

**Service Oriented Code Generator for Fast
Prototyping**
Based on Requirement Definition Schema

**By D.U.I.Hewage
149211G**

**Faculty of Information Technology
University of Moratuwa**

May 2017

Service Oriented Code Generator for Fast Prototyping

Based on Requirement Definition Schema

Software tool providing easy prototyping ability for web applications

D.U.I. Hewage
149211G
(MSCIT/14/028)

Dissertation submitted to the Faculty of Information Technology, University of Moratuwa, Sri Lanka for the partial fulfillment of the requirements of the Degree of MSc in Information Technology.

Faculty of Information Technology
University of Moratuwa
May 2017

Declaration

I declare that this thesis is my own work and has not been submitted in any form for another Masters, Degree or diploma at any university or other institution of tertiary education. Information derived from the published or unpublished work of others has been acknowledged in the text and a list of references is given.

Name of the Student:

D. U. I. Hewage

Signature of the Student

.....

Date:

Supervised by:

Mr.Chaman Wijesiriwardana

Signature of the Supervisor

.....

Date:

Dedication

This dissertation is dedicated to my beloved parents, siblings, nimshi and her parents, my teachers who gave me endless courage and support to achieve my task and goal in completing the research project.

Acknowledgement

My heartiest thanks go to my supervisor Mr. Chaman Wijesiriwardana for the guidance, assistance, encouragement, valuable advices on improving the research and providing this opportunity carry out this research project.

Also, sincerely thanks to all my teachers who taught in MSc IT degree program. Things leant from these subjects made it easier to make this research project a successful one.

Last but not least, a sincere thank goes to everyone who supported specially teams from Redot Pvt Ltd and Effro Pte Ltd Singapore for contributing their valuable time on this research.

Abstract

With the rise of the latest web technologies, it has now become the mostly used software solution for lots of business areas. Because of its flexibility and easy connections between the clients and the server made via clouds, it's the most popular technology solution provider for the industry. Having said that, new and faster approaches for programming, planning, deploying and testing are being introduced at a rapid speed. As a result of this, large number of new tools and technologies are popping up in the industry. Even though they have introduced these tools and technologies to ease up development work, still there are some areas which are time consuming and costly for the management. To be specific Database Designing, Initial Project setup, Authentication module coding, Coding the User Interfaces, Writing CRUD functions covering every use case of the applications, setting up deploying mechanism, writing test cases, and many more tasks are still done manually or taken from a previous project. Even if they have taken it from a previous project they still have to code the CRUD operations and some other things which are not automated yet.

Goal of this research is to find out a proper automating mechanism for most of the tasks which are not yet automated using available open source projects and to combine a set of task specific automating tools to act as a complete solution. Although there are some systems available which developers are using to reduce repetitive work, and can manage their work using these systems, it can be improved further and save days to weeks from their development time. This way it can avoid concerns over the cost involved with the implementation because of the time spent on these repetitive tasks.

This project proposes a customized solution for avoiding repetitive tasks in software implementation. Reducing the time spent on these tasks is the main objective of this project which ultimately leads to a software prototyping application. Since the modern approach of software implementation is model driven, proposed system will also consist the model driven approach with its solution. The proposed system is capable of source code generation. Pre-generated source code can reduce the time spent on coding,

use a model driven approach, automate validations, maintain coding standards as well as the generate UI elements which are required to display data on frontend. Using only 2 schema files written in JSON format which describes the flow of the application, models and validation systems are capable of understanding and generating code based on these definition files. Since the application is using a very high level of definition of the user requirement, it is a forward engineering approach. Keeping in mind of the latest technologies and the mobile technology evolvement, the generated code will consist of 2 sections, namely the Front-end and the Back-end. Front end consists of the UI information and the flow of the application which the final client or the customer will experience. Back end is consisting of a highly customizable REST API which supports mobile implementation as well. To ease up the implementation of this project, project is using set of open-source projects such as angular-seed, expressjs. And the solution is provided using NodeJs. Model-Driven Application Prototyping and Code Generation using Forward Engineering System (FES) supports code generating in multiple languages and supports multiple DBMSs such as MySQL, SQL, MongoDB, etc. I have tested the system with the colleagues at my work place which all of them are developers. And currently using the system for generating first and second level prototypes.

