Claims analysis provides design reasoning by documenting designer’s hypothesis about how these features may affect usability, provide a balanced view of the scenario and, extend the scenarios or elaborate the initial scenario specification [156]. Claims analysis in scenario-based design focuses on reasoning on the important design tradeoffs for a particular usability situation [156].

Claim analysis was executed for the following tasks:

1. Environment provide situations

2. Engineers have context
3. Engineers receive past solutions
4. Context and situation of the environment defines the solution
5. Engineers preserve solutions

Table 4.1: Claims analysis

FEATURE/ACTIVITY	**CLAIM (PROS & CONS)**
Engineers have context	+ It enables the system to recognise the context of interest. + It helps in easy access to solutions. - The attributes of a context may be more than one, and so care must be taken to enter all of them. - The data is structured in a particular format that must be strictly followed. - Keying in input data could be time consuming.
Environment provide situations	+ Provide the system with the condition of the environment. + Make regulation of the situation of the environment easy. - In a situation of multiple sensors, confusion may sometimes arise from different sets. - Faulty sensors will result in incorrect data on the condition of the environment.

Engineers receive past solution	+ The knowledge base is the common storage of knowledge and experiences of the analysts. Engineers having shared knowledge from a common knowledge base will result in good solutions which in turn will result in good performance. + Learn from experiences of past problems. + Have the same interpretation of situations, and ultimately, better comprehension and projection. - Updating Knowledge base could be tedious at times. - Organisations have to bear the cost of data integrity as knowledge base can contain faulty knowledge. - Engineers need regular training and experiences to maintain shared knowledge.
Context and situation of the environment defines the solution	+ It enables analysts have solutions to problems in a particular context. + It takes care of sorting. + The confusion that would have arisen from presenting all the solutions to every user is avoided. - Engineers works in teams and so they need knowledge of each other's task. - Solutions received are only a part of the complete solution to solve the problem.
Engineers preserve solutions	+ Knowledge base is updated to meet current trends. + Solutions are preserved for future use. - The engineers may lose track and subsequently have trouble retrieving incomplete entries.

4.2.4 Action Planning

In action planning, a conceptual framework is developed to solve a practical problem as described in scenarios. The problem in this scenario is hydrate formation in subsea gas pipelines. Collaborative work with domain experts in this complex environment revealed the need for a decision support system (DSS) that will reflect the changes in this environment. The primary aim of DSS is to support humans in the performance of tasks that involve decision making and the choice of appropriate actions [160] in order to overcome human limitations such as low vigilance or impaired cognitive capacities [140]. In a safety critical scenario such as hydrate formation, maintaining situation awareness of the environment is essential for effective decision-making. Machine learning techniques have been shown to be useful for dealing with uncertain knowledge in decision making [195]. For instance, case-based reasoning is effective where the general domain knowledge is difficult to extract and instead requires reasoning based on local knowledge or where it is difficult to formulate rules describing the situations [72]. CBR also helps in situations of incomplete domain data [151].

Case-based reasoning methodology presents a foundation for a new technology of building intelligent computer applications [162] but is much more useful when combined with domain models [96]. A domain model is an object model of the subsystem (problem domain). Domain analysis in addition to functional and non-functional analysis is needed to specify a system [133]. The idea behind domain modeling is that users, in describing what is of interest to them during requirements capture, are not interested in, and possibly are completely unaware of the rules that apply to the situation. The way to capture these rules is by dialogue with domain experts.

Khajotia et al. [89] used CBR to build a model for corrosion rate prediction, a flow assurance problem in oil and gas operation. The domain model in this approach was used at the CBR 'revise' stage. The CBR system was used for the automated adaptation of the solutions generated rather than the determination of solutions per se. Shokouhi et al. [175] used a domain model with CBR in a manner that the two concepts contribute to the generation of the solutions. The Shokouhi's knowledge intensive case-based reasoning

(KiCBR) system was developed to find solutions to problems associated with hole cleaning, a drilling problem in the oil and gas industry, based on either the CBR or the general domain knowledge model alone, or both of them combined.

The addition of context in CBR problem description in complex domains has shown to improve similarity assessment [200]. Contexts are not cases per se but are transformed into cases or can be used to identify cases. Zimmermann [200] used contexts just like cases in a case-based reasoning system in a mobile scenario. The user context was enclosed in cases to facilitate comparison of contexts, and provide solutions based on context-similarities.

Whilst these CBR approaches in context awareness address problems of incomplete data and domain specific problems, future problems that are situation-dependent cannot be anticipated due to lack of the facility to predict the state of the environment. Here, prior work is built on to present an approach that combines situation awareness, context awareness, case-based reasoning, and general domain knowledge in a decision support system. In combining these concepts the architecture (Figure 4.5) of this system provides the capability to handle uncertain knowledge and predict the state of the environment in order to solve specific domain problems.

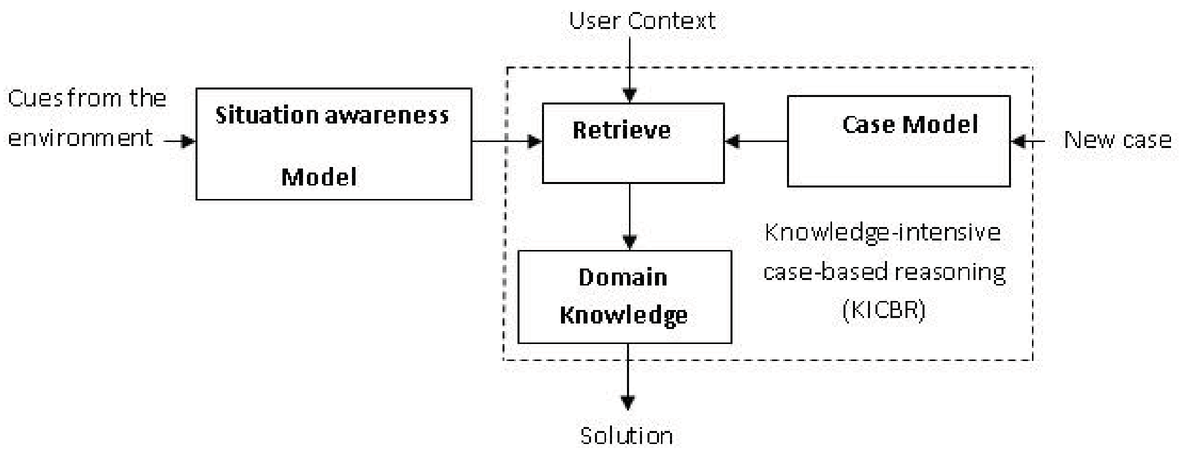

User Context
Cues from the environment
Situation awareness Model
Retrieve
Case Model
New case
Domain Knowledge
Knowledge-intensive case-based reasoning (KICBR)
Solution

Figure 4.5: Architectural Framework for Situation-aware Case-based Decision Support System (SACBDSS)

The user provides the system with information (context) that will enable the system provide support that will meet his specific needs. Environmental sensors provide the system with the state of the environment. With user context and environmental cues from sensors, the SA model provides situation awareness. With situation awareness from the SA model, the CBR system assesses similarity and provides solutions to the current situation based on past situations.

Situation Awareness Model

Situation awareness modelling is based on the environment and the user context. The system assesses the situation with the information about the environment. The assessment includes understanding the information, comparing it with an individual user context. Context modeling in this approach is based on seven attributes; goal, plan, identity, location, distance, expectation and time. The goals define the recommendations generated by the CBR system. Plan is the action plan to achieve the goal. Identity defines the entity type under consideration. Location is the geograpical position of the entity. Distance is the position of the entity relative to a point. Expectation is the preconception of the user, and time is the time to execute a plan.

The situation awareness model is the Endsley model comprising perception, comprehension, and projection. To foresee the future state of the environment, there must be an understanding of the current situation. The system keeps a finite history of the time space information on the state of the environment of the entities. To predict the future situation at a point in time, statistical inference is performed over these historical data. For example, the projection function in the SA implementation is the subsea temperature. The system kept a finite history on solar radiation, waves, and the resulting ocean depth temperatures, and statistical inference was performed on these historical data to predict the future temperature.

To reduce the operator's cognitive load, information is presented to users only if it is of relevance and significance to the them. Therefore, the situation awareness model presents for solution retrieval only the scenarios that are relevant and significant with respect to a user's context and situation. Depending on how the SA may affect the goal attainment status of the user, SA is used for retrieval of relevant solutions in one of the three ways: Situations that have no effect on goals are presented as NORMAL, a situation that may have an effect is presented as WARNING, and an extremely bad situation is presented as DANGER. A DANGER in one context may be a WARNING or a NORMAL in another context or vice versa.

The situation awareness model classifies the situation by a classification rule that assigns an appropriate message level to each recognised situation.

Case Model

In this knowledge intensive case-based reasoning (KiCBR) approach, a model-based method is implemented as a complementary tool in problem solving. A KiCBR system achieves its reasoning power through the set of previous cases combined with some other source of knowledge about a certain domain. The system is designed to solve problems based on either the case base or the SA module alone, or both of them in combination. To build the system, three knowledge models are needed:

- A taxonomy: extracting important terms from the environment.
- A causal model: building a model that describes causes and effects.
- A set of cases: concrete past problem solving experiences.

The approach for defining a taxonomical hierarchy depended on the practitioners' view of the domain, the tasks they perform and their goals for performing tasks. A systematic top-down view of the domain will lead to the use of a top-down approach. In this work, a combined top-down and bottom up approach is adopted to model both the user's context and the environment after they are defined by domain practitioners (see Section 4.3.3). A taxonomical and causal model for the hydrate domain was developed, in which all the entities are linked by binary relations. The causal model links the structured nodes together in a semantic network. The three main types of relation are: structural relations, e.g. has subclass; implication relations, e.g. causes; and associative relations, e.g. occurs in (gas type). To express the degree of coverage of the implication relations, quantifiers was added to a relation, i.e. Normal, Warning, and Danger. The "Normal" quantifier is the default if there is no hydrate threat.

Past cases in addition to domain knowledge forms the experience source in the system development. A case represents a hydrate forming situation. To make a case, all the relevant data and information are analyzed and a problematic situation is captured as a case. The case model is the library containing past situations and their unique characteristics (problem part) and their required actions with levels of impact (see appendix A) using the jcolibri's CBR tool [150]. In the participatory UCD requirements process, experts provided information on different problem situations, contexts, actions and levels of impact of actions on situations. This information was analysed and used for agile case-based development.

Cases in the case model have both context and SA features. Similarity assessment process has been defined that can run with or without the use of SA. A past case has a finite history of the time space information on the situation in a particular context. Predicting the problem solving method or the future of a new case is based on the assumption that every case obtains a history and a future and two cases with a similar history have a similar future [200]. The prediction of the problem solving method is preceded by the prediction of the situation of the environment, one of the attributes for the case's problem description.

The case's problem description is a seven dimensional vector of complex attributes from context, and the environment (Table 4.2)

Table 4.2: Context and Situation awareness Case Representation

Problem description	Attribute	Meaning	Example	Type of value
Context	Goal	To monitor hydrate	Prediction	String
	Identity	Single phase gas	1-single, 3-multi phase	Nominal
	Plan	Gas compositions	14.4 mol%CH4	Numerical
	Location	North sea	N	Char
	Distance	Ocean depth	300km frm WH	Nominal
	Time	Hour to execute plan	21:00GMT	Nominal
	Expectation	Flow rate	134MMscfd	Numerical
Situation awareness	Environment	Solar radiation	94/m SolarRad	Numerical
		Wind direction	Wndir NE	String
		Wind speed	80km/h WndSpd	Numerical
		Wellhead temperature	50C FWHT	Numerical
		Wellhead pressure	250Barg FWHP	Numerical

With user context the system retrieves from the situation awareness model information that is of relevance to the individual user. With the same context and the customized situation awareness, the system also retrieves, from the CBR library, scenarios that have happened in the past, in similar contexts, and for similar situations. The retrieval process for cases consists of pre-query processing and query-processing. The pre-query processing creates an index containing statistical information for all the cases in the case base, before queries are made. The query-processing uses information contained in the index to determine the case(s) most similar to the query. Local similarity and global similarity are calculated to form a basis for the system to retrieve similar case(s) from the case-base. Since the attributes are represented by nominal and numeric values, local similarity is calculated by using different equations. As for nominal attributes, local similarity is calculated based on single-valued nominal formula which is a common method as shown in Equation 4.1 and Equation 4.2.

$$sim(xi, yi) = \begin{cases} 1 \quad for \quad xi = yi \\ 0 \quad otherwise \end{cases} \tag{4.1}$$

$$sim(xi, yi) = \begin{cases} 1 \quad if \quad xi \leq yi \\ 1 - \frac{|xi-yi|}{|max(xi,yi)|} \quad otherwise \end{cases} \tag{4.2}$$

After a set of local similarities has been calculated for each known criteria-attribute pair, next the global similarity is calculated using nearest neighbour similarity equation as shown in Equation 4.3.

$$sim(xi, yi) = \frac{\sum_{i=1}^{n} wi \;\; sim\,(xi, yi)}{\sum_{i=1}^{n} wi} \tag{4.3}$$

where :

- n is the number of attributes of a case
- xi are the attributes of new case
- yi are the attributes in the case base

- sim(xi, yi) are the different attributes similarities between xi and yi
- wi are the weights, wi ϵ[0,1]

The CBR retrieve facility also carries out pre-processing of the context and environmental information, to decide whether the new input is significantly different from the current situation, in order to limit the number of executions of the case-based reasoning cycles [95]. A piece of information about the environment has different meanings and usages to different users depending on their individual context. Users shared context attributes except for plans and times. In the similarity assessment, the system put certain weight on the attributes. If the plan and time changes, the case-based reasoning system is able to detect the similarity value exceeding a certain threshold value, and this will trigger an event that will initiate the case-based reasoning cycle generating new action recommendations.

Domain Knowledge

Scenario-based analysis is used in developing a problem domain model [157]. Scenarios in Section 4.2 are used to develop the problem domain model. The problem description is the problem domain task description in sentences. Multiple scenarios are used to create the task descriptions. The use of multiple scenarios produces a generalised domain model. Multiple scenarios also aid the understanding of the problem domain, as it is refined beyond any single scenario that only provides partial information [156].

4.3 Agile UCD

The concept developed in Section 4.2 is used in this section to design a tool to solve a practice-based problem. Timeboxing was used in the agile, user-centred design development plan of study. A period of eight months was alloted to this second segment (agile UCD) of the three years action research-design process. The following explanation of the process summarises the overall development and evaluates the resultant outcomes.

4.3.1 Requirement Analysis

The first stage of the agile user-centred design is requirements analysis. This level provides abstract functional requirements based on a scenario and the framework in action planning. Requirement analysis describes the activity of defining the precise needs that the system must meet in terms of users and their environment with no reference to how the needs will be met by the system. The process of requirement work produces an abstract and partial descriptions of the user's task. Both "user" and "system" are actors in the Use Case diagram. Actors "represent the roles that people, other systems or devices take on when communicating with the particular Use Cases in the system" [15]. The environment provided an additional actor, "sensor".

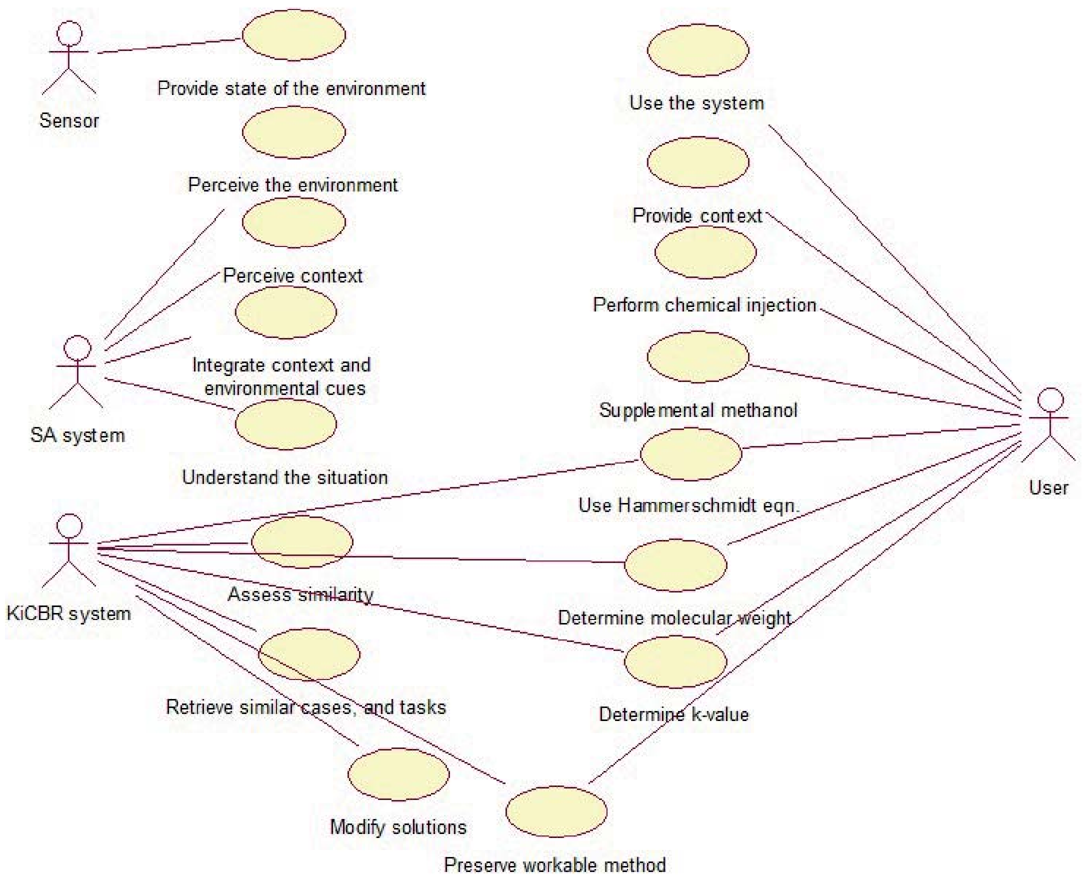


Figure 4.6: Use case modelling

From the Use Case model (Figure 4.6), sensor provides the state of the environment. The SA system perceives the environment and user's context, integrates the context and cues from the environment, and understands the situation in the domain. The KiCBR system matches the present situation with past situations to assess similarity, and then retrieves a similar past situations together with the sequence of tasks that were performed to address it. The user interacts with the system, provides the context, performs chemical injection, applies supplemental methanol.The user and the KiCBR system use Hammerschmidt equation at the aqueous phase, determine the molecular weight, determine the k-value and finally preserve workable solutions.

Each of the actors has a number of Use Cases but some of the Use Cases depend on other Use Cases, for example, the SA system's Use Case, "understand the situation" depends on the Use Case, "integrate context and cues" (Figure 4.7). Similarly, user's Use Case "apply methanol" is dependent on "perform chemical injection".

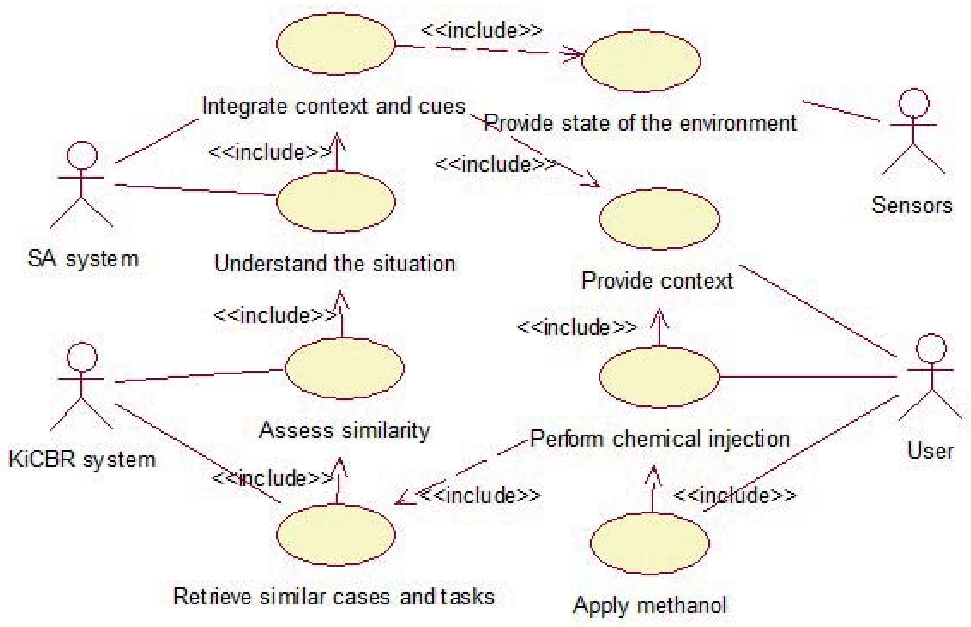


Figure 4.7: Use case relationships and interactions

In the way, the Use Case of one Actor can depend on the Use cases of other actors. For example, the SA system's "integrates context and cues" which is the inclusion Use Case to “understand the situation”, is dependent on the user’s “provide context” Use Case and the sensor’s “provide state of the environment” Use Case. The dependencies of the Use Cases shows the interrelationships between the user’s tasks and the SA system’s tasks.

The Use Case model in Figure 4.7 shows the interaction between Use Cases and the actors. Below, is the table (Table 4.3) of the requirements drawn from the Use Cases.

Table 4.3: Requirements List

No	**Requirements**	**Use case (s)**
2	To provide the system with contextual data	Provide context
4	To provide cues from the environment	Provide state of the environment
3	To recognise cues from the environment	Perceive the environment
5	To analyse the cues with respect to the context	Integrate context and environmental cues
6	To comprehend the current state of the environment	Understand the situation
7	To query the system to receive solutions to similar past problems	Use the system
8	To find out the most similar problem	Assess similarity
9	To retrieve from case base the most similar past problems with their solutions	Retrieve similar cases
10	To reuse chemical injection method to prevent hydrate formation	Perform chemical injection
11	To prevent hydrate formation by injecting methanol	Supplemental methanol
14	To save repaired solutions for future use	Preserve workable method

4.3.2 Design

The conceptual design (Figure 4.8) supported by an object-oriented Computer Aided Software Engineering (CASE) tool, Unified Modelling Language (UML) gives form to the functional requirements, and features from the users' view.

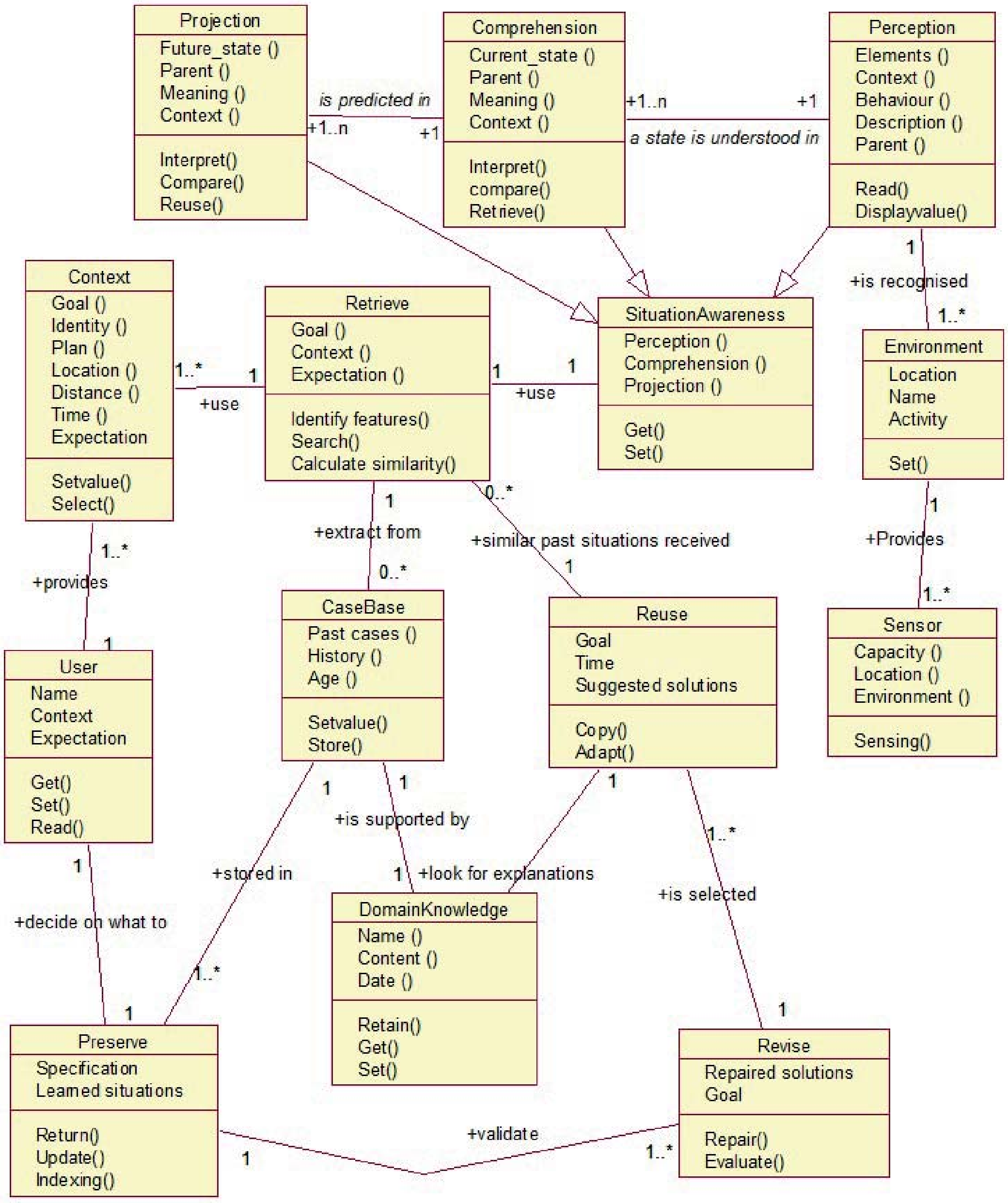


Figure 4.8: Class diagrams showing interaction standards

The design produced an information architecture, generic task model, and interaction standards development. The agile development of the classes are derived from the UCD requirements analysis. Each of the identified objects has a class, and the classes have associations that correspond to the links between objects. The class diagram in Figure 4.8 also shows the relationship between the classes, for example the class “situation awareness” is a generalization of the classes “perception”, “comprehension”, and “prediction”. One “perception” receives condition of the environment from one-to-many “sensors”. One can have one-to-many “comprehension” from one “perception”. Similarly, one “comprehension” can result in one-to-many “projection”. These identified UCD requirements are used for prototype development. The design draws on the Endsley’s situation awareness model which presents SA as a three-level mental representation: perception, comprehension, and projection. The three mental representations are specialised types of situation awareness. The SA system and the KiCBR system are two different actors in this approach. In the next approach (Section 6.4.1) the SA system and the KiCBR system are combined into one “application” actor.

