INTERNATIONAL JOURNAL OF ENGINEERING AND MANAGEMENT SCIENCES
© 2004 -17 Society For Science and Nature (SFSN). All Rights Reserved
www.scienceandnature.org

REQUIREMENT ELICITATION FOR CONTEXT AWARE SYSTEMS (SELF-ADAPTABLE ELECTRONIC SMART DEVICES)

[1]Ovass Shafi Zargar and [2]AB. Qayoom Sofi
[1]Department of Computer Applications, Amar Singh College, Srinagar, J&K, India
[2]Department of Computer Applications, Govt Degree College, Pattan, Baramulla, J&K 193121, India

ABSTRACT
Context aware systems can be described as applications that are able to read certain contextual elements, reason about them and then adjust their behaviour so that it meets in the best possible way the user needs in the current operational context. Context aware computing brings new paradigms for computing models. In Context Aware Systems, a better interaction between the device and its environment is needed. Though the benefits of working with such a system are not hard to notice, context-aware systems are not yet very common. This could be because there is still a big gap between the system behaviour and user expectations. It is actually the requirements engineering part of a system design process that takes care that the user gets what he actually wants from the new system. Requirements engineering is special in a context-aware system because in contrast to a regular system the CAS is expected to perform in a versatile environment with properties that might change or even be unknown at design time. The context aware systems operate in dynamic environments. As such, a context-aware system must continuously monitor changes in its context and react accordingly. When engineering requirements for a context aware system we have to consider context as an input for the application. The application should be able to reason about this input and then generate a context based output. This paper will study the applicability of traditional software requirement engineering methods in Context Aware/ubiquitous systems, and underline their advantages and disadvantages in the development of Context aware systems. Such an analysis gives us a fundamental for the development of a requirement engineering method best suited for Context Aware Systems
The paper will devise a suitable Model that will be applicable for Elicitation of User Requirements in Context Aware Systems.

KEYWORDS: Context based systems, Requirement Elicitation, Dynamic Environment, Context based Input, Requirement Elicitation Model

INTRODUCTION

In modern era we are surrounded by electronic gadgets and are almost completely dependent upon various electronic devices. Modern electronic devices are far different from the electronic devices used a decade ago. Most of the modern electronic devices we use today are smart in the sense that they adjust their operations whenever there in change in the operating environment. The environment in which the device operates is known as the context for that device and affects the operations of the device to large extent. If a device adjusts itself as per the changing environment and provides suitable service as per new environmental conditions, the device is said to be context aware device, and the system is known as context aware system.

"Context aware systems can be described as applications that are able to read certain contextual elements, reason about them and then adjust their behaviour so that it meets in the best possible way the user needs in the current operational context."

Types of Context

The contexts that affect the operations of context aware system are:

1. **Location**: Place where the device is being operating
2. **Time**: A particular moment at which the service is being offered
3. **Activity**: It includes the environment and everything that happens in it at a given moment of time.
4. **Identity**: subject to whom service is to be offered.

The context aware system shall be able to observe its environmental properties and then adjust its behaviour according to the properties of the environment. The environment can be observed using some sort of sensor. The properties of environment that affect the operations of context aware system are broadly classified into two classes:

- Contextual Properties.
- Non-Contextual Properties

CONTEXTUAL PROPERTIES

1. **Dynamic Environment**
   The environment in which the device operates changes with time and the system should also change its operations accordingly.
2. **Display Characteristics**
   The system should change its display characteristics as per the changing environments. For example a smart phone should adjust its display characteristics

like brightness, contrast, colour etc. with the change in lighting attributes.

3. **Bandwidth:** The context aware systems that provide online services depend largely upon bandwidth of the line. The bandwidth varies from time to time and the system should enable/disable various features that depends upon bandwidth

NON CONTEXTUAL PROPERTIES

1. **Non Frequent User Interaction**: Users don't interacts continuously with the system
2. **Different User Interface**: Different Context Aware Systems have different User Interface to perform same kind of operation.
3. **Availability**: The context aware system should be available to provide the service to its users.
4. **Limited Computational Recourses**: The computational resources are limited in context aware systems (Limited Power Supply). The context aware systems should provide better service with limited computational resources.
5. **User Interacts with system in bursts**: When user interacts with the system, it results in heavy computation.

