



VAASAN AMMATTIKORKEAKOULU
VASA YRKESHÖGSKOLA
UNIVERSITY OF APPLIED SCIENCES

Baiqi Wu

WÄRTSILÄ PROJECT PURCHASING AND LOGISTICS CONFIGURATOR

Department of Technology and Communication

2013

ABSTRACT

Author	Baiqi Wu
Title	WSPT Project Purchasing and Logistics Configurator
Year	2013
Language	English
Pages	78
Name of Supervisor	Ghodrat Moghadampour

The purpose of this thesis was to build up a Rich Internet Application (RIA) application on logistics field for Wärtsilä Oyj, which is a global company, and one of the world-leading providers in ship engines and power plants.

The Wärtsilä Project Purchasing and Logistics Pre-Configurator (WSPT) application was designed for Cost Estimation (CE), planned shipment time, Weight Estimation (WE), Volume Estimation (VE), Loading and Unloading Plan, Route Schedule, Risk Management, Freight Rate Request, Port Information Management, PDF file generation and admin functionality in order to make the logistics process more convenient and efficient.

The WSPT application user interface was built based on Vaadin Framework. The Hibernate framework was decided for database management and Struts 2 framework made entry page.

All features of the final thesis motioned above were achieved in WSPT application. The application has been demonstrated to the Wärtsilä user. The results of the tests indicated that all objectives work as customer requirements. The status of the project is waiting to change to the product version.

ACKNOWLEDGEMENT

The WSPT was done between May 2011 and January 2012. First, I would like to thank my supervisor Dr. Ghodrat Moghadampour for his continuous support and hearted assistance to complete my thesis and Bachelor's degree. When I came to Finland, I did not even know about the programming. His patient teaching, courage and immense knowledge helped me from an IT idiot to be a good IT student.

Beside my supervisor, I would like to thank Petri Helo, who gave me this chance to work with him and helped me step by step to build the project, communication with the customer and design the business logistics structure. I could not have succeeded in this project without his guidance and courage.

My sincerely thanks go also to all the people who accompanied me during my school life, Dr. Cao Gao, Dr. Yang Liu, my friends Vinh Vo, Lam Lee, Tan Wei and my parents and all other friends. Just because of you, my college life became so wonderful.

CONTENTS

ABSTRACT

LIST OF ABBREVIATION.....	3
1 INTRODUCTION.....	6
1.1 WSPT Project Scope	7
2 TECHNOLOGY OVERVIEW	9
2.1 JAVA.....	9
2.2 Vaadin Framework.....	10
2.3 Apache Tomcat	11
2.4 Hibernate	12
3 WÄRTSILÄ LOGISTICS CONFIGURATOR.....	15
3.1 Main Features Deployment	16
3.2 WSPT Use Case Diagram	17
3.3 WSPT MVC Architecture	22
3.3.1 WSPT Model Design.....	22
3.3.2 WSPT Model Functionality.....	23
3.4 Main Functions Description with Sequence Diagram.....	24
3.4.1 User Login and Register function.....	24
3.4.2 Configure and Establish New Logistics Project.....	26
3.4.3 Cost Estimation	28
3.4.4 Map View	30
3.4.5 Result Filtering	31
3.4.6 Schedule in PDF	32
3.4.7 Admin Function.....	33
3.5 WSPT Database Design	34
3.5.1 WSPT Entities	34
3.6 Architectural Overview	43
4 WSPT USER INTERFACE DESIGN.....	45
4.1 WSPT Configurator View.....	45
4.2 WSPT Configure Result View	46
4.3 Details Info View	47

4.4	PDF View Layout.....	48
4.5	Administrator View.....	49
4.6	Schedule Info View.....	50
5	IMPLIMENTATION AND DEPLOYMENT.....	51
5.1	Deployment.....	51
5.1.1	Deployment Descriptor file.....	51
5.1.2	Struts 2 configuration file.....	53
5.1.3	Application Property File.....	54
5.1.4	Hibernate Configuration File.....	55
5.2	Main Functions Implementation.....	57
5.2.1	Create new Configuration.....	57
5.2.2	Sort Result by Price.....	58
5.2.3	Result Filter.....	60
5.2.4	WSPT Map.....	61
5.2.5	Route Schedule Table.....	62
5.2.6	Generate PDF project file.....	65
5.2.7	Administrator Part Sample Implementation.....	66
5.2.8	WSPT Calculation Implementation.....	68
6	SUMMARY.....	73
7	CONCLUSION.....	75
7.1	Further Development.....	76
8	REFERENCES.....	78

LIST OF ABBREVIATION

RIA	Rich Internet Application
WSPT	Wärtsilä Logistics Preconfigurator and Transportation
PC	Personal Computer
TCE	Transport Cost Estimation
TTE	Transporting Time Estimate
VE	Volume Estimation
WE	Weight Estimation
RF	Red Flag
LPI	Loading Port Information
DPI	Destination Port Information
LP	Loading Plan
UP	Unloading Plan
RP	Route Proposal
SS	Security Situation
OS	Operating System
GWT	Google Web Toolkit
CSS	Cascading Style Sheet
CSRF	Cross-site Request Forgery
SSL	Security Socket Layer
IDE	integrated drive electronics
ASF	Apache Software Foundation
NCSA	National Center for Supercomputer Applications
ASF	Apache Software Foundation
JSP	Java Server Pages

EJB	ENTERPRISE JAVA BEANS
HQL	HIBERNATE QUERY Language
HDLCA	Hibernate Dual-Layer Cache Architecture
BB	Break Bulk
FCL	Full Container Load
FRQ	Freight Rate Request
DB	Database
MVC	Model/view/controller
DAO	Data Access Object
RFQ	Request For Quotation
EER	Enhanced Entity-Relationship
PK	Primary Key
FK	Foreign Key
ID	Identification
BAF	Bunker Adjustment Factor
CAF	Currency Adjustment Factor
GUI	Graphical User Interface
ETA	Estimated Time of Arrival
ETD	Estimated Time of Departure

LIST OF APPENDICES

APPENDIX 1. JDK Setup for Windows 7 Platform

APPENDIX 2. Apache Server Setup for Windows 7 Platform

APPENDIX 3. Eclipse – Java Development Tool on Windows

1 INTRODUCTION

With the rapid development of information technology, web applications have been increasing in recent years. Compared with the desktop application, the advantages of web application for users are:

- No need for installation and updating
- Access anywhere through the available internet
- Better compatibility on cross platform
- Save disk space and easier to use
- Data stored remotely
- Better antivirus security than on PC

Unfortunately, the increasing complicated demand to build applications has continued to exceed the ability of traditional web applications. For these reasons, RIA (Rich Internet Application) has increasingly attracted the attention of enterprises, which is great to see because of a better user experience, which can make a huge difference for big companies.

Table 1. Web, Desktop and RIA comparison /1/

	Desktop	Web Application	RIA
Rich User Experience	Y	N	Y
Interactive Responsive	Y	N	Y
Low Maintenance	N	Y	Y
High Reach	N	Y	Y

Looking at the RIAs future, the industry changes their OA from desktop to the web gradually. There are many frameworks to create RIAs written in Java, Flash or Ajax, etc.

One of the world's leading providers of ship engines and power plants company, Wärtsilä, has requested to create a web application for project purchasing and logistics management so that it can be utilized in Wärtsilä global logistics project. The purpose of this application is to help Wärtsilä Oyj preliminary and efficiency to configure a reasonable, lower cost and higher quality logistics project.

1.1 WSPT Project Scope

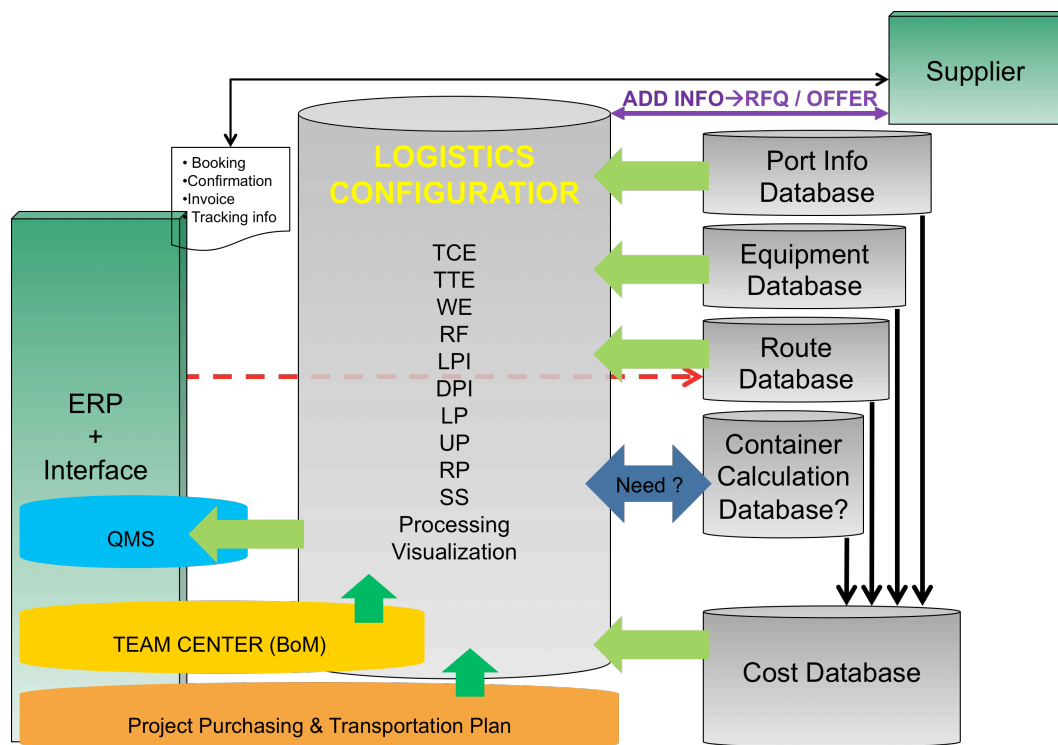


Figure 1. Preview of the business logical for WSPT

The requirements from Wärtsilä are briefly described (passive voice) by the above flow chart. The main idea of WSPT is a logistics configurator, and it should contain the following basic functions:

- Transport Cost Estimation (TCE)
- Transporting Time Estimate (TTE)
- Volume Estimation (VE)
- Weight Estimation (WE)
- Red Flag (RF)

- Loading Port Information (LPI)
- Destination Port Information (DPI)
- Loading Plan (LP)
- Unloading Plane (UP)
- Route Proposal (RP)
- Security Situation (SS)
- Booking and Tracking

These functions are quite necessary and useful in a logistics project. For example, when we want to buy a flight ticket from a flight ticket website, we always to compare the different operators in price, time or amenity etc and choose the most satisfying one. The idea is the same for the company to configure a logistics project, but it is more complicated. That is why this project is quite useful for the Wärtsilä OYj.

2 TECHNOLOGY OVERVIEW

This part is a description of technical used for this application. The programming language is Java, the platform is windows OS or Linux OS, the main RIA framework is Vaadin, the database Framework is Hibernate, and the server can be Tomcat/JBoss. There will be an introduction and definition for each part and the main features of those technologies will be described in detail below.

2.1 JAVA

Java is the most popular programming language according to the authoritative ranking by TIOBE site, because of its powerful library, simple way of thinking and perfected design structure.

Java was developed by a small group of Sun engineers called “Green Team” which was led by James Gosling in 1991. Due to its amazing development, Java has changed our world and daily life. Today, Java has permeated everywhere, from mobile devices, games and navigation systems to e-business. /2/

Java is a high level and object oriented programming language. It enables programmers to write in English letters.

There are four main principles in Java

1. **Easy to use:** The ancestor of Java is C++. Although C++ is a powerful language, it was too complex, and cannot meet all of Java's requirements. Java built on, and improved the ideas of C++, to provide a programming language that was powerful and easy to use. /3/
2. **Reliability:** The likelihood of fatal errors from the programmer mistakes can be reduced by Java. With this in mind, object-oriented programming was introduced. The Java's robustness increased by its data and manipulation were packaged together in one place. /3/
3. **Security:** As Java was originally targeting mobile devices that would be exchanging data through networks, it was built to include a high level of

security. Java is probably the most secure programming language to date. /3/

4. **Platform Independent:** Programs needed to work regardless of the machine they were being executed on. Java was written to be a portable language that does not care about the operating system or the hardware of the computer. /3/

2.2 Vaadin Framework

A framework was to be selected to implements WSPT for achieving the customer requirements and compare with other RIA frameworks. The final implementation framework is Vaadin, because it is easy to use and highly efficient.

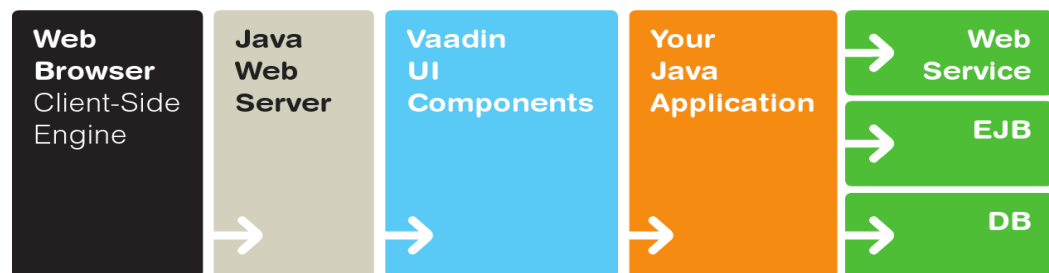


Figure 2. General Architecture of Vaadin /4/

Vaadin is a server-side AJAX web application development framework that enables the developer to build high-quality user interfaces with Java. It provides a library of ready-to-use user interface components and a clean framework for creating your own components. The focus is on ease-of-use, re-usability, extensibility and meeting the requirements of larger enterprise applications. Vaadin has been used in production since 2001 and it has proven to be suitable for building demanding business applications. /4/

Vaadin has eight main features, which make it as suitable framework for this project:

1. **Easy UI development:** The core piece of Vaadin is the Java library that is designed to make creation and maintenance of high quality web-based user interfaces easy. /4/
2. **Comprehensive Component:** Vaadin has a large collection of user interface components, controls and widgets. /5/
3. **Web Capability:** Vaadin is developed based on GWT, the Vaadin application can run on any browser without any browser plugins such as Flash Player. It also supports browser window, tab, back button, deep linking functions. /5/
4. **Customizable Look and Feel:** The developer can build custom application themes and embedded to any web page. It supports CSS styling. /5/
5. **Java Web Development:** The developer can concentrate on their Java coding, because Vaadin is Java-only. /5/
6. **Secure Web Application Architecture:** Vaadin is a server-side state management, the application code runs in the server and build-in input validation framework. The framework also supports CRSF protection and SSL protocol. /5/
7. **Extensible Component Architecture:** The developer can easily integrate the special add-ons in Vaadin Directory. /5/
8. **Tools support for Vaadin:** Eclipse IDE integration, Netbeans IDE integration and Maven support and artifacts. /5/

2.3 Apache Tomcat

Apache tomcat is one of the earliest web servers. It was developed under the ASF with Apache License, which is an open source license for free to use. Cheerfully, Tomcat sever can run very stable on most operating systems. Moreover, Java is to be closely connected to Tomcat. Tomcat was created in the earliest day of Java Servlet technology, so it is fully compliant with the Servlet and JSP. /6/

Two main components in Tomcat are:

1. **Catalina:** Tomcat's servlet container implementation, called Catalina. It used to be represented as hierarchy relations between objects in Tomcat.

The object hierarchy is described based on the configuration elements in the conf/server.xml. /7/

2. **Coyote:** The Coyote HTTP/1.1 Connector is a connector component that supports HTTP/1.1 protocol. The Coyote is responsible to render and execute servlets and JSP pages. Coyote listen to the user request from the client side and responses to it on the server side.

In additional, the Tomcat web server has much more components than uppers such as Jasper, Jasper 2, SSL, and Virtual Hosting. But, the components mentioned above are two main components to support the Tomcat running mechanism.

2.4 Hibernate

In order to satisfy customer requirements, the MySql is the choice for the database part, but at the same time, considering the complicated and large-scale storage of information need to be managed was considered. The normal way to develop database implementation will be so huge work for the programmer that the ORM evolved as a solution and Hibernate is one of the most popular ORM frameworks.

With increasing demand for database, the traditional approaches to implement database are not as easy as we think anymore, so some standards has derived from the Java Persistence API with the release of EJB to instead of previous complicated one and Hibernate is a implementation of Java Persistence API standard. /8/

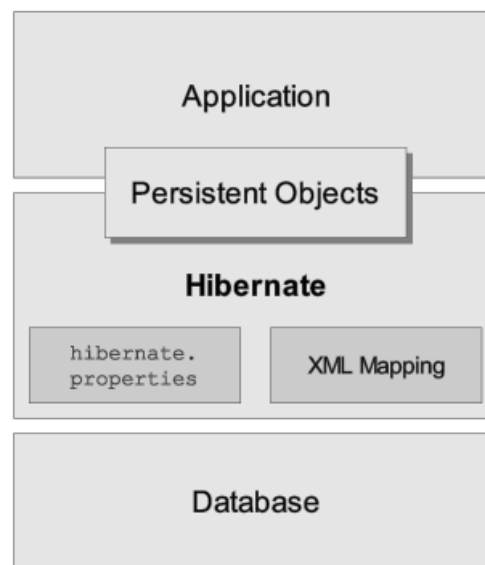


Figure 3. Hibernate Architecture /17/

Figure 3 describes the basic structure of Hibernate framework with database communication. Practically, Hibernate is a bridge between the model and control part. It uses mapping technology and JDBC for data exchange with database. There are four main features in Hibernate:

1. **Transparent persistence without byte code processing:** Hibernate provide a Java code generation library, it is used to implement Java interface that if any data changes for object transaction, they will catch the actions and propagated to database. It can save a lot of time to spend on the byte processing. /9, 1381/
2. **Object-oriented query language:** Hibernate has its own query language named HQL which is defined by EJB 3.0. The HQL can help the developer to write dynamic queries in an easier way. /9, 1381/
3. **Object / Relational mappings:** ORM is like a bridge to connect the gap between the application system and database. It makes the application and database communicate security and efficiently. /9, 1381/
4. **Automatic primary key generation:** Hibernate provides a generator to generate entity Identifiers. There are native, identity, sequence and increment etc./9, 1381/

Besides the above features, Hibernate has many other powerful functions such as J2EE perfect integration, HDLCA and so on.