4.3.3 Hydrate Situation Awareness Modeling (Prototype)

The third level of agile UCD is a proof of concept prototyping. The design implementation was in the flow assurance control domain to predict the formation of hydrate in sub-sea oil gas pipelines. To effectively predict the formation of hydrate, knowledge of the sea floor is necessary in addition to knowledge of the pipelines. The environment of sub-sea gas pipelines is the ocean water. The solar radiation that hits the surface layer of the ocean water is absorbed and mixed by waves and turbulence but decreases as it sinks downward (Figure 4.9).

The temperature decreases very rapidly and continues to fall slowly with increasing depth, making the deep ocean temperature to be between 0-3 degrees Celsius (32-37.5 degrees Fahrenheit) depending on the location and time. This situation increases the density and decreases the temperature of the seafloor until it freezes. Situation awareness of ocean depth identified through UCD is one of the features of the cases in the agile development.

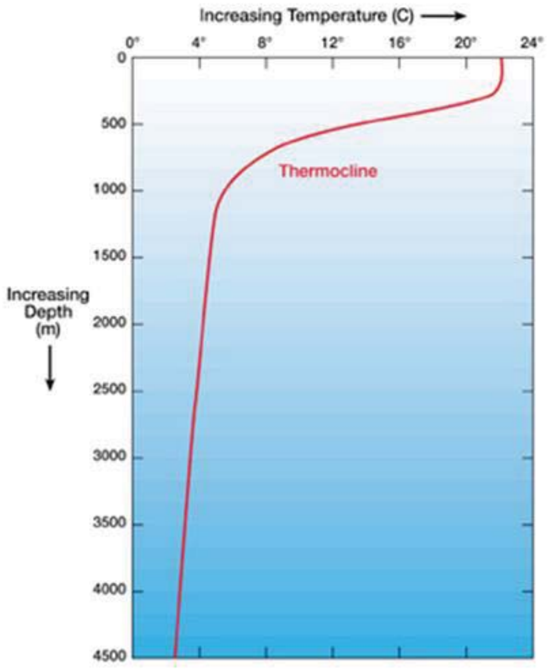


Figure 4.9: Sub-sea temperature profile

Situation awareness consists of three stages; perception, comprehension, and projection. PERCEPTION: The key elements or entities for perception from the environment recognised through UCD requirements are solar radiation and waves. The system is developed to sense the incident solar radiation, wind speed, and wind direction. Wave is determined by wind speed and wind direction. The context of users; phase type, composition, pressure, geographical location, distance below sea level, and time are also recognised. COMPREHENSION: Key parameters of the elements, such as the solar intensity, wave height, wave speed, and wave length, are identified in order to understand the current state of the ocean depth. Water waves store or dissipate energy and the wave height contains the wave's energy. A wave's energy is proportional to the square of its height (potential) e.g a 4m high wave has 4x4=16 times more energy than a 1m high wave. The wave length determines how deep the heat can sink. With the thermodynamics equation of motion for vertical mixing, the extent at which the radiation has mixed up at an identified depth was determined.

From requirements analysis, a particular temperature means different things to different users. The system is developed to provide situation awareness based on the meaning of the temperature with respect to an individual user context which comprised of identity, plan, location, distance, goal, and time. In this work, identity is single phase binary gas, distance is the depth, location is the north sea, the goal is to predict the formation of hydrate, the plans are the different compositions of the gas, time is the hour and day to execute the plans. Users share identity, distance, location, and goal but have different plans and times. The tasks of all the users involve single phase binary gas pipelines, all at the same location, with the same goal of predicting the formation of hydrate. Achieving this goal is mostly dependent on knowledge of the pressure and the temperature of the gas, and precise conditions in terms of pressure and temperature depend on the composition of the gas [176]. Therefore, the plans attribute which is the composition is not the same for all users. The times of carrying out the plans are also not the same.

PROJECTION: To predict the ocean floor temperature and its meanings at any particular point in time, statistical inference is performed on the database of the ocean water in a particular context to estimate the temperature and its implication in the near future. A problem is identified by reconciling the estimated ocean temperature with the hydrate formation temperature of the gases in the pipelines.

Each of the levels of situation awareness; perception, comprehension, and projection were iteratively developed. First, the perception module was developed and tested iteratively to ensure the system recognises context and environmental elements. The next stage of the agile development was, comprehension. The comprehension module was developed and tested iteratively to see if the current situation was understood. In a similar manner, there was an iterative development of projection to test how the future of a situation is predicted by the system.

Hydrate Case-Base Modeling

Requirements work was carried out with practitioners on the situation in the project problem domain to develop a case model. The study revealed that different situations have their individual characteristics (context) and requires different actions. Decision making in this domain is essentially to identify a problem situation and then solve the problem by performing some actions. Actions have different levels of impact on a problem situation. A case identified through UCD requirements analysis represents a hydrate forming condition at a specific gas composition in the agile CBR library. A condition has attributes such as location, identity, distance, composition, time, and situation. To make a case, all the relevant environmental and contextual data are analysed and a problematic condition is captured as a case in agile development.

The retrieved preventive and remediation measures are reused by the experts. Where these solutions do not provide all the answers to a hydrate formation threat, the solutions are adapted. Flow assurance experts are allowed to feel in control by enabling them to analyse the formation threats and manually adapt the solutions. The successful preventive and repair activities are stored for future use.

Hydrate Domain Modeling

The plan attribute comprises the volume and the pressure but without the temperature of the gas. An appropriate temperature corresponding to the volume and pressure is retrieved by the CBR. To use domain knowledge to provide explanations to the retrieved case, the system used published data [106] of pressure and volume of ten different pipelines P1-P10 to calculate the hydrate formation temperature. The pipelines had fixed volumes of gases with varying pressures. Published data was used at this stage of the work because access to oil and gas industry experts for data and knowledge acquisition was limited.

Figure 4.10 presents the hydrate formation temperatures at different pressures for P1-P6.

The goal is to determine hydrate formation situations starting out from three types of features: Direct observations (measurements), inferred parameters (values derived from observations), and interpreted events (particular concepts describing important states which require particular awareness or action). The features and situations are related through intermediate state concepts. Each relation is labeled. The situations and the case features are all represented as entities in a model, and the model-based reasoner works by finding paths from the entities representing case findings to the entities representing

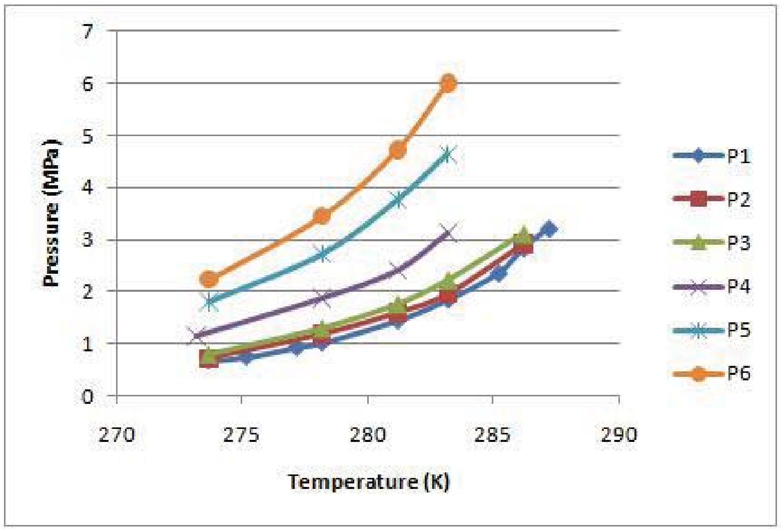


Figure 4.10: Phase envelope

situations. The goal of the model-based reasoner is to determine which situation or intermediate states are manifested by the features. Only some paths provide support for such a conclusion. In order to determine legal paths, plausible inheritance was used. This method is a generalization of normal subclass inheritance that allows inheritance of relationships over other relation types than "subclass of" relations. Plausible inheritance is governed by a set of rules declaring which relation-types can be inherited over which relation-types. Causal relationships in this work are transitive, and any relationship can be inherited over subclass of relationships. Sometimes there is more than one explanatory path from different finding to each target entity (situation). The total explanation strength for each target entity is determined. This calculated explanation strength becomes an indicator of being the possible situation. The system used the combined gas model to find the hydrate formation temperature of the gas and the results of these calculations were almost the same with retrieved gas temperatures. The strength of the indicating entities was decided by practitioners to reduce subjectiveness of these values.

4.3.4 Design Evaluations

The case base contains sixty seven cases of hydrate forming conditions. Using the linear nearest neigbour similarity framework, the case matching for the knowledge-intensive case-based reasoning (KICBR) alone, and the situation-aware knowledge-intensive case-base reasoning (SAKICBR) will be presented. The 10-fold cross-validation technique is used to evaluate the methods. In the first three test datasets, seven cases are taken out of the case base and matched against sixty one train cases, and in the remaining seven test datasets, six cases are taken out and matched against sixty one train cases. The KICBR method had a mean accuracy of 0.6 in all the ten different evaluations with number of matches ranging between 404 to 453. The accuracy of the SAKICBR method for the same number of evaluations was 0.7 with number of matches ranging between 490 to 512. The mean accuracy and the best matches are summarized in table 4.4 and table 4.5 respectively.

Table 4.4: Mean Accuracy

Evaluations	1	2	3	4	5	6	7	8	9	10
KICBR	0.64	0.65	0.64	0.62	0.60	0.60	0.66	0.64	0.67	0.61
SAKICBR	0.76	0.76	0.75	0.76	0.75	0.75	0.74	0.73	0.73	0.74

Table 4.5: Number of matches

Evaluations	1	2	3	4	5	6	7	8	9	10
KICBR	430	436	432	422	404	406	447	429	453	411
SAKICBR	512	510	503	510	504	504	500	493	490	501

In the matching results of test cases, the two methods retrieved the same best match for only a few test cases. In most of the retrievals as shown in Table 4.6, the best match for the unsolved cases are different. For example, having case 22 and case 40 as test cases, the retrieved cases as best matches for the two methods were case 30 and case 51 respectively. But in case 3, case 10, and case 62 as test cases, the KICBR retrieved case 45, case 40, and case 22 respectively as best matches. For the same test cases, the SAKICBR retrieved case 49, case 1, and case 18 respectively as best matches. Without the automated adaptation function this system requires additional human reasoning, increased participation of the engineers in evaluating the solutions and deciding if it can be reused. The engineers analysed the retrieved cases to decide on the solutions that are more relevant.

For instance, evaluating case 45 retrieved by the KICBR and case 49 retrieved by the SAKICBR as best matches for the test case 3 revealed that the risk of possible blowout in “direct heating”‘ recommended by the KICBR is high. The preventive measure recommended by the SAKICBR through case 49 is “antiagglomerant additive and/or presence of natural surfactants”. This measure allows hydrate crystals to form but size of the particles is limited and transported within the hydrocarbon phase as a suspension. According to experts, the measure requires minimal cost for separation at the processing plant, which is preferable compared to the danger prone direct heating method.

For the test case 62, engineers evaluated the solution of case 22 retrieved by the KICBR and the solution of case 18 retrieved by the SAKICBR. The solution of case 22 is “ammonia injection” and that of case 18 is “depressurization”. By expert analysis, the cost of chemical injection is huge and it is always considered as the last option. Careful analyses of the solutions by the experts revealed that the condition of case 62 is still within the scope that “depressurization”, a cost effective measure, can control.

Table 4.6: Similarity assessment

Test case	case 3	case 22	case 10	case 40	case 62
Best case by KICBR	case 45	case 30	case 40	case 51	case 22
Best case by SAKICBR	case 49	case 30	case 1	case 51	case 18

The effect of integrating situation awareness to case-based reasoning, particularly knowledge-intensive case-based reasoning was observed by changing not only the similarity but also the retrieved cases. The results shows improvement in both similarity assessment and problem solving prediction.

4.4 Business Change

Here a reflection (action taking) on this prototype was made by the team. The overall research process was evaluated and lessons related to how the prototype supports decision making were drawn. The researcher presented details of the work to the supervisory team. The work was evaluated and comments were made by every member of the team. The approach explored how situation awareness can be used to identify problems in participative case-based decision support process based on integration of elements of AR into agile user-centred design of DSS. The approach is an integration of two distinct models; a rule-based situation awareness model and a case-based problem solving model. Both models use the user context for reasoning but have separate general knowledge. Representing the knowledge of the physical world, domain information formed the basis of all three levels of situation awareness as well as case-based decision support. The situation awareness model monitored the dynamic changes in the domain and then used rules to understand the current situation. The situation awareness model narrows down its assessment of the current situation to the specific context in an individual query. Situation awareness formed at such context-based assessment produced well focused action retrievals in the CBR model. The case-based reasoning model uses context and the current situation to assess similarity and then retrieve similar past situations. Past situations in the situation library have both SA and context features. Evaluating the CBR perspective of the system, it was observed that queries with both SA and context features retrieved solutions with higher accuracy compared to queries with only context features.

To further evaluate the effectiveness of the system in supporting real-life situations, expert evaluation was conducted on retrieved cases from two systems as shown in Table 4.6. A comparative assessment on cases retrieved with queries having both SA and context features, and queries having only context features were carried out. Results showed improvement in problem solving prediction using queries with both features.

Lessons learned from the modelling approach are two fold: first, it attracts extra cost and time to build two separate models for the system. Secondly, the rule-based situation awareness model was resource constrained in assessing some situations. This was due to limited understanding of domain rules as a result of limited access to domain experts. Thirdly, running the system on a static interface was inappropriate for the context sensitive design.

Reflecting on how to address the above issues, it was realised that case-based reasoning apart from its effectiveness in problem solving is also a human cognitive process of situation assessment. It was therefore resolved that the next architecture would be a single CBR model for both situation awareness and problem solving. However, a framework for an adaptive user-interface for this system using situation-based task modeling was developed first before the next architecture. Task modeling was both a theoretical underpinning to the work, and also a practical way of modeling some of the information needed to provide decision support.

Chapter 5

Situation-Aware User Interface (SAUI)

5.1 Introduction

This chapter discusses a framework for the design of the user-interface of the architecture discussed in Chapter 4. In effect the research outlined in this chapter follows the research methodology in Figure 3.5 (Chapter 3). However, as the core diagnostic element had been undertaken previously, this chapter focuses on the agile UCD/Business change elements. The aim was to produce a situation-aware user interface (SAUI) with the ability to perceive and adapt to the user's context. The interface uses context-aware technologies to provide methods to help operators to perform their tasks in smart and intelligent ways that will reduce their cognitive loads. Context-aware computing is a trend whereby computing devices and systems serve their users beyond the traditional desktop in diverse environments [35]. Context-aware user interfaces allow systems to dynamically adapt to changes in a user's task domain by updating relevant information and service provision.

Non-adaptive user interfaces used by engineers in the hydrate domain does not solve the problem of drawing together the information required for situation-aware decision support systems in a way that minimises cognitive load. This approach discusses a framework for user interface design that exploits the environment and context from users to provide information tailored to the user's tasks in specific situations. The adaptation of the user interface to context and the presentation of a reusable sequence of tasks in the current situation reduces memory loads on operators. But it is simplistic to assume that adaptive user modelling will solve all human-computer interaction problems. A growing body of research has examined the characteristics of human-operator interaction with adaptive display and described the human performance costs such as trust, complacency, skill and performance degradation and decreased user acceptance that can occur in such interaction [135] [166]. Limbourg et al. [104] developed a language, UsiXML, to describe context-aware user interfaces. They provided tool support but concentrate on transformations between models in order to transform abstract descriptions to concrete ones, with no recognition of the fact that there could be unexpected changes of the UI when a context change occurs. Clerckx and Coninx [39] provided a mechanism to avoid these unexpected changes by incorporating context in user interface development using transformations between models [40] but the integration with the context model is done by the designer.

Mori et al. [121] describe the TERESA tool for designing user interfaces for mobile devices. Abstract models are used in order to deploy concrete user interfaces on several platforms. The approach is task centered implying that a lot of effort has been taken in visualizing the task model. A reconsideration of visual representation of task models was recently carried out by Patern'o and Zini [136]. Techniques like semantic zoom (hiding information outside the point of focus) and fish eye views (increasing the size of elements in focus) are introduced in order to improve the effectiveness of viewing and constructing task models.

To express the solution for identified user interface patterns in an abstract way, Campos and Nunes [27] provided a modelling tool for designing the UI with two levels of abstraction: a wisdom presentation model and canonical abstract prototypes. The tool applies the Wisdom model to UI patterns, easily expressing containment relationship between an object instance and other object instances within a particular environment, while the Canonical prototype is much closer to the concrete representation of the identified pattern. However, support for context-aware and multi-device user interfaces using the Canonical notation is not obvious and is therefore not considered here.

Calvary et al. [26] describe a development process to create context-sensitive user interfaces. The development process consists of four steps: creation of a task-oriented specification, creation of the abstract interface, creation of the concrete interface, and finally, the creation of the context-sensitive interactive system. The focus, however, is on a mechanism for context detection and how context information can be used to adapt the user interface, captured in three stages; recognizing the current situation, calculating the reaction, and executing the reaction.

Wu et al. [199] used Hierarchical Task Analysis (HTA) combined with scenario-based design to develop a user interface for context-aware indoor navigation applications. The approach used the HTA method to identify user, user-application, and application tasks. The work provided a framework of command interfaces for executing interaction between application tasks and user tasks. These command interfaces link users, user-applications, and application tasks. The work did not look at how individual differences influence usability. Also, no mention was made of the method of interaction between objects.

In a similar hybrid approach, Lewis and Rieman [102] combined HTA with requirement analysis by replacing the abstract and partial task elements of requirement analysis with real tasks from the task analysis. Kim et al. [90] and Liu et al. [105] combined metadata definitions with scenarios to build task knowledge structures in their work on sentence ends and interruption points in speech. Metadata provided meanings to words and sentence structures which makes features understandable for users.

Designers of user interfaces for situation-aware systems must know what changes from users or environments are related to the tasks that the users perform to achieve goals by drawing up a task model, using a notation which allows it to describe tasks for various types of situations [41]. In order to achieve a concrete user interface (UI), it is assumed that the designer adds abstract UI components to the task model. This information is platform-independent so that the rendering back-end can ultimately use this information to construct a concrete user interface for various platforms. The next step consists of creating the dialogue model. Designers can be supported by automatically generating the statuses and transitions between the various individual dialogues, so as to simplify the work of designers. The tool includes an algorithm to calculate the different dialogues and transitions between dialogues from the task specification. Designers can adjust, add or remove these transitions according to the results of a previous testing stage or the designers' experience. This way situation-aware user interface designers can manipulate transitions that would be triggered by context and situation changes.

In the case study (Hydrate formation prediction in subsea oil and gas pipelines) there will be three transition statuses, Normal, Warning, and Danger (Section 4.2.4). Normal situations represent situations where there is no problem in the domain. A warning situation represents a situation that is not normal but not yet in danger. A danger situation is a crisis situation that means there are already problems in the domain. The user interface executes reconfiguration after input variation so as to stay adapted to any of these situations that depict the current situation in the domain. Warning situations cause the presentation of preventive sequences of tasks while danger situations cause the presentation of remediation or repair sequences of task. HTA is used to describe tasks for these situations. HTA is supplemented with scenarios from Chapter 4 to stimulate design ideas. Each scenario has a setting that explicitly describes the starting state of the current and the future situations, and implicitly depicts the characters that take part in the situations in the scenario. Each scenario has actors who perform tasks to achieve goals in different situations. Requirements analysis from Chapter 4 is used to supplement the scenario-based HTA in representing interrelationships between tasks. Dialogues and transitions between dialogues are calculated from the task specifications.

5.2 Designing Situation-Aware Interfaces

This section provides a discussion of the design process (Figure 5.1). The design process supports the design of declarative abstract models, describing the situation-aware user interface.

The aggregate of the models can be serialized in order to export these models to a runtime. To test the result of these models, the corresponding user interface can be

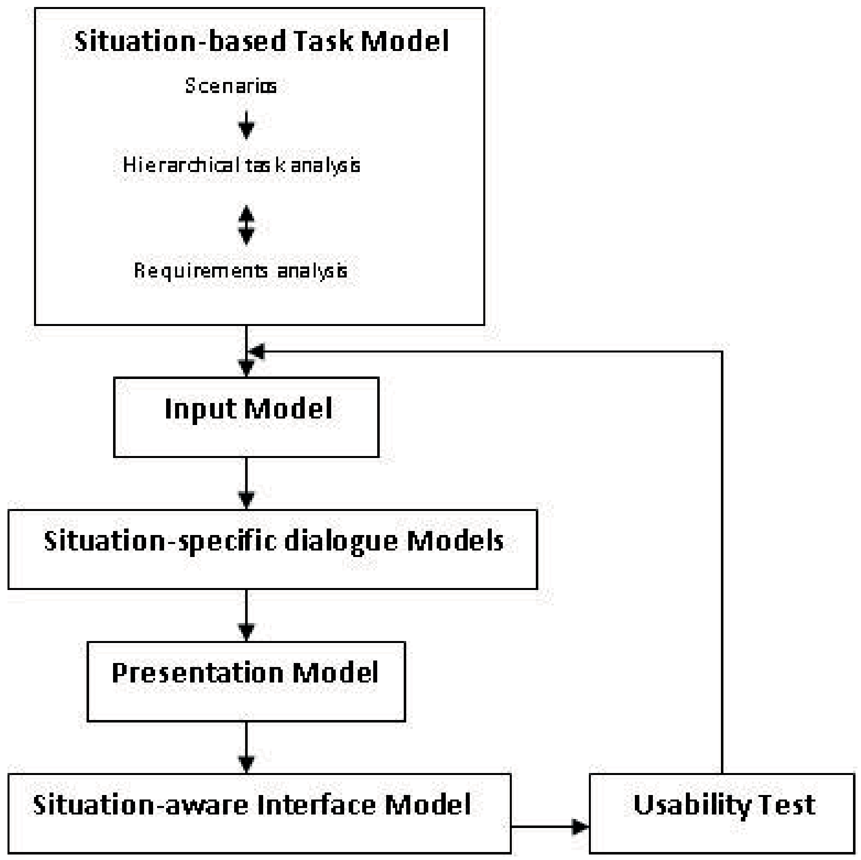

Situation-based Task Model
Scenarios
Hierarchical task analysis
Requirements analysis
Input Model
Situation-specific dialogue Models
Presentation Model
Situation-aware Interface Model
Usability Test

Figure 5.1: Situation-aware User Interface Design Process

generated in the shape of a prototype to check the usability of the system. Considering the prototype, some changes to the models in the design process can be applied to alter for instance the presentation of the user interface or how situation changes may affect the user interface.

Situation-based Task Model: First, a task model is specified describing the tasks users and application may encounter when interaction with the system is taking place. Because the goal is to develop situation-aware user interfaces that will adapt to an individual user context, task models are drawn for specific situations. In this way, the designer can describe different tasks for different situations.

