Journal of Theoretical and Applied Information Technology
31st May 2014. Vol. 63 No.3
© 2005 - 2014 JATIT & LLS. All rights reserved·

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

TROPOS BASED ADAPTATION FRAMEWORK FOR SELF ADAPTIVE SYSTEM

[1]GOWRI. R, [2]KANMANI.S, [3]PUNITHA. D

[1]Associate Professor, Department of IT, Sri ManakulaVinayagar Engineering College, Puducherry
[2]Professor, Department of IT, Pondicherry Engineering College, Puducherry
[3]Assistant Professor, Dept of CSE, Achariya College of Engineering Technology, Puducherry
E mail - [1]gowrithirugnanam@gmail.com, [2] kanmani@pec.edu, dpunitha2004@yahoo.com

ABSTRACT

Today's software complexity is increasing which in turn increase the need for software to work autonomously with least human intervention. Self-adaptive software work autonomously in unpredictable environment overcoming the failures and increasing the performance. Engineering self- adaptive software is very complex. In this paper generic adaptation framework using AOSE methodologyTropos has been proposed to develop Self-Adaptive System. In this Framework adaptation process is fully automated by making feedback loop as first class citizen. Two classes of requirements namely adaptive requirements to be monitored and evolution requirements to make changes on target system as a result of reconfiguration is used. Proposed framework is illustrated with case study Smart Travel Recommender System as it involves dynamic changes and is evaluated using existing system CARE framework and Tropos4As framework.

Keywords: *Tropos, Feedback loop, Adaptive requirements, evolution requirements, Reconfiguration*

1. INTRODUCTION

As technology advances, the requirements for software systems become ever more ambitious. We have reached a point where complexity is now one of the major challenges for the Information Technology (IT) industry. A solution that has been proposed by researchers in the past years is to design systems that adapt themselves to undesirable situations, such as new contexts, failures, suboptimal performance, etc. These solutions invariably include, even if hidden or implicit, some form of feedback loop, as in control systems[4]. Only few of them, however, consider the issue of adaptation during the whole software development process, starting from requirements engineering. A promising solution for software adaptation is to develop self-adaptive systems[SAS] that can manage changes dynamically at runtime in a rapid and reliable way.

The explosion in number and complexity of software solutions over the last several decades have pushed the limits of engineering capabilities to deal with the challenges in building, running and managing these systems. To cope with these problems, researchers have looked to a diverse set of fields (biology, control theory and artificial intelligence) to find techniques and unique solutions for these problems[3]. Self-adaptive software provides a solution to unbounded complexity by endowing software systems the ability to cope with unforeseen circumstances at run-time that were not envisioned during design time of the system.

Along this vision many existing approaches extends Tropos to engineer SAS by advocating requirements in all stages of development phases of SAS from early requirement to implementation and automatic mapping to BDI agents[1] which holds requirements at run-time. But the issue in this approach is evaluation is based on feedback from user.Unfortunately these approaches fail to accommodate dynamic changes or new requirement. The solution to this problem is evaluation should be automated to evaluate their own behavior and performance in order to re-plan and reconfigure when needed. To achieve this feedback loop should be made explicit as first class citizen.

This paper provides adaptation framework based on Tropos to increase adaptation process by making the feedback loop automated and first class citizen using Policy based reconfiguration which is based on Event-Condition-Actionat runtime.The remainder of the paper is structured as follows,In section 2 Literature Survey discussed and Section 3 describes existing Requirement based adaptation

Framework and Proposed Adaption framework based on Tropos is discussed in section 4 and Section 5 discuss the evaluation of our proposed framework.

2. LITEARATURE SURVEY

i * and Tropos, have been extended to represent requirements for system adaptation. Tropos4AS[1]extends Tropos in order to allow designers to model non-intentional elements using UML class diagrams, specifying resources that belong to an agent and the ones that belong to the environment. This framework allows the modeling of undesirable (faulty) states, which are known to be possible at runtime and should trigger system adaptation. Finally, Morandini [1] maps the goal models to the Jadex platform for run-time implementation.

In Continuous Adaptive Requirements Engineering (CARE)[2] method [Qureshi and Perini, Qureshi et al], Domain ontologies represent the knowledge about the domain and are linked to the goal model to help analysts detail the expected behavior of the system. Based on these models, the system monitors for environmental changes (that violate goals) or user requests (queries), representing them as Run-time Requirement Artifacts (RRAs).