REQUIREMENT ELICITATION IN CONTEXT AWARE SYSTEMS

The requirement elicitation process is straight forward in traditional systems. There are number of fact finding techniques which are employed for elicitation of requirements at design time like onsite observations, interviews, and questionnaires. The requirements once established remains static and thus are implemented during the coding phase. Requirements engineering is special in context aware system because in contrast to a regular system the CAS is expected to perform in a versatile environment with properties that might change or even be unknown at design time. As such, a context-aware system must continuously monitor changes in its context and react accordingly.

In order to develop requirement elicitation method for context aware system, we should think about context modelling method. The context model method to be developed should focus on:

- Identification of variouscontext attributes whose change may affect the system.
- In what manner the attributes may change.
- What are the reasons for the change in attributes?
- How shall the System respond with change in attributes?
- Who will initiate the Change?
- Response to change of one attribute may cause change in another attribute.

Therefore one important objective is to study the interdependencies between the attributes of the system.

EXISTING METHODS

The traditional requirement elicitation methods are not suitable for the development of context aware systems as they focus on the requirements that are known well in advance at design time and are static in nature. However the context aware systems operate in highly dynamic environment where the requirements change frequently and the system shall adapt itself as per changing requirements.

It is better to use goal oriented requirement engineering methods (GORE) for eliciting requirements for context aware systems. The goal oriented requirement engineering methods focus on the use of goals for the elicitation

Goal-oriented requirements engineering (GORE) is concerned with the use of goals for eliciting, elaborating, structuring, specifying, analysing, negotiating, documenting, and modifying requirements. Thus GORE methods are best suitable for the elicitation of requirements for context aware systems.

GORE methods studythe system under development in its context which is essential for pervasive services as well. Secondly,GORE methods provide number of mechanisms to identify the properties of pervasiveservices mentioned above. This is because it studies the system under development in terms of its environment.

Some of the existing Goal Oriented Requirements Engineering methods are discussed below:

GBRAM

The Goal-Based Requirements Analysis Method (GBRAM) focuses on the initial identification and abstraction of goals from all available sources of information, regardless of the scope of the knowledge base. GBRAM consists of two types of activities – goal analysis activities and goal refinement activities. Key concepts in GBRAM are goals, stakeholders.

- Those who have interests in completion of identified goals, and agents
- Those who are responsible for completion of the goals.

These are identified during goal analysis processes and elaborated in the refinement processes. GBRAM goal analysis comprises the following activities:

Exploration –
Extraction and identification of all goals from existing sources of information

Identification –
Initial specification of goals and preparation for future elaboration;

Organizing –
Classification of goals and identification of interdependencies between goals.

The result of the goal analysis is an organized goal set, which is ready for further refinement and optimization. Goal refinement concerns three specific activities:

- Refinement of the goal set to prune its size, elimination of redundancies and reconciliation of synonymous goals
- Elaboration of the goals to uncover hidden goals and requirements
- Operationalization of the goals into operational requirements.

Generally, the GBRAM method provides a very elaborate and detailed framework for the definition and elaboration of goals, although some ideas and conclusions are questionable. However, transition from goals to requirements is ill-defined. GBRAM does not provide any specific mechanisms for explicit modelling of system's environment, which is a serious disadvantage for the design of pervasive services, since, as mentioned above, the main idea of such services is to be built in the environment on various and to react on changes in the environment.

KAOS

The Keep All Objectives Satisfied is a method that tries to support the entire process of requirements engineering – starting from the definition of high-level goals which have to be achieved up to the requirements, objects and operations which are assigned to the various agents in a system.

The KAOS methodology has three constituents:

- A specification language;
- An elaboration method;
- Meta-level knowledge.

The KAOS language has a two-level structure:

1. An outer semantic net layer for the declaration of concepts,
2. An inner formal assertion layer for the formal definition of these concepts.