Input Model: When the task model is specified, the designer has to denote what kind of input can influence the interaction, i.e. the tasks. This can be done by selecting objects for input gathering (Perception Objects or POs). These objects can be aggregated by the aggregation objects (AO) and interpreted by the interpretation objects (IO). The designer can do this by linking AOs to POs and selecting from a set of predefined interpretation rules how the input has to be interpreted. The IOs represent the interpreted information at the comprehension layer. When the input model is specified, the designer has to link the IOs to task model nodes (inter-model connection). In this way, the designer can denote which tasks can be performed in which situation.

Situation-Specific Dialogue Models: Next, the tool will automatically extract a dialogue model from the task model for each situation. Afterwards, inter-model connections are added automatically between states of the dialogue model and tasks of the task model that are enabled for each particular state. The dialogue model nodes (states) of the different dialogue models are linked to denote between which states situation changes may occur.

Presentation Model: To provide the interface model with information about how the interaction should be presented to the user, designers have to compose abstract user interface components, and link these to the relevant tasks for each presentation model node. The presentation model nodes can be structured hierarchically in order to group presentation components for layout purposes. The designer can choose from several abstract user interface components such as static, input, choice, navigation control, hierarchy, and custom widget. Finally the UI components can be grouped, and structured in a hierarchical structure.

Situation-Aware Interface Model: The aggregate of all the models results in a situation-aware interface model.

Usability evaluations: Usability tests are then carried out to test and improve usability of the graphical interface with the models.

Below each of these stages is explained further based on the specific design for this study. A period of twelve months was alloted to the SAUI agile, user-centred design development. Each of the stages were time boxed for two months.

5.3 Situation-Based Task Design (Requirements level)

The first step in the situation-aware user interface (SAUI) design process, just like every other interface design, is to draw up the task model, a hierarchic structure and a way of establishing temporal relationships between various (sub) tasks. Task analysis can help designers understand what needs to be accomplished by the user, the environment, and the system and break down the major task into the simplest component parts. Designers need to know what user tasks are necessary to operate the system and also need to know which part of user input can be transferred to the system task in order to increase the level of context awareness of the system. Hierarchical Task Analysis focuses on the way a task is decomposed into subtasks and the order and conditions where these are executed. HTA is represented as a hierarchy of tasks, subtasks and plans. It provides a brief picture of user tasks and basic functional specification of the proposed application. The top down structure of HTA ensures completeness and is easy to comprehend [29] but cannot adequately address human cognition and social issues, for example, emotion [35]. Such issues may be elicited from a scenario.

Scenarios according to Carroll [30], are examples of specific experience that exist to stimulate designers' creative imagination. Scenarios and claims are lightweight instruments that guide thought and support reasoning in the design process [30]. But scenarios also have their own downsides. According to Diaper [48] scenarios can lead to errors, as a scenario, or even a set of scenarios, do not explicitly guide a designer towards a correct model of the required system. Both scenarios and task analysis are criticised for omitting the explicit representation of communication between agents engaged in collaborative tasks and also not capturing the richness of interaction that occurs in the real world compared with other methods such as requirements analysis [97].

This chapter presents the design of a task model based on situations, using a hybrid technique of combining scenarios, HTA, and requirements analysis. Designers use the set of tasks that can be identified in the task specification as a basis for the different dialogues the user interface will need to complete its tasks.

5.3.1 Hierarchical Task Analysis

Hierarchical Task Analysis (HTA) is used to decompose complex tasks identified in scenario design (described in Chapter 4, Section 4.2.2) into subtasks and the order and conditions where these are executed. The output of HTA is represented diagramatically (Figure 5.2). HTA provides a brief picture of user tasks and basic functional specification of the proposed system. The break down of tasks enabled the research to stay focused on parts of the overall task without missing the picture of overall task activities. A top down structure ensures completeness and is easy to understand [174] [29]. Also, in the task-design mapping, HTA provides a good description of all task functions for mapping on to the system [29] (see appendix B).

Flow assurance engineers monitor and prevent hydrate damage through better understanding about their formation. They have knowledge of situations in the pipelines by reading data from sensors, identifying the wellhead temperature, pressure, and the composition of the gas. Knowledge of the state of the gas in the pipeline is received through its current flow rate, understood by the integration and interpretation of relevant data. When there are indications of possible hydrate formation resulting from a short fall in expected value, engineers use their experience to recall successful preventive methods applied in similar past situations.

From Figure 5.2, there are a number of methods to prevent hydrate formation. A retrieved solution could be the method of preventing free water in the gas stream by dehydrating the gas, or elevating the temperature to vaporize more water.

A solution could be the method of increasing the gas temperature above that needed for hydrate formation at the operating pressure, decrease the pressure below that is needed for hydrate formation at the operating temperature, or introduce chemical inhibitors (Figure 5.3). A preventive solution could also be to heat or insulate traced lines to keep the temperature of flowing gas above the hydrate formation temperature within a specific range of gas flow rates or use glycol dehydrators to remove water vapour from the gas stream. Chemical injection method, supplemental glycol/methanol injection is recommended for start-ups, scheduled shutdowns and low flow conditions. The method determines safe

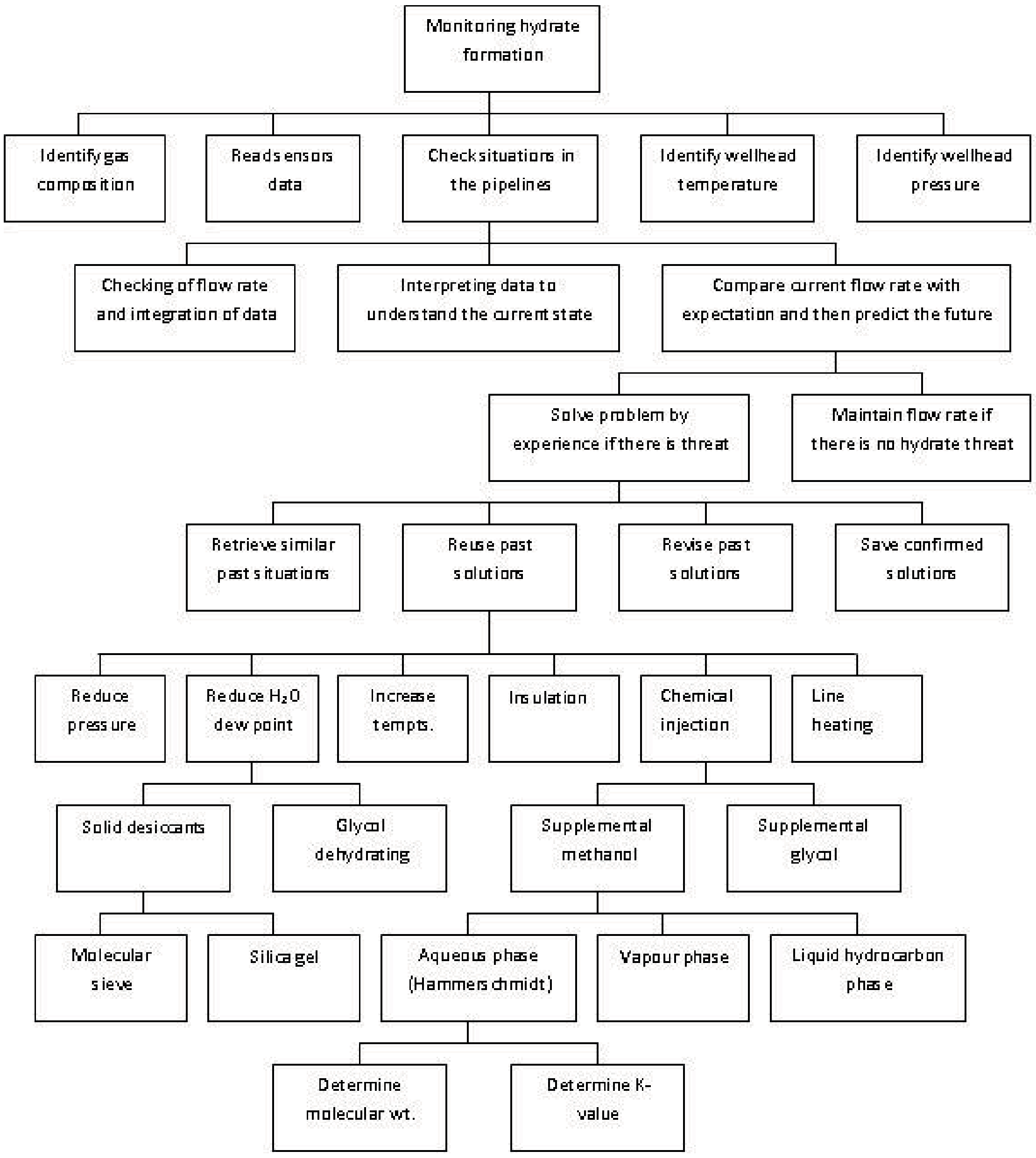

Monitoring hydrate formation
Identify gas composition
Read sensors data
Check situations in the pipelines
Identify wellhead temperature
Identify wellhead pressure
Checking of flow rate and integration of data
Interpreting data to understand the current state
Compare current flow rate with expectation and then predict the future
Solve problem by experience if there is threat
Maintain flow rate if there is no hydrate threat
Retrieve similar past situations
Reuse past solutions
Revise past solutions
Save confirmed solutions
Reduce pressure
Reduce H₂O dew point
Increase tempts.
Insulation
Chemical injection
Line heating
Solid desiccants
Glycol dehydrating
Supplemental methanol
Supplemental glycol
Molecular sieve
Silica gel
Aqueous phase (Hammerschmidt)
Vapour phase
Liquid hydrocarbon phase
Determine molecular wt.
Determine K-value

Figure 5.2: Situation-based Task Model

shut-in periods for lines to avoid pressure build-up. Even insulated lines will not prevent the gas stream from cooling down to the hydrate range within hours in shut-in conditions. A safe shut-in time can be determined from temperature data gathered during short shut-in periods. Other dehydration methods include molecular sieve, silica gel and calcium chloride towers. Methanol/glycol injection systems tie up free water and water vapour to prevent hydrate formation. The choice between methanol or glycol is one of economy.

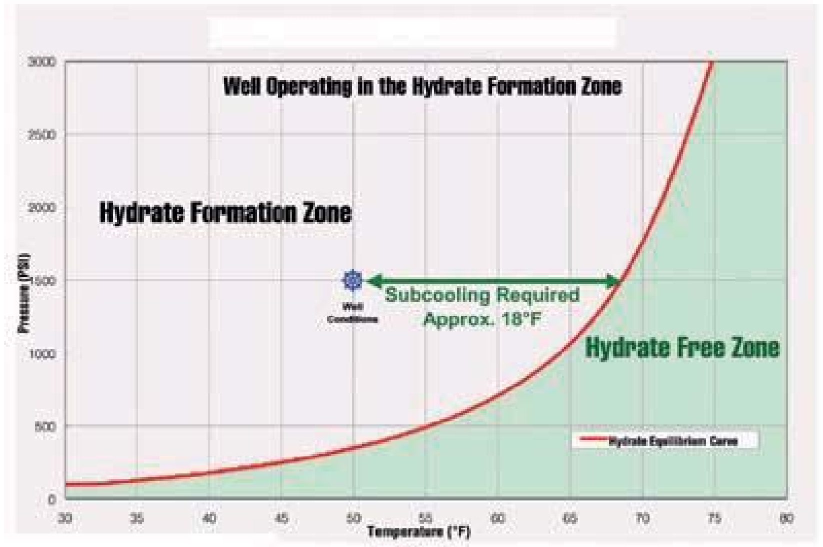


Figure 5.3: Phase envelope (Vaular et al, 2010)

Supplemental methanol is cheaper on a volume basis but practically cannot be recovered and regenerated. In gas-condensate and oil streams, much of the methanol injected is "lost" into the hydrocarbon liquids. The result is a very high injection rate compared to supplemental glycol. Glycol is more expensive but can be recovered and regenerated for reuse easily. Also, glycol is "attracted" to the water phase of the gas stream, so the injection rates are less than methanol. Engineers, by their experience, recall any or some of these hydrate methods that were applied to similar situations in the past to solve the current problem. The decomposition of the task of problem solving and learning from experience is shown in Figure 5.4.

To solve a new problem with the experience-based approach, the user queries the database. The system searches for similar past solutions by matching and comparing the current problem to old problems. Previous solutions are retrieved based on a correspondence of the new problem to some past problems. The system retrieves a set of similar cases and then evaluates the similarity between each case and the query. The most similar case(s) retrieved are presented to the user as possible scenarios for the current problem. If the solution retrieved is applicable to the problem, the user reuses the solution, and if it cannot be reused, the solution is adapted by modifying it. A modified solution is revised by testing it on practical problems. When the validity of the solution has been

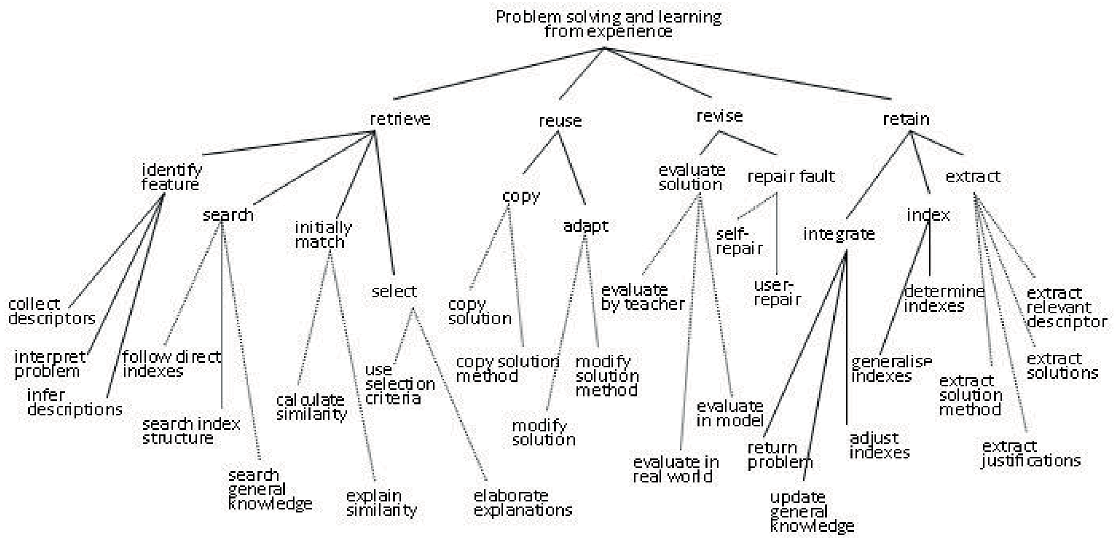


Figure 5.4: The Task-method decomposition of CBR (Aamodt and Plaza, 1994)

determined, the user retains it with the new problem as a new case in the database for future use. At this point, the case is considered to have been learned [73]. There are a number of sub-tasks under the retrieve, reuse, revise, and retain tasks (Figure 5.4) that all contribute to the effective performance of the CBR problem solving process.

5.3.2 Integrating HTA and Requirements Analysis

Data and metadata of representative tasks from HTA are mapped into the abstract model of requirements analysis in Chapter 4 to supplement requirements analysis using Use Cases of the Unified Modelling Language (UML). The “human user”, “sensors” and the “application” are actors in the Use Case diagram. Actors “represent the roles that people, other systems or devices take on when communicating with the particular Use Cases in the system” [15]. Use Cases are the different tasks performed by the human user, and the application [143].

5.4 Interface Design

The three interface design stages of the AR research-design process are input model, situation-specific dialogue models and the presentation model. For each of the stages, there was one month of design and one month of agile implementation.

5.4.1 Input Model

Input acquisition takes place at perception layer. Input comes to the application core from users' context, and sensors. These two types of inputs form the perception object (PO). Three levels of processing take place at the comprehension layer; input aggregation, input interpretation, and adaptation, represented as aggregation object (AO), interpretation object (IO), and adaptor respectively. The interpretation object (IO) ensures that mapping takes place from POs to AOs each time new services become available or when services disappear. The AOs then indicate to the IO the categories of POs from which they can use input information. The IO carries out the tasks of (1) Recalculating the mapping of AOs on POs: a service can be a supplier of input information. If this is the case, the IO can make use of this and treat the service as a perception object and link to comprehension which can make use of this input information (2) Detecting input changes: if a context and environmental change takes place, the IO will look at the adaptor in order to decide whether the change has a direct influence, so that an interdialogue transition has to be implemented (3) Invoking an interdialogue transition: the IO sends an event to the adaptor and tells it that an input change has taken place and that the interdialogue transition has to be implemented if it is evident from this interpreted information that a situation change has taken place. If a transition exists in the dialogue model to follow up this situation change, the adaptor will invoke the appropriate transition.

5.4.2 Situation-Specific Dialogue Models

A separate dialogue model is calculated automatically for these different types of situations and presented to the designer. The designer can then indicate between which statuses transitions are possible under the influence of situation changes. For example, in a three status situation, Normal, Warning, Danger, the designer can decide only to make a transition from Warning to Danger when the user interface is in the main menu status. This avoids the user interface adjusting if this is not desirable. IOs are linked to these transitions to make it clear what has to be taken into account in order to make the transition. An example of an IO is the Warning object. This object can indicate if the situation is in the Warning state, using POs and AOs. The adaptor changes the state of the UI caused by a change in context and the environment. The tool provides a design technique that can carry out prediction of possible changes in the UI following termination of a task, the implementation of a user action or a situation change. The design tool generates a UI which it derives from the tasks in the task specification. The specific presentation of the gathering of tasks is generated from a device-independent XML-based UI description which the designer can attach to the tasks.

5.4.3 Presentation Model

During the application runtime, the adaptor controls communication between UI abstract input information and the application core. The adaptor possesses information about the user's tasks and how these can be influenced by the situation. The IO encapsulates input information at such an abstract level that it only tells the adaptor that the situation change that has taken place is significant enough to adjust the status of the UI. The adaptor uses the dynamic dialogue model and the dynamic task model to decide when the UI has to be updated. These dynamic models are adjusted so that account can be taken of the current situation, if this influences the tasks the user wants to perform. The dynamic dialogue model consists of possible statuses of the UI. The difference is in the transitions that can occur. Here, a distinction is made between intra-dialogue and inter-dialogue transitions. An intra-dialogue transition is a transition between two states which is performed if the task described for the transition is performed by the user or the application. An inter-dialogue transition, by contrast, is a transition between two possible states of the UI, but can only be performed if a situation change has taken place which fulfills the conditions defined by the designer for the transition.

From the time the application is launched, the status of the UI and the application can be changed by the user, the application and the IO. The IO detects the current situation, supplied by the abstract interaction objects and then the adaptor is notified of the status in which the UI will have to be launched.

5.5 Situation-Aware User Interface Model (Prototype)

The aggregate of all the models in the agile development form a situation-aware user interface (SAUI) comprised of user interface component, situation awareness model, and the environmental sensing systems (Figure 5.5).

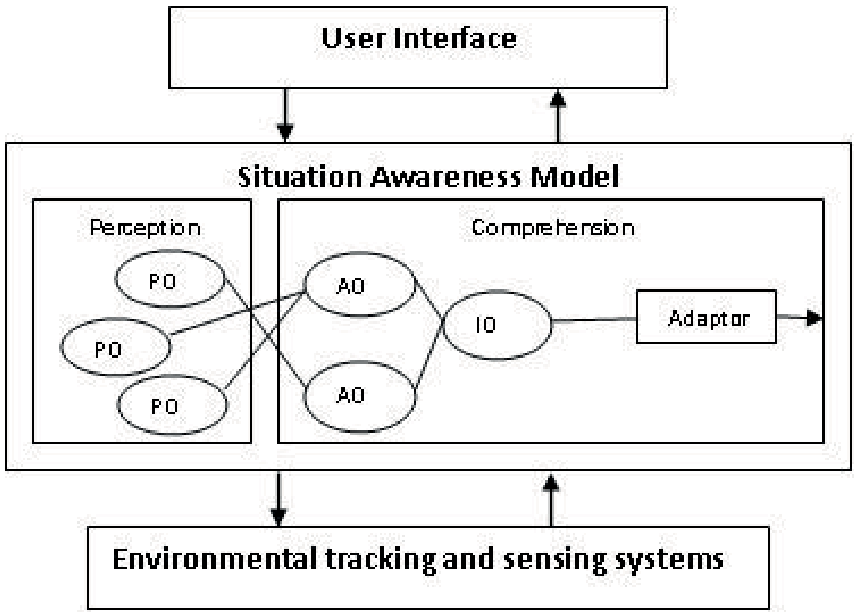
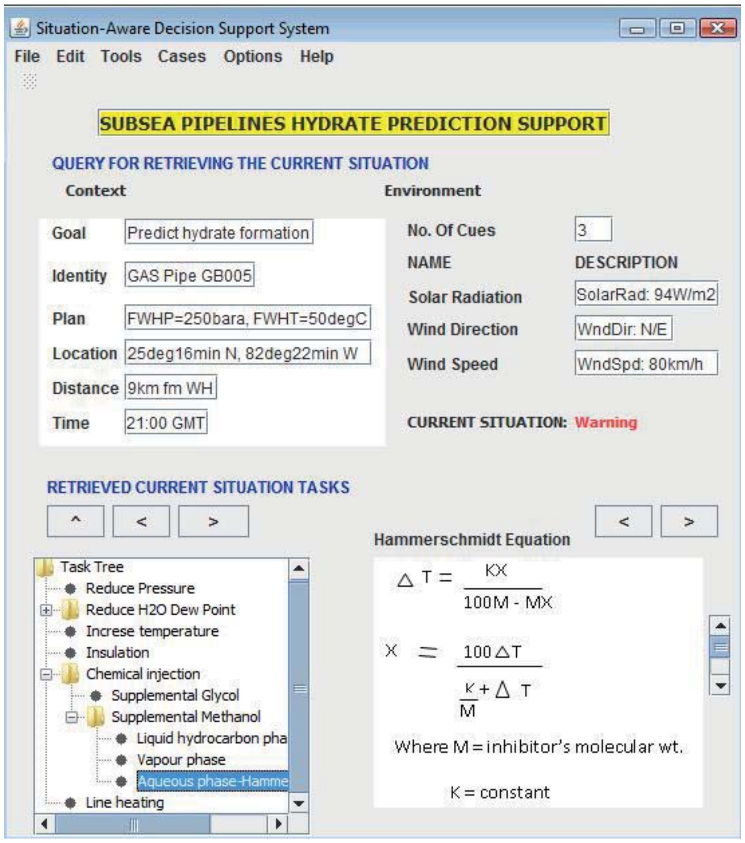

User Interface
Situation Awareness Model
Perception
PO
PO
PO
Comprehension
AO
AO
IO
Adaptor
Environmental tracking and sensing systems

Figure 5.5: Situation-Aware User Interface Design Model

The system is developed for a user to provide the system with context, and receive decision support from the system through the user interface. The system senses the environment through some sensing systems or sensors. The SA component is the application core. The SA model is the first (perception) and second (comprehension) layers of the Endsley SA model [54]. The system perceives cues from the environment to understand the current situation. Additionally, the system accepts or ignores the user's context, based on some logic. Ignoring context gives a static interface where SA will be the same on each retrieval, although different actions could be explored by the user with different information.

The system's acceptance of the user's context gives an adaptive display whose be-


Situation-Aware Decision Support System
File Edit Tools Cases Options Help
^ < >
< >

SUBSEA PIPELINES HYDRATE PREDICTION SUPPORT

QUERY FOR RETRIEVING THE CURRENT SITUATION

Context

Goal	Predict hydrate formation
Identity	GAS Pipe GB005
Plan	FWHP=250bara, FWHT=50degC
Location	25deg16min N, 82deg22min W
Distance	9km fm WH
Time	21:00 GMT

Environment

No. Of Cues 3

NAME	DESCRIPTION
Solar Radiation	SolarRad: 94W/m2
Wind Direction	WndDir: N/E
Wind Speed	WndSpd: 80km/h

CURRENT SITUATION: Warning

RETRIEVED CURRENT SITUATION TASKS

Task Tree
- Reduce Pressure
- Reduce H2O Dew Point
- Increse temperature
- Insulation
- Chemical injection
  - Supplemental Glycol
  - Supplemental Methanol
    - Liquid hydrocarbon pha
    - Vapour phase
    - Aqueous phase-Hamme
- Line heating

Hammerschmidt Equation

$$\Delta T = \frac{KX}{100M - MX}$$

$$X = \frac{100\Delta T}{\frac{K}{M} + \Delta T}$$

Where M = inhibitor's molecular wt.

K = constant

Figure 5.6: A screen shot of Situation-Aware User Interface

haviour will be customised to the user's specific need (see screen shot of the interface in Figure 5.6). In the adaptive mode at constant state of the environment, the system prompts the operator for context. Context input results in an automatic change in SA and action list. The system presents SA and a set of actions to be performed in the situations to the operator. The operator cannot generate any other option but to select the option provided by the computer to perform decision making and physically implementing the actions.