Goals are assigned weights/priorities[5] and this framework also keeps track of each goal's activation level. At runtime, polling monitors attributes and goals, comparing them to the aspiration levels, and if it identify that attribute does not reach its aspired level then it will reconfigure the system.

DeLoach and Miller propose GMoD[6]to specify goals during requirements and then use those same goals throughout system development and at runtime. The GMoDinstance model captures the dynamics of the system state while maintaining the structure of the specification model. Capture any sequential constraints among goals and Determine which goals should be created in response to events that occur at runtime.. GMoD is based on attribute-precede-triggeranalysis. If a goal B has to be achieved preceding goal A has to be satisfied, then A is precedence of B. Trigger with event name and set of parameters trigger another event

Oyenan and DeLoach proposeOMaSE[7] for a customizable agent-oriented methodology framework allowing designers to build custom agent-oriented methods using a set of method fragments, all of which are based on a common meta-model. . In OMaSE domain model is used to capture the environment as a set of object types and agents that are situated in the environment.

From the literature survey the issues we identified are:

- Lack of automated feedback loop
- Refinement of requirements at runtime.

3. EXISTING SYSTEM REQUIREMENT BASED ADAPTATION FRAMEWORK

Existing Requirement based adaptation framework[2] is based on core ontology for requirement engineering with new revised elements based on which requirement engineering for SAS is acquired. In existing framework adaptation is based on adaptive requirements including functional, non-functional requirements but also specify properties for control loop functionalities such as monitoring specification, decision criteria and adaptation actions. Run-time analysis and refinement of requirement is achieved by involving end-user at run-time thus bridging the gap between design- time and run-time. ARML adaptive requirement modeling language is used for adaptive requirement specification in system-to-be. Monitor agent monitors the service to fulfill the user request. It monitors the ontology and goal model to better match the user request. The events are logged by monitor agent. Events are informed to the evaluator agent which is responsible for finding possible action. It refers to the adaptive requirements which provide rules to select action.

The issues identified in the existing system are:

- Adaptation is based on the log maintained in monitor component.
- Reconfiguration for dynamic changes is not carried out.
- No specific methodology is used.

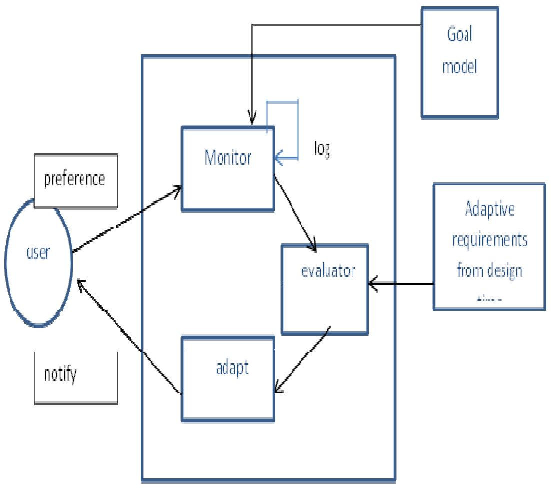

Fig 1: Requirement Based Adaptation Framework

4. PROPOSED TROPOS BASED ADAPTATION FRAMEWORK

4.1 Application

The case study Travel Recommender system has been chosen to illustrate the proposed Adaptation framework.The advantage of choosing this Case study is that it involves more dynamic changes which are needed.Travel Recommender System handles request from different customers like Business, vacation, and student and give full travel recommendation according to user preferences. Customers on confirming the ticket the recommender system should keep track of the customer context and status of the ticket booked. If any dynamic changes like cancellation of flight or delay of flight due to bad weather condition, then the system should keep on recommending the user about the travel based on preferences.

4.2 Related Work

Tropos4As[1] extendTropos by capturing environmental conditions and failure in architectural phase. Tropos4As preserves concepts of agent and goal model through all development phases until run-time through automated mapping to BDI agent. The advantage of Tropos4As is it preserves the requirement at runtime. The feedback loop in Tropos4As helps in tracing the decisions and changes from requirement to code. In Tropos4As evaluation is not automated because in adaptation process the system gets the user feedback for evaluation[1].