Finally, I have achieved the task of implementing a software solution which generates model driven, initial project setup, reusable and testable code, supports multiple databases and greatly reduce time spent on repetitive work. And in the first phase of the most web application projects in can reduce the project setup time by nearly 1.5 weeks as well as 4-10 hours of 1 module of the code.

Keywords: prototyping, model driven, code generation, forward engineering, CRUD automation, multi-language

Table of Contents

1. Introduction	1
1.1. Prolegomena.....	1
1.2. Background and Motivation.....	3
1.3. Problem domain	5
1.4. Hypothesis	7
1.4.1. Expected Features	8
1.5. Aim and Objectives.....	8
1.5.1. Aim	8
1.5.2. Objectives	8
1.6. Structure of the Thesis.....	9
1.7. Summery	9
2. Current Development and Challenges.....	10
2.1. Introduction	10
2.2. Current developments	10
2.2.1. Toward automatic generation of mvc 2 web applications	10
2.2.2. A Generator of MVC-based Web Applications.....	11
2.2.3. A Model-Driven Approach for the Fast Prototyping of Web Applications.....	12
2.2.4. XFlash—a web application design framework with model-driven methodology	13
2.2.5. Leveraging declarative languages in web application development.....	14
2.2.6. An Effective Development Environment Setup for System and Application Software	14
2.2.7. Other works.....	15
2.3. The Research Gap	16
2.4. Summery	18

3.	Technology foundation of the solution.....	19
3.1.	Introduction	19
3.2.	Technologies used for the solution.....	19
3.2.1.	Server-side JavaScript powered by NodeJs	19
3.2.2.	ExpressJs framework	19
3.2.3.	JSON	20
3.2.4.	Client-side JavaScript powered by AngularJs	20
3.2.5.	Multiple DBMS support	20
3.2.6.	Mocha and Chai	20
3.2.7.	Bitbucket	21
3.3.	Development tools.....	21
3.3.1.	Visual Code.....	21
3.3.2.	PhpMyAdmin.....	21
3.3.3.	Ampps	21
3.3.4.	mobaXterm	22
3.4.	Hosting and Deployment technologies	22
3.4.1.	AWS EC2.....	22
3.4.2.	Beanstalk.....	22
3.5.	Summery	23
4.	A new approach to fast prototyping	24
4.1.	Introduction	24
4.2.	Requirement Gathering	24
4.2.1.	Issues faced by the project teams working in the industry	24
4.2.2.	Gaps identified which hasn't been addressed by other researches	25
4.2.3.	Limitations with current development tools	25
4.3.	Hypothesis.....	26
4.3.1.	Fast prototyping	26

4.3.2.	Code generators	26
4.3.3.	Prototyping services.....	27
4.3.4.	Schema based requirement definitions	27
4.4.	Users of the system	27
4.5.	Inputs to the system.....	28
4.5.1.	Flow Definition Schema	29
4.5.2.	Data Model Definition Schema	29
4.6.	Outputs of the system.....	30
4.7.	Features	30
4.8.	Summery	31
5.	Analysis and Design of the new prototyping solution.....	32
5.1.	Introduction	32
5.2.	Research planning	32
5.2.1.	Development methodology.....	33
5.2.2.	Selection of the software process model.....	33
5.3.	Analysis of the current development workflow	34
5.4.	Requirement analysis	34
5.4.1.	Functional requirements of the solution	34
5.4.2.	Non-functional requirements of the solution	35
5.5.	Top level design architecture	35
5.6.	Module architecture.....	36
5.7.	Schema based requirement definition	38
5.7.1.	Application flow definition.....	39
5.7.2.	Data Model Definition	39
5.8.	Generated Prototype.....	40
5.8.1.	Generated client-side application design architecture	41
5.9.	Dependency management, ground up work on generated prototype	42

