ANTWERPEN MANAGEMENT SCHOOL

Applying Normalized System Theorem on Restful APIs using the C#.NET programming language

Author:
Gerco Koks

Supervisors:
Prof. Dr. Ing. Hans Mulder
Frans Verstreken, Mcs

A thesis submitted in fulfillment of the requirements for the degree of Master of Enterprise IT Architecture

May 1, 2022

Declaration of Authorship

I, Gerco Koks, declare that this thesis titled, "Applying Normalized System Theorem on Restful APIs using the C#.NET programming language" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

"Thanks to my solid academic training, today I can write hundreds of words on virtually any topic without possessing a shred of information, which is how I got a good job in journalism."

Dave Barry

ANTWERPEN MANAGEMENT SCHOOL

Abstract

Master of Enterprise IT Architecture

Department or School Name

Master of Enterprise IT Architecture

Applying Normalized System Theorem on Restful APIs using the C#.NET programming language

by Gerco Koks

The Thesis Abstract is written here (and usually kept to just this page). The page is kept centered vertically so can expand into the blank space above the title too...

Acknowledgements

The acknowledgments and the people to thank go here, don't forget to include your project advisor...

Contents

List of Figures

List of Tables

For/Dedicated to/To my...

1 Introduction

"*Pantha Rhei*" is, according to *Plato*, one of the famous philosophical statements first described by the Greek philosopher *Heraclitus*.[1] Translated as "everything flows" this statement is an unambiguous commitment to ubiquitous dynamics of everything that exists. "Life is flux", one of the constants in life is change and its best we act accordingly.

In the realms of Software Engineering the "laws of software evolution" (Lehman, 1980) refers to a series of laws described by Lehman starting from 1974. With these Laws, he describes the balance between the forces driving new developments on the one hand (a change), and the forces that slow down progress on the other hand. Based on *Heraclitus* philosophical statement we assume a software engineering project frequently will be subjected to change, possibly due to changing functional requirements and technological progress. As these changes emerges, the complexity of these software projects will gradually increase over time. If the system is not adapted appropriately the combinatorial effects of these changes will result in ever-increasing complexity and render the software system eventually obsolete, according to Lehman (Lehman, 1980).

As the competitive environments of contemporary organizations are changing continuously, the speed at which changes follow each other is also increasing. IT organization are attempting to cope with this trend by adopting agility and maturing its agile practices (Kappelman et al., 2014). Agility is defined as a measure for contemporary organizations to adept to new environments and to cope with rapid change (Neumann, 1994).

The subjects discussed in previous paragraphs depict the current challenges of software evolvability

1.1 Problem statement

1.1.1 Normalized Systems Theorems

«To-do: Link Normalized Systems to thesis and content problem statement»//

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu,

[1] https://plato.stanford.edu/entries/process-philosophy/

pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

1.1.2 Clean Architecture

«To-do: Link Clean Architecture to thesis and content problem statement»// Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

1.1.3 Normalized Systems

Companies that apply the Normalized Systems Theory research into their products are primarily using Java EE as a programming language. The company NSX for example has implemented their generation tools, modelling suite (Prime Radiant) and expander using this programming language. Java EE is still a very popular programming language for enterprise-, and IT organizations. Many software solutions are created and maintained using this programming language. The Normalized Systems Theorem is not only applicable to Java EE. The principles and design patterns that derive from the Normalized Systems Theorem are in fact applicable for any object-oriented programming languages.

Another example of a popular programming language in enterprise-, and IT organizations is C#. There is however no documented research, or proof of experiences on C# software projects using Normalized Systems Theory with the aspects of integration, expansion and rejuvenation.

1.2 Research questions

The goal of this research is to determine if the design of

2 Theoretical background

2.1 Main Section 1

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

2.1.1 Subsection 1

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

2.1.2 Subsection 2

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

2.2 Main Section 2

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

3 Research and design approach

This chapter describes the overall research design approach and contains the conceptual framework that is applicable for research assignment at hand.

