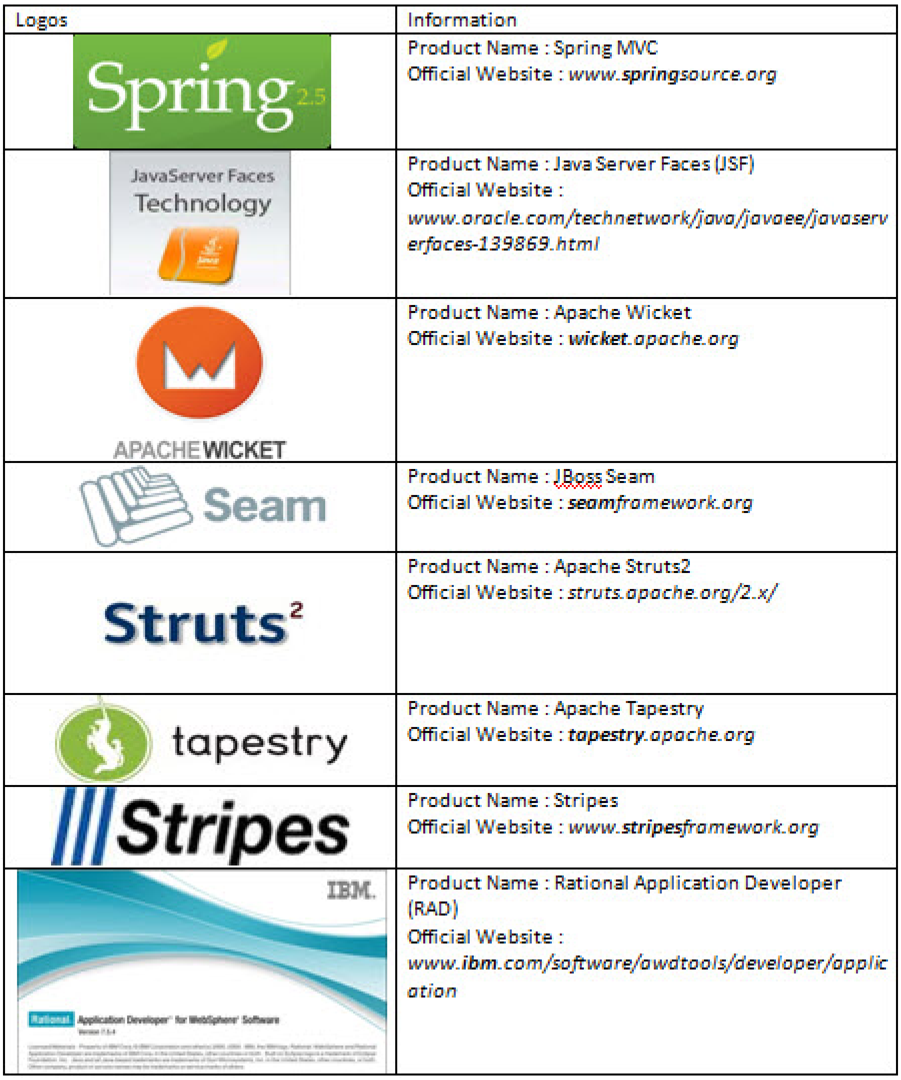
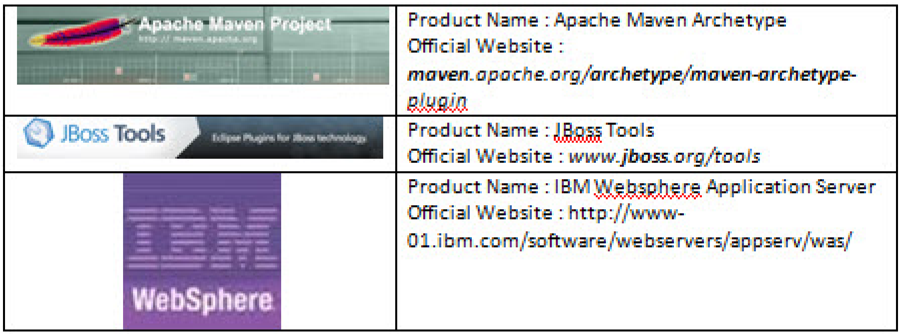
A. Appendix