5.6 Usability Evaluations

In the experimental study, domain experts were asked to evaluate the accuracy of solutions generated by the system in terms of situation classification and action recommendation. The experts determined, given a specific state of the environment and the user context, if the system correctly classified the situation or recommends appropriate actions. In querying the system to trigger a problem situation, both implicit and explicit triggers are used. Implicit triggers are the user context, whereas explicit triggers are the environmental elements. Environmental elements are the parameters used in generating test scenarios, whereas user context specifies the search space of the system and therefore influences the subsequent evolution of the system. For example, the composition of the gas specifies the particular pipeline where the problem lies. Any variation in the context data may induce the system into making the necessary adjustment in similarity assessment and action recommendation. Each distinct context of the user marks a unique situation setting. A total of 62 situations with the same goal was collected.

5.6.1 Experimental Setup

Twenty flow assurance engineers (sixteen researchers and four practitioners) participated in the experiment. The subjects were nineteen males and one female with a mean age of 36. Twelve of the subjects were Europeans, four were from Africa, and four Asians. Ten of them were familiar with other flow assurance software including OLGA, MEPO and PIPEFLO. Subjects were assigned to four groups. Two independent variables were situation awareness, and action recommendation. Situation awareness had three classifications, Normal, Warning, Danger. Action recommendation is a factor to represent how accurately the system display actions based on a entered query.

5.6.2 Apparatus

Four HP with Intel Duo Core CPU E8400 (3.00GHZ) and 19-inch LCD monitors were used. SUMISCO, the program for Software Usability Measurement Inventory (SUMI) [91] was installed on the four computers. SUMI is a method of measuring software quality from the end user's perspective. A consent form, a pretest questionnaire, and a post-evaluation form were prepared (see appendix C).

A subject querying the system for the first time is presented with a static display. The static display had a specific number of options on the task list page that subjects could move forward and backward to see the options by clicking the “Prev” and the “Next” buttons. An adaptive display tailored to the specific need of the user is presented to him from his second interaction with the system. By clicking one of the options in the list, subjects could see a magnified description of the action in a situation. If the subjects are satisfied with the recommended actions, the subjects clicked the “Accept” from the “options” drop down menu.

5.6.3 Procedures

Prior to the experiment there was a general discussion on hydrate formation situations and appropriate preventive measures or actions in different situations. During the experiment, SA and recommended actions for three different queries were presented one by one using ten different contexts. Subjects were asked to query the system with a context. After identifying hydrate forming situations and solutions based on the context, they were asked to comment on 50 areas of satisfaction in relation to the system they were evaluating. This procedure involved administering questionnaire forms to the 20 subjects. The subjects had to decide whether they agreed, undecided, or disagree with each of the 50 items in relation to the UI. The subjects completed the inventory. The items were observed to relate to a number of different meaningful areas of user perception of usability.

5.6.4 Results

The program (SUMISCO) [91] analysed and transformed the data into Global and five other subscales namely: Efficiency, Affect, Helpfulness, Control, and Learnability.

Efficiency measures the degree to which users feel that the UI assists them in their work and is related to the concept of transparency. The Affect subscale measures the user’s general emotional reaction to the UI or Likability. Helpfulness measures the degree to which the UI is self-explanatory, as well as more specific things like the adequacy of help facilities and documentation. The Control subscale measures the extent to which the user feels in control of the system, as opposed to being controlled by the system, when carrying out the task. Finally, Learnability, measures the speed and facility with which the user

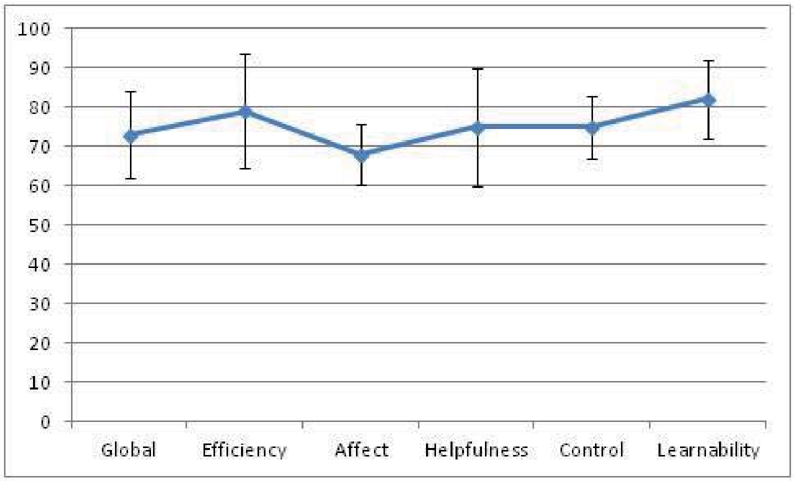


Figure 5.7: Profile Analysis

feels that they have been able to master the UI, or to learn how to use new features when necessary.

Figure 5.7 shows the median rating for each usability scale, with upper and lower confidence intervals. Feedback from users on efficiency rating indicated that users were satisfied with the UI adaptation to current situations and the presentation of reusable sequence of tasks for hydrate prevention especially chemical injection tasks and procedures. Users were also satisfied with fewer commands required for a given task performance, reduced number of clicks or keystrokes required to carry out tasks, and fewer options on the screen at one time. However, users were of the opinion that the HELP facility should have more information than already provided. The comments on the HELP facility resulted to the "Affect" low rating of 68%. The generally high usability rating (70%) of the UI implies low cognitive load on users. Instead of having to make extra effort to understand the UI, a user only needed to be focused on task performance (see Table 5.2). All the comments were noted and the response will be reflected in further work on UI.

5.6.5 Cognitive Load Reduction

Situations are classified into three levels of significance, namely Normal, Warning and Danger. Two performance measures for this function are defined, namely the accuracy of situation classification and the reduction in the cognitive load for the decision maker.

When the message level assigned by the system to a situation is different from that of the engineer, the situation classification is deemed as inappropriate. The system has a performance result of 70% in situation classification and action recommendation determined by the number of accurate predictions.

Table 5.1: The prediction accuracy in situation classification

Classifications	Number of predictions	Number of accurate predictions	Prediction accuracy (%)
Normal	28	19	67.8
Warning	23	16	69.6
Danger	11	8	72.7
Total	62	43	69.4

Context-based classification of situations contributes to the reduction of cognitive load on the engineers. Instead of having to attend to each and every situation, an engineer now only needs to pay attention to situations classified as Warning and Danger. The service information provided by the SA system ease individual's cognitive effort to analyse the situation. The system helps engineers choose actions with a reasonable amount of effort. To evaluate the system's performance in quantitative terms, the cognitive load reduction (CLR) index for a situation type c is defined formally as:

Rc = 1 - (Nc/N)

where Nc is the number of situation c requiring attention and N is the total number of recognised situations.

Table 5.2: The CLR indices for Normal, Warning and Danger situations

Total number of situations	Number of Normals	Number of Warnings	Number of Dangers
62	28	23	11
CLRI	55%	63%	82%

Number of situations requiring attention is the number of predictions in a particular classified situation that the operator has to access in order to make decision. The total number of recognised situations is the sum total of all recognised situations in a context presented to an operator. As shown in Table 5.2, the system has been effective in reducing the cognitive load of engineers by 55% for Normal, 63% for Warning and 82% for Danger. The progression in higher order represents the priority placed on safety critical situations.

5.7 Business Change

The core of the UI study is that the addition of context to input in situation-aware systems results to automatic change in SA and action list, making the UI adapt to the specific need of the individual operator. Also, that adaptation will take place only when the change in context and the environment is significant enough to result in transition between two possible statuses of the user interface. The adaptation of the interface to the current situation and the presentation of reusable tasks in the situation with reduced number of commands, clicks, and options reduced cognitive loads on operators and thereby facilitates interactions (see Table 5.2).

The work has also demonstrated a method of combining scenarios, HTA, and requirements analysis in task modelling. The approaches complement each other by using scenarios to stimulate and support reasoning in task analysis. Task analysis provides an integrated picture of tasks. Mapping real, complete and representative tasks of HTA to abstract and partial tasks of requirements analysis helps to ensure that all important users' tasks with their relationships and interactions are identified.

Lessons learned from the adaptive UI design was through comments from practitioners. They argued that an adaptive display is not an answer to problems in all situations. The argument was that a fully adaptive user interface reduces cognitive load. According these practitioners, humans tend to be less aware of changes in environmental or system states when those changes are under the control of another agent than when they make the changes themselves. Further work on UI design will be based on the study of Level of Adaptation (LOA) [53]. LOA is a level of task planning and performance interaction between an operator and the computer in a complex system with four systematic functions; monitoring, generating, selecting, and implementing [87].

The system architectural design in Chapter 4 presents a rule-based situation awareness model and a case-based problem solving model. With users' context the system retrieves from the situation awareness model information that is of relevance to the individual user. With the same context and the customized situation awareness, the system also retrieves, from the CBR library, scenarios that have happened in the past, in similar contexts, and for similar situations. The system has the advantage of critically analysing domain situations using domain rules and then solve problems in these situations based on past experiences. The architecture provides a very useful problem-solving technique in an ideal collaborative action research project. In this project, experts spent less time on providing the researcher with knowledge of the domain rules. The limited knowledge was insufficient for the formulation of reliable rules for statistical inference. Since SA is one of the features of the cases in the case-base it was observed that during evaluations the CBR model used context to assess the current situation even when some of these situations could not be understood by the rule-deficient situation model.

In the next chapter, the architecture is redesigned. The new architecture is a fusion of the CBR model and the SA model into an integrated case-based situation awareness model for SA based on experience rather than rules.

Chapter 6

Situation Awareness for both Problem Identification and Solving

6.1 Introduction

The first prototype in Chapter 4 presented a design which comprised two distinct parts: situation awareness (SA) and case-based reasoning (CBR). The SA part keeps a finite history of the time space information of the domain and uses rules to interpret cues from the environment with respect to an individual user's context, and then anticipates future situations by performing statistical inference over historical data. The CBR part seeks to accomplish a particular task with knowledge of the environment from the SA component. This chapter discusses situation awareness as a means of both problem identification and solving in context-aware case-based decision support systems. The approach is a fusion of the CBR model and the SA model into a case-based situation awareness (CBSA) model for situation awareness based on experience rather than rule, similarity assessment and problem solving prediction. In this prototype, experienced situations are stored as cases and experiences are retrieved by comparison with a current experience [73]. When retrieving the appropriate action, the operator has some expectations expressed through context that constrain the assessment. The expectations are monitored by the system while the assessment is being executed. Together with the probability of occurrence of each situation, the overall assessment value is given by the expectation value which indicates the applicability of the action and the future of the situation.

6.2 Cognitive Processes in Case-Based Situation Awareness

As defined in Chapter 2, situation awareness (SA) is a function of the operators' minds, their mental models of evolving task situations in complex, dynamic and high-risk environments. It is a state of awareness and understanding of the domain and other situation-specific factors affecting current and future goals, for the purpose of rapid and appropriate decision-making and effective performance. Representations of domain knowledge for situation awareness are stored in mental models or schema [54]. The level of SA that an operator has is dependent on the complexity of the available mental model. As an operator becomes more experienced with the domain, their mental model becomes more developed, which explains why experts are better at integrating multiple cues compared to novices [81]. The difference between the expert and the novice in their level of SA is experience-based reasoning. One of such reasoning methods is case-based reasoning (CBR). Case-based reasoning is a psychological theory of human cognition that addresses issues in memory, learning, planning, and problem solving [178]. The psychological assumptions of the case-based reasoning paradigm is that memory is predominantly episodic and so it is richly indexed such that experiences are related to each other in many complex and abstract ways. CBR builds on an understanding on how humans assess situations [162], supporting recognition-primed decision (RPD) framework proposed by Klein et al. [162]. The framework emphasises the role of experiences in human decision making processes during time critical situations. Klein et al pointed out that humans depend more on past experience rather than deliberate rational analysis of possible alternatives during time-critical decision making. In such critical situations, reasoning by humans is done by recalling memories guided by experiences of their immediate environment and factors that defines or characterised a particular situation.

In case-based situation awareness (CBSA) in Figure 6.1, an individual's ability to acquire and maintain situation awareness is a function of his/her cognitive abilities based on their context, which in turn is influenced by experience. An operator senses cues in the environment and uses them with context to understand the current situation by recalling similar past experiences.

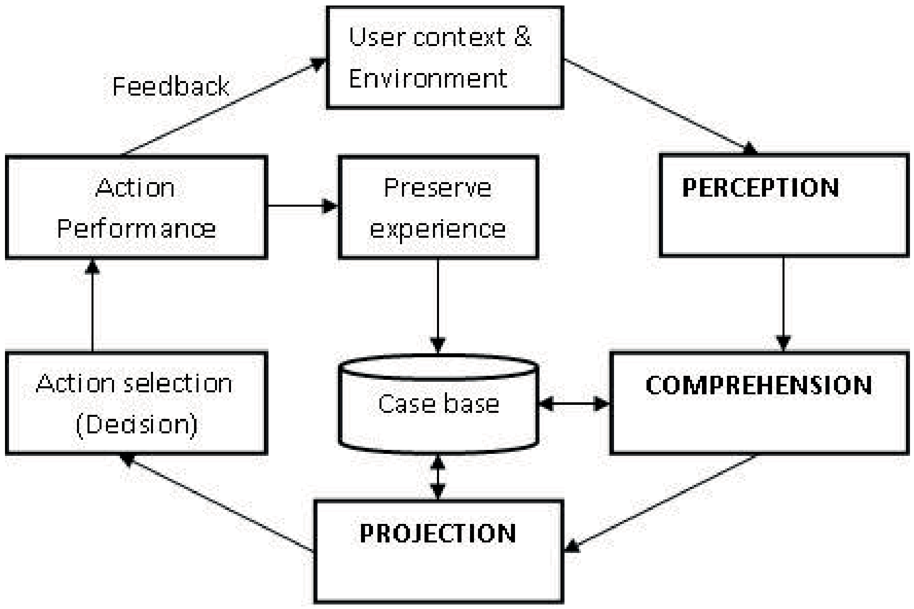


Figure 6.1: Proposed Case-based situation awareness cognitive framework

The operator's context includes goal, expectation, location, plan, identity, time and any specific needs. An operator also has the capability to predict (projection into the future) how a situation may evolve by recalling and assessing the evolution and solutions of similar past situations. Decision making and action performance are separate stages that proceed from SA but provide a feedback method to direct behaviour in order to attain a desired SA.

CBSA is a single model approach of assessing situations to form situation awareness and solve problems. The framework is the attempt of this project to integrate situation awareness, decision making and action performance into case-based reasoning. CBR is used in the three levels of situation awareness; perception, comprehension, and projection. It has no rule-based component unlike the previous approach in Chapter 4. The approach addresses the problem of lack of understanding of the domain rules. It relies solely on experiences. It takes time and commitment to provide and to understand rules. With little access to domain experts, experience can be received.

6.3 Scenarios

In this section, a scenario is discussed utilising the framework in section 6.2. Following the methodology in Chapter 3, scenario comprises problem description, diagnosis, and action planning.

6.3.1 Problem description

Following the Robertson model in Figure 3.4, natural language is used to describe a scenario in hydrate formation prediction in sub sea oil and gas pipelines as below:

An engineer monitoring the formation of hydrate plans to use a system that will understand the situation in the sub-sea oil and gas pipeline as well as provide decision support to identified situations. The system uses knowledge of past situations to understand the current situation in the pipeline by integrating the engineer's context with elements sensed from the sub-sea pipelines. The situation presented was a warning situation which consists of the corresponding tasks. Among the tasks are, reduce water dew point task, and chemical injection task. He decided to use chemical injection method to solve the problem in the absence of dehydrator. The available chemical for the engineer to use was methanol which is cheaper on a volume basis than glycol. Methanol is distributed in three phases; aqueous, vapour, and liquid. At the aqueous phase the engineer used the Hammerschmidt equation to determine the methanol molecular weight and k-value before injection.

The scenarios stakeholders are the same as in Chapter 4 (Figure 4.4). The difference between the present scenario and the first scenario (Section 4.2.2) is expressed in the first two sentences of both scenarios. In the present scenario, an engineer uses a single approach, a system that has the capability to understand the environment and also extract past experiences. The approach is in contrast with Section 4.2.2 where an engineer uses two different methods to carry out task performance; domain rules to understand the environment and experience-based system to extract past experiences.

6.3.2 Diagnosis

Similar to Chapter 4 diagnosis (Section 4.2.3) is the simplification of the problem description to a network of propositions. Answers to some “how questions” in Chapter 4 diagnosis revealed that “system” comprises of two sub-systems, “SA system” and “KiCBR system”. In the present approach, the KiCBR system and the SA system are combined to a single “Application”.

Proposition Analysis

The task scenario is simplified by partitioning the scenario into 12 propositions to identify candidate design objects as follows:

1. Engineer predicts the formation of hydrate
2. Engineer uses the application
3. Application senses the sea floor environment
4. Application integrates context and cues
5. Application understands the situation
6. Application provides decision support
7. Application uses past knowledge
8. Reduce water dew point is a task
9. Chemical injection is a task
10. Engineer carry out reduce water dew point task
11. Engineer carry out chemical injection task
12. Methanol distributes in three phases, aqueous, vapour, and liquid.

Candidate Objects Identification

Some objects identified are, engineer, hydrate formation, application, chemical injection, methanol, sea floor environment, use solution, modify solution, decision support, past knowledge, and preserve solution.

Scenario Exploration by Systematic Question-Asking

Systematic question-asking used to elaborate the propositional list are as follows:

Some why-questions were asked, for example:

1. Why is knowledge of the environment required to understand the situation?
2. Why is chemical injection a method of preventing hydrate formation?
3. Why did the Engineer not use the line heating method?
4. Why are solutions preserved?

Some how-questions were also asked, for example:

1. How are context and cues integrated?
2. How does the application retrieve past situations?
3. How does the application assess situations?
4. How does the application assess similarity?
5. How are solutions modified?
6. How are solutions preserved?
7. How can a problem be solved if no past is found?

6.3.3 Action Planning

Using the information from scenarios, action planning provides a conceptual framework to solve the identified problems. In Chapter 4, the SA model receives context and environment elements and then provides the CBR model with situation awareness. With situation awareness from the SA model the CBR system provides the operator with actions to perform. In this approach, the case-based situation awareness application uses user context and environmental cues from sensors to assess similarity and then provide situation awareness and problem solving methods (Figure 6.2).

Context defines the goal, expectation and the specific needs of the operator. State of the environment collects cues of the current situation and sends the information to the perception component. The perception component delivers data in terms of predefined objects from the context of users and the environment and converts this data into an abstraction in order to feed it into the reasoning process. Comprehension is the retrieval

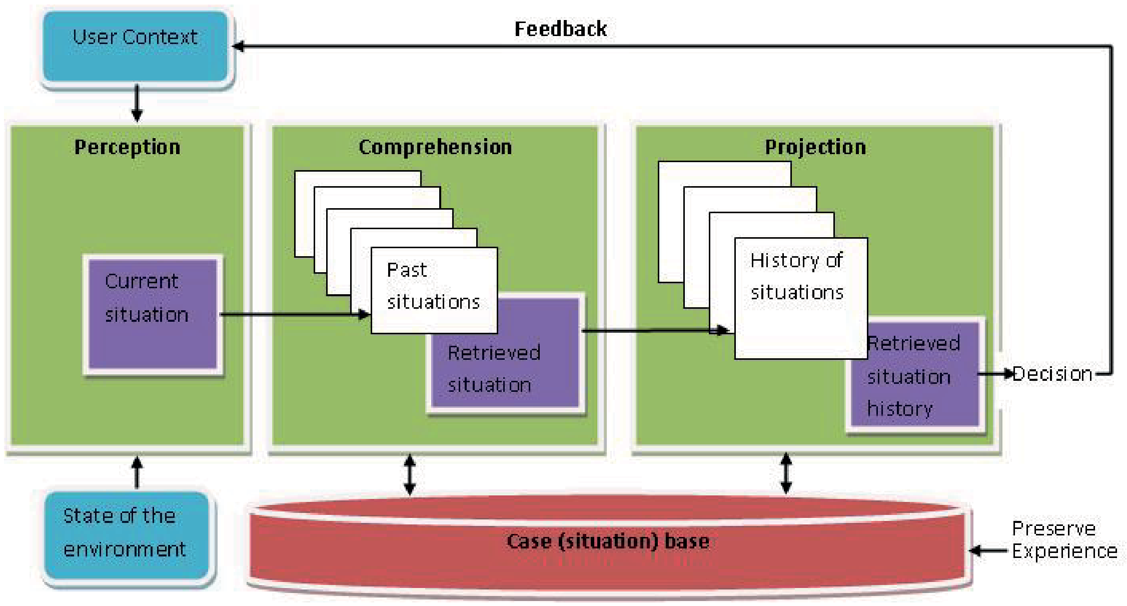


Figure 6.2: Proposed Case-Based Situation Awareness Model

component which extracts all situations of the case-base that have the highest similarity with the current situation. The Projection component is where existing knowledge is exploited by a reuse process to identify consequences of the current situation on future situations and present actions that are most suitable to avert the situations. The operator carries out decision making by selecting the appropriate action. The judgment of the operator on the future of a situation is also used to direct further perception of the system through feedback. The last component is the preserve phase which is applied after the selected action is implemented and found to be workable. A newly acquired experience is entered into the case-base in order to update the knowledge base.

Case-Base (Situation Library)

The case-base is the library containing past situations and their solutions (actions that were performed to correct the situations). Building the case-base is dependent on the definition of a case (situation). The main focus lies on an indexing of the situations in order to facilitate and speed-up the search for the most similar situations. The indexing scheme is based on links between different situations and facilitates the search for situations by walking through the case-base. Situations are linked in three different dimensions. In the first dimension, situations are organised hierarchically according to the specialisation of the situation. In the second dimension, situations at the same level of specialisation share a link representing their differences. And lastly, links denote temporal evolutions of situations. The hierarchical arrangement represents an order of situations from the most general to the most specific situation. Specialisation takes place because new instances of concepts or roles are added to the situation. In doing so, the link holds the reasons that led to the specialisation of that situation, i.e. it contains all the differences which make this situation a more specific situation. The edge between two situations holds the difference between these two situations. These links are used for generalisation of new situations. A situation is linked temporally with another, if its contents have changed significantly over time and is a direct evolution of the preceding situation. The applied action that will be appropriate for the temporally succeeding situation is stored together with the link. Due to the applicability of different actions, a situation can have multiple succeeding situations. Each applied action for a given situation is assigned multiple temporally succeeding situations, each succeeding situation together with its probability of occurrence.

Perception

The recognition of the status and the dynamics of relevant elements in the environment is the first stage in determining situation awareness.The elements are the entities. Entities are objects in the environment which have attributes. The entity class in this work is the general description of an object in the environment with relevant attributes. The data structure that encapsulates all the relevant information from the operator in the environment is the event. Events are problems defined by the environment and context. An event injection causes the case-based situation awareness model to reassess the relevant entities attributes and their relation with each other which eventually will result in a new situation awareness. This layer recognizes the state of the environment and the user context and then structures the information into a coherent shape.

Comprehension (Situation Retrieval)

The reason for situation retrieval is to extract the most similar past situation in the case-base relevant to the current situation. The best situations are searched by traversing the case-base recursively along the paths given by the hierarchical organization. Each directly linked specialization of a situation is called a child node of that situation. Starting with the top element, a child node is visited if it matches the current situations. This is done for all child nodes of a visited node. If a node has no matching child nodes, a best situation is found and added to the set of retrieved situations. Single situations from the case-base can be used multiple times because of different mappings of the individual situations. The situation in an experience case holds precondition cues from the environment which act like a pattern or schema for the system to recognise the current situation. These precondition cues mainly consist of some descriptions about the situation. Similarity assessment is conducted by matching the evidence cues of the current situation with the precondition cues. The key cues picked up by the system are used to form an evidence set. When the evidence cues match the precondition cues of an experience case, the situation will be retrieved.

Comprehension through the retrieve process as described above is by situation assessment, which enables one to compare different situations and find out which one is the most similar to the current situation or the other way round, which situations are most dissimilar.

Links are given which represent the temporal evolution of the situation. In order to select the most appropriate action, all possible evolutions of the situation are regarded by analysing the temporal successors of the retrieved situation. In order to detect dangerous situations at an early stage, the prediction considers multiple levels of successors. This can be done by combining the assessment along the prediction path using the minimum. The reason is because the uncertainty of the prediction increases with the length of the prediction path [192].

A single value p between 0 and 1 is calculated to express the assessment of the situation, where a higher value expresses a more similar situation. The assessment is based on the evaluation of different features, whereas a feature can only be fulfilled or not.

A function f(x) is defined as the expectation which assigns a value out of [0; 1] to each feature x. The overall situation assessment is defined by:

$$p = \min\{f(x) | x \in \text{fulfilled features}\}$$

The consequence of taking the minimum is that only the most important fulfilled feature counts and all less important features are ignored regardless of how many apply.

