International Journal of Engineering Research-Online
A Peer Reviewed International Journal
Email:editorijoer@gmail.com http://www.ijoer.in

Vol.4., Issue.4., 2016
(July-August)

RESEARCH ARTICLE

ISSN: 2321-7758

EXCEPTION HANDLING FROM REQUIREMENT SPECIFICATION TO IMPLEMENTATION: EXTENDING UML

RAVEENA YADAV
M.Tech., Research paper Scholar, SITM Rewari, Haryana, India

RAVEENA YADAV

ABSTRACT

In this research paper we study about Exception handling implementation. In our approach we have made an extension to UML2.0 which makes it possible to consider exception handling during applications modeling phase of the software lifecycle. Our approach makes also opportunity to integrate the exception handling system explicitly in Use case, sequence diagrams allowing developer generating the last one, the class diagram and the aimed programming language code automatically with some needed requirements. Those automatic generations adapt the exception handling system specifically to each phase making some kind of communication between all phases of software life cycle. We have proceed to build our approach with defining a new UML2.0 extension profile and we have used model driven engineering to automatize diagrams and code generation making possible to specify exception handling and automate its passage from one phase to another along the software life cycle.

KEYWORDS – Exception Handling , Types of Sterotypes , Handling Mode , Model Driven Engineering .

1. INTRODUCTION

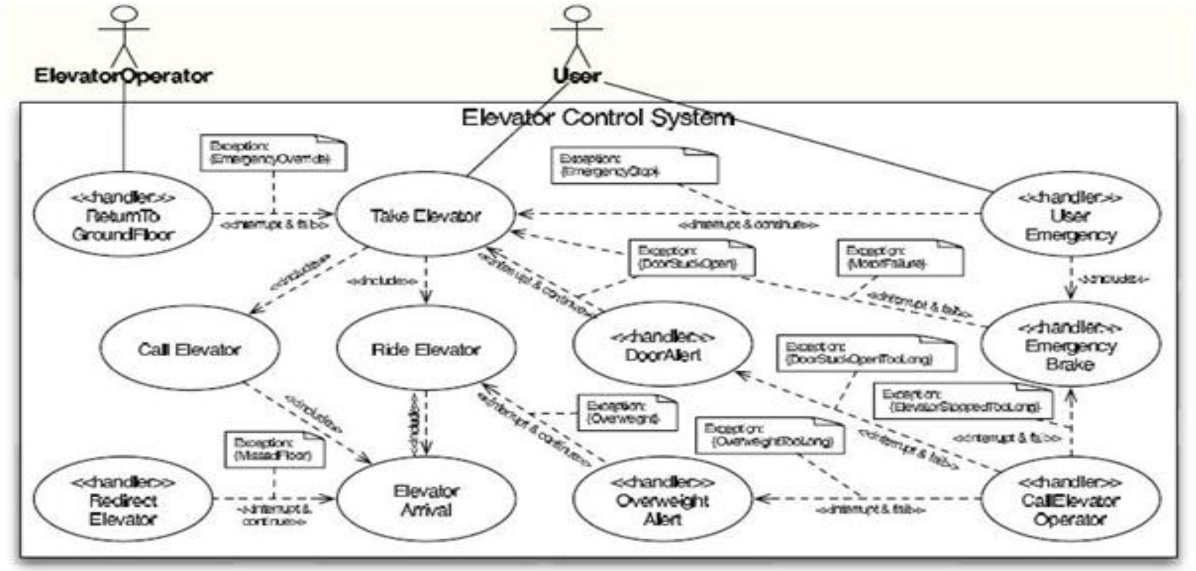
Our approach to Exception handling is frequently considered as the final task to achieve when developing programs. However dependable systems need from their stakeholder to predict all exceptional situations, because any non-considered exceptional behavior potentially makes great damages. . Exception handling is very important specially when we talk about dependent systems because many disasters were caused by non handled or non predicted exceptions. Designers, developers and software stakeholders should model exception handling system early when developing a software, because thinking about software exception on the last phase of software developing they will be found under pressure because they must focus on the normal and the exceptional behavior of the application in the same time, so any ignored exception may cause great disaster.

