User Centred Design Approach to Situation Awareness

Nuka Nwiabu
School of Computing
The Robert Gordon University
Aberdeen, United Kingdom

Ibrahim Adeyanju
Computer Science & Engineering department
Ladoke Akintola University of Technology
Ogbomoso, Nigeria

ABSTRACT
Situation awareness is a variety of cognitive processes employed by operators in a complex and dynamic environment to understand the current state of the environment in order to know its future state. Information systems designed on the principle of situation awareness adopt the technique of user-centred design (UCD) methodology. A UCD approach involves users in system development process to make the system meet end-users requirements. Such user-friendly systems result in better decision making and optimal performance. In this paper, we model situation awareness with agile user-centred development method for predicting hydrate formation during crude oil drilling. The design is implemented with Case-Based Reasoning, an experiential machine learning paradigm, to expand the scope of the decision-making module.

General Terms
Information Systems, Human Computer Interaction.

Keywords
Situation awareness, User-centred design, Case-based reasoning, Agile delopment.

1. INTRODUCTION
The attempt to find a solution to human errors using decision support systems (DSS) sometimes leads to more complexity, greater errors and more cognitive load [1]. Some findings have shown that automation of tasks has caused more problems than it has solved mostly due to decision support systems being technology-centred, i.e. a situation where systems are design without much emphasis on end-users' requirements. End-users' requirements include their information processing need. Human beings naturally have an information processing bottleneck and as such can only attend to limited volume of information at a time [1]. Decision support systems that place less emphasis on users' information processing needs do not always result in good performance [2]. An over-reliance on automation, and poor human monitoring capabilities was said to have contributed to circumstances of the accident involving an American airline, Boeing 757 that struck a mountain at Colombia on the 20th of December, 1995 while descending for landing [1].

Metzger et al [3] carried a study on training and design of methods to solve the problem of over-reliance on automation. Users can be made to overcome over-reliance on automation, and have good control over information systems either by giving them some special training or by designing systems using alternative methods or both. The result of the study suggests that design and training reduces users' complacency on automation as they provide users with the flexibility in attention strategies needed in multi-task environments.

The problem associated with user's over-reliance on automation can be addressed by adopting the user-centred design (UCD) approach in developing situation awareness systems. UCD supports user's goals and tasks, user's way of processing information to make decision, and the user's control and knowledge of the system [4]. 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 stages in the project to ensure that the system meets their requirements. Donald Norman [5] 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".

One of the techniques adopted in our proposed user-centred design is scenarios. Scenarios represents the entire activities, describing the social settings, resources, and goals of users, looking at the "big picture" of how a work is done and not a narrow description of the task [6]. Scenario analysis in this work helps in one of the fundamental tasks in object-oriented design which is developing a problem domain model [7]. Scenarios are simplified to produce a set of propositions that are subjected to objected-oriented analysis which in turn generate objects. The objects are identified, and the methods and interactions that would produce the behaviour are defined [8]. These processes are achieved by using domain knowledge and object-oriented system design skills to elaborate on the explicit knowledge provided by the scenarios. This method cannot be possible using functional specification alone. Although, traditional functional specifications are easily understood by system builders as the specifications are well laid out in non redundant order as expected in the system, they may not be user-centred. Functional specifications focus on the technology of the system and do not express the psychological or work context for the technology in use such as users actions, goals, and expectations, which could make the actions to be meaningful or problematic [8]. Scenarios are open-ended, as they are used in design; new questions emerge, which can be answered only by returning to the user [9]. 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 [8].

To integrate human computer interaction (HCI) concerns with software engineering, UCD is integrated to agile iterative development. Iterative development is one of the software development approaches since the early days of software development. Iterative approaches would have by now replaced the single-pass waterfall model but for the fact that waterfall method gives somebody a feeling of knowing exactly what the end result will be, and how much the process is going to cost, the method is in use in some quarters, especially amongst managers [10]. But, the waterfall model fails to adapt to discoveries made during the design and implementation phase. There is hardly the opportunity to make adjustments once the analysis phase is over. In normal situations, end-result of projects changes as it runs over a

period of time. The rigidity of the waterfall model do not allow for constantly evolving products and services. In fact, the waterfall model does not recognise the fact that conditions and goals change with time [11]. As the advantages of iterative methods are becoming widely known, the shortcomings of the waterfall method are becoming clearer. A generic term for a number of iterative development methods is the "agile" development.

Agile development can be 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" [12]. Agile development is an iterative method designed to solve the problems of the waterfall method. Evaluation in agile development is done many times during the project to give room for a change in direction if need be. Where there is need for a change, the constant evaluations enable the designer to redesign at an early stage, saving time and resources [13].