Projection (Reuse of Situations)

In case-based reasoning, the purpose of the reuse stage is to analyze existing knowledge contained in the retrieved cases and to generate a solution from this knowledge. In this work, case-based situation awareness, the goal is to select the appropriate action for a recognised situation. Different applicable actions are evaluated by the system and the most appropriate action is selected as the best suitable solution. Actions are organised to represent temporal relationships between different situations. When retrieving the appropriate action, the operator has some expectations expressed through context that constrain the assessment. The expectations are monitored while the assessment is being executed. If the expectations are not met, the specific action for the situation may not be executed and the future situation could be in danger. Every situation has a history. Links are given which represent the temporal evolution (history) of the situation. In order to select the most appropriate action when only one similar situation is extracted, all possible evolutions of the situation are regarded by analyzing the temporal successors of the retrieved situation. In order to detect unfavourable situations at an early stage, the prediction can consider multiple levels of successors. This can be done by combining the assessment along the prediction path using the minimum. The uncertainty of the prediction increases with the length of the prediction path. The assessment of the temporally succeeding situations is done by evaluating the different rates for each situation. Together with the probability of occurrence of each situation, the overall assessment value is given by the expectation value which indicates the applicability of the action. The higher the expectation value is, the better the action is applicable. This expectation value is calculated for each applicable action. The action with the highest overall value is selected.

If multiple situations are extracted, an assessment value for the related action of each situation is assigned. After that, the action that has the minimum value assigned across all the situations is selected.

Decision Making (Revise)

For good decision making in a given situation, an operator needs to have SA by assessing his current situation. With the SA, he can then consider the options of actions that can be performed and decide on the best options available. This process is facilitated by the operator by using the CBSA system to monitor situations in the domain and recommend possible courses of actions. The human operator then uses his expertise to choose from the options the action he considered most appropriate for the situation. In some circumstances, the actions are modified to suit the current situation. Decision making and action performance are the human operator's tasks carried out with the support of the computational situation awareness.

Preserving Experience

The last phase of case-based situation awareness is to preserve newly acquired experience and to provide it for future SA. This phase is executed later when an assessment of the applied action (solution) is known to be workable. In the reuse stage, different situations are extracted representing the situation most appropriate and the best suitable solution are generated based on these situations. In another iteration, the next set of situations with the best similarity is selected according to the situation retrieval phase. Based on this selection, it is now possible to check reflecting on the reuse stage, which temporally succeeding situations are really happening. Given this information, the probability of occurrence can now be updated for all these situations. If for a best situation none of the temporally succeeding situations did happen, a new situation must be created and integrated into the case base through the following steps: (1) Specify all objects of the current situation, that are part of the previous situation and the matching situation and all new appeared objects (2) Make the current situation conform to all these objects and their relations (3) The objects should be generalised to the level of the matching situation.

The newly created situation can then be integrated into the case-base. This implies adding the situation to the case-base and creating all links for this situation. Using key words from the user, the system automatically generalise situations in the case-base, if the branching factor of a situation is higher than a certain value. In that case, all situations at the same level as the added situation are taken into account. Generalisation is done by the system by identifying the similarities between the new situation and an arbitrary situation at the same level of specialisation. These two situations are replaced by this new generalised situation and added as child nodes.

This system only preserves positive solutions because it was primarily designed to identify classified situations. Reusing solutions as classified situations requires no adaptation. Recommending preventive actions to identified situations, which requires learning from both positive and negative outcomes in the adaptation process is an extension of the original design.

6.4 Agile UCD

The section presents the agile user-centred design process of the current approach, the third prototype. The four stages of agile UCD; requirements analysis, design, prototype, and design evaluation are alloted a total period of eight months. Each of the stages are time boxed for two months.

6.4.1 Requirement Analysis

Use Case modelling in Figure 6.3 produces abstract and partial descriptions of users task similar to the first and second prototypes. In this design, the actor “SA system” and “KiCBR system” (in section 4.3.2) are combined as one actor, “Application”.

Requirements drawn from the Use Cases are the same as represented in the earlier prototypes. The combination of two actors into a single actor does not change the re-

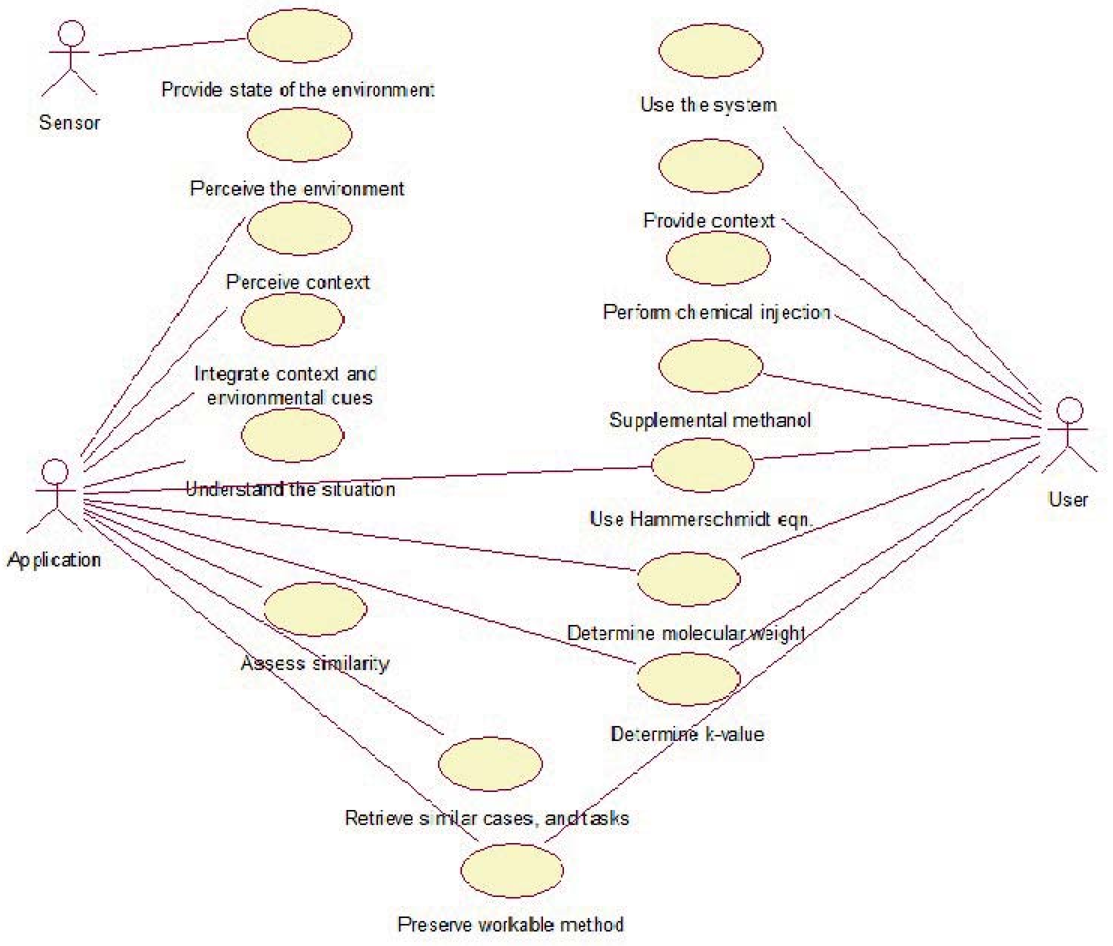


Figure 6.3: Use case modelling

quirements list.

6.4.2 Design

Although the changes in actors do not affect the requirements they do affect the interaction standards development (Figure 6.4).

The class "situation awareness" remains a generalization of the classes "perception", "comprehension", and "prediction" drawing on the Endsley's situation awareness model. Similar to the first prototype, one "perception" receives condition of the environment from one to many "sensors". One can have a one to many "comprehension" from one "perception".

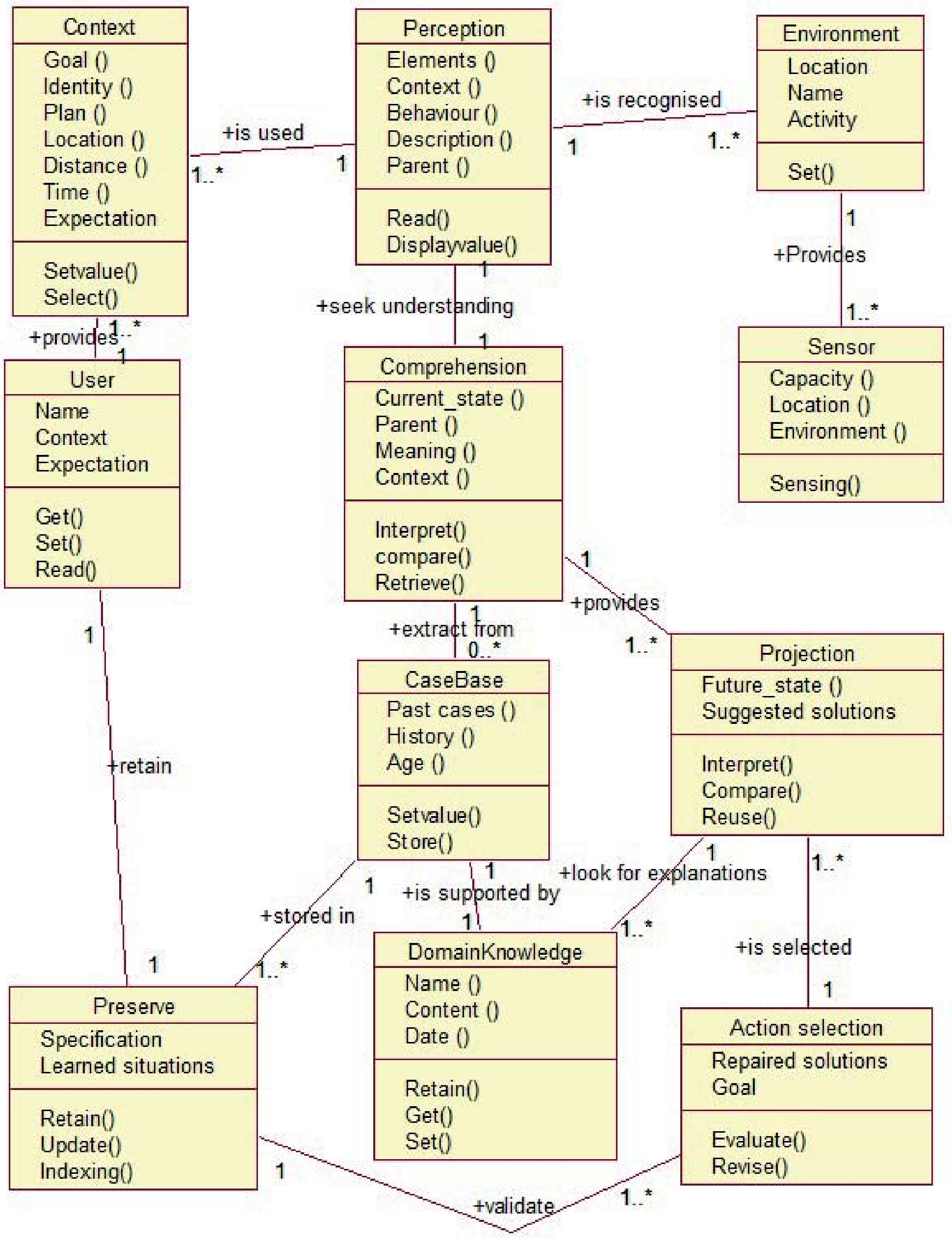


Figure 6.4: Class diagrams showing interaction standards

6.5 Prototype (Hydrate Situation Awareness Modelling in CBSA)

To understand the situation in sub-sea gas pipelines and effectively predict the formation of hydrate requires knowledge of the sea floor (the environment) in addition to knowledge of the pipelines (the domain). The environment of sub-sea gas pipelines is the ocean water. The solar radiation that hits the surface layer of the ocean water is absorbed and mixed by waves and turbulence but decreases as it sinks downward. The temperature decreases very rapidly and continue to fall slowly with increasing depth, making the deep ocean temperature to be between 0-3 degrees Celsius (32-37.5 degrees Fahrenheit) depending on the location and time. This situation increases the density and decreases the temperature of the sea floor until it freezes. Knowledge of the domain, such as the type of material the pipelines are made of, the composition of the gas flowing in the pipelines, the well head temperature, the pressure, flow rate, is also necessary.

6.5.1 Hydrate Perception in CBSA

The key elements or entities for perception from the environment are solar radiation, and waves. The system senses the incident solar radiation, wind speed, and wind direction. Wave is determined by wind speed and wind direction. The context of users; phase type, composition, pressure, geographical location, distance below sea level, and time are also recognized.

6.5.2 Hydrate Comprehension in CBSA

Situations in the case-base are the different hydrate forming conditions. Each of the gases has their different hydrate forming conditions. A particular condition comprise of temperature, pressure, phases, composition mol % in aqueous, liquid and hydrate. One of the hydrate forming conditions for methane is identified by the following attributes: temperature (2.5), pressure (3.31), phases (LA-H-V), composition mol % in aqueous (0.12), composition mol % in liquid (0.026), composition mol % in hydrate (0.14.2). The same attributes but different values holds for ethane, propane, isobutane, hydrogen sulfide, and carbon dioxide hydrate forming conditions. The hydrate forming conditions of the gases forms the context of the operators as operators works on different gases. An operator's context together with cues from the environment, such as the solar intensity, wave height, wave speed, and wave length, are used to retrieve past similar situations. A particular situation means different things to different users because of different hydrate forming conditions of the gases. With the same sea floor temperature, flow rate, wellhead temperature, wellhead pressure the system retrieve different past situations based on individual users context.

6.5.3 Hydrate Projection in CBSA

Projection is the reuse stage of case-based situation awareness. In the case study in hydrate prediction, the system analyses preventive actions contained in the experience library to generate workable actions. In each assessment to retrieve the appropriate action, we varied the expectations of users through varying context.

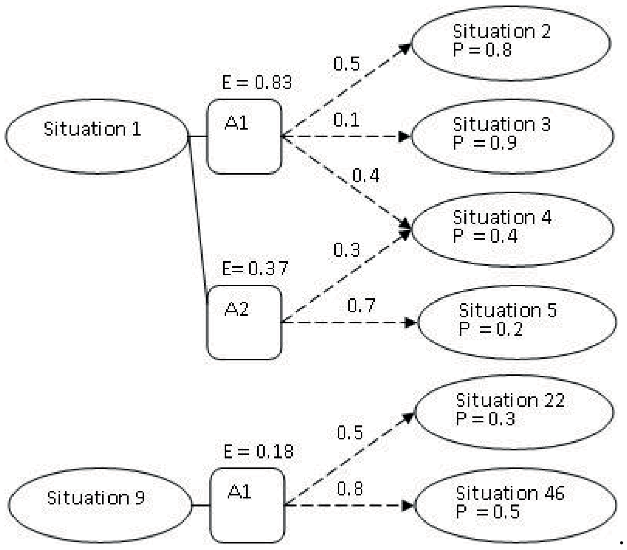


Figure 6.5: Selecting the best action by using different actions for temporal linkage of situations

In one context in Figure 6.5, the system extracted only situation 1 and found two possible actions that can be applied in the situation, A1 (methanol) and A2 (silica gel). The overall rating p of each situation, together with the probabilities of occurrence, gives the expectation value of 0.83 for action A1 and a value of 0.37 for action A2. Thus, action A1 (methanol) was selected and presented to the user.

In another context, the system extracted multiple situations. The expectation values for all the actions for all situations are calculated. This led to action A1 (methanol) for situation 1. But because action A1 can be applied in both situation 1 and situation 9, the overall minimum of that action is A1 from situation 9. The action A2 for situation 1 with an assessment value of 0.37 was extracted by the system.

6.5.4 Design Evaluations

The study investigated the number of accurate predictions of the system with past hydrate threatening situations from a North Sea oil and gas field. It also assessed the similarity between the system's recommendations and the expert solutions. Two different alternatives were evaluated: CBSA and SACBDS. Ten engineers working on flow assurance participated in the experiment. Two independent variables were system types and system accuracy. System types had two levels, CBSA and SACBDS. System accuracy is a factor to represent how accurately the system provide SA and actions based on a entered query. A query is entered by subjects into the two different systems to compare their predictions. To estimate how accurate these predictions are, the 10-fold cross-validation technique was used to evaluate the methods. The case-base contains fifty (50) past situations. Five test datasets are taken out of the case-base and matched against forty five train cases in each round of the evaluation. The result in table 6.1 provided a mean accuracy of 0.8 for CBSA, which implies that out of every 10 predictions eight are correct (see accuracy pattern in 6.6).

Table 6.1: Mean Accuracy

Evaluations	1	2	3	4	5
CBSA	0.81	0.80	0.82	0.78	0.80
SACBDS	0.66	0.61	0.65	0.69	0.82

Evaluations	6	7	8	9	10
CBSA	0.87	0.70	0.84	0.88	0.87
SACBDS	0.62	0.64	0.67	0.88	0.79

In the same experiment, the previous architecture (SACBDS) was evaluated. SACBDS had a mean accuracy of 0.6, signifying six correct predictions out of every ten predictions. It was observed that the low level of accuracy of the SACBDS is as result of the structure of the historical data used in the design. Most of the data on past situations have incomplete attributes which were difficult for the rule-based situation model of SACBDS to interpret.

In the matching results of test cases as shown in Table 6.2, the two methods retrieved

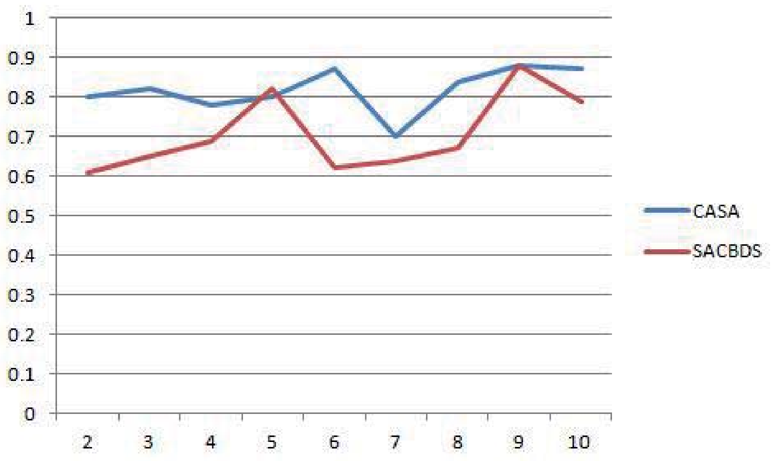


Figure 6.6: Accuracy pattern

Table 6.2: Similarity Assessment

Test Case	10	6	47	18	34	25
CBSA's Best Case	28	44	12	6	48	No case found
SACBDS's Best Case	21	44	34	19	26	Warning

the same best match (case 44) using case 6 as a test case. In most of the retrievals, the best match for the unsolved cases are different. For example, in case 10, case 47, case 18, and case 34 as test cases, the CBSA retrieved case 28, case 12, case 6 and case 48 respectively as best matches. For the same test cases, the SACBDS retrieved case 21, case 34, case19 and case 26 respectively as best matches. Using case 25 as a test case, the CBSA found no situation in the case-base that is similar to 25. Also, the SACBDS did not find any similar situation to case 25 but however, use rules to understand the situation as a Warning situation. The SACBDS recommended actions to be carried out avert the situation.

The "revise" stage is a manual adaptation level which requires additional human reasoning, increased participation of the engineers in evaluating the recommended actions. The engineers analysed the retrieved cases to decide on the actions that are more relevant.

For instance, evaluating case 21 retrieved by the SACBDS and case 28 retrieved by the CBSA as best matches for the test case 10 revealed that the two cases, 21 and 28 recommended chemical injection as preventive actions. However, two different types of chemicals are recommended by the two methods. Case 21 is supplemental methanol while case 28 supplemental glycol. By expert analysis, injected methanol concentration is normally greater than 98 wt%, while the typical glycol injected into pipelines often falls in the range 67-75 wt% making glycol to have advantage over methanol. Similar advantages were found in case 6 and case 48 over case 19 and case 26 using case 18 and case 34 respectively as test cases.

However, in using case 47 as a test case, the action recommended by the SACBDS had advantage over the one recommended by the CBSA. Engineers evaluated the action of case 12 retrieved by the CBSA and the action of case 34 retrieved by the SACBDS. The action of case 12 is "silica gel" and that of case 34 is "molecular sieves". In analysing these two actions, experts said in recent years molecular sieves have gained popularity over silica gel due to its advantages of providing extremely low dew points and high absorption of water.

The limitation of this SA modelling approach is that it relies solely on past situations in a domain. The effectiveness of the system is dependent on the availability and the number of past situations in its situation library. In some complex and safety-critical environments, operators may not be able to document all their experiences. The system will provide poor SA in an environment where few past situations are preserved, and cannot be implemented where there is none.

6.6 Business Change

The evaluation of the case-based reasoning process here is not an evaluation of case-based reasoning as a general method of reasoning, rather it is an evaluation of case-based reasoning as a method for achieving situation awareness. In this sense the most important metric for evaluating the use of case-based reasoning here, is not its cognitive validity, which has been briefly discussed in Chapter 2, but rather the assessment of situations to achieve situation awareness. Situation awareness and action recommendation can be evaluated in four ways [11]; (1) absolute accuracy, as determined by a domain expert (2) the plausibility of incorrect SA, also determined by a domain expert (3) situation awareness can be compared to other approaches; and (4) situation awareness can be compared to that of a human expert. Situation awareness can in theory be evaluated following these four methods. However, the fourth method was not really possible because the user interface of this architecture to support human cognitive situation awareness is not yet developed. The third method was possible because of the first algorithm that fits this data set implemented in Chapter 4. The use of case-based reasoning here as a method of achieving situation awareness is rooted, not only in the desire to find the best suited algorithm for a limited data set, but also in the fact that the theory on human cognition that underlies case-based reasoning is exactly one of situation assessment.

Due to commercial confidentiality when working in the oil and gas domains, all data gathered must be anonymous, thus the possibility of going back to the observed subject to ask for a re-assessment was impossible. As the system was tested exclusively on known and already classified data, it is reasonable to assume that, more or less, anybody with access to the raw data can function as a domain expert for at least the absolute accuracy method of evaluation.

The work has provided a framework and an architecture for building case-based situation awareness systems and also, how the feature of expectations can be incorporated into users context to enable the system meet the specific need of individual operators. The approach produced a paper presented at the 2012 IEEE conference on situation awareness and decision support [129].

The approach has shown that experience is a critical element for a human operator to have good situation awareness (SA) [88]. Based on this premise an experience-based SA has been developed using case-based reasoning (CBR). The case-based situation awareness system, a computational SA approach, provides a higher number of accurate predictions than the rule-based SA model.

Chapter 7

Critical Analysis of the Approach

This chapter is an assessment of the approach adopted in this research project work. The collaborative work at the design workshops and the use of the CBR technologies provided interesting experiences. A close relationship with practitioners was established and it was discovered that it is unproductive to view the researcher/client agreement (RCA) as a static initial document and as such, the need for continuous renegotiation of the dynamic relationship with the practitioners as the design evolves. The experience of managing data, action cycles, network, and co-authoring with domain practitioners are also discussed.

7.1 Design Workshops

There was an agreement to investigate and design a hydrate prediction system with practitioners working on flow assurance control in the North Sea using the above ethical considerations. After the agreement was made, there were a number of design workshops. At these workshops, people from the oil and gas industry, as well as academic hydrate specialists, participated.

The design workshops were typically planned around a particular aspect of the hydrate investigation, such as the single phase gas pipelines, and the design of the user interface of the situation-aware hydrate formation system. The topics were selected by the researcher and some senior flow assurance specialists, and the relevant practitioners were invited to debate this issue.

Domain knowledge was translated by practitioners for the researcher to understand. An example of knowledge translation during the workshops concerned the nature and order of sensors data as they appeared on electronic systems. The researcher was shown how these data are interpreted and sorted according to urgency. In this case, the role of the researcher was to support the engineers in their interpretation of how the electronic system should be designed by suggesting theoretical explanations (e.g., the distinction between boundary factors and contextual contingencies [19]) for why the differences in work practices between flow assurance and well engineering were important design considerations.

This form of domain knowledge translation, which took place during the design meetings, was conceptualized as transforming the empirical observations of current work practices and representing them through articulation. As time went by, the researcher emerged in a role at the meetings as advocate for the flow assurance engineers by pointing toward important Human Computer Interaction issues about the design that might affect crucial work practices in flow assurance. These issues concerned the order of data entry fields in the user interface, differences between the user contextual data and environmental elements, and the flows embedded in the generic design of the flow assurance control system. It was not the role of the researcher to impose on the practitioners' decisions about the design, but rather to portray possible conflicts between the system design and the existing work practices. Subsequently, the engineers would decide whether they wanted to change their existing work practices, reconfigure the application, or define a workaround to accommodate the conflict.