The objective of this master thesis is to study the exception handling system from the requirements specification phase to the final phase when developing software. This study is based on extending UML by a profile which allows the designer and even the developer modeling the aimed exception handling system on the earliest phases of developing and then receiving automatically the traces of this system on the application code according to the target programming language.

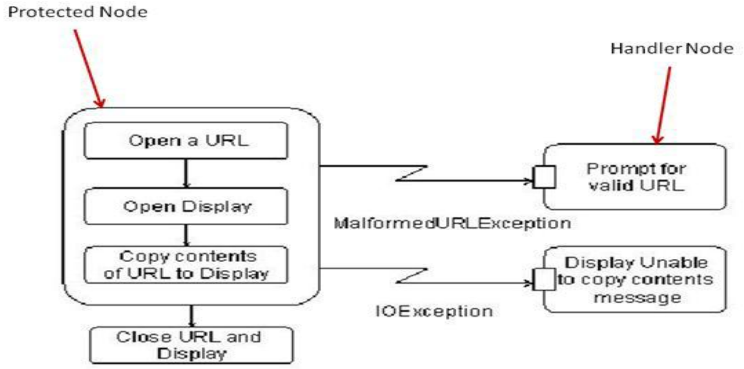
This research aims the following issues:

1) Defining a UML profile in order to support exception handling system along all the software life cycle,
2) Generating semi automatically the sequence diagrams from use case diagram,
3) Generating automatically class diagrams,
4) Generating application code,
5) Moving automatically the exception handling system when passing from a diagram to another until arriving to the generated code,
6) Controlling any changes made on the exception handling system at any software life cycle phase and applying them to other phases,
7) Specifying constraints that are based on matching the exception handling system to the programming language features in the phase of code generation.

The figure below represents a use case diagram made for an elevator system. We see in this diagram the use cases stereotyped as handlers in order to distinguish them from normal use cases, we see also comments stereotyped as Exception in order to show the name of the exception associated with a given use case, we can see also links stereotyped as interrupt and continue, others are stereotyped as interrupt and fail. These two last stereotypes are defined to express in the use case diagram the model of exception handling model such as resume and termination.



Exceptional Use Cases



The URL viewer example (extracted from [UML Superstructure])

In the specification of behavior features made by OMG, exceptions can be associated to behavior features such as operations which could raise these exceptions. Even UML offer a small supporting of exception handling, it does not permit to model it and to show it explicitly in the most important diagrams of software's life cycle such as use case diagrams, sequence diagrams and class diagrams.

The proposed UML Profile

According to a profile in the Unified Modeling Language (UML) provides a generic extension mechanism for customizing UML models for particular domains and platforms. Extension mechanisms allow refining standard semantics in strictly additive manner, so that they can not contradict standard semantics.

In order, to adapt UML to our approach we have defined a profile which contains many stereotypes that extend the UML meta classes to be able to show explicitly exception handling notations in the aimed locations. In the following we will present firstly the whole profile, then we will describe each stereotype separately.

Handling an exception may be done in different modes which can be:

- Termination,
- Resume,
- Retry,
- Propagation,
- Return Value,
- Abort All.

Implementation of the proposed approach

In order to implement our approach we were based on model driven engineering to manipulate all the UML models and diagrams which we have used and the new extensions which we have done to UML. To increase the benefits of our approach, we have proposed a tool that facilitates software development by automating its most important phases. This tool looks also after moving Exception Handling defined in the first phase of software life cycle to the next phases until arriving to implementation phase automatically

Model-driven engineering

According to Model-driven engineering (MDE) is a software development methodology which focuses on creating models, or abstractions, more close to some particular domain concepts rather than computing (or algorithmic) concepts. It is meant to increase productivity by maximizing compatibility between systems, simplifying the process of design, and promoting communication between individuals and teams working on the system.

