Online Task Manager

Group8d
Software Architecture Document
Version 1.4

Revision History

Date	Version	Description	Author
09/02/2006	1.0	Skeleton of the Software Architecture Document produced.	My-An Nguyen
19/02/2006	1.1	Sections 1-3 completed. Started section 4.	My-An Nguyen
21/02/2006	1.2	More writing and diagrams	My-An Nguyen
26/02/2006	1.3	Working on section 5	My-An Nguyen
02/03/2006	1.4	Finalizing the document	Oshadha Yohan Kariyawasan, Robin Nadeau, My-An Nguyen, Natalie Villanueva

Table of Contents

1. Introduction....3
   1.1 Purpose....3
   1.2 Scope....3
   1.3 Definitions, Acronyms, and Abbreviations....3
   1.4 References....3
   1.5 Overview....4
2. Architectural Representation....5
3. Architectural Goals and Constraints....6
4. Use-Case View....7
   4.1 Architecturally significant use cases....7
      4.1.1 Assign Task/Subtask....7
      4.1.2 Add New Task/Subtask....8
      4.1.3 Modify Task/Subtask....8
      4.1.4 Delete Task/Subtask....8
      4.1.5 Apply Category....8
      4.1.6 Add Category....8
      4.1.7 Delete Category....8
      4.1.8 Modify Category....8
      4.1.9 Add To-Do Item....9
      4.1.10 Modify To-Do Item....9
      4.1.11 Delete To-Do Item....9
      4.1.12 View Task/Subtask Progress....9
      4.1.13 Add Journal Entry....9
      4.1.14 Modify Journal Entry....9
      4.1.15 Delete Journal Entry....9
5. Logical View....10
   5.1 Overview....10
   5.2 Architecturally Significant Design Packages....11
      5.2.1 Front Controller + Commands....11
      5.2.2 Template View....12
      5.2.3 Data Mapper....13
      5.2.4 Helper....14
      5.2.5 Domain Model....15
      5.2.6 TDG....17
   5.3 Use-Case Realizations....18
6. Process View....19
7. Deployment View....20
8. Implementation View....21
   8.1 Client side....21
   8.2 Server side....21
9. Data View....22

1. Introduction

1.1 Purpose

This document provides an overview of the architecture of the Task Manager system. It embodies the significant decisions that were made concerning the architecture of the application.

1.2 Scope

This Software Architecture Document gives a global view of the architecture of the Task Manager System using the "5 + 1" view model and the template found on the SOEN344 course website. This system is developed by a team of six students registered in the SOEN390 Software Project course.

The Task Manager provides users with the ability to manage and categorize their tasks, track their progress, and create a list of general things to do. All of those features are available to the users wherever they can find a computer with an Internet connection.

1.3 Definitions, Acronyms, and Abbreviations

HTML = Hypertext Markup Language.
JSP = Java Server Pages.
TA = Teacher's Assistant.
TDG = Table Data Gateway.
UI = User Interface.

SOEN342 = Software Requirements course.
SOEN343 = Software Design course.
SOEN344 = Software Architecture course.
SOEN357 = User Interface Design course.
SOEN390 = Software Development Project course.

1.4 References

"SOEN 344, Software Architecture" SOEN344 Course Page [Website] Accessible: http://www.cs.concordia.ca/~chalin/courses/06W/SOEN344/. Accessed February 9, 2006.

"The Servlet Life Cycle" Java Servlet Programming [Website] Accessible: http://www.unix.org.ua/orelly/java-ent/servlet/ch03_01.htm. Accessed

February 26, 2006.

Fowler, Martin. Patterns of Enterprise Application Architecture. Boston: Pearson Education, 2003

1.5 Overview

In compliance with the specified purpose and scope, the rest of this document will deal with the following topics: The Architectural Representation used to describe the system, namely the “5 + 1” view model; The Architectural Goals and Constraints of the Task Manager; The Use Case View; The Logical View; The Process View; The Deployment View and The Implementation View.

2. Architectural Representation

The architecture is represented in this document as a set of five views: the Use Case view, the Logical view, the Process view, the Deployment view, and the Implementation view.

The Use Case view presents the architecturally significant use cases.

The Logical view focuses on the functional requirements of the system.

The Process view presents the allocation of the Logical view components to processes or tasks.

The Deployment view shows the allocation of the Logical view elements to hardware.

The Implementation view is the actual organization of the software. It allocates the Logical view elements to implementation components.

The Data view describes the persistent data storage perspective of the system.