The design meetings were a general knowledge translation workshop, where negotiations were prompted while volunteering theoretical explanations of concrete issues pertaining to the design while the practitioners articulated their responses. This domain knowledge translation workshop prompted a very productive design environment where users' perspectives were negotiated as a part of the design. In this way, the theoretical findings were co-constructed together with the practitioners, making the domain knowledge translation happen in a timely manner and directly impacting the user-centred design.

7.2 Design Experience

By focusing on user-centred design, understanding of the user is developed, understanding of why the system is being developed and who will be using the system. As the UCD process ensures an understanding of the users, the agile development model ensures that there was iterative work, enable faster development of a functional prototype, which are more easily communicated and tested, thus providing better input for the next iteration. The two methods complement each other. The agile iterative development is more appropriate for the user-centred design process, as evaluation in agile development is done many times during the project to give room for a change in direction if necessary. Where there is need for a change, the constant evaluations enable us to redesign at an early stage, saving time and resources. This approach requires engagement from everyone involved in this project; ourselves, and the practitioners. This highly collaborative way of working ensures that any problems that would have arisen are noticed at an early stage. This iterative process is strenuous but it yields the desire results. One of the results is the production of a user interface that will now help practitioners in their daily tasks.

The main function of the situation-aware user interface (SAUI) is to provide an adaptive checklist of the task to be performed. The adaptive task checklist helps each operator navigate through computer-presented preventive information by suggesting an optimal path and indicating the current state of performing the task. SAUI offers performance support through the customization to each user's knowledge, and preferences. Typical hypermedia systems identify a predefined course through technical information. SAUI, on the other hand, dynamically defines a unique course each time it presents decision support. The adaptive predictive components serves as the expert engineer, driving the prevention strategy based on a dynamic assessment of time, location, environment, resources, and a specific operator's knowledge and experience with the current situation. SAUI uses a collapsible checklist of steps to guide the operator through a prevention procedure. SAUI determines how to present this checklist based on a dynamic assessment of the user's context. The evaluation of SAUI revealed the difference in the needs of experienced and inexperienced operators. Inexperienced operators need an expanded outline of subtasks that reveals details. Experienced operators expand the outline if they choose, and have greater flexibility to navigate within the checklist. Inexperienced operators require more assistance in step-by-step navigation.

7.3 Using CBR Technology

Designing SA systems using CBR required less knowledge engineering task than rule-based or model-based approaches. Having access to similar problems reduced the need for problem analysis. Solution components from old problems are reused and less in-depth analysis of new problems are required. Adapted solutions to new problems are preserved for future use. The competence of the CBR system increased over time as new cases are processed and added to the case base. This is a learning process because it results in an improvement in the competence of the systems. The case base can be extended and updated as part of the larger case-base maintenance issue. Case-base maintenance involves identifying existing cases that are out of date and identifying new cases that extend the competence of the system. For instance, some cases used in the first prototype were considered obsolete in problem solving in the third prototype.

There are two levels in problem solving as formalised in knowledge based systems. The first stage involves an analysis of the problem that produces a representation of the problem that can be manipulated by the reasoning system. This representation is often a set of attribute values. The second stage involves developing the reasoning mechanism that manipulates the problem representation to produce a solution. For CBR, these reasoning processes is implemented as retrieval and adaptation. Our CBR systems involves little or no adaptation and the reasoning mechanism is simply a retrieval process with solutions being used intact or with adaptation performed by the user. In this context, these CBR systems have less advantage over systems that require full knowledge engineering.

Further more, hydrate domain does not have a workable set of predictive features to make knowledge engineering task with CBR easy. Developing CBR systems to address problems in this domain has been difficult because the important features were difficult to determine and the relative importance of features were difficult to perceive. This problem has been confounded by the fact that the features are highly context sensitive. Features that are very predictive in some contexts are not relevant in others, for example, a high temperature may be desirable in one context but undesirable in the other. In addition, different engineers solve similar problems in different ways, solutions that are considered standard by practitioners in one oil field might be considered unusual by practitioners in another oil field.

7.4 Close Relationship by Working together

Working closely with these practitioners for a long time, created some challenges. It was difficult as a researcher to detach from the practical process. The focus of the collaboration became the focus of practice. The researcher became "one of them" and the borders became opaque. Dealing with this close relationship was, at times, quite challenging [19]. For example, the researcher volunteered to facilitate some of the design workshops focusing on process and work flows in hydrate prevention in North sea gas pipelines. These meetings were aimed at articulating the existing work processes in hydrate control before and after the implementation of the new system. The facilitation consisted of preparing models concerning different flows based on the ethnographic observations. These models were then used to prompt questions and revise models outlining the various work processes within hydrate control according to the new systems. During these workshops for processes, all models were revised and evaluated. Moreover, many important issues related to the current work practices and the design were discussed and negotiated. The researcher constructed and revised the models as well as writing up the observation notes from the workshops. However, facilitating these meetings made the distinction between being a researcher and a practitioner unclear.

However, the close relationship was crucial to create the environment where the researcher, with the practitioners, co-constructed new knowledge that was both relevant for practice and theory. It was theoretically anchored because, throughout the iterative action cycles, problematic issues were challenged with various theoretical concepts such as, the theoretical concepts of contextual contingencies, situation awareness, tacit knowledge, case-based reasoning etc in the process of reaching an in-depth understanding of the particular issues and then interventions were planned based upon this reflection.

7.4.1 Renegotiating Goals

One of the essential areas that kept the close relationships necessary to turn the theoretical concept into action was the negotiation and renegotiations of the dual goals of action research. Prior work argued for the importance of negotiating the goals of AR initially through researcher/client agreement (RCA) [12]. The researcher saw the need for the agreement, and also the need for continuous negotiating and re-negotiating the dual goals. Other investigations have also pointed to the need for renegotiation [19] [10]. Practice is dynamic, and the context for research is continuously changing. In enacting action research, it is important to follow this change, thus re-formulating the research question, aim, objectives and expectations at different stages. This work proposed a flexible researcher/client agreement rather than an RCA produced in advance [45]. Continuous renegotiation of the dynamic relationship with the practitioners is essential. Davison et al. [45] together with Bjorn and Balka [19] had acknowledged that the researcher/client agreement, in reality, is of a more emergent nature, and that it is unproductive to view the researcher/client agreement as a static initial document.

7.4.2 Managing Data and Action Cycles

Another area supporting the maintenance of close practical relations, while preserving the research goal, involves the management of the action cycles. Studies in AR generally agree that clearly defined cyclic processes are essential for structured data collection [82]. While agreeing that the cyclic process, as well as structured data collection, is essential, Bjorn et al. [19] argued that the borders of the cyclical processes are emergent and changing. This means that determining when a phase will end and when the next phase will begin is inappropriate and should be negotiated. If action research takes into account the continuously changing practice, it makes it impossible to initially plan the whole project, including all the cycles and phases. In this project, the cyclic perspective on the research is managed in order to create the engagement necessary for understanding domain knowledge. This includes continuously evaluating the practical problems as well as identifying the theoretical research approach. Allowing the theoretical approach to change and move together with the context increased our chances of producing theoretical findings immediately relevant for practice. However, managing this process required that the data collection techniques were continuously re-evaluated. Without clear borders between activities, the risk of getting lost in the practical problems increases. It was essential to distinguish the researcher from the practical cycles, stepping back and reflecting upon the practical problems in light of the research cycles. In this process, the borders of the project is continuously constructed and reconstructed. While constructing and reconstructing borders, the dual cycles of practice and research is seen to be detached from each other [116].

7.5 Managing a Network of Stakeholders

The researcher's role depends on the changing context, and effort is required to sustain the relationship with practitioners both at the organizational and collaborative level. In this project there was an additional responsibility of managing a network of all stakeholders. The role of managing a network is, however, often missing from the literature on action research. The problematic issue of putting in extra work to keep the access to the field has been mentioned in previous research [146]. By the experience on this project, it is argued here that putting in the time for networking is not a choice but a condition for action research. The extra time is not necessarily well used producing designs to meet short-term practical goals. The time was best used building up engagement and networking as well as keeping the door open and noticing new opportunities.

7.6 Co-authorship with Practitioners

During the period of this project work, the ideas discussed in Chapter 3, 4, 5 and 6 were published in papers [128] [129] [130] presented at IEEE situation awareness conferences.

The process of co-authoring with practitioners in a different discipline involved translation of theoretical concepts to the practitioners. In this situation it is not appropriate to overload a co-author with a lot of reading material. Much more is described about the theoretical framework than is normally done. In this way, there was a lot of extra effort required of the researcher in writing with domain practitioners. There is much more articulation work [181] involved in the collaborative work of co-authoring papers with domain practitioners. This means that allowing the practical situation to guide the direction of research calls for a flexible theoretical perspective. The researcher therefore see stability in theory as an idealised vision, and action research is to balance the theoretical approach with the actual practical work [82]. It was challenging to co-author with domain practitioners but the process added to the data collection and interpretation techniques, since the researcher got the practitioners' perspective on the actual theoretical findings, which helped ensure that the theoretical contributions were relevant for practical situations.

Chapter 8

Conclusion and Future work

The work presented in this thesis is based on the review of the current design of context-aware case-based decision support systems (CACBDSS) and the identification of some of its drawbacks, particularly its inability in similarity assessment and problem solving prediction in ongoing situations. This thesis has presented techniques of incorporating models of situation awareness to context-aware case-based decision support systems to address problems in the current situation. The chapter outlines the contributions of the thesis and points out some directions for future work. Finally, the work is summarised and concluded.

8.1 Contributions

Some contributions are made in answering the main research question of how context-aware case-based decision support systems can reflect the current situation. Answers to the question are presented in four main contributions contained in this work. The first contribution is a research and design methodology for context-aware case-based decision support systems to meet the objective of undertaking an action-research-based evaluation of the developed tools through a series of pilots applications and evaluations. The second contribution is the architecture of situation awareness model as a problem identification component of CACBDSS. The third contribution is the framework for user interface design for situation awareness model as a problem identification component of CACBDSS.

The third contribution presents some methods to be utilised by designers in a software development environment after existing design patterns were investigated to consider how to decouple the interface and underlying data model. The final contribution is the integration of the situation awareness model and the CBR model to make situation awareness a means of problem identification and solving in CACBDSS. The second and the final contributions are products of a review and assessment to critique the way existing design notations depict the changing underlying information to the user. The ideas can be applied in any dynamic domain that requires experiential reaction to occurring situations such as medical diagnosis, air traffic control, and military command and control. In this project, the ideas are tested in the oil and gas hydrate control domain as a proof of concept.

1. In Chapter 3, the research-design methodology integrated the methodologies of action research (AR), user-centered design (UCD), and agile development (AD) to form a comprehensive research-design cycle. The integration of these different methods results in a research-design process comprising three segments; scenarios, agile user-centered design, and business change. The approach was effective in designing decision support systems that requires collaborative work with domain practitioners in order to understand the activities of the domain, capture the practitioner's requirements, refining the requirements for redesign at an early stage to save time and cost. The agile iterative development is more appropriate for the user-centered design process, as evaluation in agile development is done many times during the project to give room for a change in requirements if necessary. Domain analysis with practitioners using scenarios provided the understanding of the domain activities, the social settings, resources, and goals of users. The evolutionary and question-asking process of scenarios filled the knowledge gap about the domain and acted as a communication mechanism between the users and the user-centered design process. Methods such as interviews, surveys, and field studies are well suited for scenarios and provide good understanding of who users are, why they need the system, and in what context they are going to use the system. Combining action research with agile user-centred design is an appropriate means of achieving design and research simultaneously to solve organisational information systems problems.

2. In Chapter 4, the project's first prototype, situation awareness as a problem identification component of context-aware case-based decision support systems (CACBDSS) is presented. A system designed as three-model architecture is presented. The design combined the concept of situation awareness, context awareness, case-based reasoning, and general domain knowledge in decision support (Knowledge-intensive case-based decision support). The case-based reasoning component of the system is the part that seeks to accomplish a certain task. The situation awareness component uses the context of the user to provide relevant information about the environment to be used in the reasoning process. The general domain knowledge provides explanations to the outcome of the reasoning process. Prior work uses context in similarity assessment in order to understand the current situation. This approach uses context with features of real situations to interpret and to understand the current situation. The approach, apart from its usefulness in solving problems of incomplete data and domain specific problems, is also useful in anticipating situation-dependent problems. Enriching knowledge intensive case-based reasoning with additional knowledge (SA) made retrieved solutions immediately ready to be used in the solving of new problems.

3. In Chapter 5, a framework for the design of the user-interface for situation awareness as a means of problem identification in CACBDSS is presented. The first step in a interface design is to draw up the task model, a hierarchic structure and a way of establishing temporal relationships between various (sub) tasks. A method of combining scenarios, HTA, and requirements analysis in task modelling is demonstrated. The approaches complement each other by using scenarios to stimulate and support reasoning in task analysis. Task analysis provides an integrated picture of tasks. Mapping real, complete and representative tasks of HTA to abstract and partial tasks of requirements analysis helps to ensure that all important users' tasks with their relationships and interactions are identified. Metadata of each of the task data item was created. Metadata was created within a specific context and for specific purpose, and different purposes and different contexts have different metadata requirements. The framework produced an interface that adapts to the context of individual operators, providing methods to help operators

to perform their tasks in smart and intelligent ways using context-aware technologies. A novel approach is presented, using the Endsley's situation awareness model to design an adaptor and the dynamic dialogue model. The adaptor uses the dynamic dialogue model and the dynamic task model to decide when the user interface has to be updated. These dynamic models are adjusted so that account can be taken of the current situation, if this influences the tasks the user wants to perform. The dynamic dialogue model consists of possible statuses of the user interface. This avoids the user interface adjusting if this is not desirable. Existing user interface design does not solve the problem of drawing together the information required for situation-aware decision support systems in a way that minimises cognitive load. The user interface reconfigures automatically in order to adapt to the current situation. The adaptation of the user interface to the current situation and the presentation of a reusable sequence of tasks in the situation reduces memory loads on operators by 55%, 63%, and 84% for normal, warning, and danger respectively.

4. Chapter 6 discusses situation awareness as a means of both problem identification and solving in context-aware case-based decision support systems. The approach is a fusion of the CBR model and the SA model into a case-based situation awareness (CBSA) model for situation awareness based on experience rather than rule, similarity assessment and problem solving prediction. In this prototype, experienced situations are stored as cases and experiences are retrieved by comparison with a current experience. When retrieving the appropriate action, the operator has some expectations expressed through context that constrain the assessment. The expectations are monitored while the assessment is being executed. Together with the probability of occurrence of each situation, the overall assessment value is given by the expectation value which indicates the applicability of the action and the future of the situation. Infusing situation awareness into CBR increases the accuracy level of CBR. In situation awareness, comprehension is not simply being aware of the elements that are present, but includes an understanding of the meanings of those elements with respect to an individual's goals. It provides an organized picture of the elements with an understanding of the significance of objects and events. Representing the CBR's retrieve process as Level 2 SA (comprehension) extends the retrieve task from

extraction of cases due to pattern match to include sense making, meaning, and relevance. Similarly, the representation of the CBR's reuse process as Level 3 SA (projection) ensures that actions are not only adapted for the current situation but also for situations that will evolve from the current. The approach provided a framework and an architecture for building efficient single-model case-based situation awareness systems to meet specific needs of individual operators with minimal time and cost.

8.2 Future Work

Although situation awareness as presented in this thesis has addressed some of the issues in context-aware case-based decision support systems there are a number of possible avenues for future work which could extend this work.

Case-based reasoning technologies alone are not sufficient to provide situation awareness. In situation awareness, comprehension is not simply being aware of the elements that are present, but includes an understanding of the meanings of those elements with respect to an individual's goals. It provides an organized picture of the elements with an understanding of the significance of objects and events. The complexities in comprehension in some domains make past situations not a good predictor of future action. So, it is inappropriate to assume that situations in every domain will be understood by retrieving similar past situations. To solve such complex domain problems, future research should consider an approach that combines case-based reasoning and rules.

Additionally, to support the use of procedural information, further work will be carried out on a user interface that will custom-select links to support information that will be offered to each operator through its navigation component. The interface will not only custom-select links for an operator, but will also inform the operator on the relevance of the links that are offered. Operators who are inexperienced with a step will be offered links to fundamental concepts, and background information, while experienced operators will be offered links to more concise information that omits fundamentals that have already been mastered. A comparative study of the static and adaptive displays of the user interface will be evaluated.

Finally, the system will be implemented in the public health domain to predict the outbreak of diseases and to recommend appropriate preventive plans. The state of disease outbreak in an environment is uncertain. Similar to hydrate formation, predicting the outbreak of a disease requires the ability to be aware of the occurring situations and react experientially to them.

8.3 Summary

Context-aware case-based decision support systems that have situation awareness using action research as a domain knowledge translation process to enable the development of the knowledge was designed. The goal is to ensure findings have practical relevance for the design of predictive control systems. It is found that establishing a research context as a domain knowledge information context required a strong relationship with practice. This strong relationship supported the engagement with the actual design activity of the hydrate control system, spurring a process of crucial negotiations for the study of the generic flow assurance system.

Creating the research context for understanding domain knowledge requires four important activities: (1) negotiating and renegotiating the dual goals (action and theory),(2) managing the interlinked action cycles, (3) managing the network, and (4) creating and translating the continuously changing theoretical conceptual framework, model, and architecture.

By identifying these activities embedded in action research, some issues taken for granted in action research were questioned. Negotiating and renegotiating the dual goals of action and theory raises questions about whether or not it makes sense at all to have, as an ideal approach, the researcher/client agreement negotiated initially [45] [82], because it will change over time.

Identifying "managing the action cycles" as a crucial part of action research raises questions about the whole approach presented in many descriptions of action research, that the research project is a straightforward, clearly phased process [12] [82].

Pointing to the essential element of managing the network as a part of action research makes visible an absent issue of action research, which has not been addressed. The study found that network was essential for creating the close relationship with practice, establishing a context for domain knowledge gathering where the creation of knowledge directly impacted the design of systems.

In this project, the researcher became committed to producing timely and accessible findings that can be used in practice. By noticing issues of the design, the researcher make possible conflicts visible and available for negotiations in the actual process of design and configuration. When the researcher then engaged in discussions and negotiations with the practitioners, volunteering theoretical conceptualizations of the experienced issues, the researcher, together with the practitioners, co-construct the context of research and general knowledge. This means that nobody is the sole custodian of knowledge, since both researchers and practitioners construct and create new meaning essential for new knowledge.

Action research in modelling situation awareness comprises the dual process of transforming and aligning practitioners' practical experiences into empirical observations, as well as developing, transforming, and aligning theoretical concepts relevant to explain, interpret, and understand the empirical observations.

This work is not a conclusive list of the work required to establish action research as a means of realising a situation-aware user-centred design. Instead, this is the beginning of articulating an important benefit of action research in UCD, which has not been highlighted in previous SA and HCI papers [36]. It is hoped other SA researchers in complex domains will take up the challenge to engage in action research projects by collaborating with practitioners in participatory design processes, since this will bring new avenues for SA and HCI research in general.

Bibliography

[1] A. Aamodt. Explanation-driven case-based reasoning. in topics in case-based reasoning. In S. Wess, editor, *Springer Verlag*, pages 274–288, 1994.

[2] A. Aamodt. Knowledge-intensive case-based reasoning in creek. In P. Funk and P.A. Gonzalez Calero, editors, *Advances in case-based reasoning, 7th European Conference, ECCBR 2004, Proceedings. Madrid, Spain. Lecture Notes in Artificial Intelligence, LNAI 3155, Spinger*, pages 1–15, 2004.

[3] A. Aamodt and E. Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, 7(1):39–59, 1994.

[4] E.C. Adam. Fighter cockpits of the future. In *Proceedings of the 12th DASC.The 1993 IEEE/AIAA Digital Avionics Systems Conference.*, pages 318–323, 1993.

[5] M.J. Adams, Y.J. Tenney, and R.W. Pew. Situation awareness and the cognitive management of complex systems. *Human Factors*, 37(1):85–104, 1995.

[6] S. Adikari, C. McDonald, and J. Campbell. Little design up-front: A design science approach to integrating usability into agile requirements engineering. *LNCS*, 5610:549–558, 2009.

[7] D.F. Andersen and G.P. Richardson. Scripts for group model building. *System Dynamics Review*, 13:107–129, 1997.

[8] D.F. Andersen, J.A.M. Vennix, G.P. Richardson, and E.A.J.A. Rouwette. Group model building: Problem structuring, policy simulation and decision support. *Journal of the Operational Research Society*, 58:691–694, 2007.

[9] H.H.K. Andersen and G. Hauland. Measuring team situation awareness of reactor operators during normal operation: A technical pilot study. In *Proceedings of the first human performance, situation awareness, and automation conference.Savannah, Georgia: October 2000.*, 2000.

[10] D. Avison, R. Baskerville, and M. Myers. Controlling action research projects. *Information Technology & People*, 14:28–45, 2001.

[11] R. Bareiss. The experimental evaluation of a case-based learning apprentice. In *Workshop on Case-Based Reasoning*, page 162167, Pensacola Beach, Florida, USA., 1989.

[12] R. Baskerville and M. Myers. Special issue on action research in information systems: Making information system research relevant to practice. *MIS Quarterly*, 28 (3):329–335, 2004.

[13] R.L. Baskerville and A.T Wood-Harper. A critical perspective on action research as a method for information systems research. *Journal of Information Technology*, 11:235–246, 1996.

[14] K. Beck, M. Beedle, A. Bennekum, A. Cockburn, W. Cunningham, M. Fowler, J. Grenning, J. Highsmith, A. Hunt, R. Jeffries, J. Kern, B. Marick, R.C. Martin, S. Mellor, K. Schwaber, J. Sutherland, and D. Thomas. The agile manifesto. In *http://agilemanifesto.org*, 2001.

[15] S. Bennett, S. McRobb, and R. Farmer. *Object-oriented system analysis and design using UML.* McGRAW-HILL education, 2006.

[16] R. Bergmann, S. Breen, M. Goker, M. Manago, and S. Wess. Developing industrial case-based reasoning applications: The inreca methodology. *Lecture Notes in Artificial Intelligence, State-of-the-Art-Survey, LNAI 1612. Springer, Berlin, Heidelberg*, 1999.

[17] H. Beyer, K. Holtzblatt, and L. Baker. An agile user-centered method: Rapid contextual design. In *Agile Universe*, pages 50–59, Calgary, Alberta Canada, 2004.

[18] I. Biederman, R. J. Mezzanotte, and J.C. Rabinowitz. Scene perception: detecting and judging objects undergoing relational violations. *Cognitive Psychology*, 14:143–177, 1982.

[19] P. Bjorn and E. Balka. Supporting the design of health information systems: Action research as knowledge translation. In *Proceedings of the 42nd Hawaii International Conference on System Sciences*, 2009.

[20] K. Bodker, F. Kensing, and J. Simonsen. *Participatory IT design: Designing for Business and Workplace Realities.* The MIT Press, Cambridge, Massachusetts, 2004.

[21] C.A. Bolstad, H.M. Cuevas, C. Gonzalez, and M. Schneider. Modeling shared situation awareness. In *Paper presented at the 14th Conference on Behavior Representation In Modeling and Simulation (BRIMS), Los Angeles, CA*, 2005.

[22] R.H. Bonczek, C.W. Holsapple, and A.B. Whinston. The enveloping roles of models in decision support systems. *Decision Sciences*, 11:337–356, 1980.

[23] J. Braa, E. Monterio, and S. Sahay. Networks of action: Sustainable health information systems across developing countries. *MIS Quarterly*, 28 (3):337–362, 2004.

[24] R. Breton. Situation awareness: A review of the concept and the measurement. Technical Report DRDC Valcartier TR - 2001 - 220, Universite Laval, Canada, Defence Research and Development Canada, 2003.

[25] A. M. Busquets, R. V. Parish, S. P. Williams, and D.E. Nold. Comparison of pilots' acceptance and spatial awareness when using efis vs display formats for complex, curved landing approaches. In R.D. Gilson, D.J. Garland, and J.M. Koonce, editors, *Situation awareness in complex systems*, pages 139–170. Embry-Riddle Aeronautical University Press, Daytona Beach, FL, 1994.

[26] G. Calvary, J. Coutaz, and D. Thevenin. Supporting context changes for plastic user interfaces: A process and a mechanism. In *In Joint Proceedings of HCI and IHM 2001. Lille, France*, page 349364, 2001.

[27] P.F. Campos and N.J. Nunes. Canonsketch: a user-centered tool for canonical abstract prototyping. In *Proceedings of EHCI and DSV-IS, Germany 2004*, pages 108–126, 2004.

[28] J. Canas, A. Antoli, and J. Quesada. The role of working memory on measuring mental models of physical systems. *Psicologica*, 22, 2001.

[29] M. S. Carey, R. B. Stammers, and J. A. Astley. *In Task Analysis for Human-Computer Interaction.* Ellis Horwood, 1989.

[30] J.M. Carroll. Making use is more than a matter of task analysis. *Interacting with Computers*, 14:619–627, 2002.

[31] J.M. Carroll and M.B. Rosson. Getting around the task-artifact cycle: How to make claims and design by scenario. *ACM Transaction on Information Systems*, 10:181–212, 1992.