3.1 Research model

The research approach is based on the Design Science method. The following section describes the research model based on the Design Science research framework (Recker, 2013, (P. 107)). According to Recker Design Science has been formulated as followed:

> *'A research paradigm in which a designer answers questions relevant to human problems via the creation of innovative artifacts, thereby contributing new knowledge to the body of scientific evidence. The designed artifacts are both useful and fundamental in understanding that problem.'*

Figure 3.1 depicts a graphical view of the research approach based on the Design Science Research framework. The fundamental components of this research are two separate artifacts.

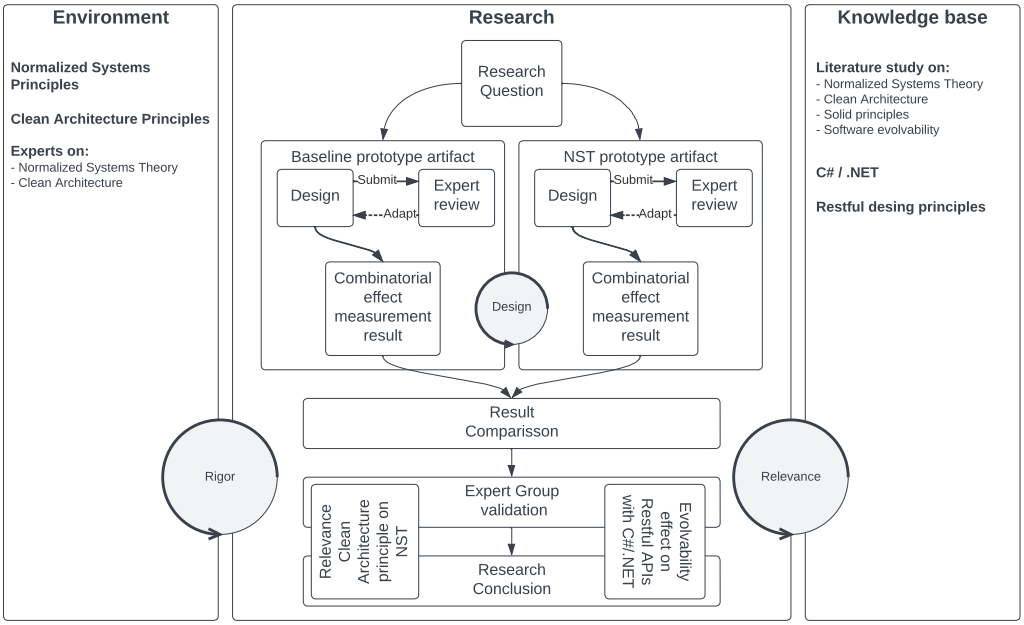

FIGURE 3.1: Research approach.

The first artifact is intended to be a working prototype of a Restful API, designed following to the Clean Architecture principles (Martin, 2018) using C#/.NET programming language. The second artifact follows the results of the baseline artifact. It is

enhanced with the design based on the Normalized Systems Theorems. The artifact uses the Software Generation concepts like expanding, rejuvenation and harvesting as proposed in the paper "On the Realization of Meta-Circular Code Generation and Two-Sided Collaborative Metaprogramming" (Mannaert, Cock, and Uhnak, 2020)

Each artifact will endure a review cycle done by field experts of on each of the given design principles. The review cycle is used to ensure that the design and implementation are according to those design principles.

Besides the review of the field experts, the artifacts will also be validated by using an automated instrument (script) that measures the number of combinatorial effects on both of the artifacts. The combinatorial effects are measured in a spectrum of changes in different area's of the artifacts. For example:

- A change in a data entity
- A change in a use case
- A change in an action
- A change in a validation
- etc...

The outcome of the measurements on combinatorial effects on both artifacts are the basis for the comparison results. These results will be discussed and validated by a control group determining the effect on the evolvability when using the Normalized Systems Theorems in a C#/.NET restful API.