4.3 Proposed Work

The proposed adaptation framework is extension of Tropos. The formal Tropos provides the semantic for goal model. The extension Tropos4As extends environmental model, goal model and failure model. The Figure 2 shows the metamodel of proposed extension of Tropos. The metamodel is extended by providing adaptation manager which controls the agent. BDI agent which holds the requirement is made more adaptive by making feedback loop explicit which captures adaptive requirement and result in evolution requirement providing the action to be carried out. The dotted box indicates the proposed extension of Tropos metamodel. The components of adaptation manager are described in section 4.4.

4.4 Improved adaptation

In our proposed framework this issue is overcome by extending Tropos based Framework more adaptive by making the adaptation loop as closed loop which monitors self changes and changes in the context. So once change has been monitored the analysis is performed based on Event–Condition-Action and Reconfiguration is performed when any failure occur switching from one task to another task. The requirements which occur dynamically are given as adaptive requirements to be monitored as input to feedback loop. Once reconfiguration carried out the resulting requirements are given as evolution requirement. This process makes feedback loop as first class citizen.

4.5 Feedback loop process

The feedback loop monitors if the system responses to the request are satisfied and also to check whether the system takes correct action if the request is not satisfied.

- *Adaptive Requirements*

Adaptive requirements are requirements to be monitored to observe variability in operational context.

- *Monitoring*

Monitoring is responsible for monitoring the self-changes and adaptive requirements

- *ECA based adaptation*

On monitoring failures Event-condition-Action based adaptation process is carried out. This process goes through the list of actions associate and selecting and executing the appropriate action based on the condition. Here in ECA rule, event is requirement failure, condition is elicited from stakeholders in terms of when to use particular adaptation action such as "this is applicable only once". Finally action is sequence of operations.

- *Policy based Re-configuration*

On any situation that none of the requirements is satisfied the adaptation process has to reconfigure to new task based on the policy through communicating with other agents through message passing. Adaptation based on re-configuration involves (i) one or more parameters modeled during system identification are chosen (ii) based on then, based on the relation of these parameters with the failed indicator, reconfiguration agent decides by how much they should be changed. (iii) The parameters are then incremented by this value (iv) the framework waits for the change to produce any effect on the indicator (v) evaluating it again after the waiting time (vi) In each cycle, proposed framework can learn from the outcome of this change, possibly evolving the adaptation mechanism and updating the model. (vii) Finally, it decides whether the current indicator evaluation is satisfactory (viii) either concludes the process or reassesses the way it was conducted in the previous Cycles and starts over.

- *Evolution Requirement*

Evolution requirement specify changes to other requirements by applying the conditions. Evolution requirements are sequence of primitive operations which have an effect on target system, telling how to change the requirement model in order to adapt.

The advantages of our proposed framework are

- Avoids problematic repeated queries by user having partial knowledge about the domain. Feedback loop is made automated which monitors the variability in requirements and helps in deciding suitable adaptation strategy.
- Reduces the workload of design engineer having partial knowledge of domain.
- Repeated testing is avoided and localizing the errors is made easy.

4.6 Proposed Framework: Illustration of Smart Travel Recommender System

4.6.1 Perform early Requirement Analysis: Actor Diagram

In smart travel recommender system the actors are user interface agent, travel agent and flight agent are represented along with the dependency between the actors. The hard goal dependency between the actors UI agent and user is provide preference and notify user. Hard goal dependency between UI agent and travel agent is recommendation, and between flight agent and travel agent flight schedule. The resource availability is flight status.

4.6.2 Early Requirement Analysis: Goal Diagram

In goal diagram the goal provide full package of actor travel agent is identified. It is decomposed in to possibleleaf goals like monitoring itinerary changes, booking of flight and notification to end-user. For each of the leaf node tasks like get flight options, get service details, invoke service and notify user are identified. The soft goal associatedreasonable cost, reliability and convenience. The goal provide full package of Smart travel agent is further decomposed using AND/OR decomposition. Its sub-goals include monitoring dynamic changes, confirmation of ticket booking and notifying confirmation to user. Adaptive requirements are captured by monitoring.