The KAOS elaboration method consists of the following steps:

- Goal elaboration: elaboration of the AND/OR graph by defining goals and their refinements until assignable requisites are reached. This process is in general a combination of top-down and bottom-up approaches.
- Object capture: identification of the objects involved in goals formulations, definition of their relationships and of their properties
- Operation capture: identification of object state transitions, which are meaningful to the goals
- Operationalization: derivation strengthening on operation pre- and post- conditions in order to ensure that all requisites are met. This process is supported by a set of formal derivation rules.
- Responsibility assignment: identification of alternative responsibilities for requisites.

PROPOSED IDEA

The requirements elicitation process for traditional systems consists of following stages:

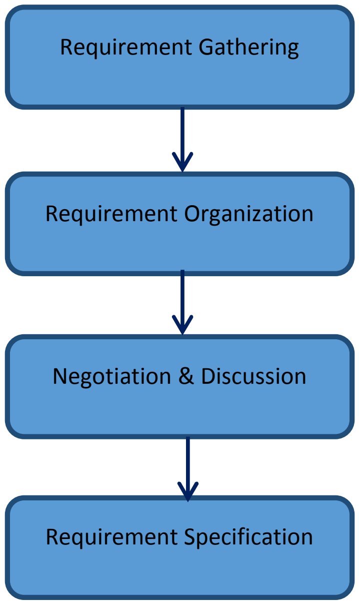
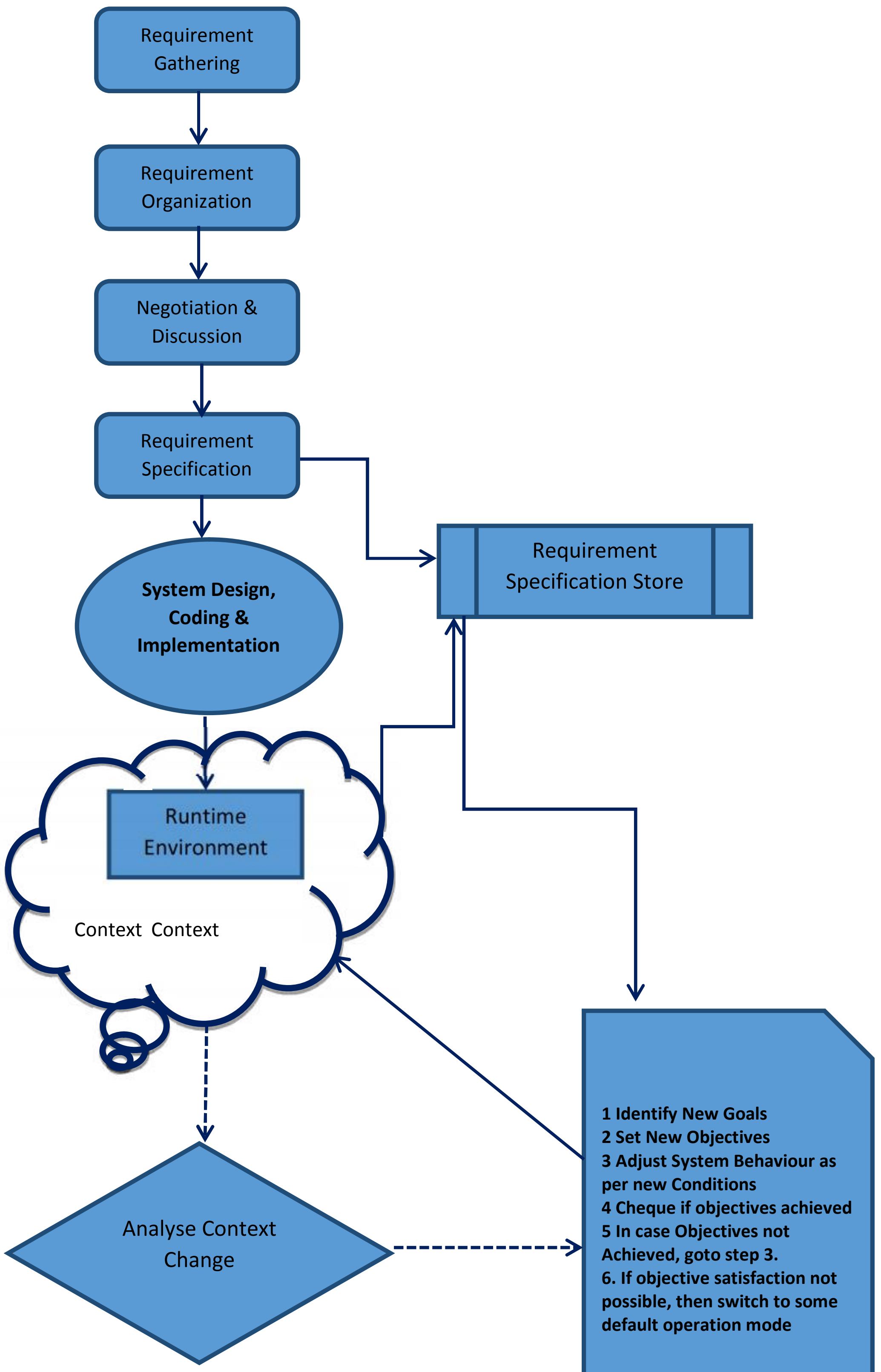