This paper argues that an approach integrating agile development concepts and user-centred design enables a situation-aware system to meet the functional and usability needs of the user. Each of these is iterative, action-based learning approaches. Drawing them together enables the building of a series of solutions based on agreed priorities, user-related activities and constant evaluation. Out of this a synergy of practice-based solutions and theoretical developments are drawn.

The remainder of the paper is as follows. The following section provides the methodology for research and design of our system. The design process and how it can be applied in a problem domain (hydrate formation) is then presented in Sections 3 and 4. The system design and related methodology are evaluated from that application in Section 5. Finally, the paper is summarized and concluded in Section 6.

2. INTEGRATED APPROACH

In this work, action Research (AR), user-centred design (UCD), and the agile development (AD) methodologies will be linked together to form a comprehensive research-design cycle. The usefulness of action research methods is that, it links theory and practice, thinking and doing, reflects on the process and the product, achieving practical as well as research objectives [14]. It addresses two challenges, "action" and "research" [15]. In other words, action research addresses social issues in a practical fashion and also makes a contribution to developing and testing theory. This is made possible through cycles of action and reflection with the outcomes of each cycle checked against set plans and goals. The integration of these different methods results in a research-design process comprising three segments; scenarios, agile development, and business change (evaluation).

The starting segment of the research-design process is the domain analysis using scenarios. Even though scenarios are generated at the first segment, they evolve throughout the project lifecycle. Scenarios are comprised of problem, diagnosis, and action planning. The second segment is a user-centred design by agile development method. It is an iterative and evolutionary development comprising of requirement analysis, design, prototype, and design evaluation. When the process of design is completed, the result is taken to the research action-taking level in the business change segment, wherein collaboration with practitioners, an intervention strategy is adopted to ensure that the design solves organisational problems.

After action-taking, there is collaboration with practitioners to evaluate the outcome of the implementation, assessing the effect of the theoretical concept on practical problem solving. Where the research questions are answered, the research process ends after evaluation. But, where the research questions are not answered, there is an adjustment in the thinking, and specifying a new direction (learning) that again pass through scenarios to agile development and then back to the business change segment. The iterative research and design cycles continues until the research questions are answered.

3. DESIGN

Here, we discuss details of the individual components of our integrated approach to a user-centred design process. These components are scenarios, requirements analysis and interaction standards development.

3.1 Scenario

Scenario-based user-centred design is used in this work because of the evolutionary and question-asking nature of scenarios which fills the gap of the author's lack of strong intuitions about the domain [8]. Also, scenarios are now the best means available for the purpose of designing today's systems [16]. Scenarios in this work identify issues to be investigated, interpret the problem (diagnosis), and draw up a theoretical framework (action planning) which indicate the future state of the organisation and the changes that would achieve such a state [14].

The problem domain is analysed by task description and by initial scenarios specification. There were three structured interviews with domain practitioners to understand the tasks together with the knowledge of why tasks are carried out the way they are carried out. This involved probing of the social, historical, and cognitive rationale of users' tasks. To diagnose the identified problem the following scenario was explored through requirements analysis.

A team of ten engineers working on the prediction of hydrate in a two-phase flow system (water + gas) with the assistance of situation-aware decision support systems investigated hydrate formation conditions of (CH4 + C2H4) and (CH4 + C3H6) binary gas mixture in the presence of pure water using "pressure search" method. The temperature under investigation ranges between 273.7-287.2k and pressure ranges between 0.53-6.6MPa. Ethylene content in the gas mixtures varied from 7.13 to 100%, and the propylene content varied from 0.66 to 71.96 mol%. Each of these variations of the percentage composition of the guest molecules constituted a context for investigation.

3.2 Requirements analysis

The object-oriented method is 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 forms the basis for the requirements analysis using the Unified Modelling Language (UML). In requirements analysis, the functional and non-functional requirements are analysed. Diary studies and personal development are used to understand the functional requirements of the users. For a complete analysis which includes non-functional requirements, information from the three structured interviews, contextual inquiry and task analysis with domain practitioners were used.

