Scenario-Based Requirement Analysis

Belfrit Victor Bataljery, 3423891- b.v.bataljery@students.uu.nl
Group 1
Universiteit Utrecht

1. Introduction
When developing software engineering, many steps are taken. One of the most critical phases is, to elicit
and analyze requirements (Saiedian, Kumarakulasingam & Anan, 2005). In term of system engineering,
requirements can be seen as what the system must do to achieve the goals. It is one of many important
factors that can determines a success of build or improve the system. Thus, capturing, analyzing,
validating, documenting requirement are a vital activities. The purpose of analyze requirement is to see
whether all requirements are clear, complete and really resolve the conflict.

To capture, analyze and validate requirements, many methods have been developed to do that kind of
activities. One of them which attract many practicers is the scenario-based analysis. According to
Sutcliffe (1998, p. 49), scenario is a “facts describing an existing system and its environment including
the behavior of agents and sufficient context information to allow discovery and validation of system
requirements”

Letier, Kramer, Magee and Uchitel (2005) see scenarios, at the level of requirement, express sequences of
interaction between intended systems and agent in the environment. Having said that, scenarios should be
positioned at very early stage of developing system. Thus, they also suggest that it is worthwhile to
perform analysis before implying the models.

One method that can be used to analyze requirement is scenario-based requirement analysis or validation.
It’s a method created by Sutcliffe in his paper in 1998. At the time he was a professor of systems
engineering and director of the Centre of Human-Computer Interface Design (HCI) at the City University,
London, UK. The early version of this method is called SCRAM (Scenario-based Requirement Analysis
Method). SCRAM combines three things, namely, concept of demonstrator, scenario and design rationale
(Sutcliffe, 1997). However, SCRAM only provided the outline guidance for Scenario-based Analysis. Thus,
this method can be seen as the second-generation of SCRAM.

In this method he used, two scenarios which are scenario structure model of system context and scenario
script of system usage. Therefore, this method also adopts scenario script derived from Inquiry Cycle
(Potts, Takahashi & Anton, 1994). When scripting the scenario, the context of activity can be investigated
by including system environment. This investigation is necessary in order to check the connection and
dependency between the system and its environment. The system environment itself can be described by
modeling the system based on scenario.

The result of this method is requirement specification. This can be in many kinds of format such as,
formal languages, list of requirements, artifact (prototype or storyboard), etc.

1.1. Diagram and step-by-step
Before entering the first step in scenario-based requirement analysis, there is prerequisite that needs to be
defined. That is to define knowledge representation schema for scenarios. It’s necessary since this method
is used to investigate the dependency between the system components and their environment.

The activities in the preliminary steps, including gathering facts, scenario initialization, and creates a
model. After that, the model is divided into an intended system and its environment. Later, the system’s
environment becomes a usage scenario and will be cross-checked against the system. This dependency will be described in two separate sections (inbound and outbound events).

The first step in this method is to analyze user’s goals (1) and check if they are supported by system function. For each user’s goals in scenario will be cross-checked with functional goals and yield a first cut requirements specification.

After finish with the first step, the next step is to analyze inbound event (2) that can derived from agent or objects in the environment. The aim is not only to establish requirement for normal event using scenario script, but also to deal with abnormal event patterns by analyze an exception and elaborate that with system requirement.

Input event (described in scenario script) and requirement function are cross-checked to see whether there are dependencies between them or not. This is followed by elaborating with requirement specification. Inbound events are also described in scenario of interaction between users and proposed system.

This analysis leads to first-cut decisions about extent of the automation and user-system boundary. It makes requirements specification has to be updated by adding functions that can deal with different inbound events.

The next two steps (step 3rd and 4th) are dealing with system output. System output is also motivated by users’ goals (in step 1), and will be described in terms of process and outline content in requirements specification. So first, we specify the output (3) to see the side effect at the stage of requirement, and continue to analyze its requirement (4) to see if that support users’ task or not.

To help analyze output requirement, Sutcliffe offered five types that can be used to classify requirement specification. Those five types are Direct Commands, Indirect commands, Input requests, Information displays, and Metaphors and interactive worlds. After deal with the system output, the process is continuing to trace the destination and use of output by different stakeholders.

In step 5, the investigation whether the system can really support users’ needs is taken. Besides that, it should not make any unacceptable demands. Moreover, the commands from system output are counted in order to recognize the dependency between social and technical system. The more commands, the closer coupling is between them. Thus, increase the dependency and lead to failure of the system.