3 WÄRTSILÄ LOGISTICS CONFIGURATOR

The purpose of this WSPT (Wärtsilä Project Purchasing and Logistics Configurator) is to provide Project Purchasing and Logistics Department in Wärtsilä more convenient and efficient ways to configure a logistics project. The main requirements in this system can be divided as follows:

- **Configure New Logistics Project:** The user gives the necessary data such as origin port, destination port, equipment needs to be transfer and planned ship date: The system can automatically help the user to configure a logistics project
- **Cost Estimation:** The system can estimate the cost information for logistics project
- **Planned Shipment Time:** Automatically calculate the pre-time schedule
- **Weight Estimation:** The weight should be established by the system when the project is established
- **Volume Estimation:** The Equipment Volume should be established also and indicate which kind of freight solution services should be used in the project
- **Loading and Unloading Plan:** The system can generate the loading plan and unloading plan
- **Route Schedule:** Quite a lot of routes can be used in a project, the user can check the route schedule by using this WSPT application
- **Risk Management:** WSPT is required to detect the logistics risks before a project is established
- **Check World Ports Information:** The user can easily to search and view the port details information from WSPT
- **Order Shipping Operators:** WSPT allows the user to make a pre-order with the operator
- **Search Operator:** The operator information can be easy get from WSPT
- **Tracking Route:** Features that tracking the transportation route on the map

- **Freight Rate Request:** The FRQ is the main idea of the application that WSPT will support to display the FRQ list by different operators that allows the user to find the best candidate to operate target logistics project
- **Project Schedule:** WSPT will generate all project information for the user which supports to print or save in local pc features
- **DB CRUD:** This is the administration feature with database management. It provides functions such as add new port, new operator, new equipment, new route schedule and so on

In order to meet Wärtsilä user requirements, the application should be available in multiple browsers, such as Firefox, IE 8+, Opera, Chrome and so on. It was decided that the application deploys in the JBoss Server when it is released as a commercial production version.

3.1 Main Features Deployment

The main function of the WSPT is to help Wärtsilä User to configure a logistics project in the easiest and fastest way with the lowest risk. The application can generate the project purchasing and transportation plan including loading plan, unloading plan, cost estimation and so on that allows the user to save it into local PC or Wärtsilä DB.

Luckily with Vaadin and Tomcat strong scalable features to enable the developer to provide the user an intuitionistic and visual using experience, for example, there is a map view feature has been embedded to WSPT.

The first version was deployed in the Tomcat Server now, but in the future, the commercial product version will be deployed into the JBoss Server. The JBoss Server was developed based on Tomcat Server, so this will not be a problem to test on Tomcat Server.

The following table displays a list of features that need to be implemented in this project and their priority. The number in priority column means the essentiality of features in this project release.

Table 2. Features and their priorities /10/

(1: urgent, 2: must this release, 3: should this release, 4: Could this release, 5: not this release, 6:

N/A)

<i>Feature Num.</i>	<i>Feature Name</i>	<i>Priority</i>
1	User Login and Register	2
2	Logistics Project Configuration	1
3	Cost Estimation	2
4	Planned Shipment Time	2
5	Weight Estimation	2
6	Volume Estimation	2
7	Loading and Unloading Plan	3
8	Route Schedule	4
9	Risk Management	4
10	Tracking Route	5
11	Freight Rate Request	3
12	Project Schedule	3
13	DB CRUD	4
14	Map View	3
15	Import Port Information	5
16	Generate Schedule File	4
17	Search Port Information	2
18	Search operator Information	2
19	Search Equipment Information	2
20	Configuration Result Filter	4

3.2 WSPT Use Case Diagram

There are two kinds of users, a normal user, and administrator who can access the WSPT application for configuration of the Logistics Project. After successful login, the user can insert the necessary data and get configuration data from the system that allows the user to search port information, generate schedule file, check route schedule, filter configure result etc.

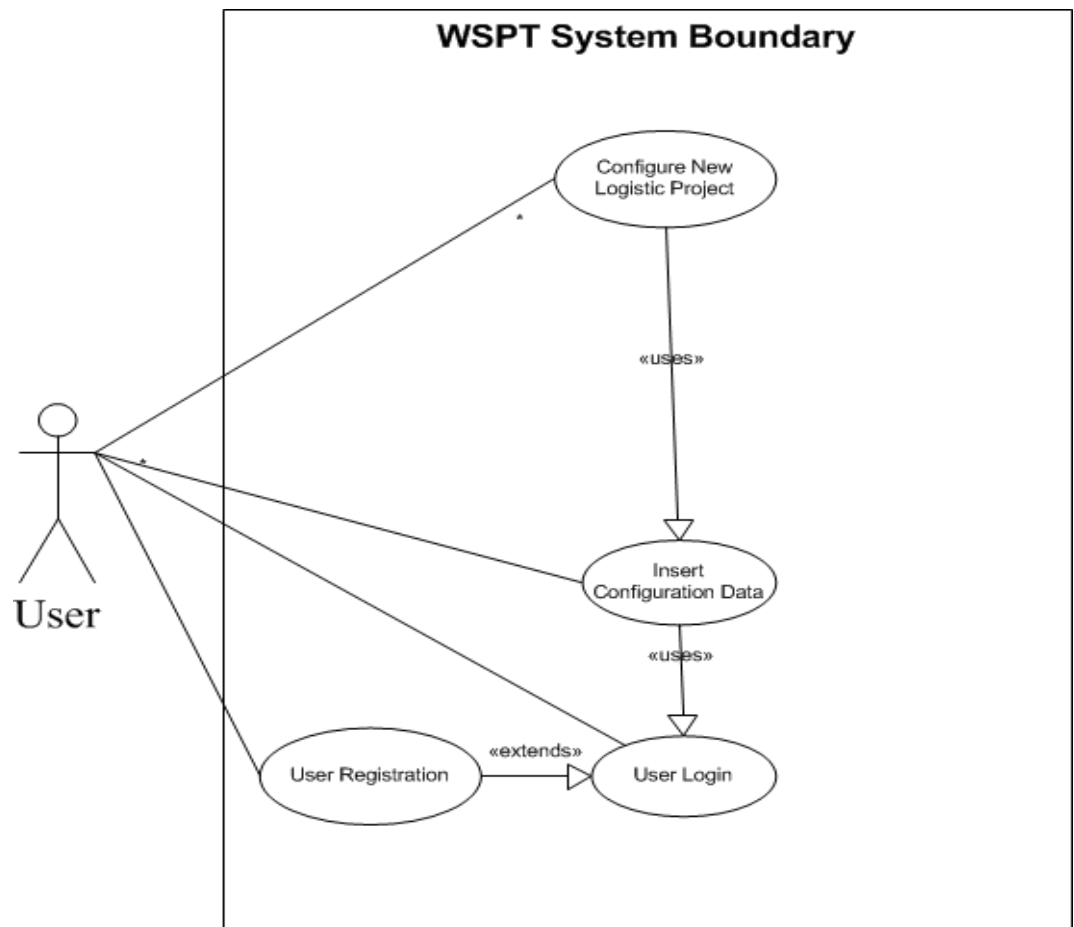


Figure 4. WSPT basic structure use case diagram

Figure 4 is the basic use case structure of WSPT application and all functions are user-based landing permission. The user enters or changes about the logistics configuration data and sends it to the server. The server side will auto configure a pre logistics project with different operators and different routes based on the user input information. The next three figures will describe the main three features in this application.

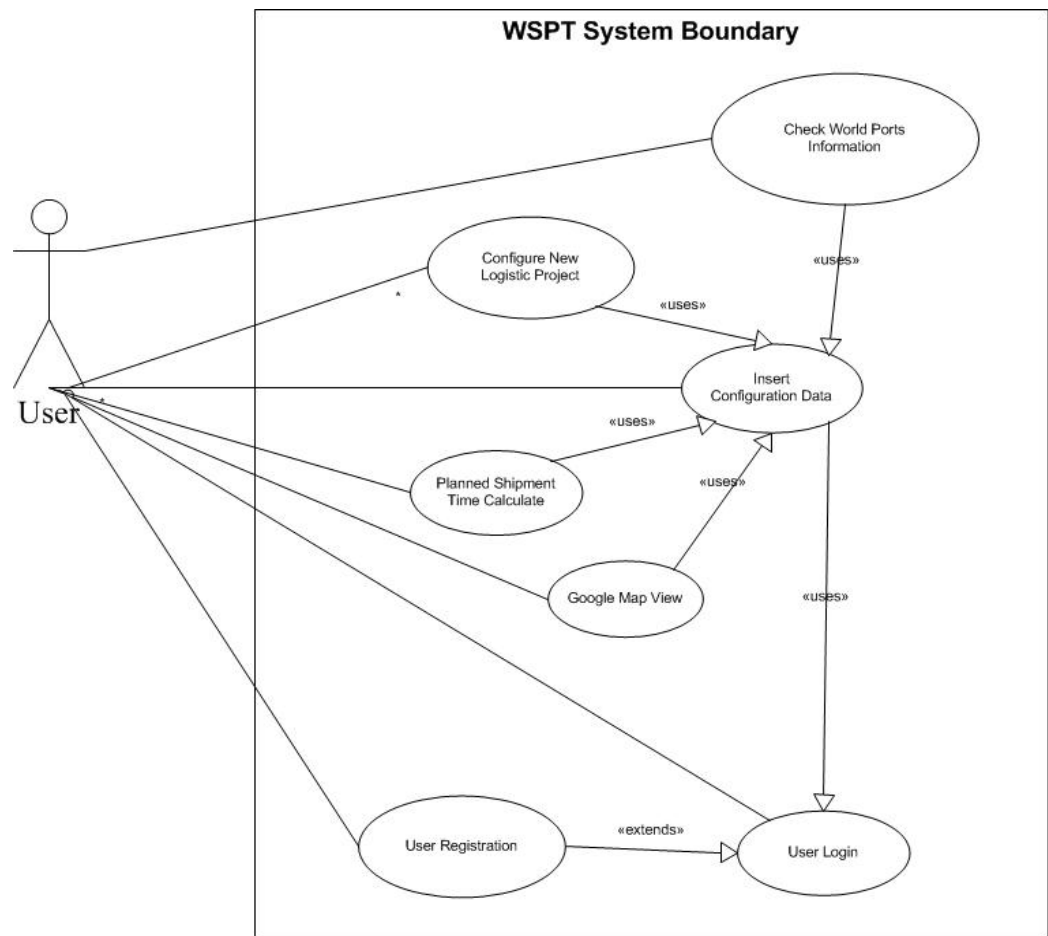


Figure 5. Insert configuration data use case diagram

The user has to insert configuration data first before he can pre-configure a new logistics project. In this feature, there are many sub-features when the user enters critical configuration information. The user can view the route map information with a Google map view and check world ports information with either the configuration panel or map view. The application can help the user to calculate planned shipment date and detect the risks in logistics configuration process.

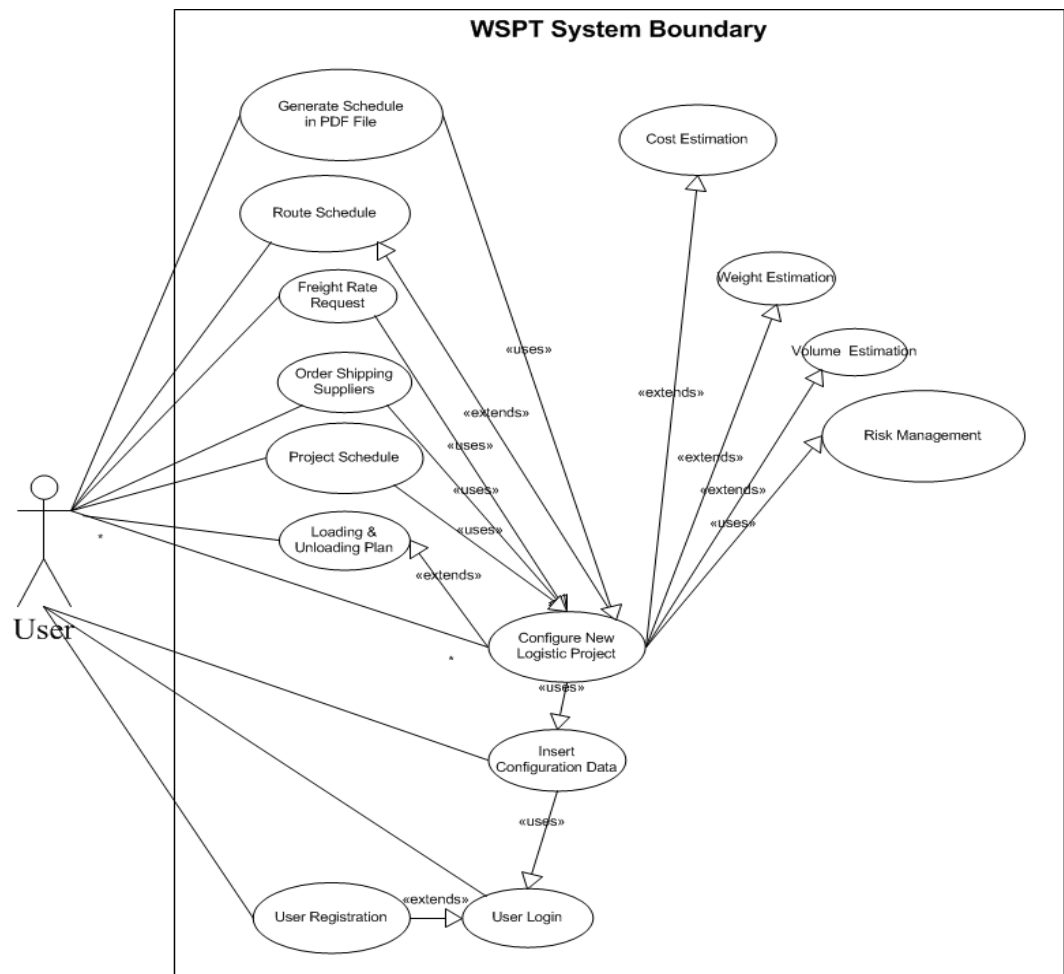


Figure 6. Configure new logistics project use case diagram

Configure new logistics project is the core function in WSPT application. When the application generates a new pre-configure logistics project plan with the user requirements, it contains cost estimation, weight estimation, volume estimation and risk management. In addition, the application also provides the features:

- Generate a PDF file, which contains logistics project information.
- Route schedule, which helps the user to check the transportation information.

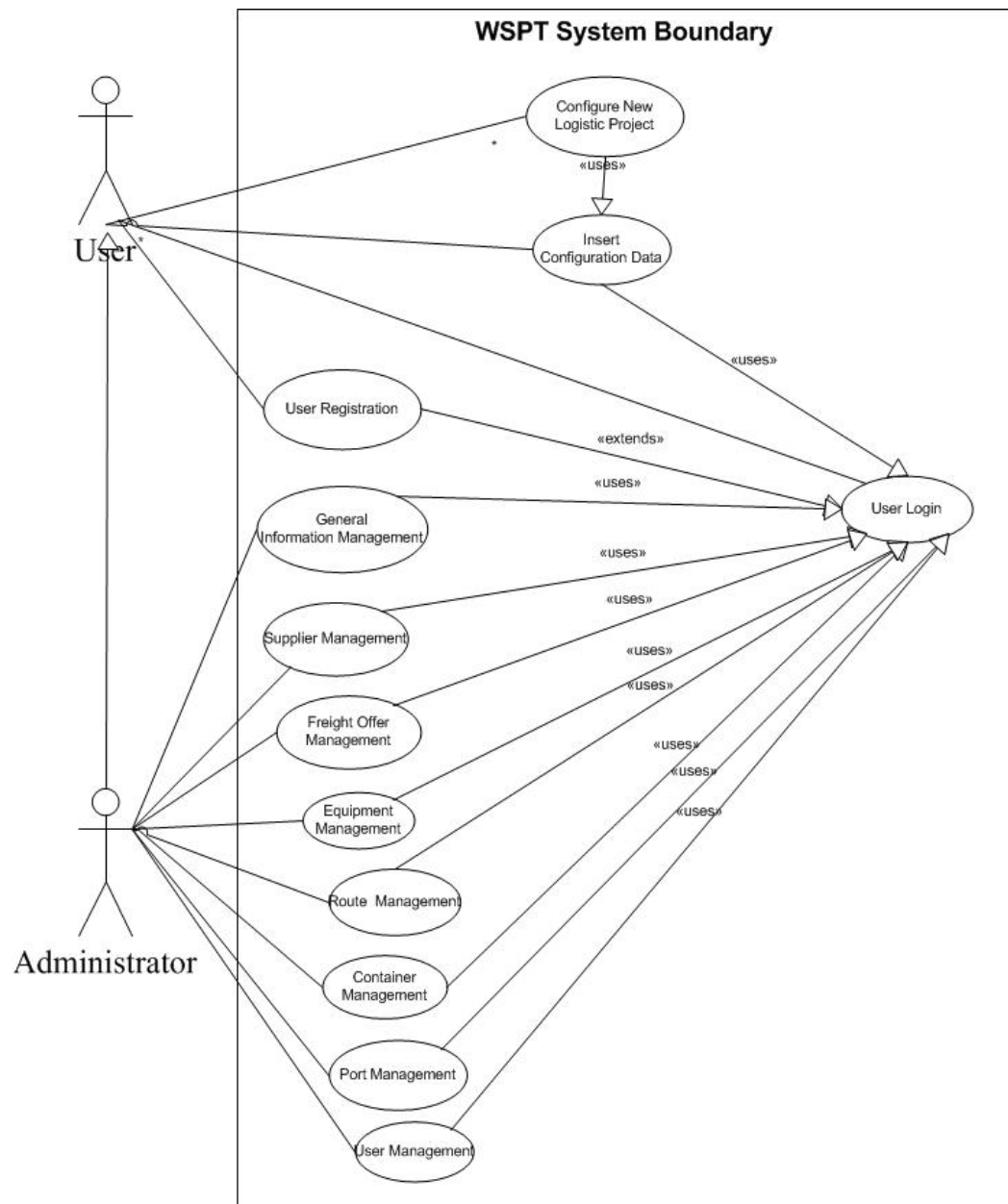


Figure 7. Administrator use case diagram

Figure 7 is a use case diagram to illustrate the relations between users and main features. In addition, the administrator not only has the normal user rights but can manage the important entities with admin rights. However, the system can help the user in a smart and easier way to configure an intelligent logistics project with cost, weight, volume, freight offer, time estimation etc. It helps the user to save time to consider a lot of valuable but superfluous factors when configuring such a

complicated project. The next section will describe the project structure, which was developed in this application.

