Estima
Software Architecture Document

Version <1.0>

Revision History

Date	Version	Description	Author
12 / 01 / 2012	1.0	Initial Software Architecture Document	D.A.U.Nanayakkara – 090342F

Table of Contents

Software Architecture Document

1. Introduction

1.1 Purpose

This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system. This document shall be used as a guideline for implementing the actual system. The use cases described and the different views used to describe the system will be useful in understanding the systems behavior and its expected working environment. This will be helpful in having great insight to actually implement the system.

1.2 Scope

This Software Architecture Document applies to the "Estima" System which will be developed using this document as a guideline.

1.3 Definitions, Acronyms, and Abbreviations

BOQ:	Bill of Quantities
Taking off:	Process of extracting details from the drawings and specifications.
TDS:	Time dimension sheets (A method of taking off)
SLS 573:	Sri Lanka Standard 573: 1999 [UDC 69(083.74)] : Method of Measurement of Building Works (first revision)
Item	Building construction activity. (eg: concreting, site clearance).
Unit	Standard unit of measurement of an item (eg: meters, kg, nos (discrete units in numbers)).
Quantity	Amount of work required for an item in standard units.
Rate	Cost of a unit of item. (may also be referred to as unit rates)
Unit rate	Rate
Amount	Total cost of an item.
Description	Explanation of work carried out under an item.

1.4 References

Sri Lanka Standard 573: 1999 [UDC 69(083.74)] : Method of Measurement of Building Works (first revision)

1.5 Overview

This architecture design document looks at the software architecture design from five different views. Use case view, logical view, process view, implementation view and data view. Using these perspective the design of the system can be better understood.

2. Architectural Representation

This document presents the architecture as a series of views; use case view, process view, deployment view, and implementation view. These views are presented as per the Rational Rose Models and Unified Modeling Language (UML).

3. Architectural Goals and Constraints

The Estima system contains the following requirements and constraints that have a significant impact on the architecture.

- System shall be able to do the calculations at least within one second.
- All performance related requirements mentioned in SRS shall be taken into consideration in designing the system.
- The system shall be designed in such a way that the users can make the system work with custom databases.

4. Use-Case View

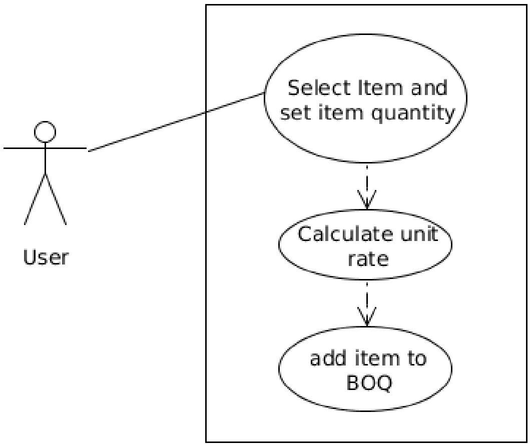
Use cases that have a significant impact on the architecture of the software are listed below. The scenarios under consideration depict important functionality of the system.

- Add item to BOQ
- Add / remove or edit item in database
- Add / remove or edit material in database
- Add / remove or edit unit rate calculations
- Save work
- Export worksheet as pdf.

4.1 Architecturally significant use cases

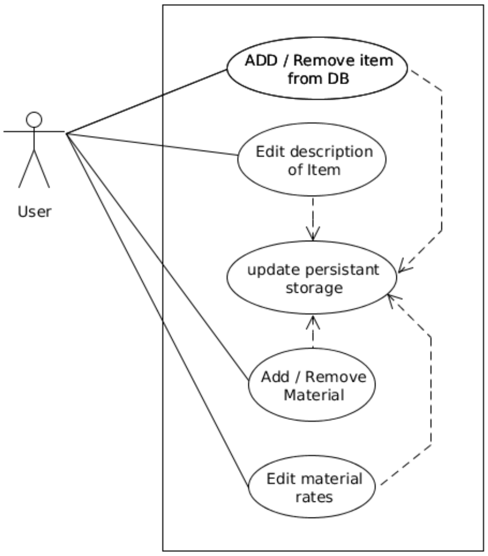
4.1.1 Scenario 1: Add item to BOQ

Description: after “taking off” values from the plan user select items as per the SLS 573 and set the quantity value. After that the system calculates the unit rate of the item using a predefined formula and multiplies the quantity by unit rate to get the amount. After that the item is added to the BOQ. For the scenario to take place the item user select should be in the database and the unit rate calculation formula should be present.