Figure: Requirement Elicitation Model for Traditional Software Systems.

The model given above is not suitable for context aware systems as the requirements may change during the runtime. Therefore the following model is proposed for the elicitation of requirements in context aware systems.

In addition to carryout requirement elicitation at design time, the process of requirement elicitation needs to be carried out at runtime also. Whenever there is any change in the context, the system shall detect the changes using sensors and analyse the change, search for the adaptation in knowledge base and behave as per changing requirements

Combo Model for Requirement Elicitation of Context Aware Systems.


Requirement Gathering
Requirement Organization
Negotiation & Discussion
Requirement Specification
Requirement Specification Store
System Design, Coding & Implementation
Runtime Environment
Context Context
Analyse Context Change
1 Identify New Goals
2 Set New Objectives
3 Adjust System Behaviour as per new Conditions
4 Cheque if objectives achieved
5 In case Objectives not Achieved, goto step 3.
6. If objective satisfaction not possible, then switch to some default operation mode

In the above figure, the design process starts with the normal requirement elicitation cycle that is applicable in traditional software development process. During the requirement elicitation processthe system analyst uses various requirement elicitation techniques like interviews, questionnaires, group discussions, onsite visits, onsite observations, etc., and creates a knowledgebase and stores information regarding different context changes and the corresponding responses to be provided by system to these changes in the knowledgebase if already known at design time. For example if a user stores some event like meeting date in his smart device's event calendar for some future date. On that date while in meeting, if a call arrives, it is obvious that the smart device shall first check the event calendar, and when finds that a person is most probably in meeting, the device shall search the knowledge base for required response and adjust its operations accordingly for example put the device on silent mode. The information about such responses can be stored in knowledge base at the design time. It is also possible that sometimes the change in context results in situation that has no response information in the knowledgebase. In such situation, the system shall analyse the change and follow the steps as:

1. Identify various context attributes whose change may affect the system operation.
2. In what ways the attributes may change.
3. What are the reasons for the change in attributes?
4. How shall the System respond with change in attributes?
5. Who will initiate the Change?
6. Response to change of one attribute may cause change in another attribute, therefore one important objective is to study the interdependencies between the attributes of the system.
7. Identify New Goals as per changing requirements.
8. Set New Objectives.
9. Analyse new context, goals, and objectives and adjust System Behaviour as per new Conditions to achieve the target objectives.
10. Cheque if objectives are achieved.
11. In case Objectives not achieved, go to step 9.
12. In case the required objectives/goals can't be satisfied, switch to some default mode of operation.