A.1. Frameworks and Related Tools List



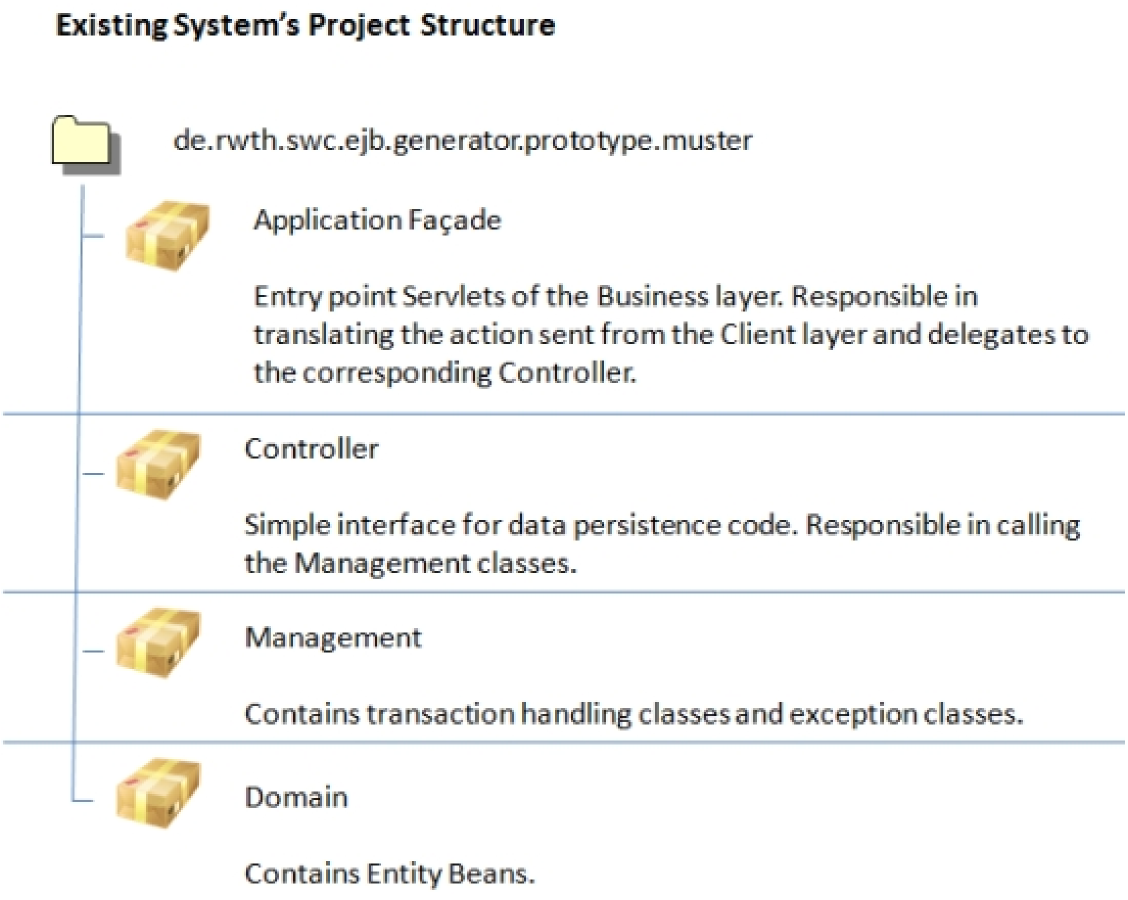
Logos	Information
Spring 2.5	Product Name : Spring MVC Official Website : *www.springsource.org*
JavaServer Faces Technology	Product Name : Java Server Faces (JSF) Official Website : *www.oracle.com/technetwork/java/javaee/javaserverfaces-139869.html*
APACHE WICKET	Product Name : Apache Wicket Official Website : *wicket.apache.org*
Seam	Product Name : JBoss Seam Official Website : *seamframework.org*
Struts²	Product Name : Apache Struts2 Official Website : *struts.apache.org/2.x/*
tapestry	Product Name : Apache Tapestry Official Website : *tapestry.apache.org*
Stripes	Product Name : Stripes Official Website : *www.stripesframework.org*
IBM. Rational. Application Developer for WebSphere Software	Product Name : Rational Application Developer (RAD) Official Website : *www.ibm.com/software/awdtools/developer/application*