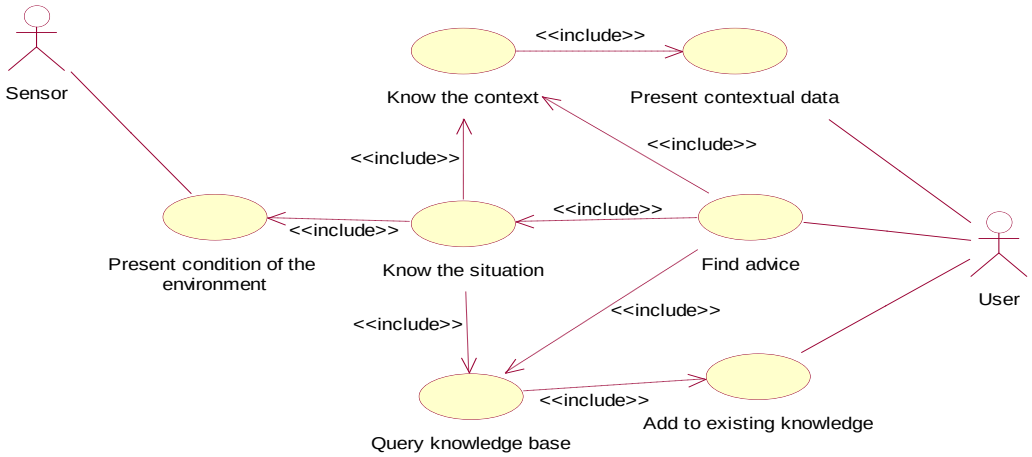

Fig 1. Use case diagram modelling of user-sensor interactions

There was harmonization of the functional and non-functional requirements in order to have the requirement specifications that served as a roadmap for the design. We constructed scenarios of interaction and functionalities detailed by 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 as shown in Figure 1.

There are two actors, "User" and "Sensor" and seven use cases in the modelling of the initial requirements. A user presents contextual data to the system to enable the system to know the context the user intends to work from. The stereotype <<include>> between the use case "know the context" and the use case "present contextual data" means that the former is the base use case dependent on the latter, the inclusion use case. The second actor "sensor" presents the condition of the environment to the system for the system to know the situation in the environment. The use case "know the situation" depend on three use cases; "present condition of the environment", "know the context", and "query knowledge base". The "find advice" use case also depend on three use cases; "know the context", "know the situation", and "query knowledge base". The "query knowledge base" which is an inclusion use case to both "know the situation" and "find advice" is a base use case to "add to existing knowledge". Carrying out requirement analysis on the elaborated scenario (see scenarios) it was discovered that the above use cases depend on other use cases for example, "know the context" not only depend on "present contextual data" but also on "validate context". Also, it was discovered that "know the situation" is a generalization of "recognise the data", "understand current situation", and "predict future situation".

3.3 Interaction standards

The design is a conceptual design supported by an object-oriented Computer Aided Software Engineering (CASE) tool, Unified Modelling Language (UML) to give form to the functional requirements, and features from the users' view. The design produced an information architecture, and interaction standards development. There is a class for each object identified, and the classes have associations that correspond to the links between objects. The class diagram in Figure 2 also show the relationship between the classes. For example, the class "situation awareness" is a generalization of the classes "perception", "understanding", and "prediction". One "perception" receives condition of the environment from one to many "sensors". One "understanding" and one "prediction" are compared to from one to many "goals".

4. PROTOTYPE

This prototype is developed to receive feedback from users to enable the author have necessary information for further design. One of the questions this work addresses is whether this design can be implemented with machine learning capabilities. Machine learning is a discipline concerned with the development of algorithms that enable the behaviour of data, such as database to be changed. The whole essence of machine learning study is that a system should be able to recognise complex patterns and make intelligent decisions based on data.

This work's prototype is implemented with Case-based reasoning (CBR), a machine learning paradigm. Case-based reasoning was chosen because of its "revise" and "retain" facilities that is useful for learning by experience. The increasing mental models of CBR will help in the building of shared mental models for the project. We used a CBR tool called jCOLIBRI framework for our implementation. jCOLIBRI runs on Eclipse, a JAVA platform for building integrated development environments (IDEs). We developed a prototype to predict the formation of hydrate in gas pipelines. Data was obtained for this work from Ma et al's publication on hydrate formation of CH4 + C2H4 and CH4 + C3H6 gas mixtures [17], and Maekawa's work on Phase equilibria for hydrate formation from binary mixtures of ethane, propane and noble gases [18].