3.3 WSPT MVC Architecture

The software architecture of WSPT is MVC, which stands for Model-View-Control. The advantage of using the MVC pattern is because it helps the developer to separate the business logical data model and view that can make the development more brief and clear. In order to develop in MVC pattern we need a development structure design for WSPT Project, such as shown below.

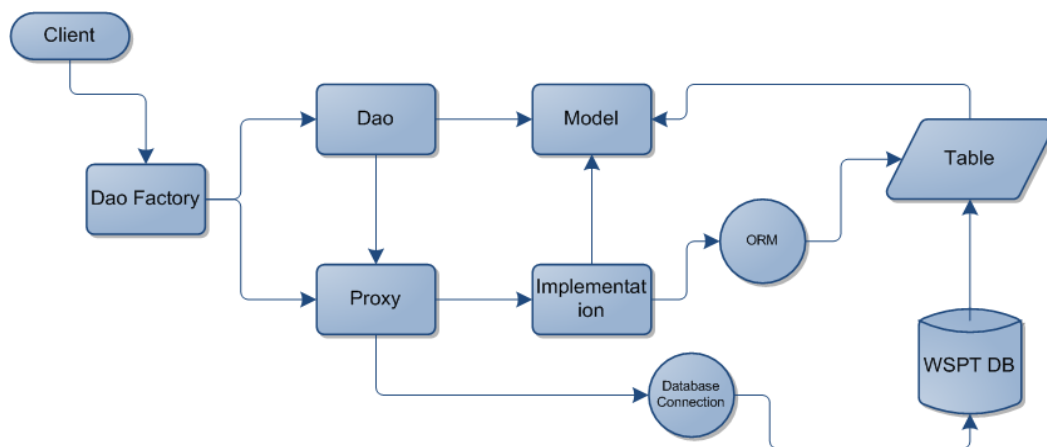


Figure 8. MVC development structure design

Figure 8 describes the MVC development structure, there is a factory object, which can access all DAO, layer functions and inject database connection to the DAO implementation layer through a proxy object. The DAO Interface object defines the data method in the WSPT application and then implements the DAO implementation object using the Hibernate ORM mechanism to retrieve data from the WSPT database.

3.3.1 WSPT Model Design

The model structure of WSPT has a lot of model objects that we need to consider it in a precise way. In order to meet Wärtsilä customer requirements, the model was designed as in Figure 10.

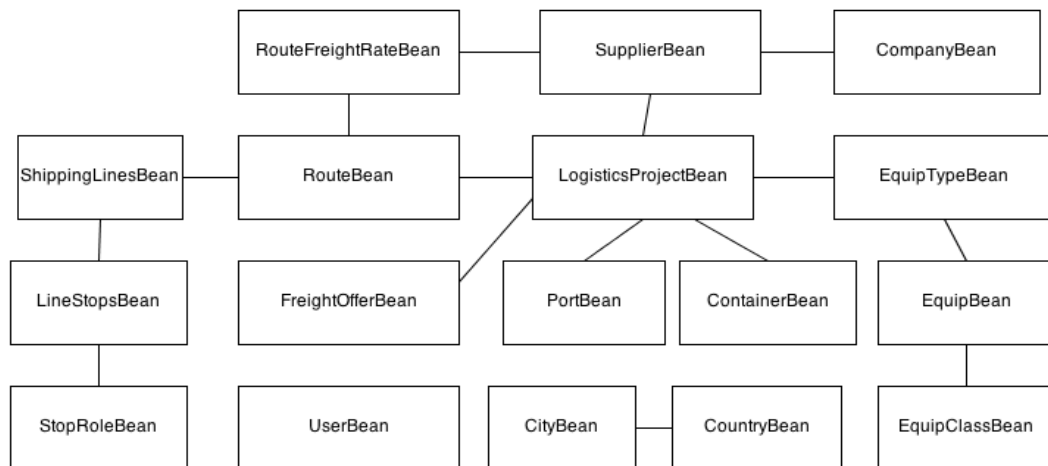


Figure 9. WSPT model design

In Figure 9, it is obvious that all models are direct or indirect to relate to LogisticProjectBean, which is the most important data object in WSPT used to configure a logistics project. UserBean is used for authorization for WSPT to separate a normal user and administrator user, but every new creation model should record the creator as well, so each model has an attribute named creator. In addition, WSPT should create time, update time that memorizes the history, and last changed date time. In LogisticsProjectBean, there are six data objects though-out whole data objects, which are ContainerBean, PortBean, EquipTypeBean, RouteBean and OperatorBean. ContainerBean decides what container should be used in the logistics project. PortBean contains essential port information that enables the application to manage the risk in transportation between two ports.

3.3.2 WSPT Model Functionality

The model contains logical components for storing and retrieving data for the WSPT application. Every model has its own DAO application program interface to connect to the WSPT server, which provides database communication for the WSPT. DAO can get the model DAO from the DAO Factory class. For example, if the user wants to reserve port information, he needs to get the port bean first. The structure to retrieve the information from the database is designed as shown in Figure 10.

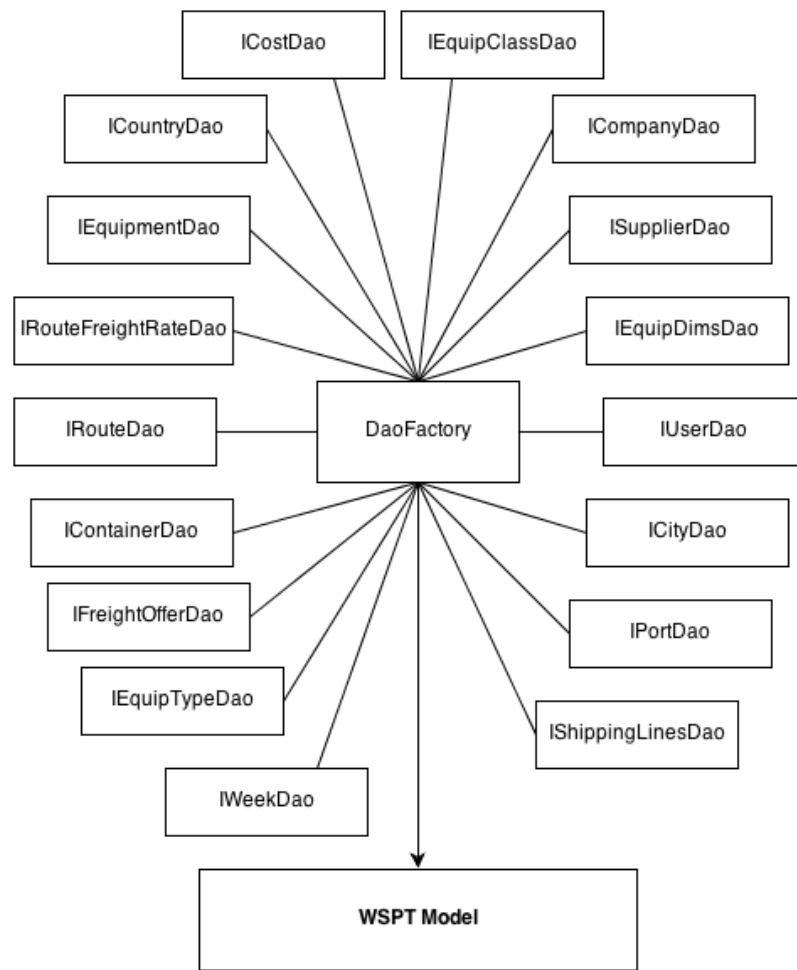


Figure 10. WSPT model functionality diagram

This is the model and model function logical implemented in the WSPT application.

3.4 Main Functions Description with Sequence Diagram

This section describes the main functions in WSPT, which has been listed in the use-case diagram before. The function logical has been drawn in Sequence Diagram by Visio 2007, and the action is described under the guild line table.

3.4.1 User Login and Register function

The user needs to login first before using the WSPT application. The user can also request a user account to access to the WSPT application. The WSPT Login actions are:

1. Enter username: The user writes the username into a text field reserved for the username.
2. Enter password: The user writes password into a text field reserved for the password.
3. Press login button: The user presses Login button to initiate the login process.
4. Login: The view passes the user credentials to the struts2 Login Action, which handles the login process with the server.
5. Response: A response is received from the WSPT Server if the login was successful or failed.
6. Show main view: The url will direct to the WSPT Main UI.
7. Display error message: If login failed, the user is notified with an error message.

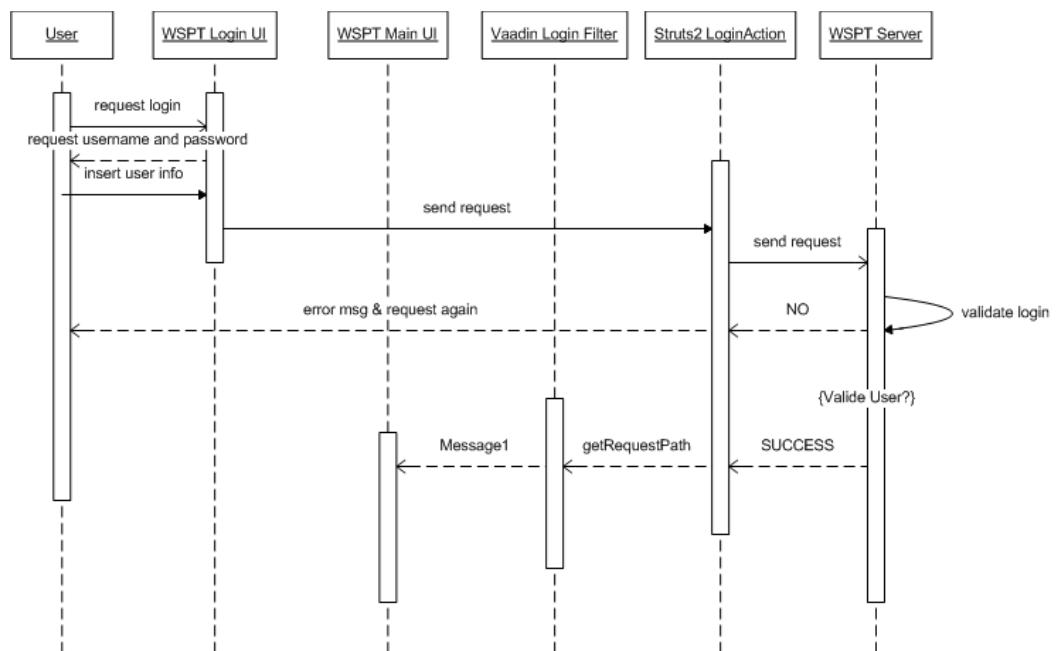


Figure 11. WSPT Login

The actions to request a WSPT user account are:

1. Request user account: When the user wants to use WSPT that he needs to request a user account.

2. Enter register information: The user writes necessary information to register a user account and send request to the server
3. Show successful message: If the register request is successful, the application will direct to the request successful page and the server will reserve the user register information to the database and waits the application administrator to verify it.
4. Display error message: If the register data is invalid, WSPT will show the error message to the user.

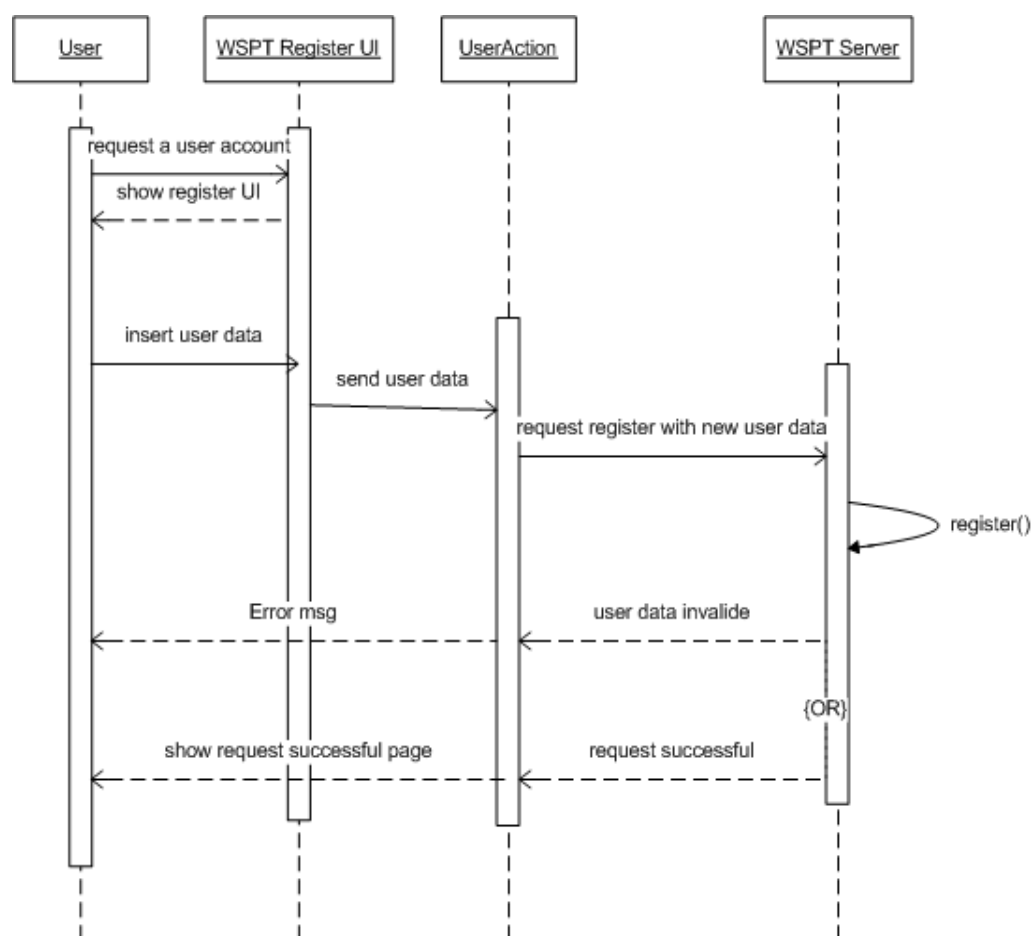


Figure 12. WSPT request a user account

3.4.2 Configure and Establish New Logistics Project

The most important function in WSPT is to configure a logistics project which can be handled by the WSPT system, the target of the WSPT is to let a green hand

to configure a low risk and high quality logistics project in an efficient and simple way. The actions to configure a new logistics project are:

1. Insert necessary configure data: When the user login is successful and the user wants to configure a logistics project, the WSPT requires the user to input the necessary configure data, such as origin place, destination place, commodity, amount and planned shipment date.
2. Press the search button: After the user presses the search button with valid input data, the system will calculate the project risk automatically at the same time.
3. Valid input data: WSPT will validate the user input data if the user type invalid data or formats to the configuration form and shows a red flag on the error place with the warning message in the warning panel.
4. Risk management: If the configuration has a risk that can be valid by WSPT, WSPT will return to the main view and show risk information in the warning message panel.
5. Display configuration result view: If the data has been valid and no risk inside, the configuration will be done at the server side and display the configuration result UI to the user.

