UML Deployment Diagram

The most appropriate architectural style for our financial management application is the repository model. The rationale for this decision is fairly straightforward. The application will have many sub-systems that are able to compute different figures and statistics based on a user’s financial account transactions. Since all sub-systems will have to draw from the same batch of data, the repository model fits best.

The repository model offers several advantages including an efficient way to share large amounts of data. The sub-systems of this model see data as a black-box, they are not concerned with how it is produced. The repository schema provides a uniform, published model for the sharing of data between sub-systems. However, the repository model carries along with it some disadvantages. The sub-systems must agree on the repository schema, which often leads to compromises. The evolution of repository data is expensive and difficult to maintain. Management policies have no specific scope within the application. The repository can be difficult to distribute efficiently in some cases.

Our deployment diagram employs 3 nodes following a 3-tier architecture, a web-server, an application server, and a database server. All of the use cases described in our design phase are allocated to the application server node. Within this node, the use cases can be distributed between the subsystems for Account, Financial Services, and External Account Manager. The Account subsystem handles the use cases for create account, login, logout, manage profile, and delete account. The Financial service subsystem handles the use cases for create transaction, schedule transaction, create budget, calculate cash flow and slippage, export data, and print data. The External account manager handles the use cases for link financial accounts, check credit score, and refresh data.