4.6.3 Late requirement Analysis:

In this phase a new actor representing the system-to-be isintroduced. The intention is to re-arrange the goaldependencies based on evolution requirement, thus delegating the goals that the system has torealize. Smart recommender system is added as new actor with its functionalities monitoring the flight status and checking the availability of service and its coordination with other actors are operationalized.

4.6.4 Architectural Design Phase:

In this phase the overall system with sub systems of Smart Travel Agent like intelligent travel agent, flight schedule agent are specified with their dependencies like flight status goal dependency between intelligent travel agent and flight schedule agent. Smart travel agent includes the sub-actors smart travel recommender system, flight schedule agent. Based on the late requirement analysis goal delegation, the goal manages recommendation and flight status is delegated to the sub-actors.

4.6.5 ECA based Adaptation:

According to our case study Smart travel Recommender system dynamic changes occur often like the internet availability to access the service may or may not available. This adaptive requirement can be specified in ECA as follows:

```
<AR1>
        <Event>
                Mon(<internet available>)
        </Event>
        <Condition>
                Event(internet
                available)==true||Return False;
        </condition>
        <Action>
                Trigger(<notify    User>    ||
<Retry>)
        </Action>
<AR1>
```

On checking the availability of internet access, another adaptive requirement is availability of service.

```
<AR2>
        <Event>
                Mon(<GetServiceDetails>)
        </Event>
        <Condition>
                Event(Service
                isavailable)==true||Return
                False;
        </condition>
        <Action>
                Trigger(<notify    User>    ||
<Relax>)
        </Action>
<AR2>
```

In the above AR2 monitors the event about the availability of service based on the new requirement provided by user at runtime. If service is available it is evaluated based on the preference and action to notify user is triggered or if service is not available action to relax the requirement is triggered.

4.6.6 Policy based Reconfiguration:

Based on the case study if Smart travel recommender system fails to satisfy the adaptive requirements, then the system should reconfigure based on the policy that is the condition provided by the user .like “notification should be provided in less than 5 min”. to satisfy this condition system reconfigure by communicating with other agents through message passing.

5. EVALUATION

Proposed adaptation Framework is evaluated on comparing with existing system requirement based adaptation framework[2] and also with related work Tropos4As[1]

5.1 Evaluation of proposed work with existing work

The proposed adaptation framework is evaluated based on the factors like human involvement in adaption, time taken to generate monitoring specification, maintenance cost, flexibility of enriching the requirements and robustness.

- *Generic*:

The proposed adaptation framework is generic and can be applied to any kind of system.

- *Automated Feedback loop*:

The proposed framework use automated feedback loop through reconfiguration on any requirement failure which increases the adaptiveness .

- *Time to generate monitoring Specification on reconfiguration*:

In existing system the monitoring specification is carried out by referring to ontology. In our proposed framework since we use BDI agent which hold requirement at run time the time it takes to create monitoring specification is comparatively less than in existing system.

- *Maintenance Cost*:

Since proposed adaptation framework is fully automated it reduces the work of designers thus reducing the maintenance cost.

- *Flexibility:*

Proposed Framework increases the flexibility and span of usage by relaxing the uncertain factors during early requirement gathering thereby increasing the adaptation process.

- Robust:

Requirements set are made Robust by making it available in any uncertain conditions.

5.2 Evaluation of proposed work with Tropos4As

Development Phases	Tropos4As Framework	Proposed Framework
Early Requirement analysis	Capture stakeholders desire as	Capture Adaptive requirement

	formal Tropos	
Late Requirement analysis	Delegation of goals	Delegation of goals based on evolution requirement
Architectural design	Extension of goal model with environmental and failure model	Extension of goal model with environmental and failure model
Detailed design	Model transformation using UML class and sequence diagrams	Model transformation using UML class and sequence diagrams
Implementation	Code generation using JADEX and Automated mapping	Code generation using JADEX and Automated mapping
Run-Time	User communicate Preferences to BDI Agent	Decision making and refinement of requirements

Table 1: Development phases of proposed framework evaluated with Tropos$4As

5.3 Inference

On evaluating proposed adaptation framework with existing works Requirement based adaptation framework and Tropos4As, its been inferred that proposed work is more efficient by reducing the time to generate specification and fully automated by making decision making at runtime.