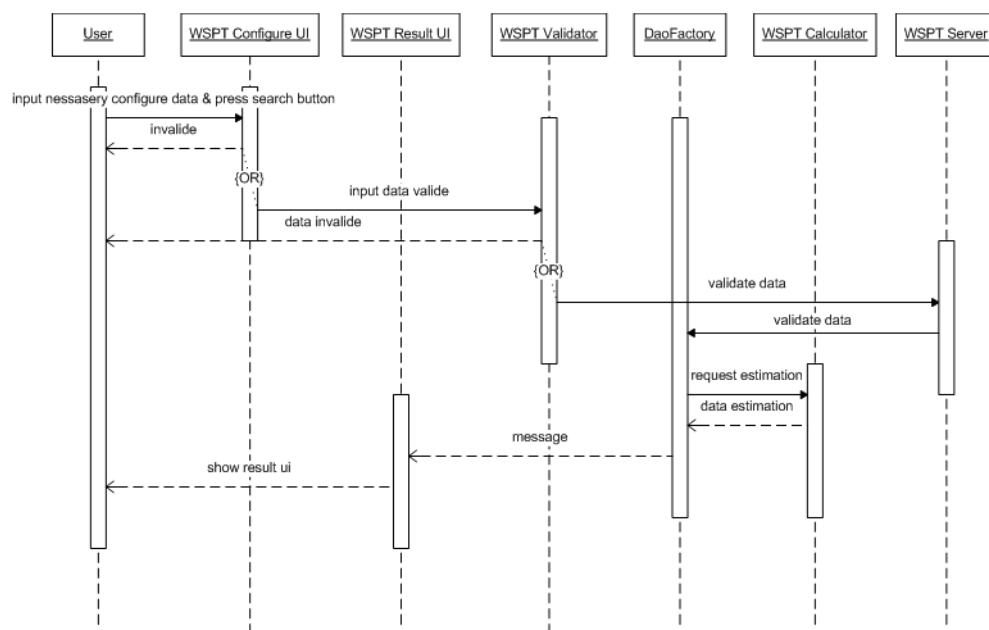


Figure 13. WSPT configure a new logistics project

After user enter the WSPT result UI that the user can create a new logistics project with his requirements and the result will rank from the lowest price to highest price which provide by different line-operator. The establish a new logistics project actions are:

1. Choose an operator: The user can view options by different operators and detailed information for this project.
2. Press setting button: After the user finds a suitable operator who meets all requirements to operate this configure project, he can press the setting button to create a new project and the system will store the project information to the database.
3. Display successful view: WSPT can save the new configuration project and display successful view if the user's operation is successful and show the project schedule in the PDF format.

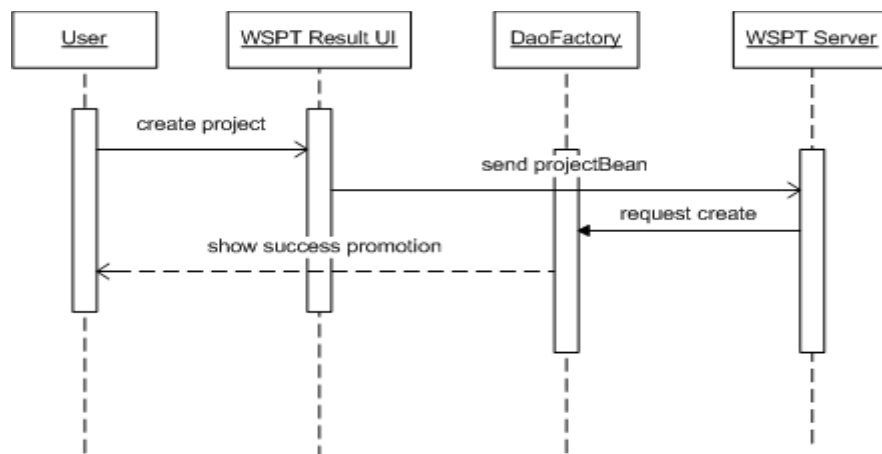


Figure 14. Establish a new logistics project

3.4.3 Cost Estimation

The WSPT application should have a cost estimation based on weight, freight rate, surcharge rate, lifting cost rate and other cost information. The aim is to configure a low risk with a high performance price ratio for the customer, the WSPT required to estimate a cost for every logistics project with a reference on incoterm standard. The cost estimation actions are:

1. Insert necessary configure data: The same as configuring a new logistics project, the user needs to insert necessary project data first.
2. Press the search button: After the system has validated all input data and sent data to the WSPT server, the server will invoke DaoFactory to use a CostDaoProxy object to achieve the cost result.
3. Display result UI with cost information: WSPT will direct to the result UI and calculate the project cost which is provided by different operators and the results will be ranked by price from low to high.
4. Add a new RFQ: The result UI will provide a RFQ function that invites operators into a bidding process on this logistics project.

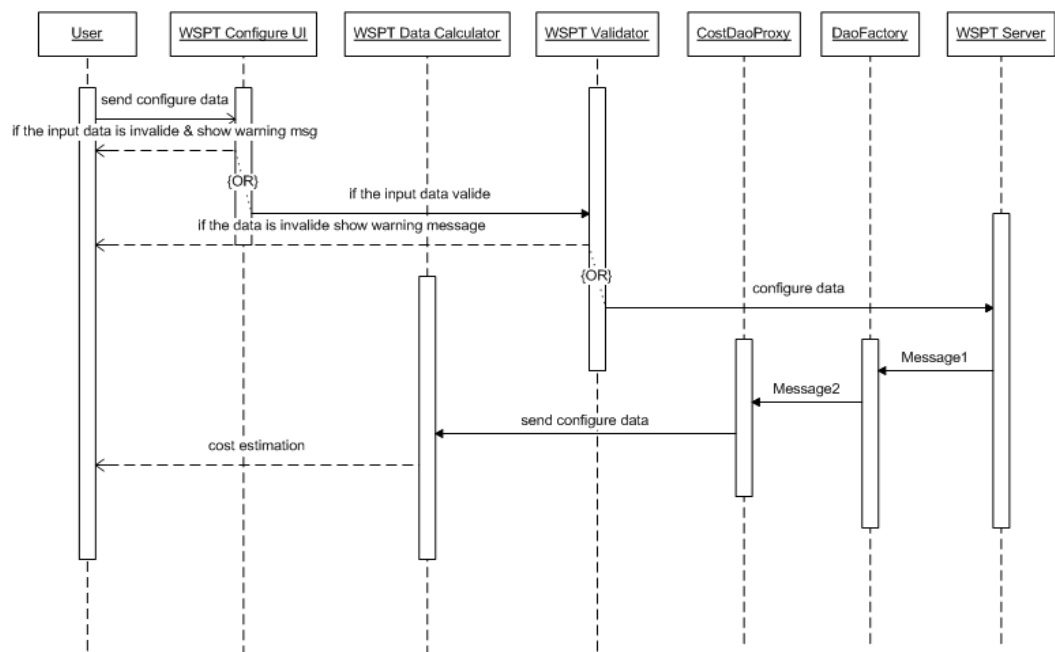


Figure 15. WSPT cost estimation

The cost estimation is a system action when the user configures a new logistic project and the volume, weight and planned shipment time have the same idea as this part. The user can get the whole project information from the result panel after configuring a new project successfully.

3.4.4 Map View

The map view is an advanced function for the WSPT system. In detail, the user can find ports information and routes information from the WSPT map view and the user can easily check and compare the available routes, from the place of origin to the destination. The user only needs to insert the origin and destination, the system can recognize the input data and provide the certain information on the right panel, then can show them immediately on the map. Below is a system action to handle a map view when user inputs the transport start to end place information. The WSPT map view actions are:

1. Insert origin and destination place: Fill the origin and destination text field in the form.
2. Send place information to Server: The WSPT has listener to listen the user input actions. The place information will be sent to the WSPT server after the text field is off focus.
3. Choose MapView Visible: The user can choose the map view to be visible or not by a checkbox and the default action is invisible.
4. Display transportation route information: The WSPT server will call WSPT Map object, and the Map object will send request to DaoFactory to get the shipline information and draw them on the MapView.

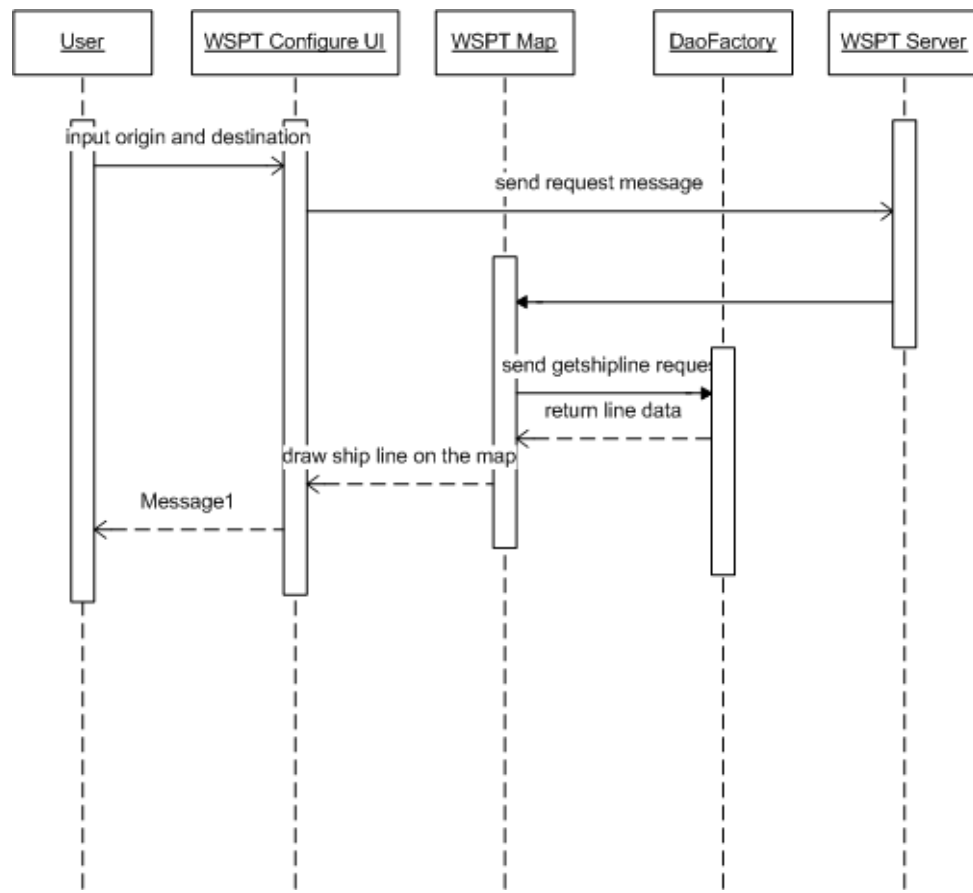


Figure 16. WSPT map view

3.4.5 Result Filtering

Result Filter is an additional function for the WSPT project. This idea comes from eBay online shop. eBay provides the filter function that meet the user requirements by selecting the suitable constraints. The same function is available in WPST, the user can filter the results by route, operator and price. The WSPT result filter actions are:

1. Insert origin and destination data: Fill the origin and destination text field in the form.
2. Choose operators: The operators, which the user wants to filter out in results.
3. Input price interval: Insert price interval constraints.

4. Click confirm button: After inserting above limiting conditions, the user can click confirm button to send filter request to the WSPT server.
5. Server response and display filter results: The server will get the filter results after receiving the user's limitation data through DaoFactory Class and send them to the Result Filter. The result filter will catch the results data and display them on the WSPT Result

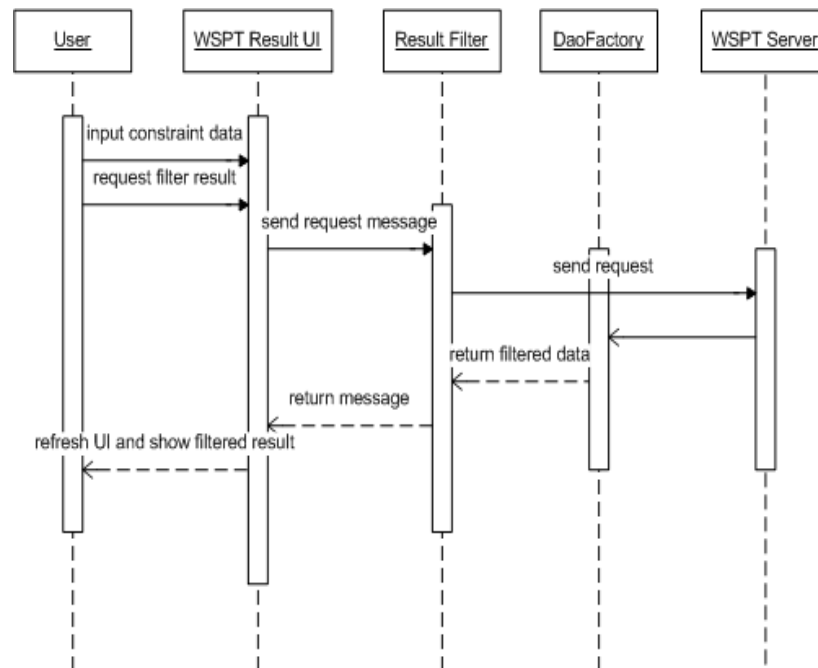


Figure 17. WSPT result filtering

The WSPT result filter is a quite useful feature that it can reduce the user's searching time, improve the efficiency of the configuration and enable the user to find the most suitable operator.

3.4.6 Schedule in PDF

The WSPT needs to be able to save the project data to the local computer or print out as paper documents when the user configures a new logistics project. In order to implement this function, WSPT plugs in iReport to the project that allows the user generate a PDF file with all project information. The user needs to press the "Generate PDF" button, and then the system will generate a file in PDF format. The WSPT PDF generator actions are:

1. Press generate PDF button: Press Generate PDF button to generate a PDF file for the configuration which chosen by the user from results
2. Send request to WSPT Resource Utility: The request will be sent to the WSPT resource utility.
3. Show PDF Window: The WSPT will show a PDF window for the user that allows the user to save or open the PDF file.

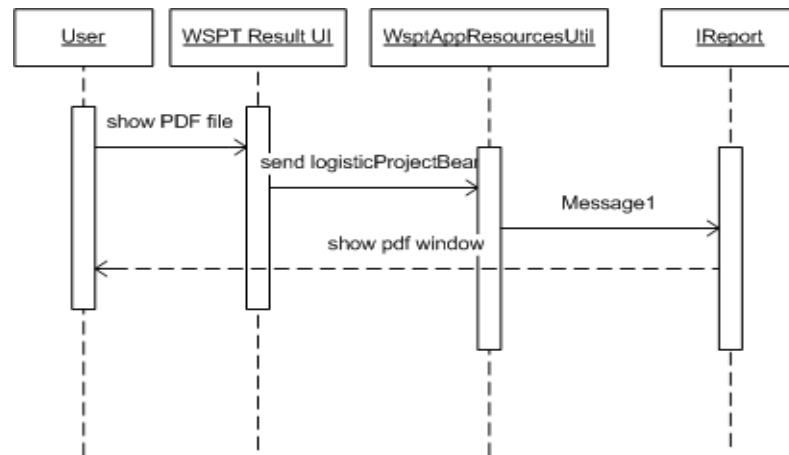


Figure 18. WSPT PDF generator

3.4.7 Admin Function

The WSPT admin function is used for managing the WSPT data information. A user who has admin right can access the WSPT admin UI and be able to storage, obtain, delete and update data. The main actions are:

1. Choose an action from the list tree: Choose a model function from the WSPT admin UI navigation tree.
2. Send CRUD request: The request will be sent to the WSPT Server with necessary data.
3. Database operation: The server will invoke DaoFactory to implement the admin request and the data will be changed in the database as well.
4. Display a successful view: The WSPT will show the successful promotion if the server successfully handles the operation.
5. Refresh view: The data view will be refreshed after the operation is done and the data will be updated at the same time.

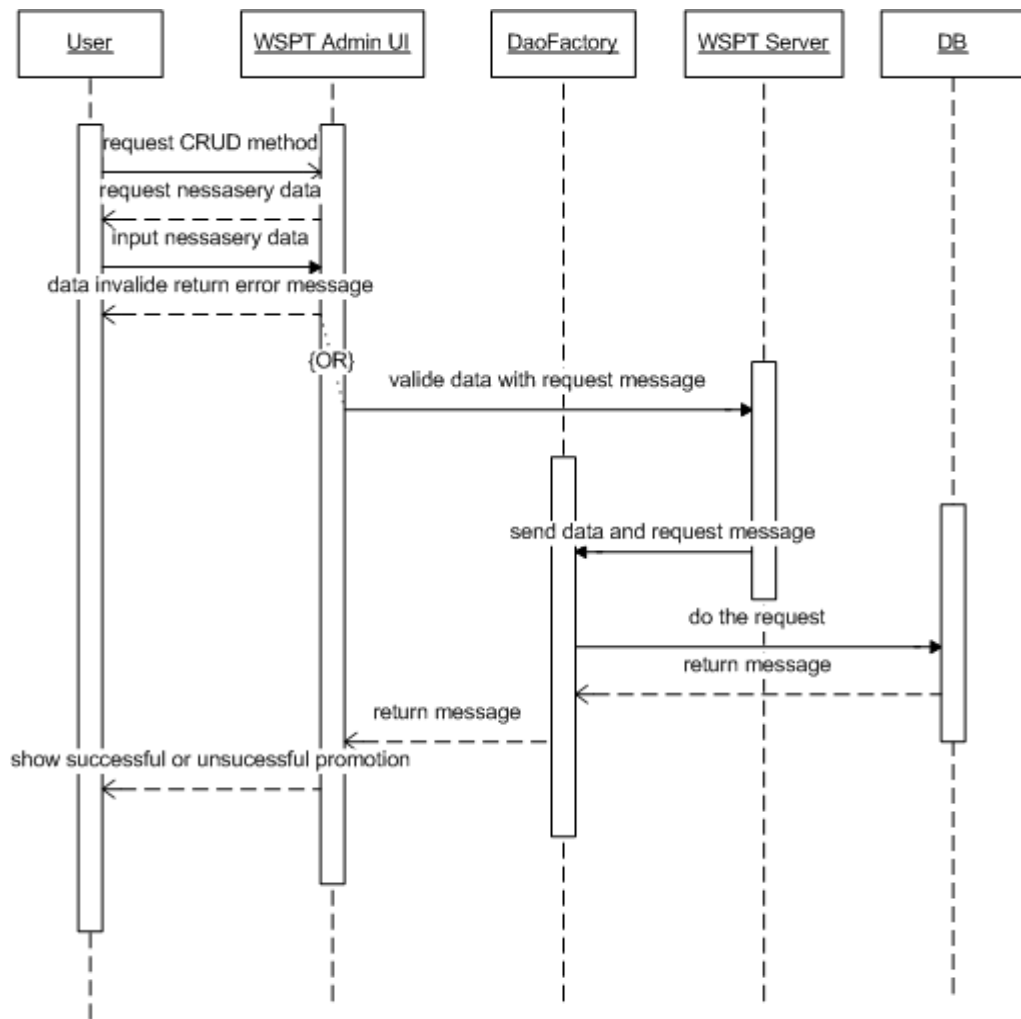


Figure 19. Admin Functions

3.5 WSPT Database Design

WSPT is a solution for logistics manager to search global transport operations. Therefore, it needs a large database to store global information. In order to meet the customers' requirements, the database was designed as shown below after discussion, the main entity and their relations will be described in detail.