Apache Maven Project	Product Name : Apache Maven Archetype Official Website : *maven.apache.org/archetype/maven-archetype-plugin*
JBoss Tools	Product Name : JBoss Tools Official Website : *www.jboss.org/tools*
IBM WebSphere	Product Name : IBM Websphere Application Server Official Website : http://www-01.ibm.com/software/webservers/appserv/was/

A.2. Project Structure

A.2.1. Existing System's Main Project Structure (Muster)



Existing System's Project Structure

de.rwth.swc.ejb.generator.prototype.muster

Application Façade

Entry point Servlets of the Business layer. Responsible in translating the action sent from the Client layer and delegates to the corresponding Controller.

Controller

Simple interface for data persistence code. Responsible in calling the Management classes.

Management

Contains transaction handling classes and exception classes.

Domain

Contains Entity Beans.

Figure A.1.: Existing System's Main Project Structure

A.3. Full-Version Comparison Tables

A.3.1. General Web Framework Criteria Comparison Table

* DI = Dependency Injection **AOP = Aspect Oriented Programming

	Spring MVC	**JSF**	**Wicket**	**Seam**
Architecture and Patterns	MVC, DI, AOP request/ response-based framework	MVC, component and event-based framework	Clear separation of presentation and logic, POJO for logic and HTML template for presentation, component-based framework, DI	Integrate AJAX, JSF or Wicket, EJB 3.0, JBPM together + Seam POJOs
Support IBM Websphere 7.0	Yes	Yes	Yes	Yes
Support EJB	Yes	Yes	Yes	Yes
Learning Curve	Very Steep	Low at the beginning level and steep for advance usage (custom component, Facelets).	Very Low	Steep
User Interface	Integrate with many view options such as JSP/JSTL, Tiles, Velocity, FreeMarker , Excel, XSL, PDF	A lot of UI component and libraries. Also custom UI components.	HTML templates and CSS	JSF components, Wicket, GWT

Validation	Spring Common Validators are mature solution. Bean Validation can be done using annotations and the error messages will appear automatically on the view.	Default validation messages need configuration .Bean Validation can be done using annotations and the error messages will appear automatically on the view.	Both server and client-side validation. Using DHTML on client-side validation.	JSF default validation messages, Wicket, GWT
Degree Complexity	Very Complex but very flexible.	Very simple configuration file (xml) using the default editor, which can manages navigation rules and beans effectively through diagrams and UI.	No configuration file, XML, or annotation (zero configuration)	Simple xml configuration files for deployment time but mainly, based on annotation-based configuration
Testability	Easy to test. The business and navigation logic are separated from presentation logic. EasyMock and Spring Mocks are supported tools for mock object creation.	Easy to test. presentation layer but Hard to test business layer independently. There are many tools support such as JSFUnit, InfoQ, and many which allow testing inside the container.	Easy to test. using WicketTester api, api for Unit test for Wicket. No need for Servlet container and including mock object.	Easy to test. since, all components are POJOs. For the integration testing, mock object creation tools can be used.

Community and Support	Very active community and supports	Very active community and supports	Very active community and supports	Very active community and supports
Tools and IDEs	Only Spring IDE available. It supports only for XML validation.	Many supporting tools such as Eclipse plugin, NetBeans, Websphere IDE, Oracle Developer.	No official tool supported. However, Wicket need no specific tools.	JBoss Eclipse plugin allow jBPM viewer, CRUD application generator, and reverse engineering.
Strengths	- One of the most flexible framework - Clear separation of business logic, model, and view - Very easy to integrate with other view technology - Easy to test	- J2EE and JSR standards - Many 3rd party libraries such as PrimeFaces, ADF Faces, IceFaces, Trinidad, RichFaces, and more.	- Very famous in term of one of the most vibrant communities (forums, mailing lists) - Very Light weight - Very low learning curve - Very clear and strict separation of presentation and logic - Very productive - No XML configuration	- JBPM - Design specifically for EJB 3.0 - JSF optimization - Full-stack framework with CRUD generation
Weaknesses	- One of the most complicate and steepest learning curve framework - Large and complex XML configuration files with poor IDE supported	- Heavy weight in term of memory consumption	- Good knowledge in OO is required - Huge amount of artifacts produced	- Not flexible (JSF, Wicket, GWT for presentation layer/ EJB for Business logic and Data layer/ work best on JBoss Web Server)