6. CONCLUSION

The work proposes an adaptation framework using AOSE methodology Tropos which provides systematic development including early requirement analysis to implantation. Here feedback loop is made explicit and it is made as first class citizen using adaptive requirement and evolution requirement. Reconfiguration prescribe changes in requirements at runtime. Our approach is generic and can be applied to any kind of system. The downside of our approach is since it takes partial fulfillment of requirements in feedback loop during reconfiguration the response time is less and this can be enhanced in the future.

REFERENCES:

[1] MirkoMorandini, "Goal-Oriented Development of Self-Adaptive Systems.", PhD Dissertation, March 31, 2011.

[2] Nauman Ahmed Qureshi, "Requirements Engineering For Self-Adaptive Software: Bridging the Gap Between Design-Time And Run-Time" PhD dissertation, 2011.

[3] MazeiarSalehie and LadanTahvildari.,"Self Adaptive Software: Landscape and Research Challenges", *ACMTransactions onAutonomous and Adaptive Systems*, March 2009.

[4] Betty H.C. Cheng, Rog´erio de Lemos, Holger Giese, Paola Inverardi,and Jeff Magee., "Software Engineering for Self-Adaptive Systems:A Research Roadmap".2009.

[5] "Towards a goal-driven approach to action selection in SAS", Wiley OnlineLibrary,May 2012.

[6] DeLoach and Miller, " A Goal Model for Adaptive complex system(GMoDS)", *International Journal on Knowledge-Intensive Multi-Agent Systems* .St. Louis, MO, October 11-14, 2009.

[7] Oyenan and DeLoach, "OMaSEmethodology framework",.Int. J. *Agent Oriented Software Engineering,* Vol. 4, No. 3, 2010.

[8] Vitor EstevaoSilvaSouza,"Requirements-based Software System Adaptation", PhD Dissertation, 2012.

[9] Danny Weyns and Michael Georgeff. "Self-Adaptation Using Multiagent Systems", *IEEE Software,* January/February 2010.

[10] Thomas Vogel and Holger Giese., "A Language for Feedback Loops in Self-Adaptive Systems: Executable Runtime Megamodels", *SEAMS 2012*, June 4-5, 2012.

[11] Fabiano Dalpiaz, Amit K. Chopra, Paolo Giorgini, and John Mylopoulos. "Adaptation in Open Systems: Giving Interaction its Rightful Place", ER 2010, 2010.

[12] Dalpiaz, F.; Giorgini, P.; Mylopoulos, J., "An Architecture for Requirements-driven Self-Reconfiguration.", *21st International Conference on Advanced Information Systems Engineering* (CAiSE '09), 2009.

[13] Mehdi Amoui Kalareh, “Evolving Software Systems for Self-Adaptation”, Phd dissertation, 2012.

[14] Alexei Lapouchnian1 Yves Lespérance2., “From Adaptive Systems Design to Autonomous Agent Design”, *Proceedings of the 4th International i* Workshop* - iStar10, 2010.

[15] Jon Whittle, Pete Sawyer, Nelly Bencomo, “RELAX: Incorporating Uncertainty into the Specification of Self-Adaptive Systems”, *17th IEEE International Requirements Engineering Conference*, 2009.

[16] Danny Weyns, “On Decentralized Self-Adaptation: Lessons from the Trenches and Challenges for the Future”, SEAMS ’10, 2010.