[32] M. Castellanos, C. Gupta, S. Wang, U. Dayal, and M. Durazo. A platform for situational awareness in operational BI. *Decision Support Systems*, 52:869–883, 2012.

[33] S. Chamberlain, H. Sharp, and N. Maiden. Towards a framework for integrating agile development and user-centred design. In *7th International Conference on eXtreme Programming and Agile Processes in Software Engineering (XP' 06), Springer Berlin / Heidelberg*, pages 50–59, 2006.

[34] P. Checkland and S. Holwell. *Information, Systems and Information Systems: Making Sense of The Field.* John Wiley, Chichester, UK, 1998.

[35] Y.M. Cheng and C. Johnson. Applying task analysis to facilitate the design of context-aware technologies. In *2nd Workshop on complexity in design and engineering*, 2007.

[36] C. Chewar and D. McCrickard. Links for a human-centered science of design: Integrated design knowledge environments for a software development process. In *38th Hawaii International Conference on System Sciences*, 2005.

[37] J. Choi. Context: From birth to design. In *International Conference on Advanced Language Processing and Web Information Technology*, 2008.

[38] P. Clark. Action research and organizational change. *London: Harper & Row*, 1972.

[39] T. Clerckx and K. Coninx. Towards an integrated development environment for context-aware user interfaces. In *Dagstuhl seminar. Mobile computing and ambient intelligence: The challenge for multimedia*, 2005.

[40] T. Clerckx, K. Luyten, and K. Coninx. The mapping problem back and forth: Customizing dynamic models while preserving consistency. In *3rd International Workshop on Task Models and Diagrams for user interface design (TAMODIA 2004), Prague, Czech Republic*, pages 33–42, 2004.

[41] T. Clerckx, K. Luyten, and K. Coninx. Designing interactive systems in context: From prototype to deployment. In *19th British HCI Group Annual Conference (HCI 2005), Napier University, Edinburgh, United Kingdom*, 2005.

[42] K. Conboy and B. Fitzgerald. Toward a conceptual framework of agile methods: A study of agility in different disciplines. In *Proceedings of the ACM Workshop on Interdisciplinary Software Engineering Research (WISER' 04)*, pages 37–44, Newport Beach, California, USA, 2004.

[43] A. Cooper and R. Reimann. *About Face 2.0: The Essentials of Interaction Design*. Wiley Publishing Inc., Indianapolis, IN, 2003.

[44] J. Davison, J. Thompson, D. Deeks, and M. Lejk. Pisosia, a stakeholder approach to assist change in information systems development projects and aid process improvement. *Software Qual*, 14:25–36, 2006.

[45] R.M. Davison, M.G. Martinsons, and N. Kock. Principles of canonical action research. *Information Systems Journal*, 14:65–86, 2004.

[46] M. R. W. Dawson. *Understanding Cognitive Science*. Blackwell, Oxford, UK, 1998.

[47] A.K. Dey. Understanding and using context. *Personal and Ubiquitous Computing*, 5:5–7, 2001.

[48] D. Diaper. Task scenarios and thought. *Interacting with Computers*, 14:629–638, 2002.

[49] V.T. Dijk. Context model in discourse processing. In H.V. Oostendorp and S.R. Goldman, editors, *The construction of mental representations during reading*, pages 123–148. Lawrence Earlbaum Associates, Mahwah, NJ., 1999.

[50] C. Dominguez. Can SA be defined? In M. Vidulich, C. Domonguez, E. Vogel, and G. Millan, editors, *Dominguez, C. (1994). Can SA be defined? In M. Vidulich, C. Domonguez, E. Vogel, & G. Millan (Eds), Situation awareness: Papers and annotated bibliography (AL/CF-TR-1994-0085; pp. 5-15). Wright-Patterson AFB, OH: Armstrong Laboratory.*, AL/CF-TR-1994-0085, pages 5–15. Wright-Patterson AFB, OH: Armstrong Laboratory., 1994.

[51] J.L. Drury, L. Riek, and N. Rackliffe. A decomposition of UAV-Related situation awareness. In *1st ACM SIGCHI/SIGART Conference on Human-Robot Interaction*, 2006.

[52] F. T. Durso, C. A. Hackworth, T. R. Truitt, J. Crutchfield, D. Nikolic, and C. A. Manning. Situation awareness as a predictor of performance for en route air traffic controllers. *Air Traffic Control Quaterly*, 6(1):1–20, 1998.

[53] M. Endsley and D. Kaber. Level of automation effects on performance, situation awareness and workload in a dynamic control task. *Ergonomics*, 42:462–492, 1999.

[54] M.R. Endsley. Toward a theory of situation awareness in dynamic systems. *Human Factors and Ergonomics Society*, 37:32–64, 1995.

[55] M.R. Endsley. Measurement of situation awareness in dynamic systems. *Human Factors. 37(1), 65-84.*, 37(1):65–84, 1995b.

[56] M.R. Endsley. A comparative analysis of sagat and sart for evaluations of situation awareness. In *Proceedings of the Human Factors and Ergonomics society 42nd Annual meeting*, 82-86, 1998.

[57] M.R. Endsley. Direct measurement of situation awareness: Validity and use of sagat. In M.R. Endsley and D.J. Garland, editors, *Situation analysis and measurement*, pages 147–173. Lawrence Earlbaum Associates, Mahwah, NJ., 2000.

[58] M.R. Endsley. Situation models: An avenue to the modeling of mental models. In *Proceedings of the 14th Triennial Congress of the International Ergonomics Association and the 44th Annual Meeting of the Human Factors and Ergonomics Society*, 2000.

[59] M.R. Endsley, B. Bolte, and D.G. Jones. *Designing for Situation Awareness: An Approach to User-Centered Design.* Taylor & Francis, 2003.

[60] M.R. Endsley and D.J. Garland. *Situation awareness analysis and measurement.* Lawrence Earlbaum Associates, Mahwah, NJ., 2000.

[61] M.R. Endsley and D.J. Garland. *Theoretical underpinnings of situation awareness: A critical review.* Lawrence Erlbaum associates, 2000.

[62] M.R. Endsley, L.D. Holder, B.C. Leibrecht, D.J. Garland, R.L. Wampler, and M.D. Matthews. Modeling and measuring situation awareness in the infantry operational environment. *U.S. Army Research Institute for the Behavioral and social Sciences. Infantry Force Research Unit*, 2000.

[63] M.R. Endsley, R. Sollenberger, and E. Stein. The use of predictive displays for aiding controller situation awareness. *Human Factors and Ergonomics society*, 43:51–56, 1999.

[64] Y. Feng, T. Teng, and A. Tan. Modelling situation awareness for context-aware decision support. *Expert Systems with Applications*, 36:455–463, 2009.

[65] J. Ferreira, J. Noble, and R. Biddle. Agile development iterations and ui design. In *AGILE Conference (Agile' 07), IEEE Press*, pages 50–58, 2007.

[66] J.W. Forrester. Policies, decisions, and information sources for modelling. In J.D.W. Morecroft and J.D. Sterman, editors, *Modelling for Learning Organizations*, pages 51–84, Portland, 1994.

[67] D. Fox, J. Sillito, and F. Maurer. Agile methods and user-centered design: How these two methodologies are being successfully integrated in industry. In *Agile conference*, 2008.

[68] D.M. Gaba and S.K. Howard. Situation awareness in anaesthesiology. *Human Factors*, 37(1):20–31, 1995.

[69] J.J. Gibson. *The ecological approach to visual perception.* Houghton-Mifflin, 1979.

[70] J.D. Gould and C.H. Lewis. Designing for usability: key principles and what designers think. *Communications of the ACM*, 28 (3):300–311, 1985.

[71] L.J. Gugerty and W.C. Tirre. *Situation awareness analysis and measurement*, chapter Individual differences in situation awareness, pages 249–276. Lawrence Earlbaum Associates, Mahwah, NJ., 2000.

[72] M. Gupta and S. Mukherjee. Towards situation awareness in integrated air defence using clustering and case-based reasoning. In *Pattern recognition and machine intelligence third international conference*, 2009.

[73] B.U. Haque, R.A. Belecheanu, R.J. Barson, and K.S. Pawar. Towards the application of case based reasoning to decision-making in concurrent product development (concurrent engineering). *Knowledge based systems*, 13:101–112, 2000.

[74] T. Heinbokel, S. Sonnentag, M. Frese, W. Stolte, and F. C. Brodbeck. Don't underestimate the problems of user centredness in software development projects - there are many. *Behaviour & Information Technology*, 15 (4):226–236, 1996.

[75] J.M. Henderson and A. Hollingworth. High-level scene perception. *Annual Review of Psychology*, 50:243–271, 1999.

[76] K.C. Hendy. Situation awareness and workload: Birds of a feather? In *AGARD AMP Symposium on situation awareness: Limitations and the enhancements in the aviation environment*, 1995.

[77] M. Hertzum. Making use of scenarios: A field study of conceptual design. *International Jounal of Human-computer studies*, 58 (2):215–239, 2003.

[78] B. Hilburn. Measuring the right stuff: A comprehensive approach to aerospace sa assessment. In *Proceedings of the first human performance, situation awareness, and automation conference, Savannah, Georgia*, 2000.

[79] P. Holt, S. Hogg, and J. Aitchison. A case study of usability engineering in the defence industry. In T. Spiliotopoulos, P. Papadopoulos, D. Martakos, and G. Kouroupetroglou, editors, *Integrating Usability Engineering for Designing the Web Experience: Methodologies and Principles*. IGI Global, Hershey, PA, 2009.

[80] M. Hoogendoorn, R.V. Lambalgen, and J. Treur. An integrated agent model addressing situation awareness and functional state in decision making. In *The 4th International conference on principles and practice of multi-agent systems(PRIMA2011), Wollongong, Australia*, 2011.

[81] M. Hoogendoorn, R.V. Lambalgen, and J. Treur. Modeling situation awareness in human-like agents using mental model. In *Twenty-second International Joint Conference on Artificial Intelligence*, 2011.

[82] J. Iversen, L. Mathiassen, and P.A. Nielsen. Managing risk in software process improvement: An action research approach. *MIS Quarterly*, 28 (3):395–433, 2004.

[83] S.M. Jenner, R.J. Molloy, E.A. Byrne, and B. Strauch. Automation and situation awareness in transportation accidents. In *Proceedings of the first human performance, situation awareness, and automation conference, Savannah, Georgia*, 2000.

[84] P. Johnson-Laird. *Mental models: Towards a cognitive science of language, inference,and consciousness.* Cambridge, MA: Harvard university press., 1983.

[85] D.G. Jones and M.R. Endsley. Examining the validity of real-time probes as a metric of situation awareness. In *Proceedings of the 14th Triennnial Congress of the International Ergonomics Association and the 44th Annual meeting of the Human Factors and Ergonomics Society*, 2000.

[86] R.E.T. Jones, E.S. Connors, M.E. Mossey, J.R. Hyatt, N.J. Hansen, and M.R. Endsley. Modeling situation awareness for army infantry platoon leaders using fuzzy cognitive mapping techniques. In *19th Conference on behaviour Representation in Modelling and Simulation*, 2010.

[87] D.B. Kaber and M.R. Endsley. The effects of level of automation and adaptive automation on human performance, situation awareness and workload in a dynamic control task. *Theoretical Issues in Ergonomics Science*, pages 1–40, 2004.

[88] S.J. Kass, K.S. Cole, and C.J. Stanny. Effects of distraction and experience on situation awareness and simulated driving. *Transportation Research Part F*, 10:321–329, 2007.

[89] B. Khajotia, D. Sormaz, and S. Nesic. Case-based reasoning model of co2 corrosion based on field data. In *NACE international corrosion conference & expo*, 2007.

[90] J. Kim, S.E. Schwarm, and M. Ostendorf. Detecting structural metadata with decision trees and transformation-based learning. In *HLT/NAACL*, pages 137–144, 2004.

[91] J. Kirakowski and M. Corbett. Sumi: the software usability measurement inventory. *British Journal of Educational Technology*, 24(3):210–212, 1993.

[92] G. Klein. *Analysis of situation awareness from critical incident reports.* Lawrence Earlbaum Associates, Mahwah, NJ., 2000. Situation awareness analysis and measurement.

[93] L. Kleinrock. Normadicity: Anytime, anywhere in a disconnected world. *Mobile networks and applications*, 1:351–357, 1996.

[94] A. Kofod-Petersen. *A case-based approach to realising ambient intelligence among agents*. PhD thesis, Department of Computer and Information science, Norwegian University of Science and Technology, 2007.

[95] A. Kofod-Petersen and A. Aamodt. Contextualised ambient intelligence through case-based resoning. In *8th European conference on case-based reasoning*, volume 4106, 2006.

[96] A. Kofod-Petersen and A. Aamodt. Case-based reasoning for situation-aware ambient intelligent: A hospital ward evaluation study. In *LNAI 5650*, pages 450–464, 2009.

[97] K. Kuutti. *Workprocess: Scenarios as a preliminary vocabulary. In J. M. Carroll (Ed.), Scenario based design.* Wiley, 1995.

[98] M. Kyng. *Creating Contexts for Design.* John Wiley., 1995.

[99] J.C. Lee, D.S. McCrickard, and K.T. Steven. Examining the foundation of agile usability with extreme scenario-based design. In *Proceeding of Agile 2009, IEEE Computer society*, 2009.

[100] K. Lewin. Frontiers in group dynamics. *Human Relations*, 1 (1):5–41, 1947a.

[101] K. Lewin. Frontiers in group dynamics ii. *Human Relations*, 1 (2):143–153, 1947b.

[102] C. Lewis and J. Rieman. *Task-Centered User Interface Design.* Blustein, 2008.

[103] S. Liao. Case-based decision support system: Architecture for simulating military command and control. *European Journal of operational Reaearch*, 123:558–567, 2000.

[104] Q. Limbourg, J. Vanderdonckt, B. Michotte, L. Bouillon, and V. Lopez-Jaquero. *USIXML: a Language Supporting Multi-Path Development of User Interfaces.* Kazman and Palanque, 2004.

[105] Y. Liu, E. Shriberg, A. Stolcke, B. Peskin, J. Ang, D. Hillard, M. Ostendorf, M. Tomalin, P. Woodland, and M. Harper. Structural metadata research in the ears program. In *ICASSP*, volume 5, 2005.

[106] C.F. Ma, G.J. Chen, F. Wang, C.Y. Sun, and T.M. Guo. Hydrate formation of (ch4 + c2h4) and (ch4 + c3h6) gas mixtures. *Fluid Phase Equilibria*, 191:41–47, 2001.

[107] Y. Ma, D.V. Kalashnikov, and S. Mehrotra. Towards managing uncertain spatial information for situational awareness applications. *IEEE Transactions on Knowledge and Data Engineering*, 20:1408–1422, 2008.

[108] L.E. Maggart and R. Hubal. A situation awareness model. In S.E. Graham and M.D. Matthews, editors, *Paper from 1998 Infantry situation awareness workshop*, pages 19–28. U.S. Army Research Institute for the Behavioral and Social Science, 1999.

[109] M. Malmsten and H. Lindstrom. User-centred design and agile development: Rebuilding the swedish national union catalogue. *The Code4Lib Journal*, 6:12–15, 2008.

[110] R. Mark. *Discussion: Scenarios as engines of design.* John Wiley & Sons Inc., 1995.

[111] A.B. Markman and D. Gentner. Thinking. *Annual Review of Psychology*, 52:223–247, 2001.

[112] S.P. Marshall. Using eye tracking to understand situation awareness. In *Proceedings of the first human performance, situation awareness, and automation conference, Savannah, Georgia*, 2000.

[113] A. Martin, R. Biddle, and J. Noble. The xp customer role in practice: Three studies. In *Proceedings of ADC 2004*, Salt Lake City, 2004.

[114] L. Mathiassen. Collaborative practice research. *Information Technology & People*, 15 (4):321–345, 2002.

[115] B. McGuinness and L. Foy. A subjective measure of SA: The crew awareness rating scale (CARS). In M.R. Endsley and D.B. Kaber, editors, *Proceedings of the first human performance, situation awareness, and automation conference*, Savannah, Georgia, 2000.

[116] J. McKay and P. Marshall. The dual imperatives of action research. *Information Technology & People*, 4 (1):46–59, 2001.

[117] H. Merisalo-Rantanen, T. Tuunanen, and M. Rossi. Is extreme programming just old wine in new bottles: A comparison of two cases. *Journal of Database Management*, 16 (4):41–61, 2005.

[118] L. Miller and D. Sy. Agile user experience sig. In *Extended abstracts of Conference on human factors in computing systems (CHI' 09), ACM Press*, pages 2751–2754, 2009.

[119] R.J. Molloy. Automation-induced loss of situation awareness in accidents in surface transportation accidents. In *Proceedings of the first human performance, situation awareness, and automation conference, Savannah, Georgia*, 2000.

[120] N. Moray. Mental model in theory and practice. In D. Gopher and A. Koriat, editors, *Attention and Performance XVII*, pages 223–258. The MIT press, 1996.

[121] G. Mori, F. Patern‘o, and C. Santoro. Design and development of multidevice interfaces through multiple logical descriptions. *IEEE Transactions on Software Engineering*, 30(8), 2004.

[122] E. Mumford. The story of socio-technical design: Reflections on its successes, failures and potential. *Information Systems Journal*, 16:317–342, 2006.

[123] E. Mumford and M. Weir. *Computer Systems Work Design: The ETHICS Method*. Associated Business Press, London, 1979.

[124] B. Nardi. *Some reflections on scenarios.* Number ISBN: 0-471-07659-7. John Wiley & Sons Inc., 1995.

[125] C. Neisser. *The Perceived self: Ecological and Interpersonal Sources of Knowledge.* Cambridge University Press, New York, NY, 1993.

[126] C. Nodder and J. Nielsen. *Agile usability: best practices for user experience on agile development projects.* Nielsen Norman Group, Fremont, CA., 2008.

[127] D. A. Norman. *User-Centered System Design: New Perspectives on Human-Computer Interaction.* Lawrence Earlbaum Associates, Hillsdale, NJ., 1986.

[128] N.D. Nwiabu, I. Allison, P. Holt, P. Lowit, and B. Oyeneyin. Situation awareness in context-aware case-based decision support. In *IEEE International conference on Cognitive Methods in Situation awareness and Decision Support, Miami Beach, FL*, pages 9–16, 2011.

[129] N.D. Nwiabu, I. Allison, P. Holt, P. Lowit, and B. Oyeneyin. Case-based situation awareness. In *In IEEE International Conference on Cognitive Methods in Situation Awareness and Decision Support*, pages 22–29, New Orleans, LA, USA, 2012.

[130] N.D. Nwiabu, I. Allison, P. Holt, P. Lowit, and B. Oyeneyin. User-interface design for situation-aware decision support systems. In *IEEE International Conference on Cognitive Methods in Situation Awareness and Decision Support, New Orleans, LA*, pages 332–339, 2012.

[131] H. Obendorf. Scenario-based usability engineering techniques in agile development processes. In *CHI 2008 proceedings*, 2008.

[132] H. Obendorf, A. Schmolitzky, and M. Finck. Xpnue defining and teaching a fusion of extreme programming and usability engineering. In *HCI Educators Workshop 2006 inventivity: Teaching theory, design and innovation in HCI, Limerick, Ireland.*, 2006.

[133] P. Oldfield. Domain modelling. *Appropriate process movement*, 2002.

[134] B. Oyo, D. Williams, and E. Barendsen. Integrating action research and system dy-

namics: Towards a generic process design for participative modelling. In *Proceedings of the 42nd Hawaii International Conference on System Sciences*, 2009.

[135] R. Parasuraman, T. Sheridan, and C. Wickens. A model for types and levels of human interaction with automation. *IEEE Transactions on Systems, Man, and CyberneticsPart A: Systems and Humans*, 30:286–297, 2000.

[136] F. Patern'o and E. Zini. Applying information visualization techniques to visual representations of task models. In *3rd International Workshop, Tamodia04, Prague, Czech Republic, November 15-16, 2004*, pages 105–112, 2004.

[137] J. Patton. Hitting the target: adding interaction design to agile software development. In *Proceedings of Conference on Object Oriented Programming Systems Languages and Applications (OOPSLA' 02), ACM Press*, 2002.

[138] J. Payne, J. Bettman, and E. Johnson. *The Adaptive Decision Maker.* Cambridge University Press, 1993.

[139] R.W. Pew. The state of situation awareness measurement: Heading toward the next century. In M.R. Endsley and D.J. Garland, editors, *Situation awareness analysis and measurement*, pages 33–47. Lawrence Earlbaum Associates, Mahwah, NJ., 2000.

[140] C. Pott, A. Johnson, and F. Cnossen. Improving situation awareness in anaesthesiology. In *ACM International conference proceeding series*, volume 132, 2005.

[141] J. Preece, H. Sharp, and Y. Rogers. *Interaction Design: Beyond Human Computer Interaction.* John Wiley & Sons. Inc., New Jersey, 2002.

[142] M. Press. Situation awareness: Lets get serious about the clue-bird. draft, 1986.

[143] M. Priestley. *Practical object-oriented design with UML.* McGraw-Hill education, 2003.

[144] C. Prince and E. Salas. *Situation awareness analysis and measurement*, chapter Team situation awareness, errors, and crew resource management: Research integra-

tion for training guidance, pages 325–347. Lawrence Earlbaum Associates, Mahwah, NJ., 2000.

[145] A.R. Pritchett and R.J. Hansman. Use of testable responses for performance-based measurement of situation awareness. In M.R. Endsley and D.J. Garland, editors, *Situation analysis and measurement*, pages 189–209. Lawrence Earlbaum Associates, Mahwah, NJ., 2000.

[146] R. Rapoport. Three dilemmas of action research. *Human Relations*, 23 (6):499–513, 1970.

[147] J. Rasmussen, A.M. Pejtersen, and L.P. Goodstein. *Cognitive System Engineering.* Wiley Interscience Publication: John Wiley and Sons Inc., 1994.

[148] J. Raz, A. Juhola, J. Serrat-Fernandez, and A. Galis. *Fast and efficient context-aware services.* Number ISBN: 0-470-01668-X. John Wiley & Sons Inc., 2006.

[149] P. Reason. Action research and organization development. *Journal of management inquiry*, 2006.

[150] J.A. Recio-García, B. Díaz-Agudo, M.A. Gómez-Martín, and N. Wiratunga. Extending jCOLIBRI for textual CBR. In *6th International Conference on CBR, ICCBR 2005, volume 3620 of LNCS*, 2005.

[151] M.M. Richter. *Case-based reasoning technology: From foundation to application*, volume 1400, chapter 1, pages 1–15. Springer, Berlin/ Heidelberg, 1998.

[152] W.R.T. Ricks, J.E. Jonsson, and J.S. Barry. Managing approach plate information study (maplist): An information requirements analysis of approach chart use. Technical Report NASA Technical Paper 3561, NASA Langley Research Center, Hampton, VA, 1996.

[153] A.R. Rita, P.L. Powell, and F.B. James. Uncertainty in decision-making: An adductive perspective. *Decision Support Systems*, 12:279–320, 1995.

[154] S. Robertson. *Generating object-oriented design representations via scenario queries.* John Wiley & Sons Inc., 1995.

[155] D.M. Rosenthal. Introspection. Technical report, MIT Encyclopaedia of the Cognitive Sciences (MITECS), 1998.

[156] M. Rosson and J. Carroll. *Narrowing the specification-Implementation gap in scenario-based design.* John Wiley & Sons Inc., 1995.

[157] M. Rosson and J. Carroll. The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications. *The Human Factors and Ergonomics*, pages 1032–1050., 2002.

[158] M. B. Rosson and J. M. Carroll. Usability engineering: Scenario-based development of human-computer interaction. *Morgan Kaufman, New York, NY*, 2002.

[159] W.B. Rouse and N.M. Morris. On looking into the black box: Prospects and limits in the search for mental models. *Psychological Bulletin*, 100:349–363, 1986.

[160] A.P. Sage. *Decision support systems engineering.* Wiley-Interscience, 1st edition, 1991.

[161] E. Salas, D.E. Sims, and C.S. Burke. *Is there a big five in team work?* SAGE, 2005.

[162] A.M. Salem. Case-based reasoning technology for medical diagnosis. *World academy of science, engineering and technology*, 31, 2007.

[163] N. Sanford. Whatever happened to action research? In A. Clark, editor, *Experimenting with Organizational Life: The Action Research Approach*, pages 11–18. New York: Plenum, 1976.

[164] N.B. Sarter and D.D. Wood. Situation awareness: A critical but ill-defined phenomenon. *International Jounal of aviation psychology*, 1:45–57, 1991.

[165] N.B. Sarter and D.D. Wood. How in the world did I ever get into that mode: Mode error and awareness in supervisory control. *Human Factors*, 37(1):5–19, 1995.

[166] P. Satchell. *Innovation and automation.* Aldershot, UK: Ashgate, 1998.