3. Architectural Goals and Constraints

There are a number of goals and constraints that impact the design and architecture of the Task Manager.

1. The Task Manager is developed as part of the SOEN390 course, which, in turn, is intended to be the amalgamation of concepts studied in SOEN342, SOEN343, SOEN344, and SOEN357. Therefore, the system's design will follow a layered, client-server architecture and apply a proper subset of Web Enterprise Application patterns, in addition to using good UI design practices.

2. The development environment provided consists of the Java software development kit (J2SE 1.4.x), Eclipse, Apache Tomcat Server (4.1.x), MySQL database, and common web server. This environment strongly suggests that the Task Manager should, and will, be implemented as a web application using JSP and Servlets.

3. The configuration of the layered architecture is determined by the TaskManBase.zip provided by Stuart Thiel. The login functionality is also provided by him and, therefore, must be integrated within the design.

4. The Task Manager application is dependent on the server on which it runs, Stu. Its performance must be optimized for that server, but its availability is dependent on Stu's up-time.

5. The Stu server has some security policies that limit the implementation and the features of the application. Namely, Stu does not allow one to send emails from it; therefore, the Task Manager cannot send email notifications. Also, the JSP cannot directly access the database, nor can it call any methods that directly access the database.

6. Since the Task Manager is an online application, users must have at least a basic Internet connection and a web browser that supports pages with JavaScript, Cascading Style Sheets and HTML 4.01 Transitional.

7. The Task Manager must ensure a degree of security and privacy; therefore, all users are required to log in with their unique username and password.

8. The Task Manager must ensure that concurrent operations on the system do not cause any errors in the data or transactions.

9. All functional and non-functional requirements mentioned in the Vision Document and Supplementary Requirements and Specification Document must be taken into consideration as the architecture is being developed.

4. Use-Case View

This Use Case view will be used to present the architecturally significant use cases and identify the set of scenarios and use cases that represent important and fundamental functionalities of the system.

4.1 Architecturally significant use cases

The architecturally significant use cases are those highlighted in blue in the following diagram:

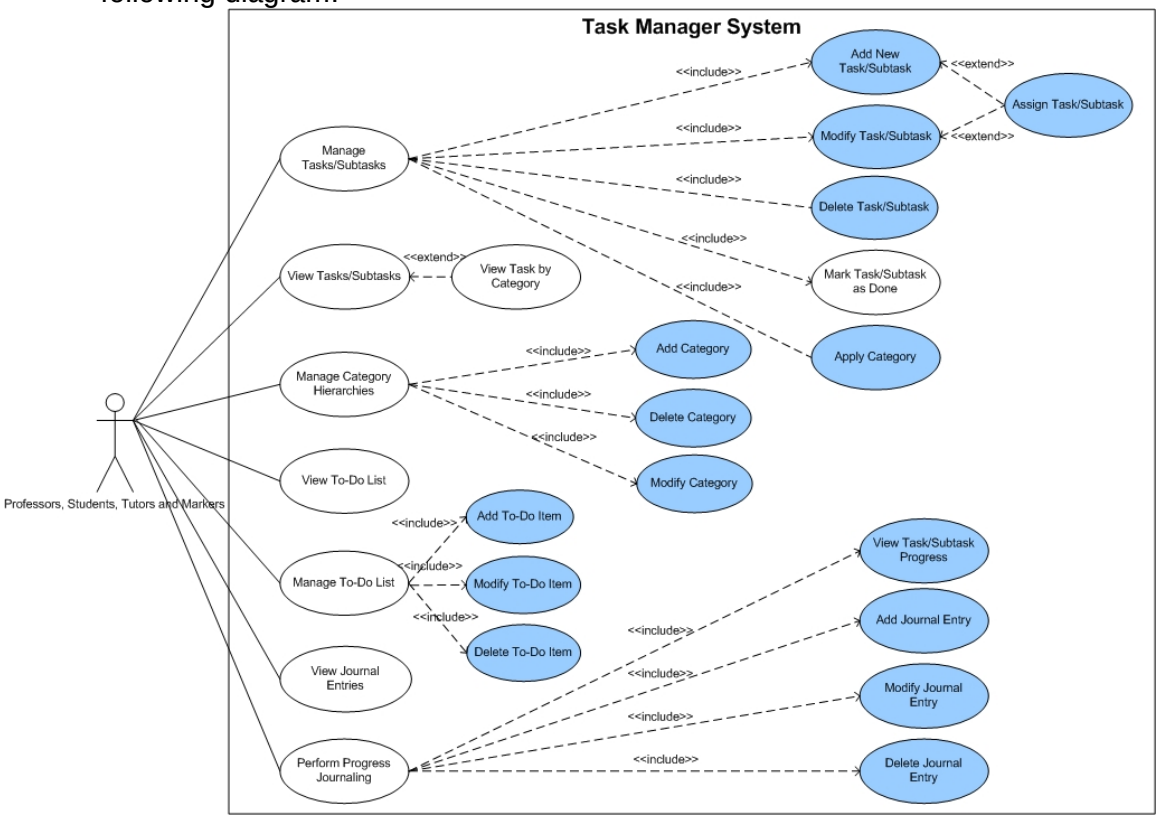


Figure 1. Architecturally Significant Use Case Diagram

4.1.1 Assign Task/Subtask

This use case occurs when a user of the system wishes to assign a task/subtask to himself or another user. The user delegates a task to a user and the assignee must be aware of the new assignment.

4.1.2 Add New Task/Subtask

This use case occurs when a user wants to create a new task/subtask. The user must be able to break down a big task into smaller, more manageable tasks and, for each task, track its progress.

4.1.3 Modify Task/Subtask

This use case occurs when users want to modify a task/subtask that they have created or that had been assigned to them by someone else. In modifying the task/subtask, the system will allow the users to enter their progress and, thereby, manage and track their advancement.

4.1.4 Delete Task/Subtask

This use case allows users to delete a task or subtask that they have created. The users cannot delete a task that they have not created. If he/she deletes a task that has subtasks, then the task and all of its subtasks are deleted too.

4.1.5 Apply Category

See group8d's Use Case Model – "Use Case: Apply a Category to a Task"

4.1.6 Add Category

See group8d's Use Case Model – "Use Case: Manage Categories"

4.1.7 Delete Category

See group8d's Use Case Model – "Use Case: Manage Categories"

4.1.8 Modify Category

This use case allows users to modify a category that they have created. They can change the category's name and specify a different location for it in the category hierarchy.

4.1.9 Add To-Do Item

This use case lets users add a new To-do item to their list of To-do items.

4.1.10 Modify To-Do Item

This use case allows users to modify a To-do item.

4.1.11 Delete To-Do Item

This use case allows users to delete a To-do item.

4.1.12 View Task/Subtask Progress

This use case occurs when users wish to view their progress for a particular task/subtask. The progress is calculated from the journal entries' inputted information.

4.1.13 Add Journal Entry

See group8d's Use Case Model – "Use Case: Manage Journal Entry"

4.1.14 Modify Journal Entry

See group8d's Use Case Model – "Use Case: Manage Journal Entry"

4.1.15 Delete Journal Entry

See group8d's Use Case Model – "Use Case: Manage Journal Entry"

5. Logical View

This Logical View of the Task Manager is used to visualize how the functional requirements are satisfied in the system.

5.1 Overview

The Task Manager follows the client-server architecture and the server's architecture is based on the layered style. The application is decomposed into six packages, which are split between three layers in the following manner:

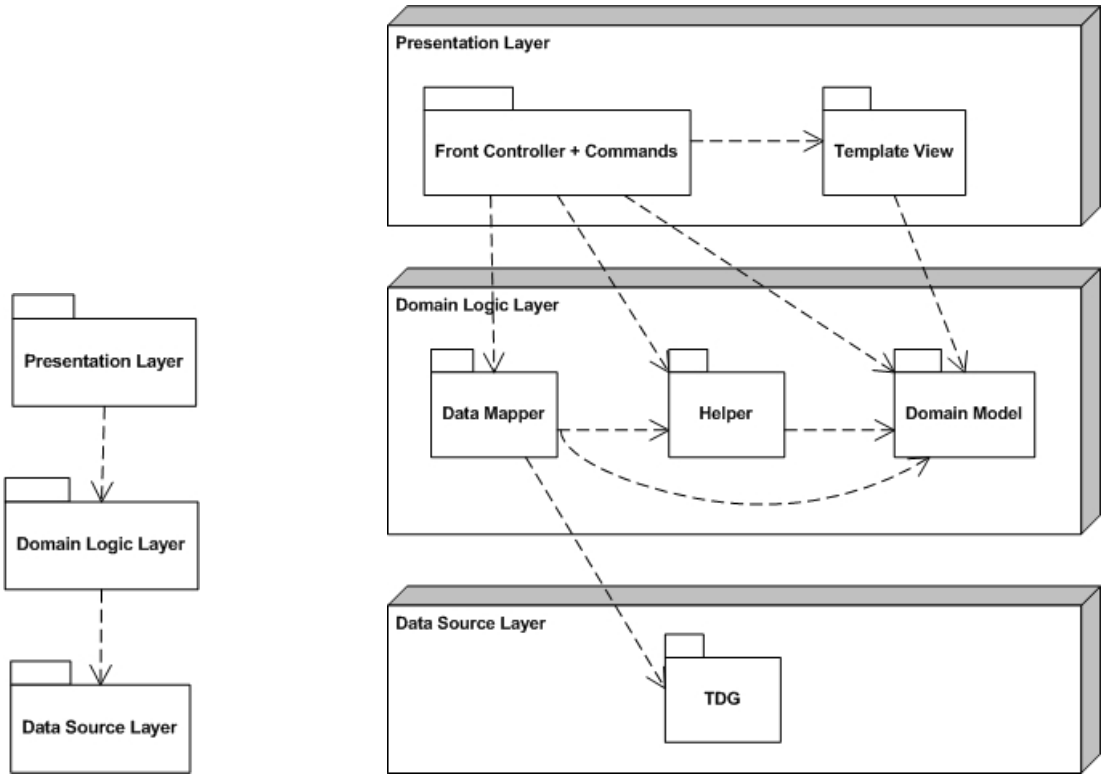


Figure 2. Logical View – Packages

Each layer is represented as a package itself.

Presentation Layer

The Presentation Layer manages the interactions between the user and the Task Manager system. It displays the information to its user, accepts commands from the user, and passes on the requests to the Domain Logic Layer. No business logic operation should be performed in this layer and this layer is only dependent on the components of the Domain Logic Layer.

Domain Logic Layer

The Domain Logic Layer contains components that are related to the business logic of the system, such as data processing and database manipulation. This layer acts as a bridge between the Presentation Layer and the Data Source Layer and is only dependent on the components of the Data Source Layer.

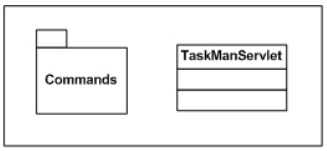
Data Source Layer

The Data Source Layer abstracts transactions with the database. This layer is responsible for storing, retrieving, and modifying data.

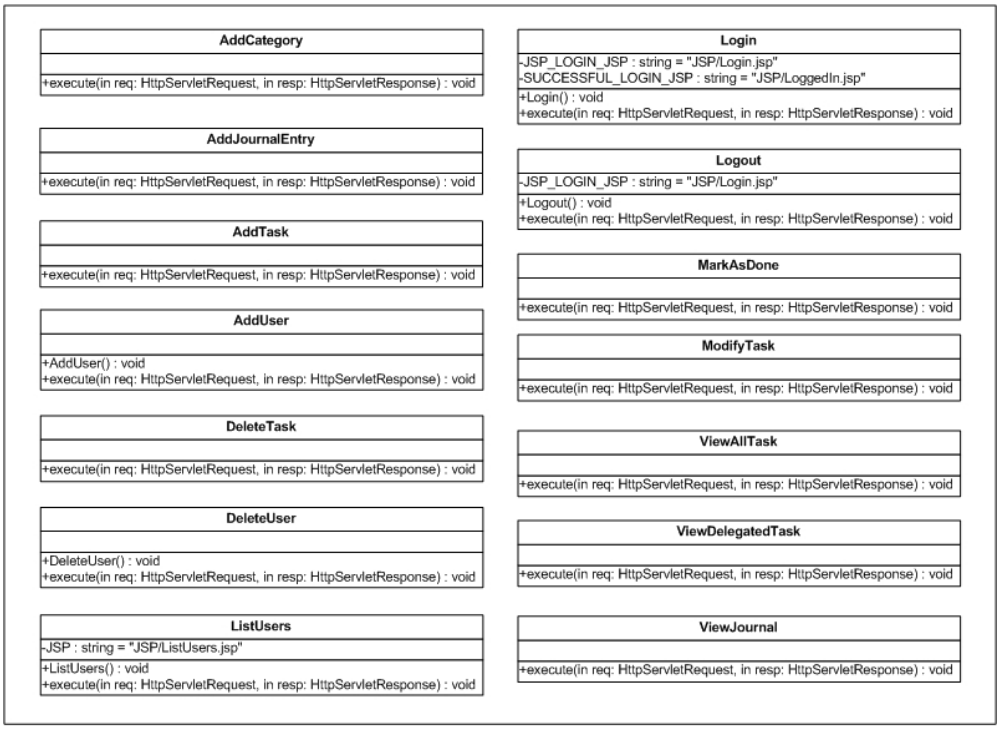
5.2 Architecturally Significant Design Packages

5.2.1 Front Controller + Commands

This package receives all requests from the Task Manager website and dispatches to the appropriate commands to handle the request.