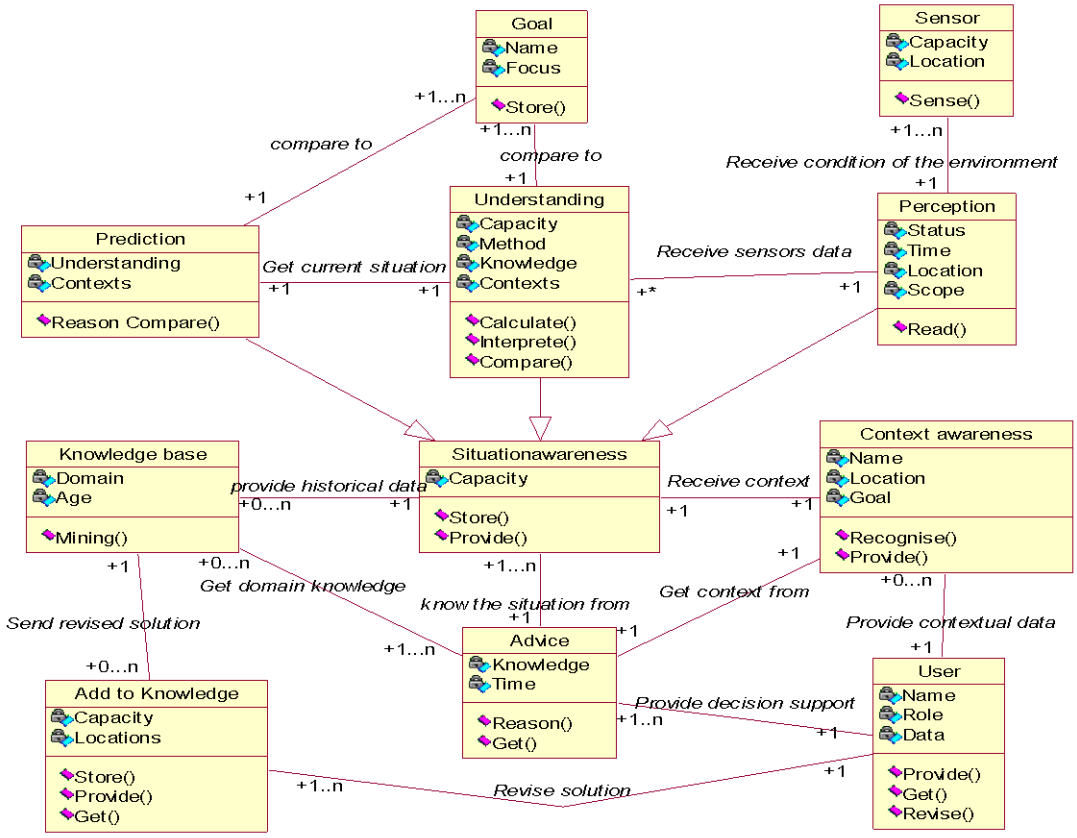

Fig 2: Class diagrams showing interaction standards

5. DESIGN EVALUATION

There are some direct interactions with end users to aid our requirements capture and also to enable us formulate a realistic usability evaluation plan. Performance evaluation of the system has been carried out to determine the effect of the theoretical concept on practical problem solving. Usability evaluation of the system by qualitative interaction with users was also carried out. The users form the focus group that provided feedback on the ease of use of the system. The response rate was 100%. Surveys were conducted on 10 users, 10 successfully completed their questionnaires and returned.

Table 1: Profile Analysis

Scale	UF	Ucl	Medn	Lcl	LF
Global	79	67	63	49	49
Efficiency	86	76	69	63	47
Affect	68	61	58	54	47
Helpfulness	79	68	65	61	46
Control	76	69	65	62	55
Learnability	78	75	72	68	62

Evaluating the user interface in Figure 3 with 10 users, the SUMI responses of each of the users involved in the study or evaluation are represented in ASCII characters. The 50 SUMI questions are grouped into 5 with each group of 10 questions coded as a block. The result of the interpretation of these responses by SUMISCO program is as shown in Table 1.

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.

The result is generally of good rating. The interface is very simple to understand with 72% median rating. The next high rating was the "efficiency" of the system with 69% score. This fairly high score on efficiency is due to much emphasis placed on the accurate prediction of hydrate formation during the design process. SUMISCO scored "Affect" the lowest. Affect represent the degree by which users like the system. Users were interviewed in respect of the low score. The feedback from users was that the interface should be further developed to contain buttons like SAVE, PRINT etc. The number (8) user was particularly interviewed for his general low rating of the system, and the feedback was that there was no HELP facility on the interface. Apart from the usability feedbacks from users, the system also has some performance drawbacks.