3.5.1 WSPT Entities

The WSPT entities define a representation of data in the application of physical and logical views of the data. This is the first version of WSPT entities design and

the following entity tables describe the contents as relation of entities, which have been implemented in the WSPT application.

The equipment is divided into different categories by equipment class entity. This entity will be a foreign key of equipment entity. They have relation one to many for each other. One equipment class entity can have many equipment entities, but one equipment entity can only have one equipment class.

Table 3. Equipment Class Entity

<i>Key</i>	<i>Attribute</i>	<i>Comment</i>
PK	EQUIP_CLASS_ID	Equipment class ID.
	class_name	The name of equipment class.
	Timestamp	Create date time.
	Updatetime	Update date time.
FK (USERS)	CREATOR	The creator.

The equipment entity contains information about the Wärtsilä equipment, such as equipment name and equipment class. The next entity is equipment type entity. One piece of equipment might have hundreds of types, so the relation between equipment entity and equipment type entity is one to many.

Table 4. Equipment Entity

<i>Key</i>	<i>Attribute</i>	<i>Comment</i>
PK	EQUIP_ID	Equipment ID.
	equip_name	The name of equipment.
	timestamp	Create date time.
	updatetime	Update date time.
FK (USERS)	CREATOR	The creator.
FK (EQUIPMENT CLASS)	EQUIP_CLASS	The equipment class ID of equipment belongs to.

Wärtsilä equipment has different types and this entity contains the detailed information of each type, such as its dimensions, weight and freight rate offer type (BB/FCL). It has two foreign keys, which are equipment id and freight rate offer id. The equipment foreign key defines which the equipment type belongs to which equipment, and freight rate offer defines the default freight rate offer for this equipment.

Table 5. Equipment Type Entity

<i>Key</i>	<i>Attribute</i>	<i>Comment</i>
PK	EQUIPTYPE_ID	Equipment ID.
	type_name	The name of equipment type.
	ed_height	The height of equipment.
	Ed_length	The length of equipment.
	Ed_width	The width of equipment.
	timestamp	Create date time.
	updateTime	Update date time.
	weight	The weight of equipment.
FK (USERS)	CREATOR	The creator.
FK (EQUIPMENT)	EQUIP	The ID of equipment, which contain this type.
FK (FREIGHT RATE OFFER)	FREIGHTOFFER	The freight rate offer ID provide by this equipment.

The container entity contains the international standards container information, which will be used in the WSPT logistics project configurator. The container information is determined by door height, door width, inner height, inner length, inner width, max gross weight, max payload, standards, tare weight. The max gross weight is the total weight of a shipping container including the contents, typically expressed in kilograms. The max payload means the max payload for the container. Based on equipment dimensions, weight, the WSPT application container will help the user to select a suitable container automatically.

Table 6. Container Entity

<i>Key</i>	<i>Attribute</i>	<i>Comment</i>
PK	CONTAINER_ID	Container ID.
	door_height	The container door height.
	door_width	The container door width.
	inner_height	The container inner height.
	inner_length	The container inner length.
	inner_width	The container inner width.
	max_grossWeight	The maximum gross weight of container.
	Max_payload	The maximum payload
	standards	Container standards.
	tareweight	The tare weight of container.
	timestamp	Create date time.
	updateTime	Update date time.
FK (USERS)	CREATOR	The creator.

The company entity saves the operator company contact information. It is a foreign key of Operator Entity. The contact information includes company name,

company web site, email, fax number and phone number. The company information helps the user to get more information about the operators.

Table 7. Company Entity

<i>Key</i>	<i>Attribute</i>	<i>Comment</i>
PK	COMPANY_ID	Company ID.
	address	The address of company.
	company_name	The name of company.
	company_site	Company website address.
	email	Company email.
	fax	Fax number.
	phone_number	Company phone number.
	timestamp	Create date time.
	updatetime	Update date time.
FK (USERS)	CREATOR	The creator.

The operator entity contains the operator contact information, which is a very important entity to the logistics project. The operator is immediately responsible for the logistics transportation part. The entity contains the operator contact information, such as email, fax, phone number and Skype.

Table 8. Operator Entity

<i>Key</i>	<i>Attribute</i>	<i>Comment</i>
PK	OPERATOR_ID	Operator ID
	email	The email address of operator.
	fax	The fax number of operator
	name	The name of operator.
	phone_number	The phone number of operator.
	skype	The skype account of operator.
	timestamp	Create date time.
	updatetime	Update date time.
FK (COMPANY)	COMPANY	The ID of company.
FK (COUNTRY)	COUNTRYCODE	The country code.
FK (USERS)	CREATOR	The creator.

The port entity is one of the most important entities in the application and every part in the project is related to this entity directly or indirectly. It contains all critical information about the port, such as harbor information, location information, contact information, restrictions and port dimensions. It is a foreign key for route entity as original port and destination port. The pier depth, crane capacity and handling capacity, which are the risk management factors in the application, are also included.

Table 9. Port Entity

<i>Key</i>	<i>Attribute</i>	<i>Comment</i>
PK	PORT_ID	Port ID.
	currency	The currency of port location.
	customes_operations	Port operation description.
	dry_dock	The dry dock of port.
	harbour_type	The harbour type of port.
	harbour_size	Harbour size.
	latitude	Port latitude value.
	longitude	Port longitude value.
	low_cargoPier_depth	Lower bound of cargo pier depth.
	up_cargoPier_depth	Upper bound of cargo pier depth.
	low_crane_capacity	Lower bound of crane capacity.
	up_crane_capacity	Upper bound of crane capacity.
	low_quayside_handling_capacity	Lower bound of quay side handling capacity.
	up_quayside_handling_capacity	Upper bound of quay side handling capacity.
	max_size	Upper bound of transfer cargo.
	port_authority	Port authority.
	port_cert	Port certificate.
	port_e_mail	Port email.
	port_fax	Port fax.
	port_name	Port name.
	port_phone	Port phone number.
	port_site	Port website.
	prefer_shiplines	Port shiplines.
	railway_size	Port railway size information.
	regular_shiplines	Regular shiplines on this port.
	repaires	Repair years
	shelter	Port shelter
	surveyor_operating	The name of surveyor of port
	timeZone	Port time zone.
	War_contactInfo	Nearest Wärtsilä contact information.
	timestamp	Create date time.
	updateTime	Update date time.
FK (CITY)	CITY	The ID of city.
FK (COUNTRY)	COUNTRY	The country code.
FK (USERS)	CREATOR	The creator ID.

The freight rate entity can provide calculation parameters for the loading cost, surcharge rates, storage cost and freight rate to calculate the total cost in the application. It has a foreign key to the operator that means that each operator can have a different freight rate for the logistics project.

Table 10. Freight Rate Entity

<i>Key</i>	<i>Attribute</i>	<i>Comment</i>
PK	FREIGHT_RATE_ID	Freight rate ID.
	lifting_fee	Lifting fee
	other_cost	The sum of other costs.
	other_cost_dsc	The other cost description.
	rate_baf	The bunker adjustment factor rate surcharge.
	rate_caf	The currency adjustment factor rate surcharge.
	rate_war	War risk surcharge.
	freight_rate	Freight rate.
	whs_fee	Warehouse fee.
	timestamp	Create data time.
	updateTime	Update date time.
FK (COMPANY)	ROUTE	The ID of Route.
FK (OPERATOR)	OPERATOR	The ID of Operator
FK (COUNTRY)	COUNTRY	The country code.
FK (USERS)	CREATOR	The creator.

The freight offers entity contains standard logistics offer types information, which are FCL (Full Container Loader) and BB (Break Bulk). The logistics loaded type is determined by this entity and it will automatically help the user to find a suitable offer type by cargo dimensions, weight and so on.

Table 11. Freight Offer Entity

<i>Key</i>	<i>Attribute</i>	<i>Comment</i>
PK	FREIGHT_OFFER_ID	Freight offer ID.
	name	Freight offer name.
	short_name	Short name.
	timestamp	Create data time.
	updateTime	Update date time.
FK (USERS)	CREATOR	The creator.

The country entity table has country information. It is responsible for providing country information for the WSPT system. The country code is named based on ISO3166.

Table 12. Country Entity

<i>Key</i>	<i>Attribute</i>	<i>Comment</i>
PK	CODE	Country code from ISO3166.
	name	Country name.
	timestamp	Create data time.
	update time	Update date time.
FK (USERS)	CREATOR	The creator.

The city entity is related to the country entity and provides city information for the system. It has the city name and foreign key to the country code.

Table 13. City Entity

<i>Key</i>	<i>Attribute</i>	<i>Comment</i>
PK	CITY_ID	City ID.
	name	City name.
	timestamp	Create data time.
	update time	Update date time.
FK (COUNTRY)	COUNTRYCODE	The country code.
FK (USERS)	CREATOR	The creator.

The route entity contains original port and destination port information by port id. It is a key entity when the user configures a new logistics project. The route description, distance, name and proposal information will be saved in this entity and used for route schedule and ship lines entity.

Table 14. Route Entity

<i>Key</i>	<i>Attribute</i>	<i>Comment</i>
PK	ROUTE_ID	Route ID.
	route_description	Route description.
	route_distance	Route distance.
	route_name	Route name.
	route_proposal	Route proposal.
	timestamp	Create date time.
	update time	Update date time.
FK (PORT)	DEST_PORT	Destination port.
FK (PORT)	ORIGIN_PORT	Origin port.
FK (USERS)	CREATOR	The creator.

The Ship lines entity is used to establish the route schedule table for the project. It has ship line name, route information and ETA (estimated time of arrival). The foreign key of route means one route can have different schedule, the relation be-

tween the route and ship lines is one to many. The ETA will define the arrival date in this route schedule.

Table 15. Shiplines Entity

<i>Key</i>	<i>Attribute</i>	<i>Comment</i>
PK	SHIPLINE_ID	Shipping line ID
	line_name	Shipping line name
	timestamp	Create date time
	updatetime	Update date time
FK (USERS)	CREATOR	The creator
FK (ROUTE)	ROUTE	The route ID of shipping line
FK (WEEKDAYS)	ETA_WEEKDAY_ID	The estimated time of arrived in weekdays

The ship line stop entity contains the transport station information in a ship line and allows the system to check the logistics stop stations from the origin to the destination. Detailed information for the ship line entity was not available when it was being made. The ship line stop entity will define how many stop stations a ship line will be passing and the stop information, such as the stopping time and the transport days for the next station. The STOP_PORT is current port and the STOP_ROLE represent the role of the current station.

Table 16. Shipline Stop Entity

<i>Key</i>	<i>Attribute</i>	<i>Comment</i>
PK	LINESTOP_ID	Line stop ID.
	stop_weight	The weight of the line stop.
	stop_days	The days will stop in this port station.
	transportation_days	Haulage time from this port to the next.
	timestamp	Create date time.
	updatetime	Update date time.
FK (PORT)	STOP_PORT	The ID of stop port.
FK (STOP_ROLE)	STOP_ROLE	The ID of stop role.
FK (SHIPLINES)	SHIPPING_LINE	The ID of shipping line.
FK (USERS)	CREATOR	The creator.

The stop role entity is used to distinguish the role of the port in a shipline such as start port, stop port and destination port.

Table 17. Stop Role Entity

<i>Key</i>	<i>Attribute</i>	<i>Comment</i>
PK	STOP_ROLE_ID	Stop role ID.
	role_name	Stop role name.

The user entity contains application user information, and it is also responsible for the system authority. The password has been encrypted by BCrypt method for a security reason so that no one can read the password from the database. The role attribute defines the permission level of this user. Each entity has an attribute named creator to notify database manager who is the last to modify it.

Table 18. Users Entity

<i>Key</i>	<i>Attribute</i>	<i>Comment</i>
PK	USERNAME	User name.
	password	Password.
	address	Street Address.
	birthday	User birthday.
	email	User email.
	fax	Fax number.
	firstname	The firstname.
	lastname	The lastname.
	gender	Gender.
	gsm	User phone number.
	isconfirmed	Confirmed statues.
	job_title	Job title.
	phone	Home phone.
	postal	Post Code.
	role	User system role.
	timestamp	Create time.
	update time	Update time.
FK (COUNTRY)	COUNTRY	Nationality.
FK (USERS)	CREATOR	The creator.

The main task of logistics project entity is to maintain the project information and statues. This entity represents an intact logistics project, which has loading plan, un-loading plan, cost estimation, size estimation, weight estimation, planned shipment start and planned shipment end. In addition, each critical entity will be configured to this entity, such as creator, transportation of the equipment, loading port information, unloading port information, the information of container used in this logistics project and the transport schedule. In addition, the information of this entity can be rendered as a PDF file with this application.

Table 19. Logistics Project Entity

<i>Key</i>	<i>Attribute</i>	<i>Comment</i>
PK	LOGISTICS_PROJECT_ID	The ID of logistics project.
	configureType	Weight unit
	configureValue	Weight value
	draft	vessel draft
	loading_plan	Project loading plan
	unloading_plan	Project unloading plan
	planned_shipment_start	Planned shipment start time.
	planned_shipment_end	Planned shipment arrived time.
	project_name	Project name.
	total_cost	Total cost
	amount	The total number of transport commodities.
	timestamp	Create date time
	updatetime	Update date time
FK (USERS)	CREATOR	The creator
FK (EQUIPMENT TYPE)	EUQIP_TYPE	The equipment type
FK (PORT)	LOADING_PORT	Origin port.
FK (PORT)	UNLOADING_PORT	Destination port.
FK (CONTAINER)	CONTAINER	The container using in project.
FK (ROUTE FREIGHT RATE)	ROUTE_FREIGHT_RATE	Route freight rate for project.

3.6 Architectural Overview

The architectural diagram in Figure 20 describes the WSPT communication framework.

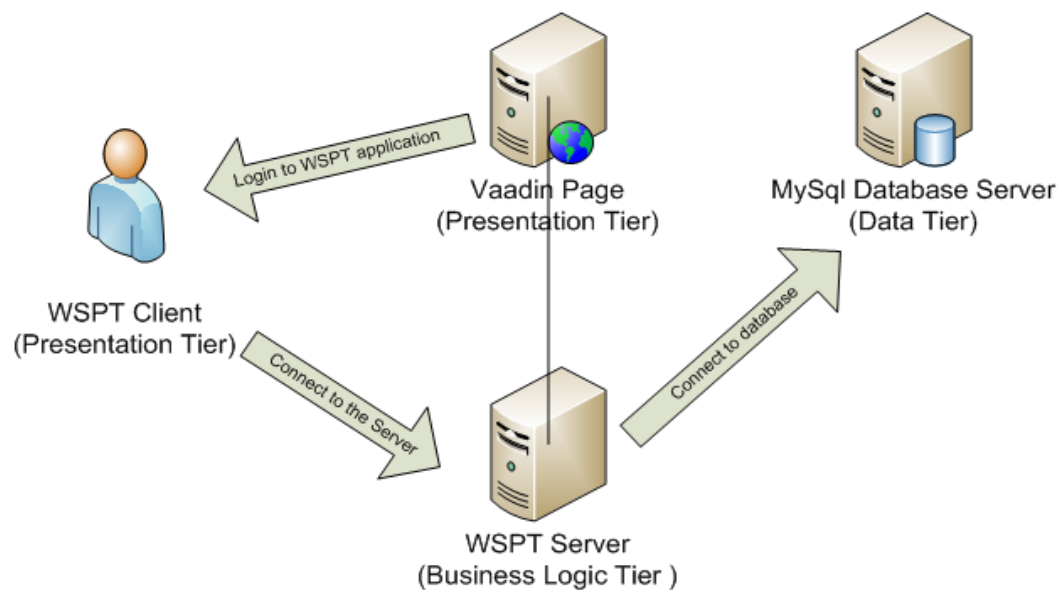


Figure 20. WSPT Architectural Diagram

The WSPT has a typical Rich Internet Application architecture, it has three layers in the application, and they are presentation layer (View), WSPT Server (Control), WSPT Database Server (Model). The user can control the complicated business logical and update data to the database through the presentation tier.