The final step is stakeholder cost-benefit analysis (6). The goal is to see how people react to the required system. In addition, the scenario structure model is used to assess the impact of system output and coupling between technical and social systems.

Figure 1: Outline of method
2. Example

To give a deeply insight about this method, the real case study is given. It was in 1991 when the London Ambulance Service (LAS) want to change how they dispatch an ambulance to the incident scenes. The goal was to replace the manual system with the computer-based one. They called it CAD (Computer Aided Dispatch). Unfortunately, the system was ceased on October 1992 due to series of failures and the system was reverted back to the manual one.

In preliminary step gathering facts, scenario initialization, and create model was taken. For the case, there are three agents in CAD system (Public people, Dispatcher, Crew), while in manual system there were more two agents (Telephone Operator and Control Assistant).

It seems like CAD tried to eliminate two agents (telephone operator and control assistant) and replace that with the new computer system. Furthermore, the CAD scenario was initialized based on the scenario in manual way.

The first step in this method is to analyze the goals. When initialized the new system (CAD), there are four stakeholders (Public, Ambulance crews, Dispatchers, and Managers) and from each of them, their goals should be met with functional goals in LAS.

After goal analysis, the next step is to trace the source of all potential input events. There is one inbound event, which is the emergency call from public. Because in one accident, there can be more than one call, so it is important to indentify caller, location and nature.

Besides that, the system also has to deal with side effect from unexpected event at the level of requirement. By its nature, it is complex and unpredictable, so the system needs to aware about it and able to give right output. The forms of system output in CAD system were a message displays, dialog boxes, synthesized speech, etc. In case of LAS-CAD, there are two output systems. One is a direct command from dispatcher to crew and the other is mandatory request to crews to report status and location.

Finish with inbound and outbound analysis, the further step is to investigate between social and technical systems. The commands that identified in system output are counted in order to recognize the dependency between social and technical system. The more commands, the closer coupling is between them. Thus increase the dependency and lead to failure.

Finally, the stakeholders need to do cost-benefit analysis to summarize previous investigation and to see the personal reaction to the new system.

2.1. Case Analysis

After doing analysis, apparently there were several reasons why the systems failed. Firstly, when analyzed the goals, it seems like many activities could only be done by human activities. Moreover, the system was likely support only the goals for dispatchers.

Secondly, it was important to identify the call from public since in one incident it was possible to got more than one call. Although it was important to get rid of duplicates call, it was missed when built CAD.

Another thing was the discrepancy between stakeholder information needs and system output. For example when the crew wants to know an updates of traffic and road situation, it was not available in the system. The crew only had an access to the dispatcher. Moreover, in manual system, the process of crew identification existed, but it was not exist in the new system. So, dispatcher can gets wrong information from crew. As a result, incorrect output from dispatcher is the main reason why the system failure.
Finally, in case of CAD, the system was closely coupled as a result from event messages between crews and dispatchers.

### 3. Related literatures

So, since getting the requirements is important factor in system engineering, method selection should be considered carefully. In the area of software development, Yang and Tang (2003) defined requirement engineering as discipline addressing the customer need and new product development. One activity in requirement engineering is requirement analysis, which is purpose and functionalities of system are elicited and modeled (Leite & Freeman, 1991). So, it can be said that in terms of system engineering, requirement analysis is the early stage of requirement engineering.

Besides scenario-based analysis, there are also others method that can be used to decompose requirements, e.g., goal-based analysis, use case-driven, and even the combination between scenario and goal analysis.

Kim, Kim, Park and Sugumaran (2004), suggested that combine several methods is the best way to analyze requirement. The reason is because each method has its own strengths and weaknesses. For example in use case-driven analysis, it cannot effectively support the requirements elicitation. Meanwhile, goal-based and combination goal-scenario analysis are good to see high-level system requirements, but it makes difficult to indentify an initial goal. (Anton, 1996)

Despite it advantage, that it ground specific detail of argument and reasoning, it also has a weaknesses. Weidenhaupt, Pohl, Jarke, & Haumer (1998) claimed that, this method can’t identify overall requirement because the relationship between scenarios is not captured. In addition, this method doesn’t really support all real-time attributes.

When looked on the case of LAS-CAD, this software system deals with real-time situation. This type of software system is unique because it is driven by asynchronous event and also it has to deal with timeline. Saiedian et al. (2005) suggested one approach that constructs a formalized view of scenario that generates timed specifications which is needed for real-time systems. This approach is called timed automata. In addition his research indicated that combine the message sequence chart by Regnell, Andersson and Bergstrand (1996) with time automata can closely depicting real-time system for complete scenario-based analysis.

### References