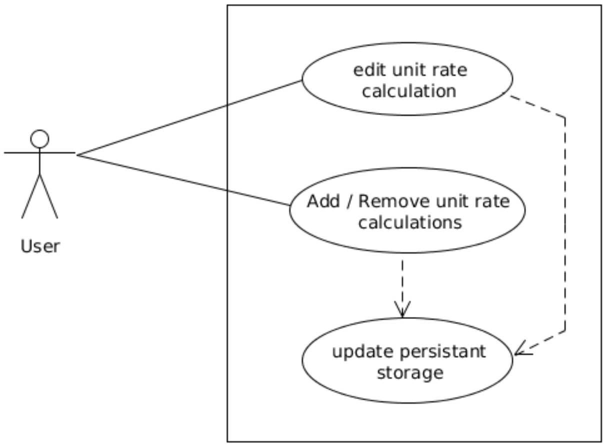
4.1.2 Scenario 2: Add / remove or edit item in database

Description: when the user need to add new item to the database or remove item from the database user selects the desired item and execute the operation. Then the system shows a warning message and after user confirmation updates the persistent storage reflecting the user intension. Similar to this material is also added and removed from the database. Other than that user can edit description of an existing item. The system issues a warning and after that update the database if user confirmation is received.



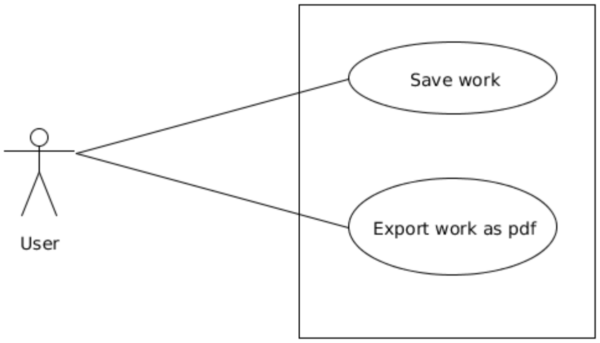
4.1.3 Scenario 3: Add / remove or edit unit rate calculations

Description: When user need to edit a certain unit rate calculation formula he selects the relevant item and edits the material quantities and materials needed for the item and saves the changes. Then the system issues a warning and after confirming the persistent storage is updated.


edit unit rate calculation
Add / Remove unit rate calculations
update persistant storage
User

4.1.4 Scenario 4: Save work and Export worksheet as pdf.

Description: when user want to save his current work he use the save project option. Then the system ask for the intended location of the secondary storage that the user intend to store the project. After getting user location from user system stores the project files in persistent storage for later use. When user wants to export the worksheet to printable material he uses this option. When selected the system exports the BOQ in to pdf and stores in a user desired location.



5. Logical View

5.1 Overview

5.1.1 Application Layer

Application layer includes the UI framework related packages. GUI’s that interact with users and that represent models in the business logic as views are in this layer

5.1.2 Business Logic layer

Core functionalities of the system are implemented in this layer. Packages that contain business logic and does the actual intended work is in this layer. BOQ engine class which gets user inputs and does the automation of the BOQ creation process is done in this layer