4 WSPT USER INTERFACE DESIGN

WSPT UI (user interface) design is one of the core tasks in project. The aim of UI design is to make web application intuitive and easy to use.

As a RIA application, the presentation tier is on multiple browsers such as Firefox, Internet explorer, Google chrome, Opera and so on. Fortunately, the Vaadin is adequate for all of them, but we still needed to test them on different browsers even on different operating systems.

Moreover, the WSPT user interface design needs to consider user experience. Skyscanner.fi was a good UI model for this project, the interaction is easy for the user to search and buy a suited flight ticket. The same as WSPT, it is also required to provide the easiest and fastest way for the user to configure a logistics project at a suited price with lower risk and good operator.

The main parts of UI design created with the Vaadin framework are described next.

4.1 WSPT Configurator View

The configurator view is used by the WSPT user for configuring a new logistics project. The user can insert necessary configuration data at the main search field and have a choice to enable pre-config information panel and WSPT map view. The pre-config information panel will be responded after the user inserts the sensitive value, for example the origin place information will be shown on the panel after the user fills in the “FROM” text field. The WSPT map can show the location and location information in an intuitive way on the map view. The user clicks “Search” button after fill all the data is in the form. The system will evaluate the data and send configure request to the WSPT Server. Figure 21 is configurator view designed by Vaadin Technology.

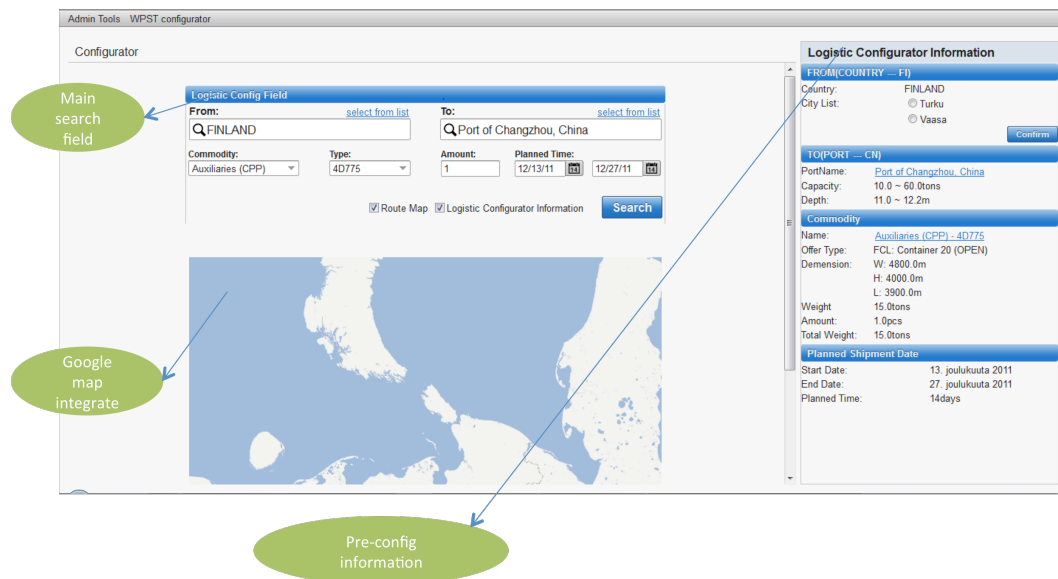


Figure 21. Configurator view

The WSPT will show a warning message on the warning message board when the user does some invalid actions. For example, if the user leaves an essential field empty and presses the search button, the title will be changed from black to red and invalid actions will be described on the warning board, as shown in Figure 22.

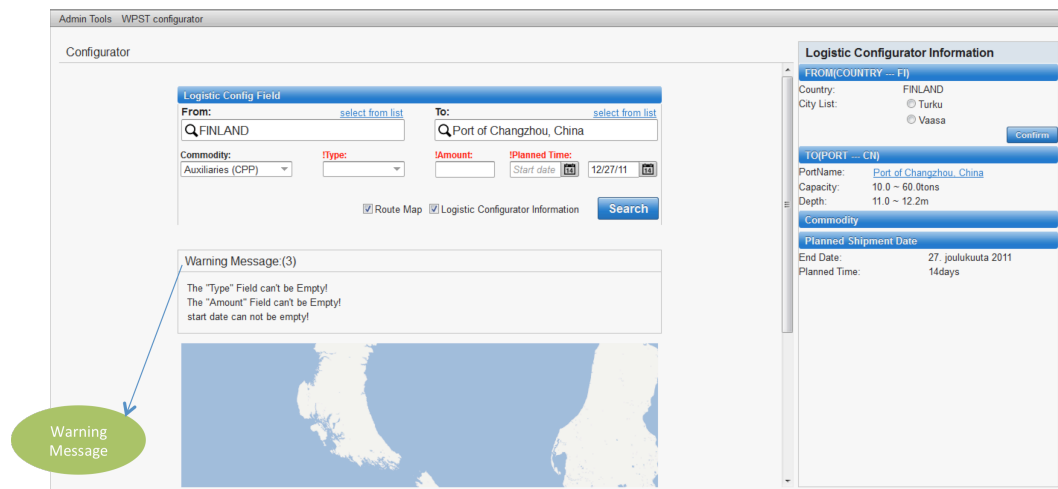


Figure 22. Error configurator view

4.2 WSPT Configure Result View

The result view is displayed after the user presses the search button. It was designed in two parts separately. The one is the search result panel, and another is

configuration filter panel. Different operators use the search result panel to show the freight rate request information for the new creation logistics project. The result is default sorted by price as we see in the picture, and the configuration filter panel can let the user filter the result.

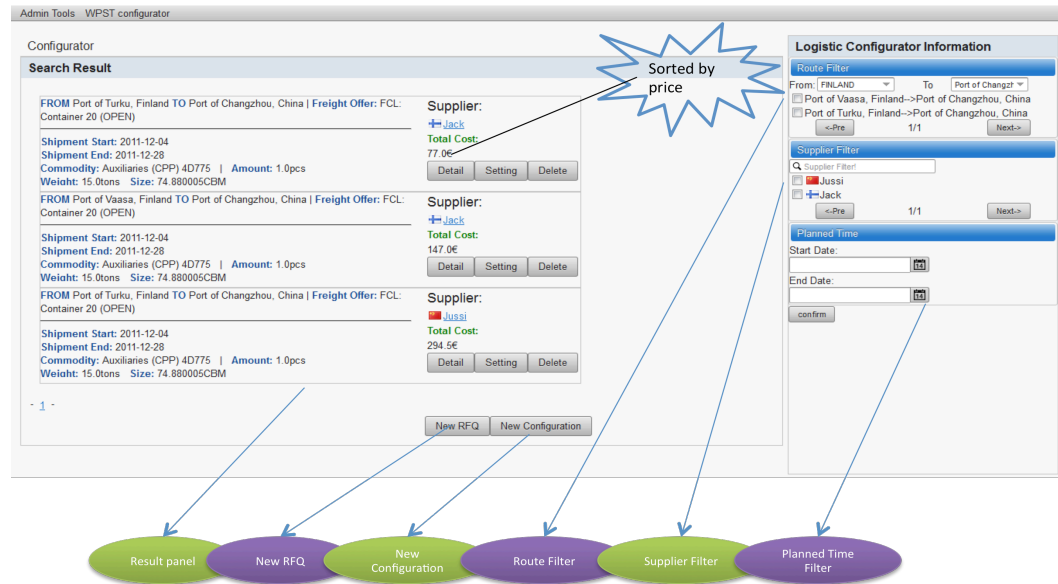


Figure 23. Result view

The “New RFQ” button is used to create a new request freight quotation for the configuration project, and the “New Configuration” lets the user to configure a new logistics project.

4.3 Details Info View

The details info view is used to display the project detail information. The dialog will be shown after the user clicks the “Detail” button on the result item. The detail information contains route information, commodity information, and cost information and operator information. The view is designed as shown below in Figure 24.

After press details button, the application will show project details information

Generate an preconfig report in pdf format that allow user could save & print out.

Route Information

Loading Port: [Port of Turku, Finland](#) Unloading Port: [Port of Changzhou, China](#)

Planned Shipment Start: 04-12-2011 Planned Shipment End: 28-12-2011

Distance Between Ports: 23456.0 Established Time to Arrival: 24days

Commodity Information

Commodity: [Auxiliaries \(CPP\) 4D775](#) Amount: 1.0pcs

Class: Controllable pitch propeller system (CPP) Freight Offer: FCL

Unit Weight: 15.0tons Unit Size: 7.488E10CBM

Total Weight: 15.0tons Total Size: 74.880005CBM

Cost Information

Lifting Rate: 1.0€/tons Transpotation Rate: 1.0€/tons

Surcharge: 3.0 Lift Cost: 15.0€

Transpotation Cost: 15.0€ Total Cost: 77.0€

Supplier Information

Supplier Name: Jack Company: ANL Netherlands B.V

Fax: Email: linyusos@gmail.com

Phone: +358 466115345 Skype: linyusos@gmail.com

Generate PDF

Figure 24. Details view

The “Generate PDF” button is able to generate the project information in PDF format.

4.4 PDF View Layout

Figure 25 is the Logistics Preconfig Report designed by iReport Designer. The document format is PDF. The content consists of four parts information as details view.

WARTSILA Logistics Preconfig Report

Unsetting Project

Route Configuration

Loading Port	Port of Turku, Finland
Unloading Port	Port of Changzhou, China
Planned Shipment Start	Sun Dec 04 22:05:00 EET 2011
Planned Shipment End	Wed Dec 28 22:05:00 EET 2011
Route Distance	23456.0
Established Time to Arrival	-24

Commodity Configuration

Commodity	40775
Class	Auxiliaries (CPRP775)
Amount	1.0
Freight Offer	FCL: Full Container Load/Container 20
Dimensions (W*L*H)	W:4800.0 L:3900.0 H:4000.0
Unit Size	7.488E10
Unit Weight	15.0
Total Size	74.880005
Total Weight	15.0

Unsetting Project

Cost Configuration

Lifting Rate	1.0
Transportation Rate	1.0
Surcharge	3.0
Lifting Cost	15.0
Transportation Cost	15.0
Total Cost	77.0

Supplier Information

Supplier Name	Jack
Company	ANL Netherlands B.V
Country	FINLAND
Fax	null
Email	imyyoon@gmail.com
Phone	+358 406115345

Sun Dec 04 22:53:45 EET 2011 Page 1 of 2

Sun Dec 04 22:53:45 EET 2011 Page 2 of 2

Figure 25. PDF Layout

The user can get this document after press “Generate PDF” button on result panel.

4.5 Administrator View

The administrator view consists of a left side bar and a right content panel. The left side bar classifies the functions into different classes, and the user can choose an admin operation by clicking the function item on the sidebar, then the operation view will be displayed on the right content panel. Figure 26 is an example view for adding a new port.

Admin Tools WPST configurator

Route Management

- New Port
- Port Management
- Route Management

General Information Form

Port Name *	Port Certificate	Port Authority
Country	City	
Time Zone	Currency	GPS Location
Port Phone	Port Fax	+Longitude +Latitude
		Port E-Mail Port Site

Regular Shipline Information

Prefer Shipline Information

Customer Operations

Equipment Management

Supplier Management

User Management

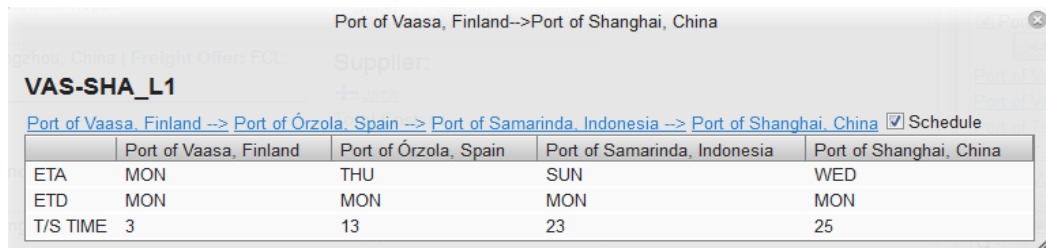
Others

Figure 26. Admin view

To access the admin view need the user has the admin right first or the application will allow the user to enter inside, and the admin operation is related to the WSPT database operation, so the admin behavior will be recorded to the database at the same time. For example, the creator, create date information will be stored when the user creates a new port.

4.6 Schedule Info View

This is an additional function view, which was requested by the customer. The user can check the route schedule when clicking the route item on the filter panel. If the route has schedule information, there will be a checkbox on the left of the route information. The users can check or close the route schedule table.



	Port of Vaasa, Finland	Port of Órzola, Spain	Port of Samarinda, Indonesia	Port of Shanghai, China
ETA	MON	THU	SUN	WED
ETD	MON	MON	MON	MON
T/S TIME	3	13	23	25

Figure 27. Shipment schedule table view

The route schedule information view is quite useful for the customer, and the schedule is a significant factor considered by the user.

5 IMPLIMENTATION AND DEPLOYMENT

The implementation and deployment will be described in this part. The main configurations deployment and functions implementation are illustrated separately..

5.1 Deployment

As motioned before, the WSPT project is developed on Struts 2, Hibernate and Vaadin framework. The following subheadings describe the architecture and configuration file for the deployed WSPT project in detail.

5.1.1 Deployment Descriptor file

The deployment descriptor file describes how to deploy a web application in a servlet container. /16/ Below is the WSPT application's web.xml.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app id="WebApp_9" version="2.4"
xmlns="http://java.sun.com/xml/ns/j2ee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee
http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd">
  <display-name>WSPT Application</display-name>
  <context-param>
    <description>Vaadin production mode</description>
    <param-name>productionMode</param-name>
    <param-value>true</param-value>
  </context-param>
```

Snippet 1. Enable production mode in web.xml

The <context-param> tag provides parameters to the entire context for WSPT application. The parameters are meant for developing this application in the production mode.

```
<listener>
  <listener-class>
    com.wartsila.logistics.wspt.listener.ApplicationListener
  </listener-class>
</listener>
```

Snippet 2. WSPT application listener initialization

The parameters in listener tag indicate an application listener class, which is used to initialize the application data.

```
<filter>
  <filter-name>struts2</filter-name>
  <filter-class>
    org.apache.struts2.dispatcher.ng.filter.StrutsPrepareAndExecuteFilter
  </filter-class>
</filter>
<filter-mapping>
  <filter-name>struts2</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
<filter>
  <filter-name>VaddinLoginFilter</filter-name>
  <filter-class>
    com.wartsila.logistics.wspt.filter.VaddinLoginFilter
  </filter-class>
  <init-param>
    <param-name>redirectPath</param-name>
    <param-value>/login.jsp</param-value>
  </init-param>
</filter>
<filter-mapping>
  <filter-name>VaddinLoginFilter</filter-name>
  <url-pattern>/VaadinvizApplication/*</url-pattern>
</filter-mapping>
```

Snippet 3. Filter configuration for Struts 2 and Vaadin

WSPT has two filters for different framework. One is used for Struts2, and another is used for Vaadin. All the url requests will be sent to Struts2 listener first to check the user login information or register information. The Vaadin login filter will save the login information and request user data into application context. The page will redirect to the application main view if the user data is validated by the login filter.

```
<servlet>
  <servlet-name>wspt_vaddin Application</servlet-name>
  <servlet-class>
    com.wartsila.logistics.wspt.servlet.JQueryServlet
  </servlet-class>
  <init-param>
    <description>Vaadin application class to start</description>
    <param-name>application</param-name>
    <param-value>
      com.wartsila.logistics.wspt.app.WsptApplication
    </param-value>
  </init-param>
  <init-param>
```

```

        <description>
            Application widgetset
        </description>
        <param-name>widgetset</param-name>
        <param-value>
            com.wartsila.logistics.wspt.app.widgetset.WsptWidgetset
        </param-value>
    </init-param>
</servlet>
<servlet-mapping>
    <servlet-name>wspt_vaddin Application</servlet-name>
    <url-pattern>/VaadinVizApplication/*</url-pattern>
</servlet-mapping>

```

Snippet 4. Servlet configuration

The WSPT `<servlet>` is a container tag which is used to initialize the application and catch all requests sent to the WSPTApplication class from url which are specified as “application url /VaadinVizApplication/* ”.

```

<welcome-file-list>
    <welcome-file>index.html</welcome-file>
</welcome-file-list>
</web-app>

```

Snippet 5. Index file in web.xml

The `<welcome-file-list>` contains the welcome files when no filename is provided in the URL.

5.1.2 Struts 2 configuration file

The main task of Struts2 configuration file is used to login and registration function for WSPT. The following snippet will describe the configuration file.

```

<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC "-//Apache Software Foundation//DTD
Struts Configuration
2.0//EN" "http://struts.apache.org/dtds/struts-2.0.dtd">
<struts>
    <constant name="struts.enable.DynamicMethodInvocation"
        value="false" />
    <constant name="struts.devMode" value="true" />
    <constant name="struts.custom.i18n.resources"
        value="ApplicationResources" />
    <!--enable vaddin application with struts2 -->
    <constant name="struts.action.extension" value="action" />
    <!--enable vaddin application with struts2 -->

```

Snippet 6. Enable action tail in Struts 2

In order to enable debug function on Struts2 framework, the constant value of `struts.devMode` is set to `true`. More noteworthy is that the constant value of `struts.action.extension` should have “action” value, because the Struts2 filter and Vaadin framework should catch different requests separately and this setting avoids the conflict between the two filters.