[167] B. Schilit and M. Theimer. Disseminating active map information to mobile hosts. *IEEE Network*, 8:22–32, 1994.

[168] H.J. Scholl. Action research and system dynamics: Can they benefit from each other? In *Proceedings of the 37th Hawaii International conference on system sciences*, 2004.

[169] K. Schwaber and M. Beedle. *Agile software development with SCRUM.* Prentice Hall, 2002.

[170] W.A. Scott. *Agile analysis and design.* Agile modeling: Effective practices for eXtreme programming and unified processes, 2002.

[171] A. Seffah and J. Gulliksen. *Human-Centered Software Engineering - Integrating Usability in the Development Process.* Springer, Heidelberg, 2005.

[172] A. Segev. Identifying the multiple contexts of a situation. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), v 3946 LNAI, Modeling and Retrieval of Context - Second International Workshop, MRC 2005, Revised Selected Paper*, 3946:118–133, 2006.

[173] W.L. Shebilske, B.P. Goettl, and D.P. Garland. Situation awareness, automaticity, and training. In M.R. Endsley and D.J. Garland, editors, *Situation awareness Analysis and Measurement.* Lawrence Earlbaum Associates, Mahwah, NJ., 2000.

[174] A. Shepherd. *In Task Analysis for Human-Computer Interaction(Ed, Daiper, D.).* Ellis Horwood, 1989.

[175] S.V. Shokouhi, A. Aamodt, P. Skalle, and F. Sørmo. Determining root causes of drilling problems by combining cases and general knowledge. *LNAI*, 5650:509–523, 2009.

[176] A. Sinquin, T. Palermo, and Y. Peysson. Rheological and flow properties of gas hydrate suspensions. *Oil and Gas Science and Technology*, 59(1):41–57, 2004.

[177] P. Skalle, J. Sveen, and A. Aamodt. Improved efficiency of oil well drilling through case-based reasoning. *LNCS*, 1886, 2000.

[178] S. Slade. Case-based reasoning: A research paradigm. *AI Magazine*, 12(1):42–55, 1991.

[179] K. Smith and P.A. Hancock. Situation awareness is adaptive, externally directed consciousness. *Human Factors*, 37(1):137–148, 1995.

[180] I. Sommerville. *Software Engineering*. Addison Wesley, 8th edition, 2006.

[181] S.L. Star and A. Strauss. Layers of silence, arenas of voice: The ecology of visible and invisible work. *Computer Supported Cooperative Work (CSCW): An International Journal*, 8:9–30, 1999.

[182] G.I. Susman and R.D. Evered. An assessment of the scientific merits of action research. *Administrative Science Quarterly*, 23 (4):582–603, 1987.

[183] D. Sy. Adapting usability investigations for agile user-centered design. *Journal of Usability Studies*, 2 (3):112–130, 2007.

[184] Y.T. Tenney, W.H. Rogers, and R.W. Pew. Pilot opinions on high level flight deck automation issues: Toward the development of a design philosophy. Technical Report Report 4669, NASA Langely Research Center, 1995.

[185] H. Teo. An activity-driven model for conftext-awareness in mobile computing. In *International conference on human-Computer interaction with mobile devices and services*, 2008.

[186] S. Ting, S. Zhou, and N. Hu. A computational model of situation awareness for MOUT simulation. In *International conference on Cyberworlds*, 2010.

[187] J.G. Trafton. Predicting complex information using qualitative mental models. In *First human performance, situation awareness, and automation conference, Savannah, Georgia*, 2000.

[188] E. Trist. Engaging with large-scale systems. In A. Clark, editor, *Experimenting with Organizational Life: The Action Research Approach*, pages 43–75. New York: Plenum, 1976.

[189] E. Tulving. Precis of elements of episodic memory. *The Behavioral and Brain Sciences*, 7 (2):223–268, 1984.

[190] E. Turban, J.E. Aronson, and T. Liang. *Decision support systems and Intelligent systems.* Prentice Hall, 7th edition, 2004.

[191] R.M. Turner. Model of explicit context representation and use for intelligent agents. In *International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-99)*, 1999.

[192] S. Vacek, T. Gindele, J.M. Zollner, and R. Dillmann. Situation classification for cognitive automobiles using case-based reasoning. In *IEEE Intelligent vehicle symposium, Istanbul, Turkey*, 2007.

[193] M.A. Vidulich. Testing the sensitivity of situation awareness metrics in interface evaluations. In M.R. Endsley and D.J. Garland, editors, *Situation awareness analysis and measurement*, pages 227–246. Lawrence Earlbaum Associates, Mahwah, NJ., 2000.

[194] M.A. Vidulich, M. Stratton, M. Crabtree, and G. Wilson. Performance-based and physiological measures of situational awareness. *Aviation, Space and Environmental Medicine*, 65:7–12, 1994.

[195] H. Wang, J. Zhou, Z. He, and J. Sha. An uncertain information fusion method for fault diagnosis of complex system. In *International conference on machine learning and cybernetics*, 2003.

[196] G.F. Wilson. Strategies for psychophysiological assessment of situation awareness. In M.R. Endsley and D.J. Garland, editors, *Situation awareness analysis and measurement*, pages 175–188. Lawrence Earlbaum Associates, Mahwah, NJ., 2000.

[197] T. Wood-Harper. Research methods in information systems: Using action research. In E. Mumford, editor, *Research Methods in Information Systems*, pages 169–191, North-Holland, Amsterdam, 1985.

[198] D.D. Woods, J. O'Brien, and L.F. Hanes. *Handbook of Human Factors and Ergonomics*, chapter Human Factors Challenges in process Control: The Case of Nuclear Power Plants. Wiley, New York, 1987.

[199] H. Wu, A. Marshall, W. Yu, and Y.M. Cheng. Applying hta method to the design of context-aware indoor navigation for the visually-impaired. In *4th Intl. conference on Mobile Technologies, Applications and Systems*, 2007.

[200] A. Zimmermann. Context-awareness in user modelling: Requirements analysis for a case-based reasoning application. In *International conference on case based reasoning*, 2003.

[201] D. Zimmermann and L. Grotzbach. A requirement engineering approach to user centered design. *HCI 2007. LNCS, Springer, Heidelberg*, 4550:360369, 2007.

Published Papers

- N.D. Nwiabu, I. Allison, P. Holt, P. Lowit, and B. Oyeneyin. Situation Awareness in Context-Aware Case-Based Decision Support. In IEEE International Conference on Cognitive Methods in Situation Awareness and Decision Support, Miami Beach, FL, pages 9-16, 2011.

- N.D. Nwiabu, I. Allison, P. Holt, P. Lowit, and B. Oyeneyin. User-Interface Design for Situation-Aware Decision Support Systems. In IEEE International Conference on Cognitive Methods in Situation Awareness and Decision Support, New Orleans, LA, pages 332-339, 2012.

- N.D. Nwiabu, I. Allison, P. Holt, P. Lowit, and B. Oyeneyin. Case-Based Situation Awareness. In IEEE International Conference on Cognitive Methods in Situation Awareness and Decision Support, New Orleans, LA, pages 22-29, 2012.

Appendix A

Hydrate preventive task data

Table A.1: Preventive task data

No	**Gas composition**	**Pressure (MPa)**	**Tempt. (K)**	**Prevention**
1	100%C2H4	0.665	273.7	Thermal method - Line heating
2	*	0.739	275.2	*
3	*	0.920	277.2	K-value, Line heating
4	*	1.010	278.2	K-value, Molecular sieve
5	*	1.439	281.2	*
6	*	1.838	283.2	Supplemental methanol
7	*	2.345	285.2	Hammerschmidt eqn.
8	*	2.830	286.2	*
9	*	3.210	287.2	K-value, Depressurization
10	5.60%C2H4 + 94.40%C2H4	0.712	273.7	Thermal method - Line heating
11	*	1.178	278.2	*
12	*	1.592	281.2	Molecular sieve
13	*	1.956	283.2	Supplemental methanol
14	*	2.916	286.2	*

15	34.09%CH4 + 65.91%C2H4	0.784	273.7	Thermal method - Line heating
16	*	1.292	278.2	*
17	*	1.755	281.2	Silica gel
18	*	2.220	283.2	Supplemental glycol
19	*	3.115	286.2	Glycol dehydrating, Depressurization
20	64.28%CH4 + 35.72%C2H4	1.146	273.2	Thermal method - Line heating
21	*	1.875	278.2	*
22	*	2.406	281.2	Hammerschmidt eqn.
23	*	3.120	283.2	Hammerschmidt eqn., Depressurization
24	85.69%CH4 + 14.31%C2H4	1.800	273.7	Thermal method - Line heating
25	*	2.714	278.2	*
26	*	3.758	281.2	Thermal, Depressurization
27	*	4.640	283.2	Methanol, Depressurization
28	92.87%CH4 + 7.13%C2H4	2.230	273.7	Thermal method - Line heating
29	*	3.448	278.2	Methanol, Depressurization
30	*	4.720	281.2	Molecular wt.
31	*	6.002	283.2	Depressurization, Molecular wt.
32	28.04%CH4 + 71.96%C3H6	0.529	273.7	Thermal method - Line heating
33	*	1.081	278.2	Supplemental methanol
34	*	1.515	281.2	Supplemental glycol
35	*	1.963	283.2	Silica gel

36	92.40%CH4 + 7.60%C3H6	1.081	273.7	Glycol dehydrating
37	*	1.765	278.2	Line heating
38	*	2.501	281.2	K-value
39	*	3.161	283.2	Methanol, Depressurization
40	96.60%CH4 + 3.40%C3H6	1.421	273.7	Thermal method - Line heating
41	*	2.381	278.2	Line heating, Silical gel
42	*	3.287	281.2	Glycol, Depressurization
43	*	4.121	283.2	*
44	99.34%CH4 + 0.66%C3H6	2.531	273.7	Thermal method - Line heating
45	*	3.681	278.2	Molecular sieve, Glycol
46	*	5.179	281.2	Supplemental glycol
47	*	6.585	283.2	Depressurization, glycol

Appendix B

Task Knowledge Structure

Below is the Task Knowledge Structure (TKS). TKS as presented here, is the summary of the design generation technique encompassing the process of modelling the system task and task knowledge that users posses.

User:

0. user to understand hydrate formation situation

- *1. specify gas type (user's task)*
- *2. Provide the context (user's task)*
    - *2.1. provide composition*
    - *2.2. provide wellhead pressure*
    - *2.3. provide wellhead temperature*
    - *2.4. provide location*
    - *2.5. provide distance*
    - *2.6. provide time*
- *3. recognise the context (system task)*
    - *3.1. obtain context attributes*
    - *3.2. compare attributes with valid attributes*
    - *3.3. declare context*
- *4. provide state of the environment (sensor task)*

    4.1. specify wind direction

    4.2. specify wind speed

    4.3. specify solar radiation

5. assess situation of the environment with context (system task)

    5.1. obtain context

    5.2. perceive

    5.3. comprehend

    5.4. project

6. present situation awareness (SA) (system task)

    6.1. discard irrelevant SA

    6.2. select relevant SA based on context

    6.3. display SA

7. query case base with SA and context information (user's task)

8. assess similarity (system task)

    8.1. obtain query

    8.2. case matching

    8.3. ignore non-similar cases

    8.4. identify similar cases

    8.5. identify best match

9. retrieve similar cases (system task)

10. present similar past cases (system task)

11. find explanations using domain rules (user-system task)

    11.1. obtain solutions

    11.2. search for meanings

    11.3. present meanings

12. find alternative solutions using domain rules (user-system task)

    12.1. pick variables

    12.2. select appropriate problem-solving model

    12.3. calculate

13. reuse solutions (user's task)

14. modify solutions (user's task)

15. preserve solutions (user-system task)

```
Plan 0: do 1 - 13 in that order
        if solutions can not effectively solve the problem
        then 14
        else 15
Plan 1: do 1 - 11 in that order
        if no similar past problem is found
        then 12
        else 13 - 15
```

Appendix C

SUMI scores

SUMI Scoring Report from SUMISCO 7.38

Time and date of analysis: 12:53:05 on 19-10-2011

Files used in this analysis:

SUMI English (UK) Language Items

SUMI Version 2.1 Scoring Keys

distributions from 1998 standardisation

weights from 1998 standardisation

population parameters from 1998 standardisation

Data file analysed: sample.ASC: Hydrate formation prediction system: 19/10/2011

Number of users analysed: 10

```
Profile Analysis

Scale         UF   Ucl   Medn Lcl   LF

Global         89   77   73   69   59

Efficiency   96   86   79   73   57

Affect         78   71   68   64   57

Helpfulness   89   78   75   71   56

Control        86   79   75   72   65

Learnability 88   85   82   78   72
```

Note:

The Median is the middle score when the scores are arranged in numerical order. It is the indicative sample statistic for each usability scale.

The Ucl and Lcl are the Upper and Lower Confidence Limits. They represent the limits within which the theoretical true score lies 95% of the time for this sample of users.

The UF and LF are the Upper and Lower Fences. They represent values beyond which it may be plausibly suspected that a user is not responding with the rest of the group: the user may be responding with an outlier.

Individual User Scores

```
User Globa Effic Affec Helpf Contr Learn
 1     78  83    79  67    86     87  001  (AC)
 2     70  70    67  78    72     71  002  (L)
 3     79  83    68  76    84     86  003
 4     80  91    70  80    79     82  004
 5     69  64    71  63    74     81  005
 6     67  71    64  68    66     77  006
 7      71  76    64  74    74    82  007
 8     58  57    61  67    68     72  008  (GE)
 9     80  89    71  78    78     77  009
 10     74  82    64  76    77     82  010
```

Any scores outside the interval formed by the Upper and Lower Fences are potential outliers. The user who produced an outlier is indicated in the right hand column. The initial letter of the scales in which outliers are found are indicated in parentheses.

Item Consensual Analysis

In the following table, the numbers in the row labelled 'Profile' are the observed responses of the actual users to each item.

The numbers in the row labelled 'Expected' are the number of responses expected on the basis of the standardisation database.

The Goodness of Fit between the observed and expected values is summarised using Chi Square, and these statistics are presented on the line below the expected values.

The number at the end of the Goodness of Fit line is the total Chi Square which applies to that item. The greater the value of the total Chi Square, the more likely it is that the obtained values differ from what is expected from the standardisation database.

Each total Chi Square marked with

*** is at least 99.99% certain to be different

** is at least 99% certain to be different

* is at least 95% certain to be different

Total Chi Square values without asterisks are not likely to differ much from the standardisation database.

In this output, the SUMI items which differ most from the standardisation are presented first.

```
I would not like to use this software every day.

Item 22     Agree Undecided Disagree

Profile       1     8         1

Expected     2.15  1.68      6.17

Chi Sq       0.61  23.68    4.33    28.62***


This software seems to disrupt the way I normally like to arrange my work.

Item 16     Agree Undecided Disagree

Profile        0       9      1

Expected      0.97    2.39    6.64

Chi Sq        0.97    18.33  4.79   24.09***


I would recommend this software to my colleagues.

Item 2         Agree Undecided Disagree

Profile          1     9        0

Expected       5.98  2.68    1.35

Chi Sq         4.15  14.95  1.35   20.44***
```

Using this software is frustrating.

Item 27	Agree	Undecided	Disagree	
Profile	0	7	3	
Expected	1.71	2.14	6.15	
Chi Sq	1.71	11.05	1.61	14.37***

There have been times in using this software when I have felt quite tense.

Item 32	Agree	Undecided	Disagree	
Profile	0	5	5	
Expected	3.78	1.52	4.7	
Chi Sq	3.78	7.97	0.02	11.77**

Working with this software is mentally stimulating.

Item 17	Agree	Undecided	Disagree	
Profile	0	8	2	
Expected	4.05	3.46	2.49	
Chi Sq	4.05	5.96	0.09	10.11**

The software has at some time stopped unexpectedly.

Item 4	Agree	Undecided	Disagree	
Profile	0	3	7	
Expected	4.69	1.06	4.25	
Chi Sq	4.69	3.57	1.77	10.04**

Getting data files in and out of the system is not easy.

Item 49	Agree	Undecided	Disagree	
Profile	0	9	1	
Expected	1.38	4.1	4.52	
Chi Sq	1.38	5.84	2.74	9.96**

The software hasn't always done what I was expecting.

Item 41	Agree	Undecided	Disagree	
Profile	1	6	3	
Expected	4.65	2.26	3.09	
Chi Sq	2.86	6.17	0.0	9.04*

I enjoy my sessions with this software.

Item 7	Agree	Undecided	Disagree	
Profile	3	7	0	
Expected	5.81	2.83	1.36	
Chi Sq	1.36	6.13	1.36	8.85*

Either the amount or quality of the help information varies across the system.

Item 43	Agree	Undecided	Disagree	
Profile	0	5	5	
Expected	2.96	5.21	1.82	
Chi Sq	2.96	0.01	5.52	8.5*

The speed of this software is fast enough.

Item 29	Agree	Undecided	Disagree	
Profile	10	0	0	
Expected	5.61	1.67	2.73	
Chi Sq	3.44	1.67	2.73	7.84*

I think this software is inconsistent.

Item 21	Agree	Undecided	Disagree
Profile	0	6	4
Expected	1.39	2.58	6.03

Chi Sq	1.39	4.51	0.68	6.58*

I find that the help information given by this software is not very useful.

Item 8	Agree	Undecided	Disagree	
Profile	1	7	2	
Expected	2.22	3.21	4.57	
Chi Sq	0.67	4.47	1.44	6.58*

I sometimes don't know what to do next with this software.

Item 6	Agree	Undecided	Disagree	
Profile	0	1	9	
Expected	3.2	1.73	5.07	
Chi Sq	3.2	0.31	3.05	6.56*

It is easy to make the software do exactly what you want.

Item 39	Agree	Undecided	Disagree	
Profile	8	2	0	
Expected	4.16	3.44	2.39	
Chi Sq	3.54	0.61	2.39	6.54*

Error prevention messages are not adequate.

Item 38	Agree	Undecided	Disagree	
Profile	1	8	1	
Expected	2.49	4.05	3.47	
Chi Sq	0.89	3.86	1.76	6.51*

The software has a very attractive presentation.

Item 42	Agree	Undecided	Disagree
Profile	2	4	4

Expected	5.63	2.72	1.65	
Chi Sq	2.34	0.61	3.34	6.29*

I have to look for assistance most times when I use this software.

Item 50	Agree	Undecided	Disagree	
Profile	0	4	6	
Expected	1.32	1.46	7.22	
Chi Sq	1.32	4.41	0.21	5.93

The software documentation is very informative.

Item 15	Agree	Undecided	Disagree	
Profile	0	7	3	
Expected	3.48	4.99	1.53	
Chi Sq	3.48	0.81	1.42	5.72

I prefer to stick to the facilities that I know best.

Item 20	Agree	Undecided	Disagree	
Profile	2	5	3	
Expected	4.21	2.08	3.71	
Chi Sq	1.16	4.09	0.14	5.39

It is relatively easy to move from one part of a task to another.

Item 44	Agree	Undecided	Disagree	
Profile	4	4	2	
Expected	6.9	1.54	1.55	
Chi Sq	1.22	3.91	0.13	5.26

It is easy to forget how to do things with this software.

Item 45	Agree	Undecided	Disagree

```
Profile       0       4       6

Expected    2.41    1.78    5.81

Chi Sq      2.41    2.79    0.01   5.2
```

This software is awkward when I want to do something which is not standard.

```
Item 24    Agree Undecided Disagree

Profile       0       6       4

Expected   3.06   4.09   2.85

Chi Sq     3.06   0.89   0.47   4.42
```

The software has helped me overcome any problems I have had in using it.

```
Item 28    Agree Undecided Disagree

Profile       0      6      4

Expected    2.76   5.0     2.25

Chi Sq      2.76  0.2     1.37   4.33
```

I think this software has made me have a headache on occasions.

```
Item 37     Agree Undecided Disagree

Profile        0      4      6

Expected     2.42   2.09   5.49

Chi Sq       2.42   1.75   0.05   4.21
```

I sometimes wonder if I am using the right command.

```
Item 11    Agree Undecided Disagree

Profile       1       1       8

Expected    3.55    1.59    4.86

Chi Sq      1.83    0.22    2.02    4.07
```

Learning to operate this software initially is full of problems.

Item 5	Agree	Undecided	Disagree	
Profile	0	1	9	
Expected	2.16	1.85	5.99	
Chi Sq	2.16	0.39	1.52	4.07

Working with this software is satisfying.

Item 12	Agree	Undecided	Disagree	
Profile	4	6	0	
Expected	5.38	3.25	1.37	
Chi Sq	0.35	2.32	1.37	4.04

This software occasionally behaves in a way which can't be understood.

Item 46	Agree	Undecided	Disagree	
Profile	1	5	4	
Expected	3.22	2.52	4.26	
Chi Sq	1.53	2.44	0.02	3.99

I will never learn to use all that is offered in this software.

Item 40	Agree	Undecided	Disagree	
Profile	0	4	6	
Expected	2.83	2.76	4.41	
Chi Sq	2.83	0.56	0.57	3.97

There is too much to read before you can use the software.

Item 25	Agree	Undecided	Disagree	
Profile	0	1	9	
Expected	1.61	2.21	6.18	
Chi Sq	1.61	0.66	1.29	3.57

The instructions and prompts are helpful.

Item 3	Agree	Undecided	Disagree	
Profile	6	4	0	
Expected	6.23	2.13	1.64	
Chi Sq	0.01	1.64	1.64	3.29

It is easy to see at a glance what the options are at each stage.

Item 48	Agree	Undecided	Disagree	
Profile	6	4	0	
Expected	5.58	2.38	2.04	
Chi Sq	0.03	1.11	2.04	3.18

This software is really very awkward.

Item 47	Agree	Undecided	Disagree	
Profile	2	2	6	
Expected	0.67	1.76	7.57	
Chi Sq	2.65	0.03	0.33	3.01

The way that system information is presented is clear and understandable.

Item 13	Agree	Undecided	Disagree	
Profile	6	4	0	
Expected	6.3	2.2	1.5	
Chi Sq	0.01	1.48	1.5	3.0

This software responds too slowly to inputs.

Item 1	Agree	Undecided	Disagree	
Profile	0	1	9	
Expected	1.9	1.53	6.57	
Chi Sq	1.9	0.18	0.9	2.98

I feel safer if I use only a few familiar commands or operations.

Item 14	Agree	Undecided	Disagree	
Profile	3	4	3	
Expected	4.04	1.87	4.09	
Chi Sq	0.27	2.41	0.29	2.97

If this software stops it is not easy to restart it.

Item 9	Agree	Undecided	Disagree	
Profile	0	6	4	
Expected	1.62	3.86	4.53	
Chi Sq	1.62	1.19	0.06	2.87

I keep having to go back to look at the guides.

Item 30	Agree	Undecided	Disagree	
Profile	0	3	7	
Expected	1.99	2.23	5.78	
Chi Sq	1.99	0.27	0.26	2.51

Learning how to use new functions is difficult.

Item 35	Agree	Undecided	Disagree	
Profile	0	2	8	
Expected	1.54	2.45	6.01	
Chi Sq	1.54	0.08	0.66	2.28

The software allows the user to be economic of keystrokes.

Item 34	Agree	Undecided	Disagree
Profile	8	2	0
Expected	6.14	2.5	1.35

```
Chi Sq       0.56    0.1    1.35    2.02
```

It takes too long to learn the software commands.

```
Item 10      Agree Undecided Disagree
Profile         0      1       9
Expected     1.07    1.75    7.18
Chi Sq       1.07    0.32    0.46    1.85
```

There are too many steps required to get something to work.

```
Item 36      Agree Undecided Disagree
Profile         1       1      8
Expected     2.07    1.99  5.94
Chi Sq       0.56    0.49   0.72    1.76
```

Tasks can be performed in a straight forward manner using this software.

```
Item 26      Agree Undecided Disagree
Profile         8       2       0
Expected      6.95    1.72    1.33
Chi Sq        0.16    0.05    1.33     1.53
```

I can understand and act on the information provided by this software.

```
Item 23      Agree Undecided Disagree
Profile         8      2      0
Expected     7.23    1.83   0.94
Chi Sq       0.08    0.02   0.94    1.04
```

It is obvious that user needs have been fully taken into consideration.

```
Item 31      Agree Undecided Disagree
Profile         3       5       2
```

Expected	4.05	3.61	2.34	
Chi Sq	0.27	0.54	0.05	0.86

There is never enough information on the screen when it's needed.

Item 18	Agree	Undecided	Disagree	
Profile	2	3	5	
Expected	1.72	2.24	6.04	
Chi Sq	0.05	0.26	0.18	0.49

The organisation of the menus or information lists seems quite logical.

Item 33	Agree	Undecided	Disagree	
Profile	8	1	1	
Expected	7.16	1.61	1.23	
Chi Sq	0.1	0.23	0.04	0.37

I feel in command of this software when I am using it.

Item 19	Agree	Undecided	Disagree	
Profile	6	2	2	
Expected	5.68	2.64	1.68	
Chi Sq	0.02	0.16	0.06	0.24