5.9.1.	Version controlling	43
5.9.2.	Dependency management	43
5.9.3.	Compiling and building	43
5.9.4.	Testing and continuous integration.....	44
5.10.	Summery.....	44
6.	Implementation.....	45
6.1.	Introduction	45
6.2.	Overall solution	45
6.3.	Implementation of the Solution.....	45
6.3.1.	Preparation	46
6.3.2.	Programming.....	46
6.4.	Requirement definition processing stored in schema files.....	46
6.4.1.	System flow definition mechanism.....	46
6.4.2.	Data Model Definition	49
6.4.3.	Seed projects	50
6.5.	Language drivers for source code generators.....	50
6.5.1.	Multiple server-side languages support	51
6.5.2.	Modularized architecture	52
6.6.	Generated sample user interfaces	53
6.7.	Summery	54
7.	Evaluation of the solution.....	55
7.1.	Introduction	55
7.2.	Case study 1 – Feasibility for a library management system.....	55
7.2.1.	Problem definition	55
7.2.2.	Requirement analysis	56
7.2.3.	Design and Implementation	56
7.2.4.	Evaluation	56

7.3.	Case study 2 – Service Oriented Prototyping and Code generation	58
7.3.1.	Problem definition	58
7.3.2.	Requirement specification	59
7.3.3.	Design and implementation	59
7.3.4.	Evaluation	59
7.4.	Other evaluations.....	61
7.4.1.	Participants.....	61
7.4.2.	Testing environment	63
7.4.3.	Test cases	63
7.4.4.	Data analysis	66
7.5.	Aim.....	67
7.6.	Objectives.....	68
7.7.	Drawbacks and limitations	68
7.8.	Summery	69
8.	Conclusion.....	70
8.1.	Introduction	70
8.2.	Conclusion.....	70
8.3.	Future works.....	71
8.4.	Summery	72
	References.....	73
	Appendix A.....	75
	Interfaces of the generated prototypes	75
	Appendix B.....	77

Table of Tables

Table 2-1 List of evaluation criteria for existing prototyping tools.....	17
Table 2-2 Evaluation done for existing prototyping tools	17
Table 7-1 Evaluation conducted in the case study 1.....	58
Table 7-2 Evaluation Conducted in the case study 2.....	60

Table of Figures

Figure 1-1 Web application and a mobile application communicating with the same web server API.....	7
Figure 4-1 Inputs to the proposed system	28
Figure 4-2 Outputs of the proposed system	30
Figure 5-1 Execution plan for the proposed system	32
Figure 5-2 Evolutionary prototyping methodology used for proposed system	33
Figure 5-3 Top level design architecture of the proposed system	35
Figure 5-4 Modules contained in the file manager	36
Figure 5-5 Modules contained in the schema processor.....	36
Figure 5-6 Modules contained in the language driver	37
Figure 5-7 Modules contained in the seed project.....	38
Figure 5-8 Generated prototype top level design.....	40
Figure 5-9 Detailed view of the data communication between generated client-side and the server-side application.....	41
Figure 5-10 Process of the client-side application working with gulp task runner	42
Figure 6-1 Sample Application Flow Definition Schema	48
Figure 6-2 Sample Application Data Model Definition Schema.....	49
Figure 6-3 Generated prototype structure	50
Figure 6-4 Language Driver Structure.....	51
Figure 6-5 Language Driver Constructor.....	51

Figure 6-6Example of generated modularized architecture.....	52
Figure 6-7Generated UI source code based on data model	53
Figure 6-8Sample generated UI.....	53
Figure 6-9 Sample generated UI.....	54