```
<package name="user" namespace="/user" extends="struts-
default">
  <action name="User_add"
class="com.wartsila.logistics.wspt.web.view.UserAction"
method="add">
    <result name="success">/index.jsp</result>
    <result name="input">/user/user_register.jsp</result>
  </action>
  <action name="login"
class="com.wartsila.logistics.wspt.web.view.LoginAction">
    <result name="URLRedirect"
type="redirect">${loginDirection}</result>
    <result name="success">/index.jsp</result>
    <result name="input">/login.jsp</result>
  </action>
</package>
</struts>
```

Snippet 7. Registration action configuration in Struts2

There are two actions under package tag, add action is used to user registration and login action is used for user login. The actions have `<result>` tag and the name attribute are the return value from WSPT action classes.

5.1.3 Application Property File

The application property file contains the validation data and page text information especially used in the registration form.

```
errors.invalid=${getText(fieldName)} is invalid.
errors.required=${getText(fieldName)} is required.
errors.number=${getText(fieldName)} must be a number.
errors.range=${getText(fieldName)} is not in the range
${min} and ${max}.
errors.stringrange=${getText(fieldName)} must be at least
${minLength} characters and less than ${maxLength} charac-
ters.
#Registration Form Element
username= User Name
firstname= First Name
lastname= Last Name
password= Password
confirmPassword= Confirme Password
gender= Gender
email= E-Mail
```

```

postal= Postal
address= Street
birthday= BirthDate
confirmPassword.isconfirm=${getText('password')} not same as
${getText('confirmPassword')}!
#END_ Registration Form Element

```

Snippet 8. Application properties file

5.1.4 Hibernate Configuration File

The WSPT application was developed with the Hibernate framework, and the “hibernate.cfg.xml” is the Hibernate configuration file, which contains the information needed to connect to the persistence layer and mapping documents.

```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC "-//
//Hibernate/Hibernate Configuration DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-configuration-
3.0.dtd">
  <hibernate-configuration>
    <session-factory name="">
      <property name="hibernate.connection.driver_class">
        org.gjt.mm.mysql.Driver
      </property>
      <property name="hibernate.connection.password">
        password
      </property>
      <property name="hibernate.connection.url">
        jdbc:mysql://localhost:3306/wpst_version_one
      </property>
      <property
        name="hibernate.connection.username">root</property>
      <property name="hibernate.dialect">
        org.hibernate.dialect.MySQLDialect
      </property>
      <property name="connection.pool_size">1</property>
      <property
        name="hibernate.current_session_context_class">
        org.hibernate.context.ThreadLocalSessionContext
      </property>
      <property name="dialect">
        org.hibernate.dialect.MySQLDialect
      </property>
      <property name="show_sql">
        false
      </property>
      <property
        name="hibernate.connection.zeroDateTimeBehavior">
        convertToNull
      </property>
      <property name="hbm2ddl.auto">update</property>
    </session-factory>
  </hibernate-configuration>

```

Snippet 9. Hibernate configuration file

The upper file content shows the data source name and JDBC details, which is required to make JDBC connection to the MySql database.

```

<mapping
class="com.wartsila.logistics.wspt.model.CityBean"/>
<mapping
class="com.wartsila.logistics.wspt.model.CountryBean"/>
<mapping
class="com.wartsila.logistics.wspt.model.EquipTypeBean"/>
<mapping
class="com.wartsila.logistics.wspt.model.EquipmentBean"/>
<mapping
class="com.wartsila.logistics.wspt.model.LogisticsProject
Bean"/>
<mapping
class="com.wartsila.logistics.wspt.model.PortBean"/>
<mapping
class="com.wartsila.logistics.wspt.model.RouteBean"/>
<mapping
class="com.wartsila.logistics.wspt.model.RouteFreightRate
Bean"/>
<mapping
class="com.wartsila.logistics.wspt.model.UserBean"/>
<mapping
class="com.wartsila.logistics.wspt.model.FreightOfferBea
n"/>
<mapping
class="com.wartsila.logistics.wspt.model.OperatorBean"/>
<mapping
class="com.wartsila.logistics.wspt.model.EquipClassBea
n"/>
<mapping
class="com.wartsila.logistics.wspt.model.CompanyBean"/>
<mapping
class="com.wartsila.logistics.wspt.model.ContainerBean"/>
<mapping
class="com.wartsila.logistics.wspt.model.ShippingLines
Bean"/>
<mapping
class="com.wartsila.logistics.wspt.model.LineStopsBean"/>
<mapping
class="com.wartsila.logistics.wspt.model.StopRoleBean"/>
<mapping
class="com.wartsila.logistics.wspt.model.WeekBean"/>
</session-factory>
</hibernate-configuration>

```

Snippet 10. Hibernate model-mapping configuration

This is mapping documents information, which is related to the WSPT model classes. All of the information will be stored in the session factory.

All of the above are deployment files for WSPT application and the main logical and code implementation will be described in detail in the next section.

5.2 Main Functions Implementation

This part contains the main logical and code implementation. Each subheading has at least one code snippet to show the implementation and describe the app function in details. Moreover, there are two algorithms for price sorting and shipment schedule in this section.

5.2.1 Create new Configuration

A new configuration is the core function in WSPT application. Users need to fill out the requested field on the ConfiguratorSearchFieldPanel to create a new logistics configuration. The brief implementation code of this method is described below.

```
/**
This is a panel for configurator field
*/
class ConfiguratorFieldPanel extends Panel {
    public ConfiguratorFieldPanel(String caption) {
        super(caption);
        buildView();
    }
    public void buildView() {
        VerticalLayout vl_content = new VerticalLayout();
        setContent(vl_content);
        setMargin(true);
        setSpacing(true);
        setSizeFull();
        setStyleName("borderless light");
        //instantiates google map component
        googleMapComponent= new WsptGoogleMapComponent();
        googleMapComponent.setVisible(false);
        p_searchField = new ConfiguratorSearchFieldPanel("Logistics
Config Field");
        Label l_space = new Label();
        l_space.setHeight("30px");
        vl_content.addComponent(l_space);
```

```

        vl_content.addComponent(p_searchField);
        vl_content.setComponentAlignment(p_searchField,
        Alignment.TOP_CENTER);
        Label l_space1 = new Label();
        l_space1.setHeight("30px");
        vl_content.addComponent(l_space1);
        //instantiate warning message panel
        initialWarningMsgPanel();
        vl_content.addComponent(warningMsgPanel);
        // add google map component
        vl_content.addComponent(googleMapComponent);
        vl_content.setComponentAlignment(warningMsgPanel,
        Alignment.TOP_CENTER);
    }
}

```

Snippet 11. The configuration field layout

Moreover, this function also provides the map view and warning notification features, which will be explained in detail in the follow sections. It helps the user to configure a safe and intuitive logistics project in a precise way.

5.2.2 Sort Result by Price

Sorting configuration result is needed when the configuration results are displayed on the result panel. The outcomes are so confusing that we need a sorting method to make them more readable and suitable for user requirements. The total cost is the main consideration by the user, so I decided to implement a sorting function by price for this application. In the future, more sorting functions will be developed for production.

The first of all we should get the result parameters and sort them with the function, which provided by `WsptDataCalculator` class. The price sort function is implemented with merge sort algorithm shown below.

```

/**
 * sort configuration results by price from low to high
 */

```

```

public static void sortRouteFreightRateBy-
Price(RouteFreightRateBean> list, LogisticsProjectBean pro-
jectbean, int left, int right) {
    int i, j;
    //sorting is starting from the middle value
    Float middle;
    i = left;
    j = right;
    //calculate the total cost for the middle item
    middle = totalCostCalculation(projectbean, list.get(left));
    while (true) {
        // if the left i is lower than right position and the total
        //price is lower than middle price the i will plus one
        while ((++i) < right - 1 && totalCostCalculation(projectbean,
list.get(i)) < middle);
        // if the the j is larger than left number and the total
        //price is bigger than middle the j will be minus one
        while ((--j) > left && totalCostCalculation(projectbean,
list.get(j)) > middle);
        // if I is larger or equals to j the loop will break
        if (i >= j)
            break;
        // switch I and j position if I bigger than j
        swap(list, i, j);
    }
    // swap left and j position if left bigger than j
    swap(list, left, j);
    //if left is lesser than j sorting start from left to j and
    // and recursive the sort function, vise versa
    if (left < j)
        sortRouteFreightRateByPrice(list, projectbean, left, j);
    if (right > i)
        sortRouteFreightRateByPrice(list, projectbean, i, right);
}

```

Snippet 12. WPST price sorting algorithm

In order to achieve the aim to sort the result with the merge sort algorithm we needed four parameters: the list of RouteFreightRateBean object, the logistics projectbean object, the sort interval attribute. The general idea of sorting is that the parent tree will separate in the left part, and right part by middle value, which is calculated with the totalCostCalculation method and then traversing the each part in a recursive way as sub branches, the element will swap to the right position in the result list until all elements is ranked from the lowest to the highest by price values. /11/

5.2.3 Result Filter

The result filter helps user to find the result, which meets his requirements. The result filter consists of three filter panels, which contains route, operator and price information. The filter class includes the following panels.

```
//customer route fileter panel will catch two objects from user
//insert, the same as operator and price
CustomerRouteFilterPanel cofp = new CustomerRouteFilter-
Panel(objs[0], objs[1]);
final CustomerOperatorFilterPanel csfp = new CustomerOperatorFil-
terPanel();
CustomerPriceChangePanel cdcp = new CustomerPriceChangePanel();
```

Snippet 13. Three filter components

With the view of the CustomerRouteFilterPanel, there are two parameters to initial this class which are the obj[0] and the obj[1]. The obj[0] is the departure place object, and the obj[1] is the destination place object. The result panel is making up of these three filter panels.

```
List<RouteBean> routeList = cofp.getSelectedRouteList();
List<OperatorBean> operatorList = csfp.getSelectedOperatorList();
p_resultPanel.setFRQPageComponent(routeList, operatorList);
```

Snippet 14. Get Freight Rate Request page

The main idea of the filter function for WSPT is to reset the result information, which is provided by those three panels, the list of routes information comes from CustomerRouteFilterPanel and CustomerOperatorFilterPanel provides the list of operators information. At last, the CustomerDataChangePanel is offering the interval shipment date information, which is required by the user.

5.2.4 WSPT Map

A feature worth mentioning in the WSPT application is the map view component. It helps the user to view the route information in an intuitionistic way. The map plugin will catch the place information and sign it on the map with a special mark.

For example, the start place icon is like  and the end place icon is .

```
// original place information get from from combobox
Object original_place = cb_from.getValue();
la_from.setValue("<span style='font-size:14px;font-weight:bold;'>From:</span>");
// when user insert the right information to the text field
// the application will remove the warning message from the panel
removeWarningMsg(cb_from);
injectedDataToConfiguratorInfo(original_place, "FROM");
// the original object will be injected to google component and
//mark it out on the goolge map view
googleMapComponent.markPlaces(cb_from.getValue(), "FROM");
// draw available ship lines
if (cb_from.getValue() != null andand cb_to.getValue() != null) {
    googleMapComponent.drawShipLines(cb_from.getValue(),
        cb_to.getValue());
}
```

Snippet 15. Drawing the original place point on WSPT map

The original place is the departure place. The WSPT application will get the original place value from “original ComboBox” and check destination value is not null, and then ship line is drawn on the map with red color.


```

Object dest_place = cb_to.getValue();
la_to.setValue("<span style='font-size:14px;font-
weight:bold;'>To:</span>");
// when user insert the right information to the desitination
//combobox field the application will remove the warning message
//original the panel
removeWarningMsg(cb_to);
injectedDataToConfiguratorInfo(dest_place, "T0");
// the destination object will be injected to google component
//and mark it out on the goolge map view
googleMapComponent.markPlaces(cb_to.getValue(), "T0");
if (cb_from.getValue() != null andand cb_to.getValue() != null) {
    googleMapComponent.drawShipLines(cb_from.getValue(),
    cb_to.getValue());
}

```

Snippet 16. Drawing the destination place point on WSPT map

Meanwhile the ComboBox contains the destination information, which needs to implement the same logic as the original one. One more thing is that the user can view the port detail information after clicking the place mark on the map.

5.2.5 Route Schedule Table

It is well known that the route schedule is an important factor in a logistics project, in order the user to search conveniently the schedule information for different routes, the route schedule table is required in WSPT application, and below is briefly code and explanation.

```

ShippingLinesBean shipline;
// initiates the shipline schedule table by shipline objects
ShiplineScheduleTable shiplineSheduleTable= new ShiplineSched-
uleTable(shipline);
// and set shipline schedule table object to the schedule table
//combobox in order to retrieve it latter in the application.
cb_schedule_table.setData(shiplineSheduleTable);

```

Snippet 17. WSPT shipline schedule table

As we see above, ShippingLinesBean is a prerequisite for achieving the route schedule table.

```
//add stops set to ArrayList and the object are sorted by
//stop weight
Set<LineStopsBean> lineStopSet = shipline.getLine_stop();
List<LineStopsBean> lineStopList = WsptUtilFacto
ry.getDataCalculator().sortLineStopsBean(lineStopSet);
```

Snippet 18. Add shipline stops to schedule table

Firstly, we get line stops from the ship line object, which is an instance of ShippingLinesBean and then sort line stops by WSPT data calculator.

```
startDay=shipline.getEtd().getDay_id();
```

Next, we get the start date from ship line with its ETD attribute.

```
//loop line stop list to get each lineStrop information
for(LineStopsBean lineStop:lineStopList){
this.addContainerProperty(lineStop.getStop_port().getPort_name(),
String.class, null);
this.setColumnHeader(lineStop.getStop_port().getPort_name(),
lineS
top.getStop_port().getPort_name())
//add estimated time arrival to the eta data list
eta_dataList.add(etaCalculate(lineStop));
//add estimated time departure to the etd data list
etd_dataList.add(etdCalculate(lineStop));
// add time estimated to the time data list
time_dataList.add(timeCalculate(lineStop));
}
```

Snippet 19. Add ETA (estimated time arrival), ETD (Estimated time departure) and time usage calculation.

Thirdly, loop the line stops are looped and each line stop within ETA, ETD and duration time are calculated.

```

/**
 * ETA calculate class is a tools to calculate the eta date in
 * weekday format
 */
public String etaCalculate(LineStopsBean lineStop){
    // start day equals to start day plus time data divided by 7
    int _startDay =(startDay+time_data)%7;
    WeekBean weekday;
    // if startday equals to 0 the weekday will be defined as
    //sunday
    if(_startDay==0){
        weekday=DaoFactory.getIWeekDaoInstance().getWeekDayById(7);
    }else {
        weekday=
        DaoFactory.getIWeekDaoInstance().getWeekDayById(_startDay);
    }
    return weekday.getDate_short_name();
}

```

Snippet 20. Logical of established time arrival

Above is the ETA calculating method, and the ETD will be totting-up to the next stop ETA calculation.

```

/**
 * ETD calculate class is a tools to calculate the etd date in
 * weekday format
 */
public String etdCalculate(LineStopsBean lineStop){
    int endDay=(startDay+time_data+lineStop.getStop_days())%7;
    WeekBean weekday;
    if(endDay==0){
        weekday=DaoFactory.getIWeekDaoInstance().getWeekDayById(7);
    }else {
        weekday=
        DaoFactory.getIWeekDaoInstance().getWeekDayById(startDay);
    }
    return weekday.getDate_short_name();
}

```

}

Snippet 21. Logical of estimated time departure

The ETD is calculated based on the ETA plus the stop days. The ETA method will return the abbreviation of week date value as etaCalculate method, and the time date method is used for maintaining current values to provide the next ETA and ETD calculation. The route schedule algorithm is shown in flow chart.

Schedule calculation: Estimated Time of Arrival

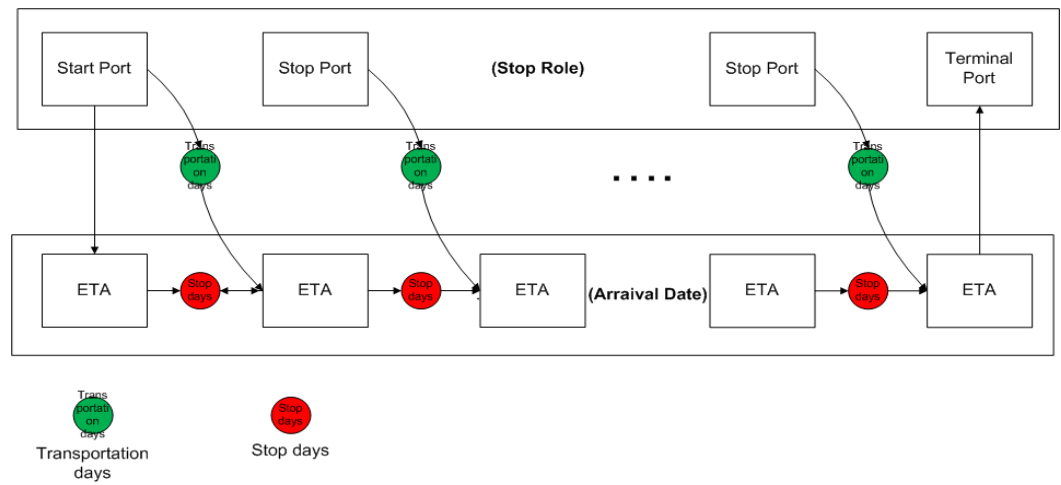


Figure 28. Estimated schedule table algorithm

The green marks represent transportation days, and the red marks represent stop days. The next ETA value is coming from previous one as the formula:

$$ETA_c = ETA_p + D_t + D_s \quad (1)$$

ETA_c = Current ETA, ETA_p = Previous ETA, D_t = Transportation Days, D_s = Stop Days

5.2.6 Generate PDF project file

Generate PDF function was developed with iReport API for Java. It helps the user not only to save data in the database but also to store in the local PC or printing as real documents. It is helpful for the user to manage the configuration data.