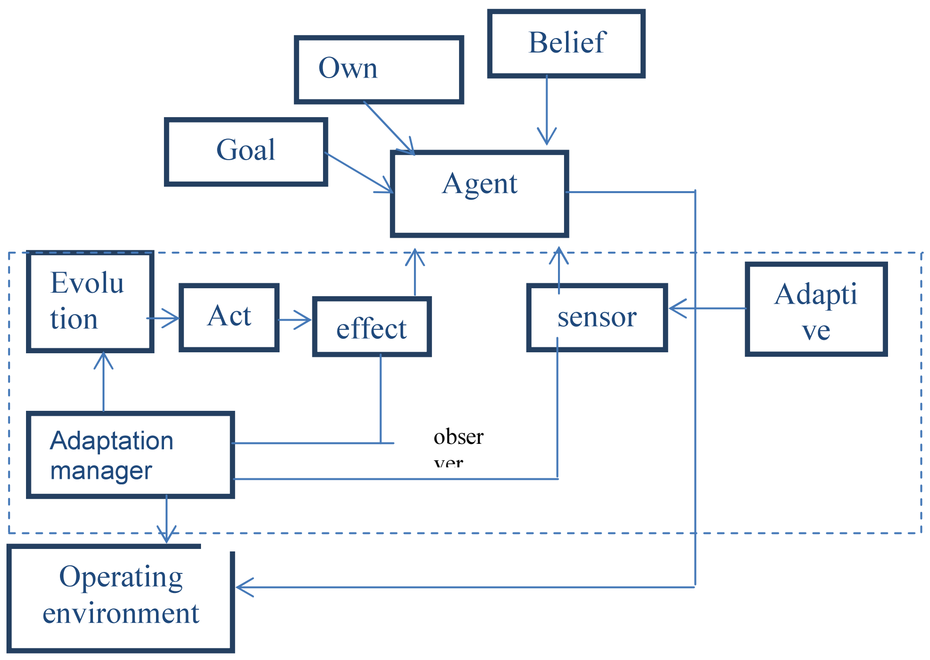

Fig 2: Meta-model of proposed extension of Tropos

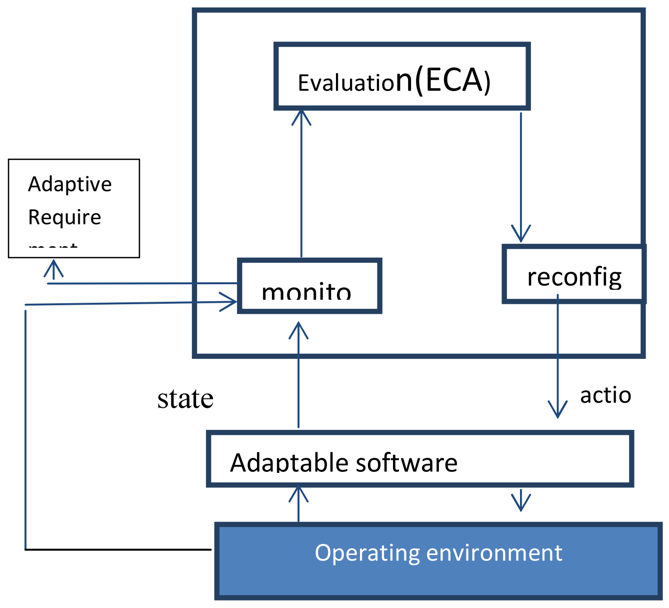

Fig 3: Proposed Tropos Based Adaptation Framework

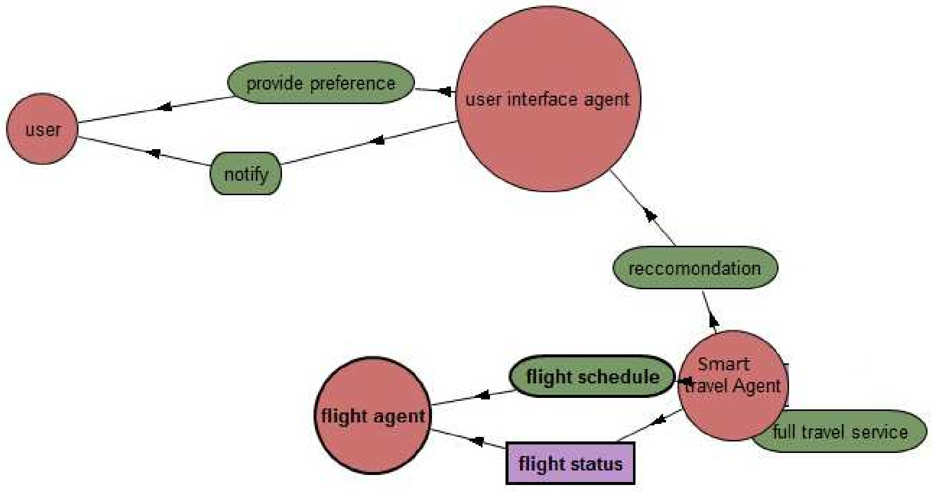

Fig 4: Early Requirement analysis: Actor Diagram

