A FRAMEWORK FOR EARLY ROBUSTNESS ASSESSMENT

Jianyun Zhou, Tor Stålhane
Department of Computer and Information Science, NTNU
Sem Sælandsveien 7-9, 7491 Trondheim, Norway
{jianyun,stalhane}@idi.ntnu.no

ABSTRACT
Time pressure and quality issues bring new challenges for developing web-based systems. The ability to analyze quality early in the development lifecycle is crucial. Among the techniques suggested by the literature, few of them actually support early quality activities when little information about the system is available. We take robustness as a critically important quality attribute, and present a framework for performing robustness assessment during analysis and architecture design stages. Firstly we use Jacobson’s analysis method to identify a set of objects. Then for each object, we apply a simplified FMEA to find robustness-related failure modes, possible causes, their effects, and furthermore, we identify possible ways to prevent or reduce robustness failures. In the end, we illustrate the proposed method through an example from a simple web-based Internet Bookstore system.

KEY WORDS
Robustness, assessment, web-based systems, FMEA

1. Introduction
The World Wide Web is rapidly becoming a popular medium of software applications. Web-based systems have penetrated into every area of our life, such as business, education, entertainment, and manufacturing. It brings new challenge for today’s software professionals. On one side, they work under time pressure to complete system development ahead of the competitors. Delivery delays often lead to the loss of revenue and reputation for the organization and in turn result in the loss of market shares and endanger the future of the organization. On the other hand, the quality is also important. Troublesome or error-prone systems can result in unsatisfied users, loss of revenue, and loss of market shares. Web-based system users are always looking for systems that serve them in a reliable way, providing quick and useful services.

To achieve fast development of high-quality systems, good engineering methods and approaches are important. In the literature, much emphasis is put on early quality assurance activities. As errors and misconceptions found in later phases of the development cycle are expensive and time-consuming to fix, it is evident that a meticulous analysis of the system and its behaviour should be carried out as early as possible [1]. The quality attribute considered in this paper is robustness. Many may confuse the concept of reliability and robustness. While reliability concerns the internal faults of the system or component, robustness concerns interaction faults. For a system, interaction faults refer to the operational environment, such as unexpected user input. For each component in the system, interaction faults refer to the failure of interaction components, which may be caused by reliability problem. We consider robustness as one of the most important factors for a successful web system: for the first, web-based systems are accessed via HTTP protocol, which has made such systems available for almost everyone. It is difficult to control the input profile of end users. They must therefore be able to tolerate errors and abnormal interactions from the user environment. Secondly, web-based systems are often not separately developed. They must therefore be able to tolerate errors and abnormal interactions caused by internal failures.

As web-based system is robustness-critical, it is necessary to carry out a set of activities – robustness assessment activities – to assure robustness in the early phases of the development process. We need to look at situations when the system is used in unspecified way, or some components are not working as expected. The robustness assessment focuses on the prevention of robustness failures or the reduction of chances for such failures.

In this paper we intend to propose a general framework for early robustness assessment, based on Jacobson’s analysis method [2] and FMEA (failure mode and effect analysis) [3]. The rest of paper is organized as follows. The second section compares the concept of reliability and robustness, and then briefly reviews some existing reliability techniques, which results in the selection of FMEA as our method for robustness assessment. Section 3 provides a short introduction to Jacobson’s analysis method. By combining this method with FMEA, we obtain our robustness assessment methodology. Section 4 illustrates the method by applying it to a simple e-commerce example. In Section 5 we conclude the paper and discuss future research directions.
2. Robustness and FMEA: Basic Concepts

2.1 Robustness and Reliability

Before we conduct any robustness related activities, the concept needs a clear definition. According to [4] and [5], robustness guarantees the maintenance of certain desired system features despite fluctuations in the behavior of its component or its environment. Robustness, in [6], is the degree to which a system or component can function correctly in the presence of invalid input or stressful environmental conditions. In a more general sense, [7] states that a robust system operates correctly across a wide range of operational conditions. The concept of robustness, used in this paper, is similar to the one used in [7]. We consider that a system or component which is totally correct with a complete specification is robust, in that its behavior is predictable for all possible operational environments. Two necessary elements are a complete specification and correctness (100% reliable).