Table A.1.: Framework Comparison Matrix Part 1

	Struts 2	**Tapestry**	**Stripes**	**JSP/Servlet (Current Solution)**
Architecture and Patterns	MVC, DI, request/ response-based framework	DI , component-based framework, POJO for logic and HTML template for presentation, Clear separation of presentation and logic	MVC, action-based framework	request/ response based, command pattern (GoF)
Support IBM Websphere 7.0	Yes	Yes	Yes	Yes
Support EJB	Yes	Yes	Yes	Yes
Learning Curve	Average	Steep	Very Low	Very Low
User Interface	Integrate with many view options such as JSP/JSTL, Tiles, Velocity, FreeMarker, Excel, XSL, PDF	HTML templates and CSS	JSP with Stripes components	Simple JSP, HTML, CSS, JSTL and other taglibs
Validation	Provides basic validation using XML and more powerful validation using OGNL	Tapestry default client-side validation is very powerful even without customization.	No client-side validation.	Javascript validation but currently, no validation at all.

Degree Complexity	Average complexity for both XML and annotations.	Low. No external configurations (Annotation only).	Low. No external configurations (Annotation only). ActionBeans are auto-discovered.	No configuration required.
Testability	EasyMock and TestStruts2 are supported tools for mock object creation.	Hard to test since, page classes are abstract.	Servlet API Mocks and MockRound Trip are supported tools for mock object creation.	Hard to do the Unit Testing because of dependencies and not clear separation of view and logic.
Community and Support	Not very active communities with poor organized documentation.	Not very active communities with poor organized (conceptual rather than pragmatic).	Small communities. Not good documentation. No books. Not actively developed as other project.	There are plenty of learning resources such as books, websites, tutorials.
Tools and IDEs	EclipseWork Eclipse plugin supported.	Spindle Eclipse Plugin allow debug, auto-completion, and tools for fast artifact creation.	No official tool supported. However, Stripes need no specific tools.	There are plenty of tools and IDEs such as Eclipse, NetBeans.

Strengths	- Widely used - Very easy to integrate with other view technology - The interceptors provides high reusability of the common code in the system.	- Very good choice in term of CPU and memory usage optimization - Live class reloading (no redeploy, no restart - Very productive	- No XML configuration - Easy to manage large complex form using type conversion, binding, validation - Very low learning curve - Very good documentation	- Very simple and required only basic knowledge of - web development to understand.
Weaknesses	- Poor organized document and resource	- No backward compatibility in Tapestry 5.0 for lower version.	- Very small community and not actively developed	- Produced a lot of action classes, Dependencies between classes.

Table A.2.: Framework Comparison Matrix Part 2

Architecture and Patterns	Brief description about the architecture of the framework and design patterns using by the framework.
Support IBM Websphere 7.0	Does the framework support working with IBM Websphere 7.0 (existing restrict environment)?
Support EJB	Does the framework support integration with EJB 3.0 (existing restrict environment)?
Learning Curve	The slope of learning curve of the framework. The criteria that determine the slope of learning curve are architecture simplicity and learning time consumption.
User Interface	List of possible technologies, which can integrate with the framework in order to develop User Interface.
Validation	Brief description about form validation support of the framework.
Degree of Complexity	Determine the complexity of the artifacts produced by the framework especially, configuration method.
Testability	Description of test method and difficulty provides by framework and support testing tools.
Community and Support	Description of framework's community activity and learning resources.
Tools and IDEs	List of framework's support tools/IDEs and descriptions.
Strengths	Strengths of the framework.
Weaknesses	Weakness of the framework.

Table A.3.: Description of the Criteria

Bibliography

[AH03] ABDELNUR, ALEJANDRO and STEFAN HEPPER: *Java Portlet Specification 1.0.* http://jcp.org/aboutJava/communityprocess/final/jsr168, 2003.

[All08] ALLEN, DAN: *Seam in Action.* Manning, 2008.

[BDS08] BROWN, DONALD, CHAD MICHAEL DAVIS and SCOTT STANLICK: *Struts2 in Action.* Manning, 2008.