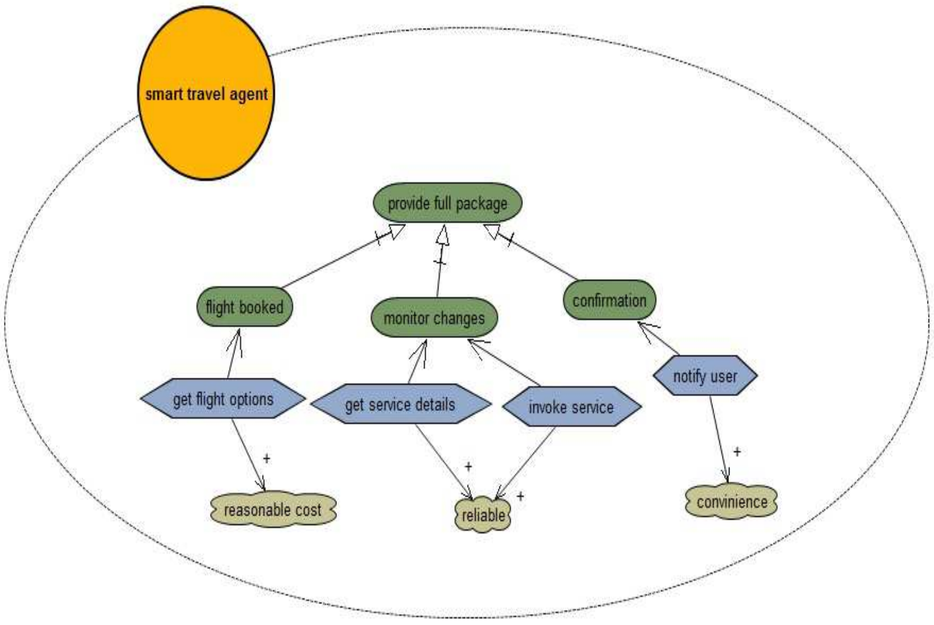

Fig 5: Early Requirement analysis: Goal Diagram

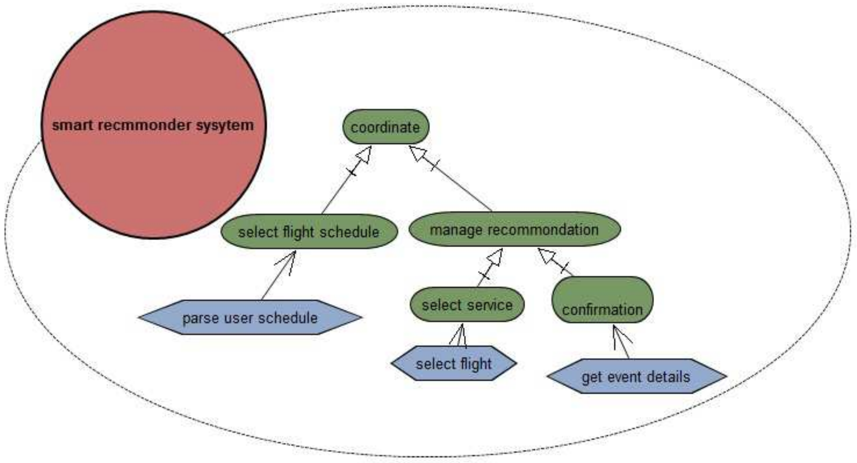

Fig 6: Late Requirement analysis

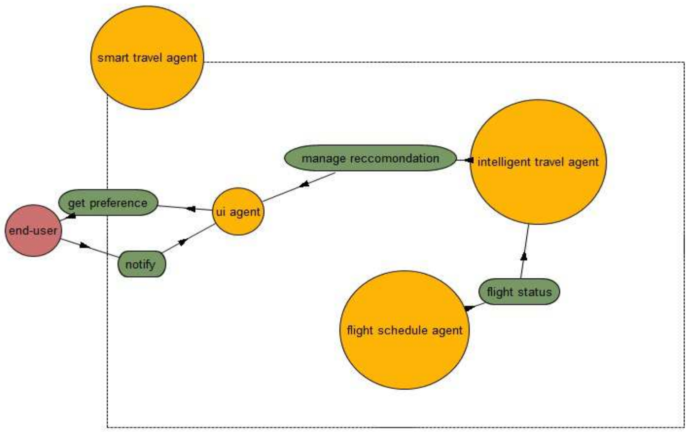

Fig 7: Architectural design