To formalize the difference between robustness and reliability, we introduce the operational environment partition model from [8]. The total operational environment of a system or component can be divided into four parts: SD, AED, FD, and UD.

<table>
<thead>
<tr>
<th>SD</th>
<th>AED</th>
</tr>
</thead>
<tbody>
<tr>
<td>FD</td>
<td>UD</td>
</tr>
</tbody>
</table>

Figure 1 A partition over all operational conditions

**SD**: standard domain refers to the set of all operational conditions for which a system satisfies its specification.

**AED**: anticipated exceptional domain denotes the set of all operational conditions for which correct exception results are produced.

**FD**: failure domain, refer to the set of all operational conditions for which the behavior of the system contradicts the specification or exceptional specification.

**UD**: unanticipated domain contains the set of all operational conditions which are not included in the specification.

Reliability is related to failure domain. The smaller the failure domain, the more reliable. The root cause is the design faults within the boundary of the systems. When FD = {}, the system is said to be correct regardless whether UD is empty or not. A robust system requires both FD = {} and UD = {} are satisfied. It is difficult, if not impossible, to achieve for any real systems. The concept of robustness considered in this paper, has a more narrow scope, in the sense that we excludes the internal faults of the system or component, and only deal with the operational, interaction-related faults.

2.2 Related Work and FMEA

We have discussed similarity and differences between reliability and robustness. Methods and techniques developed for reliability can be easily adapted to improve robustness. Such methods can be divided into three categories: reliability estimation, reliability prediction and reliability assessment.

Traditional techniques, such as reliability growth models [9], belong to the first category. They use data from testing performed at the end of the development lifecycle to predict operational reliability. It is not possible to use them earlier.

Reliability prediction is often done on the architecture level and component basis. Techniques can be classified as state-based model, path-based model, and additive model [10]. Though they are claimed to be architecture-based, most of them cannot be applied in early stages. For example, the state-based model [11] uses control graphs to depict application architecture, while control graphs can in most case only be extracted from the code.

Reliability assessment has a wide range of methods. Most of them can be used early in the development process, such as fault tree analysis and failure mode and effect analysis. At the analysis and architecture design stages, a quantitative reliability assessment is not feasible, but a study of the effect of a range of failures is invaluable for prevention or reduction of failures. As the development process progresses, more details will become available and it is then possible to study failure causes and their prevention in a more detailed manner.

To perform early robustness assessment for web-based systems, we use failure mode and effect analysis (FMEA) method. FMEA is a powerful tool used by reliability engineers for systems analysis. The method is a “bottom-up” approach as opposed to fault tree analyses which is “top-down”. FMEA breaks the system down into components or subsystems. Then, for each component failure modes, possible causes and effects on the rest of the system are analyzed. The results are documented in a specially designed worksheet. A complete described of the process can be found in [12]. FMEA technique has been mostly used for hardware reliability [13]. For software, there are few successful applications to date [14]. Some derivations of classical FMEA have been proposed for software, such as in [15]. They fail, however, to capture the essence of a software “component” in a well-defined way. It is infeasible to define such component in the beginning of the preliminary design phase. In this paper we describe an attempt to decompose the system in a different way by taking advantage of Jacobson’s analysis method [2].

3. Robustness Assessment Method

3.1 Jacobson’s Analysis Method
Ivar Jacobson introduced the concept of robustness analysis to the world of OO in 1991 [2]. It is an intermediate level of design, between Use Cases and the software design level. By analyzing each use case, robustness analysis identifies a set of objects that will participate in the use case, and classifies them into one of the following three stereotypes:

1. Boundary objects, which the actors use when communicating with the system
2. Entity objects, which are usually objects from the domain model
3. Control objects, which serve as the “glue” between boundary objects and entity objects

Figure 2 shows how to represent these three types of objects in a robustness diagram. Rules for interaction among these objects are illustrated in Figure 3.

Figure 2 Stereotype symbols

Figure 3 Interaction rules

1. Actors can only talk to boundary objects
2. Boundary objects can only talk to Control objects and Actors.
3. Entity objects can only talk to Control objects.
4. Control objects can talk to boundary objects, other Control and Entity objects, but not Actors.