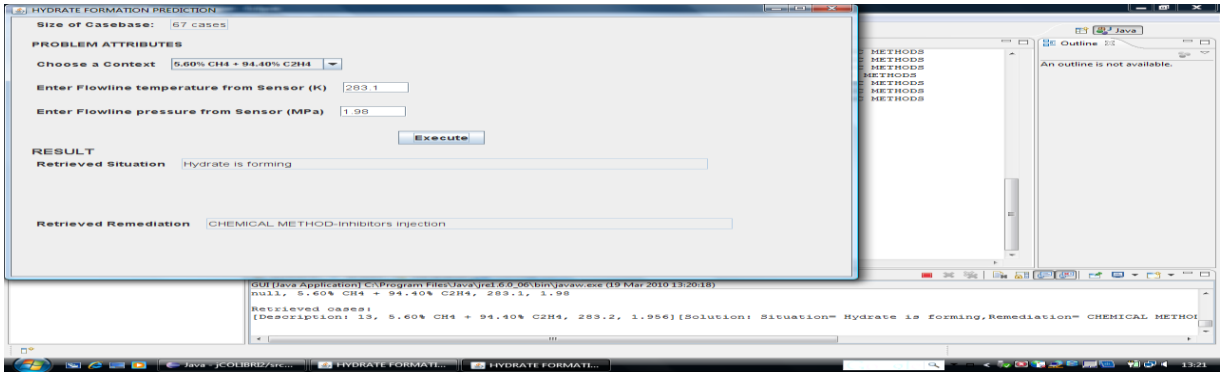

Fig. 3 Screenshot from our User interface prototype

The drawbacks of this first prototype are:

- It cannot understand situation from other scenarios.
- It is not flexible to accommodate context outside the predefined contexts.
- It cannot predict the location and time of hydrate formation.

Based on the above drawbacks, and the usability feedbacks from users, the agile development is currently undergoing refinement to ensure that the system satisfies the specified requirements required at the business change segment. The refined prototype will soon be ready for implementation at the next stage (business change segment).

6. CONCLUSIONS

The utilisation of user centred design methods ensures that there is collaboration between researchers and practitioners. But in most real-life situations, domain practitioners are always too busy for an effective collaborative work. In situations where researchers cannot exploit all the required methods due to non availability of the practitioners or where information are being withheld by practitioners, the domain analysis may provide inaccurate or incomplete information for requirements tasks.

By focusing on user-centred design, one develops understanding of the user, and thereby an understanding of why you are developing the system and who will be using the system. As the UCD process ensures an understanding of the users, the agile development model ensures that you can work iteratively, enable faster development of functional prototype, which are more easily communicated and tested, thus giving you better input for the next iteration.

7. REFERENCES

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

[2] Selcon, S. 1990. Decision support in the cockpit: Probably a good thing? Human Factors society.

[3] Metzger, U., Duley, J., Rovira, E., Parasuraman, R. 1999. Effects of training on monitoring of an automated system. Proceedings of the Human Factors and Ergonomics society 43rd annual meeting.

[4] Parker, C. and Sinclair, M. 2001. User-centred design does make a difference. The case of decision support systems in crop production. Behavour & Information Technology. Vol. 20. pages 449-460.

[5] Norman, D.A. 1986. User-Centered System Design: New perspective on Human-computer Interaction. Lawrence Earlbaum Associates. Hillsdale, N.J.

[6] Nardi, B. 1995. Some reflections on scenarios. John Wiley & Sons Inc. ISBN: 0-471-07659-7. pages 387-399.

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

[8] Robertson, S. 1995. Generating object-oriented design representations via scenario queries. John Wiley & Sons Inc. pages 279-308.

[9] Mark, R. 1995. Discussion: Scenarios as engines of design. John Wiley & Sons Inc. pages 361-386.

[10] Larman, C. and Basili, V.R. 2003. Iterative and incremental development. A brief history. Vol 36. pages 47-56.

[11] Lindstrom, H. and Malmsten, M. 2008. User-centred design and agile development: Rebuilding the Swedish national union catalogue. The Cod4Lib Journal. Vol 6. pages 12-15.

[12] Ambler, S.W. 2002. Agile Modeling: Effective Practices for XP and the UP. New York: John Wiley & Sons.

[13] Schwaber, K. and Beele, M. 2002. Agile software development with SCRUM. Printice Hall.

[14] Baskerville, R. and Wood-Harper, A. 1996. A critical perspective on action research as a method for information systems research. Journal of Information Technology. Vol. 11. pages 235-246.

[15] Avison, D., Baskerville, R., Myers, M. 2001. Controlling action research projects. Information technology & people. Vol. 14. pages 28-45.

[16] Rosson, M. and Carroll, J. 1995. Narrowing the specification-Implementation gap in scenario-based design. John Wiley & Sons Inc. pages 247-278.

[17] Ma, J., Zhang, G. 2008. Team situation awareness measurement using group aggregation and implication operators. IEEE 3rd International conference on intelligent system and knowledge engineering

[18] Maekawa, T. 2006. Phase equilibria for hydrate formation from binary mixtures of ethane, propane and noble gases. Fluid Phase Equilibria. Vol. 243. pages 115-120