A modeling paradigm for MDE is considered effective if its models make sense from the point of view of the user and can serve as a basis for implementing systems. The models are developed through extensive communication among product managers, designers, and members of the development team. As the models approach completion, they enable the development of software and systems.

As it pertains to software development, model-driven engineering refers to a range of development approaches that are based on the use of software modeling as a primary form of expression. Sometimes models are constructed to a certain level of detail, and then code is written by hand in a separate step.

Return Value stereotype As an exception could be handled by returning a value to the invoker of the method that raises the exception we have defined the stereotype Return value in order to allow to the designer expressing this handling mode and finding a graphic notation which could be manipulated easily.

Retry stereotype This stereotype extends the meta class Message (see Figure.9). It represents the retry mode of exception handling. It allows the designer to express graphically that a given handler obliges a software user to modify some parameters after the occurrence of an exception and then re-execute the software.

Conclusion and perspectives We have performed in this work some extensions to UML in order to give opportunity to designers to deal with exception handling in the early phases of software life cycle. We have done our proposed extensions according to a defined profile that extends UML in order to introduce exception handling concepts, notations and terminology in UML diagrams. This feature does not exist in standard UML. Extensions we have done have been applied to use case and sequence diagrams.

With the extensions we have done, use case and sequence diagram will be enriched by exception handling notations, a thing that provides explicitness of exception handling which was always considered as a strength piece of code. With our proposed UML extensions, designers and developers can manipulate and deal with exception handling from the early phases of software life cycle and with some level of explicitness and graphic notations.

References

[1]. Goodenough, J.B.: Exception handling: Issues and a proposed notation. Communications of the ACM 18(12), 683{696 (1975).

[2]. Shui, A., Musta_z, S., Kienzle, J.: Exceptional use cases.In: Briand, L.C., Williams, C. (eds.) MoDELS 2005. LNCS, vol. 3713, pp. 568{583. Springer, Heidelberg (2005).

[3]. Shui, A., Musta_z, S., Kienzle, J.: Exception-Aware Requirements Elicitation with Use Cases. In: Dony, C., Knudsen, J.L., Romanovsky, A., Tripathi, A.R. (eds.) Advanced Topics in Exception Handling Techniques. LNCS, vol. 4119, pp. 221{242. Springer, Heidelberg (2006).

[4]. Musta_z, S., Sun, X., Kienzle, J., Vangheluwe, H.: Model-Driven Assessment of Use Cases for Dependable Systems. In: Nierstrasz, O.Whittle, J., Harel, D., Reggio, G. (eds.) MoDELS 2006. LNCS, vol. 4199, pp. 558{573. Springer, Heidelberg (2006)

[5]. Zia, M., Musta_z, S., Vangheluwe, H., Kienzle, J.: A Modelling and Simulation Based Process for Dependable Systems Design. In: Software and Systems Modeling, pp. 437{451 (April 2007).

[6]. Romanovsky, A.: On Exceptions, Exception Handling, Requirements and Software Lifecycle. 2007 IEEE.

[7]. Si Alhir, S: Guide to applying the UML, page 350. Springer, 2002 / SiAlhir, Sinan (2002). Guide to applying the UML. Springer.

[8]. Oddleif, Halvorsen., Ragnhild Kobro, Runde., Oystein, Haugen.:Time Exceptions in Sequence Diagrams MoDELS 2006 Workshops, LNCS 4364, pp. 131{142, 2007. Springer.

[9]. Nelio, Cacho., Thomas, Cottenier., Alessandro, Garcia.: Improving Robustness of Evolving Exceptional Behaviour in Executable Models WEH '08, November 14, Atlanta, Georgia, USA, 2008. ACM.

AUTHOR BIOGRAPHY

Mrs. yadav pursuing her M.Tech in Computer science And Engineering From SITM Rewari, she has his bachelor of technology from R.P.S College Of Engineering M/Garh, Haryana, India in Computer Science And Engineering. Currently she is pursuing Research in Exception handling from requirement specification to implement :Extending UML