As we discussed in section 2.1, two important elements of robustness are specification completeness and correctness. Robustness analysis plays an important role in the completeness check. It provides a practical way to help one address all the necessary courses in the use case. Moreover, the identified objects and the essential relationship between the three stereotypes offer us an opportunity to conduct robustness assessment.

Let’s take a closer look at the three types of objects when applied for web-based systems:

1. Boundary objects are the objects that the users will use to interact with the system. These are elements that compose a web page, such as hypertext, forms, menus, buttons, and so on.
2. Entity objects often map to the database tables and elements in legacy systems. They represent resources required by use case execution.
3. Control objects embody mostly application logic. They serve as mediator between the users and the stored data. This is where one captures the frequently changing business rules and policies.

The contribution of Jacobson’s analysis method to our robustness assessment is two-fold. Firstly, it provides a systematic method for decomposing the system into objects. Secondly, as Control objects capture application logic and manage all interactions between Boundary objects and Entity objects, they serve as natural placeholders for robustness assessment using the FMEA.

3.2 The Proposed Method

In Section 2.2 we have chosen FMEA to do robustness assessment at the analysis stage. While it is the easiest and most cost-effective time to improve robustness, the difficulty lies in little information that is available at that time. To do FMEA we need to decompose the system into well-defined components, and perform FMEA on them. The analysis artifacts, provided by Jacobson’s method, support such a decomposition and analysis in a practical way. A five-step method is developed as follows:

**Step 1:** Define the robustness requirements of the system.
**Step 2:** Divide the system into subsystems by important use cases. For each use case, use Jacobson’s method to identify Boundary, Control and Entity objects. Complex logic can be partitioned into several Control objects according to the layered model presented in [16].
**Step 3:** Prepare a list of Control objects for each use case.
**Step 4:** For each Control object, fill in the FMEA worksheet, which is showed in Figure 4. The entries in the worksheet are described later in this section.
**Step 5:** Review failure modes in the worksheet and prioritize those items pertaining to a particular goal, such as customer’s satisfactions, organization’s reputation. Prioritization is based on developers’ opinions and experience since other information about the system is not yet available.

**FMEA Worksheet**

<table>
<thead>
<tr>
<th>FMEA Worksheet</th>
<th>System:</th>
<th>Performed by:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use case:</td>
<td>Date:</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Control object</th>
<th>Robustness failure mode</th>
<th>Possible cause</th>
<th>Local effect</th>
<th>System effect</th>
<th>Preventive means</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1)</td>
<td>(2)</td>
<td>(3)</td>
<td>(4)</td>
<td>(5)</td>
<td>(6)</td>
</tr>
</tbody>
</table>

Figure 4 FMEA worksheet
The FMEA worksheet proposed here is a revised version of one described in [12]. Different from many other applications of FMEA method, we focus on identifying means to eliminate or reduce the chance of robustness-related failures, rather than ranking their seriousness. The entries in the FMEA worksheet are as follows:

**Control object:** The name of the Control object is given in column (1).

**Robustness failure mode:** All robustness-related failure modes of this control object are identified in column (2). A robustness failure is defined as non-fulfillment of robustness requirement identified in step 1.

**Possible cause:** Possible causes of the failure mode are written in column (3). Since we are concerned with robustness and not reliability, internal faults of Control objects are not considered - only those stemming from outside the boundary of the objects are rated as causes of robustness failure. They are classified as operational fault (or interaction fault) in [17], which may be attributed to reliability failures of interacting objects.

**Local effect:** The main effect of the identified failure mode on the subsystem (the correct function of the use case) is recorded in column (4).

**System effect:** The main effect of the identified failure mode on the primary function of the system is recorded in column (5).

**Preventive means:** Possible ways to prevent or reduce the effect of identified robustness failure are described in column (6).

### 4. Robustness Assessment: an Example

In this section we illustrate the proposed method by an example. The system considered is a simplified Web-based Internet Bookstore [18]. In order to keep it short, we present robustness assessment only for one use case.

Robustness requirements for a web-based system will vary. In a general case we define the minimum robustness level - that a system should always responds to the user’s action either by correct results or appropriate prompt. Figure 5 shows Jacobson’s analysis diagram for the selected use case “Search by author”. The User types the name of an author on the Search page and then presses the Search button. The system searches the Catalog and retrieves all the Books with which that author is associated. The system then retrieves the important details about each Book. The system displays the list of Books on the Search Results Page, with the Books listed in reverse chronological order by publication date.