On execution of default operation, if a user is not satisfied with the system, then he responds himself by changing some settings/mode of operation, the device shall monitor the changes made by user explicitly and save the current change in context and response provided by user in the knowledgebase, so that in future such contextual change will provide the same response that user explicitly adjusts as per his requirements. In such manner the knowledge base will keep on updating and will provide response in a better possible manner to any changes in the context in future. So the system will follow self-learning and self-adjust mechanism to provide context aware responses. The figure shown above uses some labels for arrow. The meaning of the labels is given below:

1. **Label A**
   Store the already known responses to certain contextual changes in the knowledgebase at design time.
2. **Label B**
   In case knowledgebase doesn't provide suitable response for the contextual change at runtime, the user will himself make required changes/settings in the device. The device shall update the knowledgebase so that in future the same response will be provided for the same contextual change automatically by the knowledge base without intervention of user.
3. **Label C**
   When there is context change in the environment, the devise will analyse the change, search the knowledgebase for the required response and then make necessary adjustments in case knowledge base provides necessary information regarding the adjustment of device operation in response to given contextual change.
4. **Label D**
   The Runtime adjustment routine adjusts to new contextual conditions and returns control back to the runtime environment for normal operations.

Example

Consider Mr. x has a meeting scheduled on 28-06-2017. He saves the event in his smart device along with location, time and other venue details. On the said date when Mr. x is in meeting, suppose a call arrives. As there is already an event saved, the smart device will use GPS to detect the location and compare the stored location of event with the location returned by GPS. If the location matches, then the smart device puts on silent/descent mode, without disturbing participants.

Now consider the situation where call is from parents of Mr. x or from the office of Mr. x and the call is very urgent and shall not be ignored. The smart device will not be able to differentiate between types of calls. One solution is to classify the contacts in the device according to groups. For illustration High priority group, mid priority group and low priority groups. If call arrives during the meeting hours, the device will first search the incoming call contact details in contact book. If the number is from unknown source, the device will reject the call without doing any necessary action. In case the caller is in contact book and falls under low priority group, the device will put silent mode on without immediate notification to Mr. x, later after meeting he will be notified about the missed call. In case the call is from mid priority caller, the device will put vibration mode on and give immediate notification by vibration to Mr. x without noticing other participants in the meeting. Finally if the call is from high priority group the device will either produce a ring or the divert the call automatically to personal assistant of Mr. x who will act according to the current situation.

CONCLISION

A context-aware system is a system that changes its behaviour as per the changes in the environment. It is therefore the job of the requirements engineers to somehow hide the complexity while trying in the same time to capture the needs of the user in the most pragmatic and efficient way. I discuss the methods already proposed for the elicitation of requirements in context aware systems and also their merits and demerits. As far as requirements engineering is concerned, I have tried to show that there is more complexity involved but this challenge can be addressed in some cases just by re-thinking traditional requirement engineering methods.

At the moment there is no proved method on what is the right way to choose, as new methods have been suggested there are very few real world systems that make use of these methods and it is hard to understand their advantages or disadvantages. Finally I tried to devise a new model that extends the traditional model of requirement elicitation and tried to fit it for the elicitation of requirements for context aware systems. The model proposed used the concept of self-learning and self-adaptation approach for providing context aware responses to the users of system.

REFERENCES

L. Kolos-Mazuryk, G.-J. Poulisse and P. van Eck (2005). Requirements Engineering for Pervasive Services. OOPSLA’05 workshop on Creating Software for Pervasive Services. San Diego, USA, 2005

D. Bolchini and J. Mylopoulos (2003). From Task-Oriented to Goal-Oriented Web Requirements Analysis. In: Fourth International Conference on Web Information Systems Engineering. Roma, Italy. pp. 166-175. IEEE Computer Society. doi:10.1109/WISE.2003.1254480.

J. Zhang and B. H. C. Cheng, “Using Temporal Logic to Specify Adaptive Program Semantics,” Journal of Systems and Software, Vol. 79, No. 10, 2006, pp. 1361-1369. doi:10.1016/j.jss.2006.02.062

D. M. Berry, B. H. C. Cheng and J. Zhang, “The Four Levels of Requirements Engineering for and in Dynamic Adaptive Systems,” The 11th International Workshop on Requirements Engineering Foundation for Software Quality, Porto, 13-14 June 2005.

N. Robinson, “A Requirements Monitoring Framework for Enterprise Systems,” Requirements Engineering, Vol. 11, No. 1, 2005.

Adaptive Systems: A Research Roadmap,” Software Engineering for Self-Adaptive Systems, Springer-Verlag, Berlin, 2009.