5.1.3 Data Management Layer

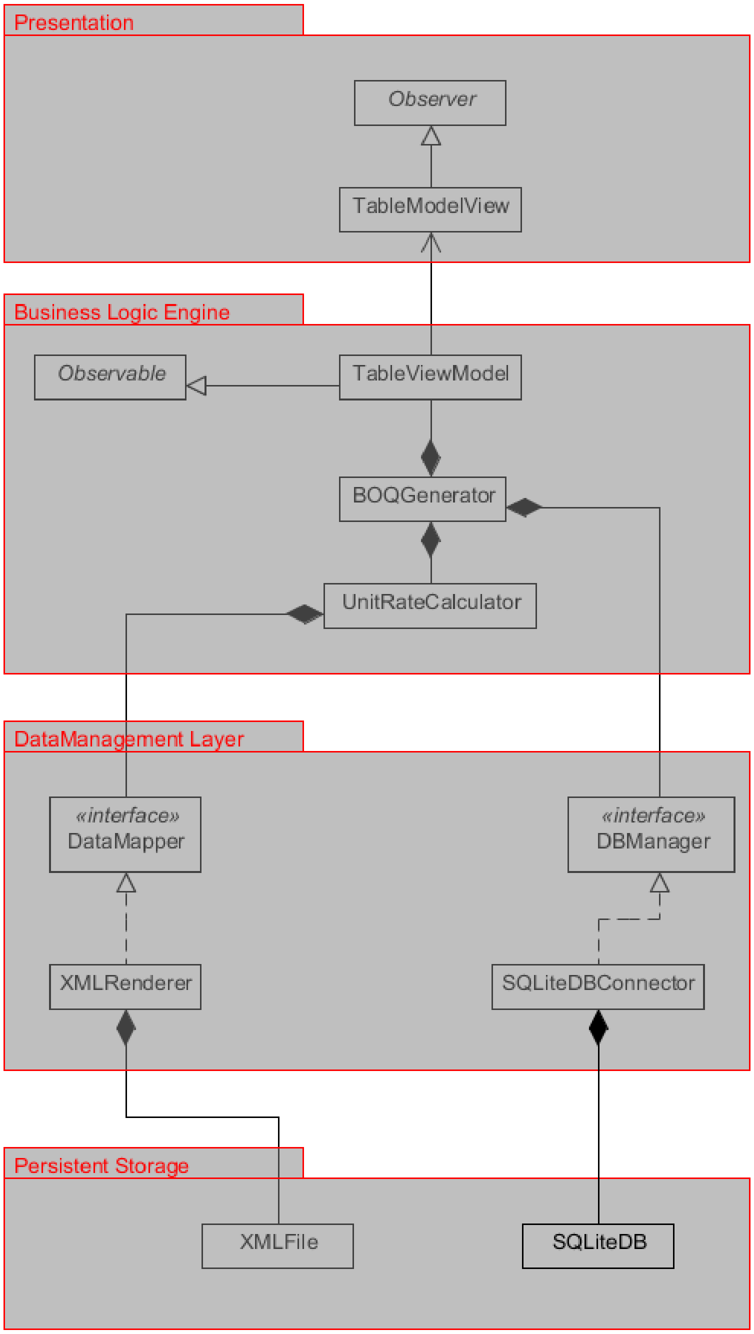
Data management layer works as an intermediary to get data and store data to the database as needed by the logical layer. This layer hides actual data storage methods from the upper layers.

5.1.4 Persistent Data Layer

This layer includes actual files and databases that the files are stored. Database connectors and data mapper classes in the data management layer communicate with the files in persistent data layer.

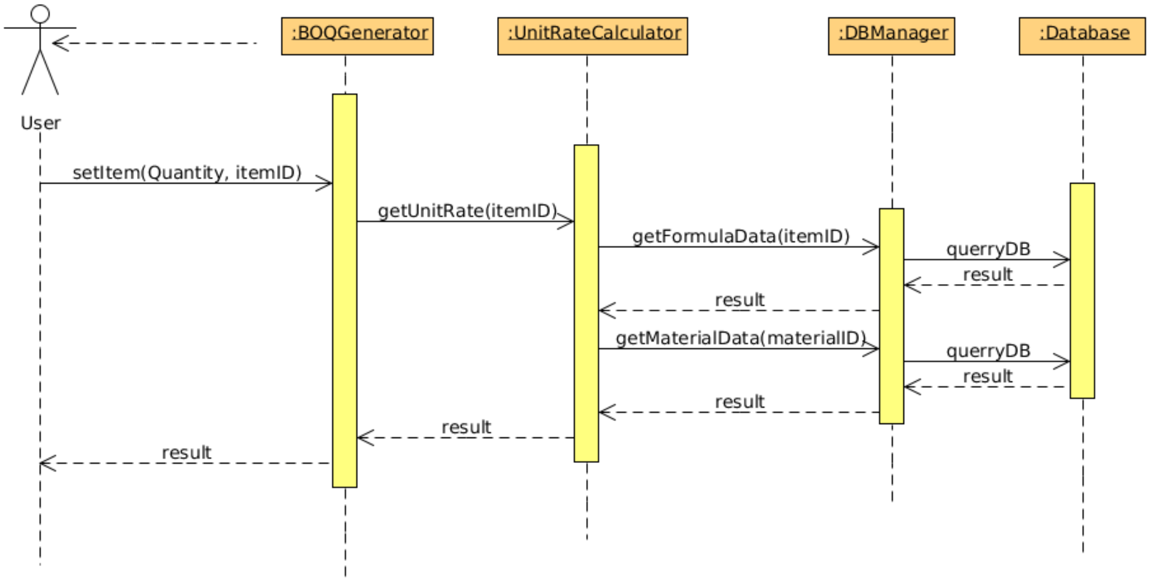
5.2 Architecturally Significant Design Packages

Following illustration shows the main classes that are vital in the architecture of the system. Package that each class falls under is also shown.