The application logic of this use case is captured by “Search on Author”, “Retrieve Details” and “Display” Control objects. They are candidates for robustness assessment. Relating to the robustness requirements defined earlier, we can make the FMEA worksheet for these Control objects as shown in Table 1.

### 5. Conclusion and future work

In this paper we have presented an approach to early assessment of robustness for web-based software systems. The main goal of this assessment is to identify measures that will increase the system’s robustness. Based on a formal definition of robustness, we explored the similarity and difference between reliability and robustness. As they are distinguished only by different types of faults, reliability techniques can be applied to improve the robustness of the system. In particular, we consider the FMEA (failure mode and effect analysis) method. As stated in [19], this is a flexible approach that is adaptable to almost every aspect of a system.

The proposed assessment framework integrates Jacobson’s analysis method, a systematic way to decompose the system, and a simplified version of FMEA. A five-step methodology is developed. As a result, we identify preventive measures against robustness failures, which in turn will contribute to specification completeness. To demonstrate our approach, we have applied the method to an example from a Web-based Bookstore system.

This work opens several possible directions for future research. An immediate applicable task is to evaluate the presented framework. Some qualitative approaches, such as case studies, may be applied. Furthermore, although FMEA is highlighted in this paper as an early assessment method, it is useful in every stage of the software development process. Our main interest in the future is to explore the continuous application of FMEA over the lifecycle of a web project and look for methods and techniques to support and ensure the total effectiveness of the FMEA process.

### 6. Acknowledgement

This paper is a part of the WebSys (Web-based systems – reliability versus time-to-market) project. The WebSys project is sponsored by the Norwegian Research Council.
<table>
<thead>
<tr>
<th>Control object</th>
<th>Robustness failure mode</th>
<th>Possible causes</th>
<th>Local effect</th>
<th>System effect</th>
<th>Preventive means</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search on Author</td>
<td>No response is produced at all</td>
<td>Error user input</td>
<td>Fail to respond to user’s interaction</td>
<td>Prevent further use of the system</td>
<td>Control user input and prevent serious errors from entering the object; Search Page detects the failure of the object and interacts with Display to prompt the user appropriately</td>
</tr>
<tr>
<td></td>
<td>Information found is incorrect</td>
<td>Error user input</td>
<td>Incorrect data is presented to the user</td>
<td>Users move to other systems if they suspect the quality of the system</td>
<td>Control input from Search on Author and prevent serious errors from entering the object; Search on Author detects the failure of the object and interacts with Display to prompt the user appropriately</td>
</tr>
<tr>
<td>Retrieve Details</td>
<td>No response is produced at all</td>
<td>Error output of Search on Author</td>
<td>Fail to respond to user’s interaction</td>
<td>Prevent further use of the system</td>
<td>Control input from Search on Author and prevent serious errors from entering the object; Search on Author detects the failure of the object and interacts with Display to prompt the user appropriately</td>
</tr>
<tr>
<td></td>
<td>Information found is incorrect</td>
<td>Error output of Search on Author</td>
<td>Incorrect data is presented to the user</td>
<td>Users move to other systems if they suspect the quality of the system</td>
<td>Control input from Search on Author and prevent serious errors from entering the object; Search on Author detects the failure of the object and interacts with Display to prompt the user appropriately</td>
</tr>
<tr>
<td>Display</td>
<td>No response is produced at all</td>
<td>Error output of Retrieve Details</td>
<td>Fail to respond to user’s interaction</td>
<td>Prevent further use of the system</td>
<td>Control input from Retrieve Details and prevent serious errors from entering the object; Retrieve Details detects the failure of the object and interacts with Display to prompt the user appropriately</td>
</tr>
<tr>
<td></td>
<td>Incorrect information is displayed</td>
<td>Error output of Retrieve Details</td>
<td>Incorrect data is presented to the user</td>
<td>Users move to other systems if they suspect the quality of the system</td>
<td>Control input from Retrieve Details and prevent serious errors from entering the object; Retrieve Details detects the failure of the object and interacts with Display to prompt the user appropriately</td>
</tr>
</tbody>
</table>

Table 1 FMEA for Search by Author

References