3.2 Conceptual framework

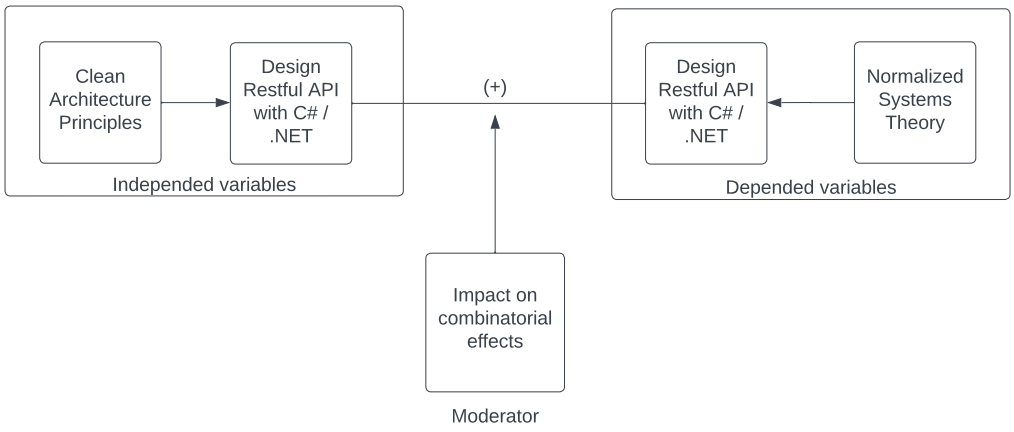

FIGURE 3.2: Overall conceptual framework.

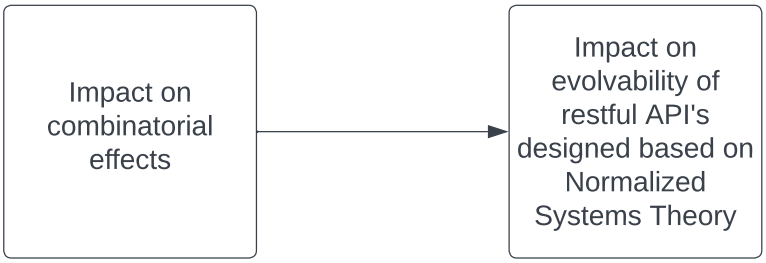

FIGURE 3.3: Conceptual framework.

A Frequently Asked Questions

A.1 How do I change the colors of links?

The color of links can be changed to your liking using:

`\hypersetup{urlcolor=red}`, or

`\hypersetup{citecolor=green}`, or

`\hypersetup{allcolor=blue}`.

If you want to completely hide the links, you can use:

`\hypersetup{allcolors=.}`, or even better:

`\hypersetup{hidelinks}`.

If you want to have obvious links in the PDF but not the printed text, use:

`\hypersetup{colorlinks=false}`.

Bibliography

Kappelman, Leon et al. (Dec. 2014). "The 2014 SIM IT Key Issues and Trends Study: Appendix". In: *MIS QUARTERLY EXECUTIVE* 13.4. ISSN: 1540-1960.

Lehman, M.M. (1980). "Programs, Life Cycles, and Laws of Software Evolution". In: *Proceedings of the IEEE* 68.9, pp. 1060–1076. ISSN: 0018-9219. DOI: 10.1109/PROC.1980.11805. URL: http://ieeexplore.ieee.org/document/1456074/ (visited on 04/25/2022).

Mannaert, Herwig, Koen De Cock, and Peter Uhnak (2020). "On the Realization of Meta-Circular Code Generation and Two-Sided Collaborative Metaprogramming". In: p. 11.

Martin, Robert C. (2018). *Clean Architecture: A Craftsman's Guide to Software Structure and Design*. Robert C. Martin Series. London, England: Prentice Hall. 404 pp. ISBN: 978-0-13-449416-6.

Neumann, Seev (1994). *Strategic Information Systems: Competition through Information Technologies*. New York : Toronto : New York: Macmillan College Publishing Co. ; Maxwell Macmillan Canada ; Maxwell Macmillan International. 258 pp. ISBN: 978-0-02-386690-6.

Recker, Jan (2013). *Scientific Research in Information Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN: 978-3-642-30047-9 978-3-642-30048-6. DOI: 10.1007/978-3-642-30048-6. URL: http://link.springer.com/10.1007/978-3-642-30048-6 (visited on 04/30/2022).