[Ber04] BERGSTEN, HANS: *Improving JSF by dumping JSP.* O'Reilly on Java.com, 2004.

[DH09] DASHORST, MARTIJN and EELCO HILLENIUS: *Wicket in Action.* Manning, 2009.

[FRBS04] FREEMAN, ERIC T, ELISABETH ROBSON, BERT BATES and KATHY SIERRA: *Head First Design Patterns.* 2004.

[Hig06] HIGHTOWER, RICHARD: *Facelets fits JSF like a globe.* IBM, 2006.

[HSD10] HOLZINGER, ANDREAS, KARL HEINZ STRUGGL and MATJAZ DEBEVC: *Applying Model-view-controller(MVC) in Design and Development of Information Systems - An Example of Smart Assistive Script Breakdown in an e-Business Application.* 2010.

[IBM11] IBM: *IBM Websphere Application Server V7 - Product Overview.* Websphere Software, 2011.

[Jos05] JOSHI, ANAND PRAKASH: *Design with the JSF Architecture.* IBM, 2005.

[KMR$^+$11] KING, GAVIN, PETE MUIR, NORMAN RICHARDS, SHANE BRYZAK, MICHAEL YUAN, MIKE YOUNGSTROM, CHRISTIAN BAUER, JAY BALUNAS, DAN ALLEN, MAX RYDAHL ANDERSEN, EMMANUEL BERNARD, NICKLAS KARLSSON, DANIEL ROTH, MATT DREES, JACOB ORSHALICK, DENIS FORVEILLE, MAREK NOVOTNY and JOZEF HARTINGER: *Seam - Contextual Components. A Framework for Enterprise Java 2.2.1.Final.* IBM Software, 2011.

[ldt11] TEAM, STRUTS2 EJB3 DEPENDENCY INJECTION LIBRARY DEVELOPMENT: *Struts2 EJB3 Dependency Injection.* http://code.google.com/p/struts2ejb3-jboss-plugin (Access: 22. August 2011), 2011.

[LL07] Ludewig, Jochen and Horst Lichter: *Software Engineering.* dpunkt.Verlag GmbH, 2007.

[Man05] Mann, Kito D.: *Java Server Faces in Action.* Manning, 2005.

[Nad08] Nadar, Kumarsun: *A simplified framework for building and testing dynamic Web pages.* IBM, 2008.

[PRL07] Panda, Debu, Reza Rahman and Derek Lane: *EJB3 in Action.* Manning, 2007.

[PRV05] Perovich, Daniel, Leonardo Rodriguez and Andres Viganaga: *Architecture of Component based Information Systems over the J2EE Platform.* 2005.

[Rai11a] Raible, Matt: *Comparing Web Frameworks.* http://static.raibledesigns.com/repository/presentations/ComparingJavaWebFrameworks ApacheConEU2007.pdf, 2011.

[Rai11b] Raible, Matt: *Web page of Raible Designs, based on Google Trends statistic (Access: 2. August 2011).* http://raibledesigns.com/rd/entry/re_which_is_the_hottest, 2011.

[Shi04] Ship, Howard M. Lewis: *Tapestry in Action.* Manning, 2004.

[Str11] Struts2, Apache: *Comparing Struts1 and 2.* http://struts.apache.org/2.0.14/docs/comparing-struts-1-and-2.html (Access: 22. August 2011), 2011.

[Tho11] Thomas, Peter: *Seam / JSF vs. Wicket: performance comparison.* http://ptrthomas.wordpress.com/2009/01/14/seam-jsf-vs-wicket-performance-comparison (Access: 27. September 2011), 2011.

[Tur10] Turnaround, Zero: *Java EE Productivity Report 2011 (Access: 2. July 2010).* http://zeroturnaround.com/java-ee-productivity-report-2011, 2010.

[WB05] Walls, Craig and Ryan Breidenbach: *Spring in Action.* Manning, 2005.

[WCF$^+$07] Wahli, Ueli, Henry Cui, Craig Fleming, Maan Mehta, Marco Rohr, Pinar Ugurlu, Patrick Gan, Celso Gonzalez, Deniel M. Farrell, Andreas and Heerdegen: *Rational Application Developer V7 Programming Guide.* Rational Sofrtware, IBM Redbooks, 2007.