```

//generate PDF file
HorizontalLayout hl_btns= new HorizontalLayout();
Button btn_showPdf= new Button("Generate PDF");
btn_showPdf.addListener(new Button.ClickListener() {
    public void buttonClick(ClickEvent event) {
        try {
            // Retrive the byte array resources and insert logistics
            //project object bean to the pdf template, define the
            //filename and generating format
            ByteArrayResource bar= new ByteArrayResource
            (WsptAppResource
            sUtil.getLogisticProjectJasperReportPDFbyteArray(1
            ogisticProjectBean), "test", "application/pdf", getApplication());
            //open pdf file and add report name to pdf report.
            getApplication().getMainWindow().open(bar,
            "ProjectReport");
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
    }
});

```

Snippet 22. WSPT PDF generation

When the user click the “Generate PDF” button, the application sends the request to the WSPT server and the server side will return the byte array data within PDF format. Finally, the application will open a new window to phase the byte array data and display PDF file on it.

5.2.7 Administrator Part Sample Implementation

Below is an abstract class for administrator components. The entire admin component needs to extend this abstract class, and this class will be explained in detail in the following.

```

/**
 * The admin tools abstract class is used for initializing the ad
 * min components in WSPT application.

```

```

* It defines the relationship between admin components and
* navigation bar
*/
public abstract class AbstractAdminToolsComponent extends CustomComponent implements Navigator.View, ViewClassNameInterface {
    private static final long serialVersionUID = -
1989405338640508841L;
    // the administrator navigation bar
    AdminToolsNavigationBar adminToolsNavigationBar;
    private HorizontalSplitPanel adminMainsplit;
    // horizontal split navigator and main information panel
    public void init(Navigator navigator, Application application)
    {
        buildView();
    }
}

```

Snippet 23. The abstract class of admin tools component

Firstly, AbstractAdminToolsComponent extends the CustomerComponent, which are a class provided by Vaadin API and implements interfaces, which are navigator View and ViewClassNameInterface. There are two key components from the above code consisting of this abstract administrator class, they are AdminToolsNavigationBar and HorizontalSplitPanel. The AdminToolsNavigationBar is a navigator component that provides the admin functions for the admin user to choose and HorizontalSplitPanel is displayed the main admin operation content view.

Each admin component has the same structure for extending this abstract class. So we give the buildView method to initialize the View structure that is uniformed by the implementation. There are two abstract methods, which are buildAdminToolView and buildComponentView in the end. They are used to build the customer admin component views, where the GUI is as in Figure 29.

5.2.8 WSPT Calculation Implementation

In this part, the implementation of the comparison, estimation and calculation methods mainly used in WSPT project are introduced.

- Get the maximum float data between two float values.

```

/*
 * Util function in project to get maxium Data between two float
 * data
 */
public static Float getMaximumFloatData(Float fa, Float fb) {
    if (fa != null andand fb != null) {
        boolean condition = (fa >= fb);
        return condition ? fa : fb;
    } else if (fa == null andand fb != null) {
        return fa;
    } else if (fb == null andand fa != null) {
        return fb;
    }
    return null;
}

```

Snippet 24. Get maximum date between two decimal numbers

The idea of the method is to compare two float values and return the bigger one. It is used to find the maximum value, such as crane capacity, quayside capacity and so on.

- The number of days interval

```

/* calculate days between two dates*/
public static long daysBetween(Date max, Date min) {
    return (max.getTime() - min.getTime()) / 86400000;
}

```

Snippet 25. Calculate difference between two dates

Because the `getTime` method in `java.util.Date` returns the milliseconds, so we needed to have the difference value divided by 8640000 which is equivalent of one day.

$$8640000 \text{ (milliseconds)} = 24(\text{h}) * 60(\text{min}) * 60(\text{second}) * 1000(\text{milliseconds})$$

- Weight Estimation

```
/* Estimated weight by weight unit and amount of cargos*/
public static Float weightEstimate(Float unitWeight, Float
amount) {
    Float weight = null;
    if (unitWeight != null andand amount != null) {
        weight = amount * unitWeight;
    }
    return weight;
}
```

Snippet 26. Estimate the weight of cargo transported

The weight estimated by this method in WSPT, the formula is as simple as we know, but the unit of weight is kilogram:

$$W_e (kg) = N_a \times W_u \quad (2)$$

W_e = Estimated Weight, N_a = Value of amount, W_u = Weight value per unit

- Size Estimation

```
/* Estimated size by equipment information and amount of cargos*/
public static Float sizeEstimate(EquipTypeBean equipType, Float
amount) {
    Float size = null;
    if (equipType.getEd_height() == null || equipType.getEd_width()
== null || equipType.getEd_length() == null) {
        return null;
    } else {
        Float m_length = equipType.getEd_length() / 1000;
    }
}
```



```

        Float m_width = equipType.getEd_width() / 1000;
        Float m_height = equipType.getEd_height() / 1000;
        size = amount * (m_height * m_length * m_width);
    }
    return size;
}

```

Snippet 27. Estimate the size of cargo transpoted

Similarly as the weight estimation, the size is calculated by the cubature formula the unit of which is cubic meter. But the data value from the database is millimeter so we divided the metric value into 1000.

- Total cost calculation

```

/* Calculate the total cost by logistics project and freight
rate*/
public static Float totalCostCalculation(LogisticsProjectBean
projectbean,RouteFreightRateBean freightrate) {
    Float totalCost = null;
    // there are two different freght offer, which are FCL and BB.
    if(projectbean.getEquip_type().getFreightOffer().getShotName()
.equals("FCL")) {
        Float surcharge =  freightrate.getRate_baf() +
        freightrate.getRate_caf() +
        freightrate.getRate_war();
        totalCost = projectbean.getEstab_weight()*
        (freightrate.getFreight_rate() * (1 + surcharge) +
        freightrate.getLifting_fee()) + freightrate.getOther_cost();
    } else if (projectbean.getEquip_type().getFreightOffer().
getShotName().equals("BB")) {
        totalCost = projectbean.getEstab_weight()*
        (freightrate.getFreight_rate() +
        freightrate.getLifting_fee()) + freightrate.getOther_cost();
    }
    return totalCost;
}

```

Snippet 28. Estimate the total cost of a logistics project plan

The core calculation in the WSPT project is the total cost and the application should check the freight offer type at first, and decide which cost formula is suitable to calculate the project. There are two kinds of cost formula to calculate the price of the project, one is used for FCL, and another is BB. The formula will be explained below.

BB (Break Bulk) cost estimation function:

$$C_t = W_e \times (R_f + C_l) + C_o \quad (3) / 12 /$$

FCL (Full Container Load) cost estimation function:

$$C_t = W_e \times (R_f \times (1 + R_s) + C_l) + C_o \quad (4) / 12 /$$

$$R_s = R_{war} + R_{baf} + R_{caf} \quad (5) / 12 /$$

C_t = Total cost, W_e = Weight estimation, R_f = Freight rate, R_s =

Surcharge rate, C_l = Lifting cost rate, C_o = Other cost

- Check is quayside capacity overloaded.

```
/* Validate if the transported cargo is overload query side capacity*/
public static boolean isQuerysideCapacityOverload(
    LogisticsProjectBean projectbean, PortBean origin_port,
    PortBean dest_port) {
    Float maximumQuerySideCapacity = getRouteMaximumQuerySide
        Capacity(origin_port, dest_port);
    return (projectbean.getEstab_weight() > maximumQuerySideCapacity) ? true : false;
}
```

Snippet 29. Determine whether query side capacity is overload

In order to reduce the risk in logistics project configuration, this method is used to verify that the configuration data will not have potential risks such as quayside capacity is overloaded etc.

- Get standard container

```
/* Automatically get suitable container for the project based on
transported equipment information*/
public static ContainerBean getSuitContainer(EquipTypeBean
equipType) {
    return DaoFactory.
        getIContainerDaoInstance().getStdContainer(equipType);
}
```

Snippet 30. Determine the suitable container for a logistics plan

The method `getSuitContainer` is helping the user to auto-configure a suitable and standard container for the project if the cargo can be transport in FCL way.

6 SUMMARY

The results of the thesis are presented in the following. The WSPT has been developed on Vaadin framework and deployed on Tomcat7 Server. All objectives of the final thesis were tested on different browsers. The testing of the main features can be described as the following testing table:

Table 20. The application-testing table part 1

<i>Test Case</i>	<i>Description</i>	<i>Predictions</i>	<i>Result</i>
V-1-01	User Login	1. Insert username and password. 2. The system checks whether user account is validate.	OK.
V-1-02	User Registration	1. Insert the required user information. 2. Check the input data is validated. 3. Redirect to the login page if register successfully.	OK.
V-1-03	Check the map function and pre configuration	1. Click "Route Map" checkbox. And the map view should display successfully. 2. Choose the "Logistics Configurator Information" check box, the configuration panel should show out on the right side.	OK.
V-1-04	Create a new logistics project	1. Insert all required information with valid input data. 2. The map view and the configuration panel should be updated when the data input has been changed. 3. Press the search button. 4. Go to the result view.	OK.
V-1-05	Warning function	1. If user insert validate data or the data which is inserted contains risks inside. 2. The warning panel should displayed with detail warning information and number of warnings that indicate user could configure a project step by step.	OK.
V-1-06	Show configuration project detail information	1. Click the "Detail" button on the result item. 2. The detail dialog will displayed successfully with the project detail information.	
V-1-07	Setting project attribute	1. Click "Setting" button on the result item. 2. The project-setting dialog will show out. 3. User can modify the data, which is editable. 4. Press "Update" button, the information should update into system also.	OK.

Table 20 indicates that the basic user functions had been successfully achieved, and the following Table 21 describes the test results of configuration part.

Table 21. The application-testing table part 2

V-1-08	Delete a project	<ol style="list-style-type: none"> 1. Click "Delete" button on the result item. 2. The item which be chosen will be removed from the result panel. 	OK.
V-1-09	Generate PDF file	<ol style="list-style-type: none"> 1. After user click the "Details" button, the detail dialog will be showed out. 2. Click the "Generate PDF" button on the dialog. 3. Pop up PDF dialog that allows user open or save the file. 	OK.
V-1-10	Create a new RFQ	<ol style="list-style-type: none"> 1. Click the "New RFQ" button. 2. Pop up a dialog that let user choose a route exist in the project, which is responsible for the new RFQ. 3. Click "Next" button. 4. Insert the required information and press "Add" button to confirm the actions. 5. The new result should be update on the result panel. 	OK.
V-1-11	Check the filter function	<ol style="list-style-type: none"> 1. Choose the specified port and operator on the filter panel. 2. Press the confirm button. 3. The filtered results should be displayed on the result panel. 	OK.
V-1-12	Schedule Table	<ol style="list-style-type: none"> 1. Click the route link under the route filter. 2. Pop up a route information dialog and selected the "Schedule" checkbox. 3. The route schedule should be visible on the dialog. 	OK.
V-1-13	Check the admin functions	<ol style="list-style-type: none"> 1. Testing create function on admin view for each part. 2. Testing data table for each part. 3. Testing delete feature on the data table. 4. Testing upgrade function on the data table. 	OK.

All of the main features mentioned above in the application have been implemented successfully. There is a short introduction video of WSPT application available on Youtube /18/.

7 CONCLUSION

The WSPT project is an excellent guideline for someone who wants a development project with Vaadin framework. Nowadays, the corporate executives have realized the importance of mobility and scalability of the application, so the RIA is more and more used in industry management. The project aim was to help Wärtsilä improve their logistics management.

This was a long-term project, which required six months to achieve the features mentioned previously in this document. In the end, all the main functions have been implemented in this version.

The core function of WSPT has been implemented. The user can preview a logistic project with this application, and the user can choose the best line operator to handle the required logistic project. The biggest challenges of the application were the route map, route schedule and cost estimation.

The route map feature needs to integrate Google map to the Vaadin Framework, but the add-on of Google map for Vaadin cannot meet the requirements. Another challenge was implementing the logical of the ship lines logical on the Google map with different departure and destination places. Fortunately, the challenge solved by the database design.

The second challenge was the route schedule table. To be realistic, it is rare to find a direct ship line from one port to another, and it is hard to get the transport information. How to show all stop ports information and estimate the total shipping time is explained in chapter 5.2.5.

In a logistic project, there are two different offer types to do a shipment. One is BB (Break Bulk), and another is FCL (Full Container Load). The project cost estimation is based on offer types. The WSPT application gives a solution for users that they do not need to consider the offer type because the offer type is decided by the size of the equipment. The solution for this problem is the total cost estimation explained under section 5.2.8.

In general, the WPST UI design is very user-friendly and can guide the user step by step to configure a logistics project with lower risk and preview of the results beforehand. For example, in the traditional way if Wärtsilä needs to configure a new logistics project, they need to do a lot of research on many aspects, such as choosing operator, port constraint information, the nearest Wärtsilä operator etc. This application helps a Wärtsilä user to preview a logistics project and integrates all the necessary information for configuring a project to satisfy user requirements.

7.1 Further Development

Presently the demo version of the WSPT project has been developed successfully, but it still remains some unimplemented features, which can be improved in the future. Nevertheless, the WPST can provide the Wärtsilä user a configure function for a logistics project, the data information, which on the persistence layer in the application is inserted by the Wärtsilä user manually, so the information cannot be updated automatically. In order to avoid this situation, WSPT has been integrated the Struts2 framework inside to make the application more scalable and smart. After-wards, the WSPT should have applications for the port operator and suppliers that allow the users to update their latest information on time. For example, the operator users can modify their new freight rate information to the WSPT system. Furthermore, more features in the next step should establish a system that provides more features to improve communication between the Wärtsilä users with other users, such as port operators, suppliers etc. There is a list of features for future development in the product version:

- Connect application database to the Wärtsilä center database for back up purpose
- Deploy the project to a cloud server
- Website displays Wärtsilä the latest logistics project and its status that allows operators to bid the project online
- The system also needs to generate an electronic invoice between the operator and supplier who is responsible for the logistics projects.

- The supplier can register into WSPT as a supplier user, which can check all the public information, such as project information, route information, port information, and so on.
- After Wärtsilä has configured a logistics project successfully, the system can intelligently alert the user tasks in the project and provide the tracking route function after the project-commenced operation.

Overall, the WSPT application is an innovation system on supplier chain management because of its attractive functions and concepts.

8 REFERENCES

- /1/ Jax Magazine (2012). [WWW]. Accessed on Internet: <URL: http://www.jaxmag.com/itr/online_artikel/psecom,id,828,nodeid,147.html>
- /2/ Oracle Technology Network (2012). [WWW]. Accessed on Internet: <URL: <http://www.oracle.com/technetwork/java/javase/overview/javahistory-index-198355.html>>
- /3/ About.com website. (2012). [WWW]. Available on the Internet: <URL: <http://java.about.com/od/gettingstarted/a/whatisjava.htm>>
- /4/ Marko Grönroos. (2012). Book of Vaadin . 4th Edition 1st Revision Edition. Vaadin Ltd.
- /5/ Vaddin website. (2011). [WWW]. Available on the Internet: <URL: <https://vaadin.com/features>>
- /6/ Datadisk. (2012). [WWW]. Available on the Internet :<URL: http://www.datadisk.co.uk/html_docs/java_app/tomcat6/tomcat6_apache.htm>
- /7/ Apache website. (2012). [WWW]. Available on the Internet: <URL: <http://tomcat.apache.org/tomcat-5.5-doc/catalina/funcspecs/mbean-names.html>>
- /8/ Jeff Linwood., Dave Minter. (2010). Beginning Hibernate
- /9/ Dreamtech Press. (2007). Java Server Programming (J2EE1.4)
- /10/ James Tauber Priority Levels. (2005). [WWW]. Available on the Internet: <URL: http://jtauber.com/blog/2005/01/03/priority_levels/>
- /11/ Open Data Structures - Section 11.1.1 - Merge Sort. [WWW]. Available on the Internet: <URL: <http://opendatastructures.org/versions/edition-0.1c/ods-java/node3.html>>
- /12/ Exporthelp. (2012). [WWW]. Available on the Internet: <URL: http://www.exporthelp.co.za/modules/16_logistics/sea_freight_calculations.html>

- /13/ The Java Tutorials. (1995). [WWW]. Available on the Internet:<URL: <http://docs.oracle.com/javase/tutorial/getStarted/cupojava/win32.html> >
- /14/ Compare Print RIA Diagram. (2009). [WWW]. Available on the Internet: <URL: <http://www.fileden.com/files/2009/5/21/2450464/Compare%20Print%20-%20vaadin.pdf>>
- /15/ TIOBE Programming Communication Index for March 2012. (2012). [WWW]. Available on the Internet :<URL: <http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html>>
- /16/ Metawerx- Reference Guide for Tomcat. (2012). [WWW]. Available on the Internet: <URL: <http://wiki.metawerx.net/wiki/Web.xml>>
- /17/ Introduction to hibernate framework architecture. [WWW]. Available on the Internet: < URL: <http://viralpatel.net/blogs/introduction-to-hibernate-framework-architecture/> >
- /18/ WSPT application preview. [WWW]. Available on the Internet: <URL: <https://www.youtube.com/watch?v=5HTCvIgN6GY>>