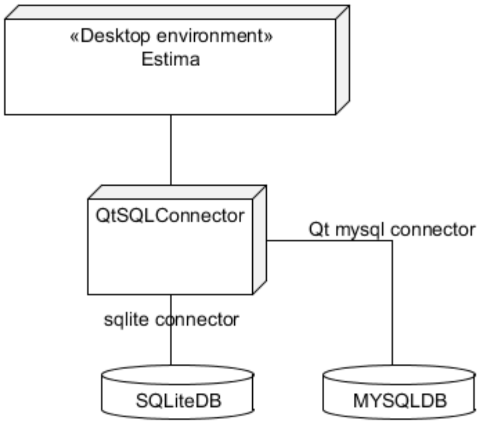
6. Process View

Following sequence diagram displays the key process of the system which is vital for the software architecture. The process related to the scenario 1 of the use cases.



7. Deployment View

"Estima" system is intended to work on desktop environment. And it has SQL connector modules that connect the system to local databases and remote databases. The deployment view is as follows.

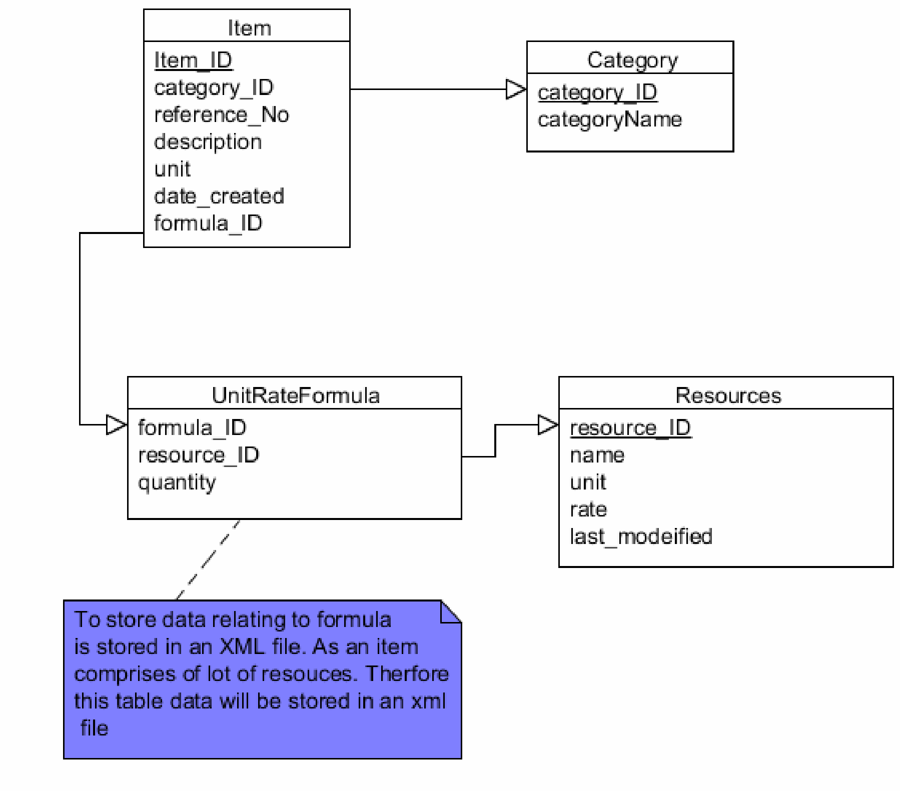


8. Implementation View

"Estima" system is a standalone desktop application. Application is provided as a binary executable file. Qt framework dependent files are converted into native C++ before compilation. Only MySQL server is needed in case where users opt to use remote database.

9. Data View

The database model used to store data is as follows. Unit rate formula table is not implemented in the database. Opposed to that xml files are implemented for each item for ease of use (for user). So that he can use those xml files in two different applications by copying the xml files.



To store data relating to formula is stored in an XML file. As an item comprises of lot of resouces. Therfore this table data will be stored in an xml file

10. Size and Performance

The software is designed to be a single user application. For better performance and less memory usage configuration less SQLite database is used.

11. Quality

Software is designed using design patterns whenever possible. Databases are connected using adapter patterns for easy extensibility and maintainability. MVC (Model-View-Control) is used to accomplish highly responsive GUI’s.