Front Controller Package



Commands Package within the Front Controller Package

Figure 3. Contents of the Front Controller + Commands Package

Class	Description
TaskManServlet	Receive all the requests from the users and dispatches to the appropriate commands to handle them.
AddCategory	Provides the functionality to add a new category.
AddJournalEntry	Provides the functionality to add a new journal entry for a task.
AddTask	Provides the functionality to add a new task.
AddUser	Provides the functionality to add a new user.
DeleteTask	Provides the functionality to delete a task.
DeleteUser	Provides the functionality to delete a user.
ListUsers	Provides the functionality to view a list of users.
Login	Provides the functionality to log in to the Task Manager.
Logout	Provides the functionality to log out of the Task Manager.
MarkAsDone	Provides the functionality to mark a task as completed.
ModifyTask	Provides the functionality to modify a task.
ViewAllTask	Provides the functionality to view all tasks assigned to the user currently logged on.
ViewDelegatedTask	Provides the functionality to view all tasks that the user currently logged on has assigned to others.
ViewJournal	Provides the functionality to view the journal entries associated with a task.

Each command is independent from each other, but each depends on a helper object and the corresponding Domain Model and Data Mapper classes.

5.2.2 Template View

This package renders the information onto an HTML web page using JSP.

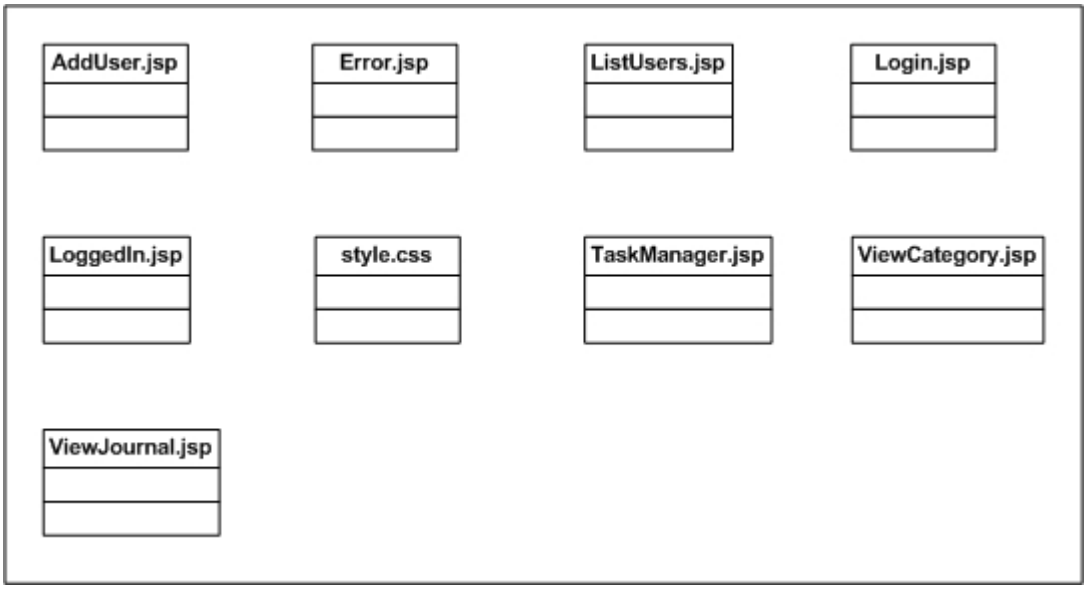


Figure 4. Contents of the Template View

File	Description
AddUser.jsp	Provides the user interface for adding a new user.
Error.jsp	Provides the user interface when an error has occurred in a transaction.
ListUsers.jsp	Provides the user interface for listing users.
Login.jsp	Provides the user interface for logging in.
LoggedIn.jsp	Provides the user interface for after a user has logged in.
style.css	Determines the style and look of each page.
TaskManager.jsp	Provides the user interface for the main Task Manager page.
ViewCategory.jsp	Provides the user interface for viewing categories.
ViewJournal.jsp	Provides the user interface for viewing a journal.

5.2.3 Data Mapper

This package acts as a go-between between the Domain Model and the Gateway or database itself by reconciling differences between the object oriented and the relational model of data.

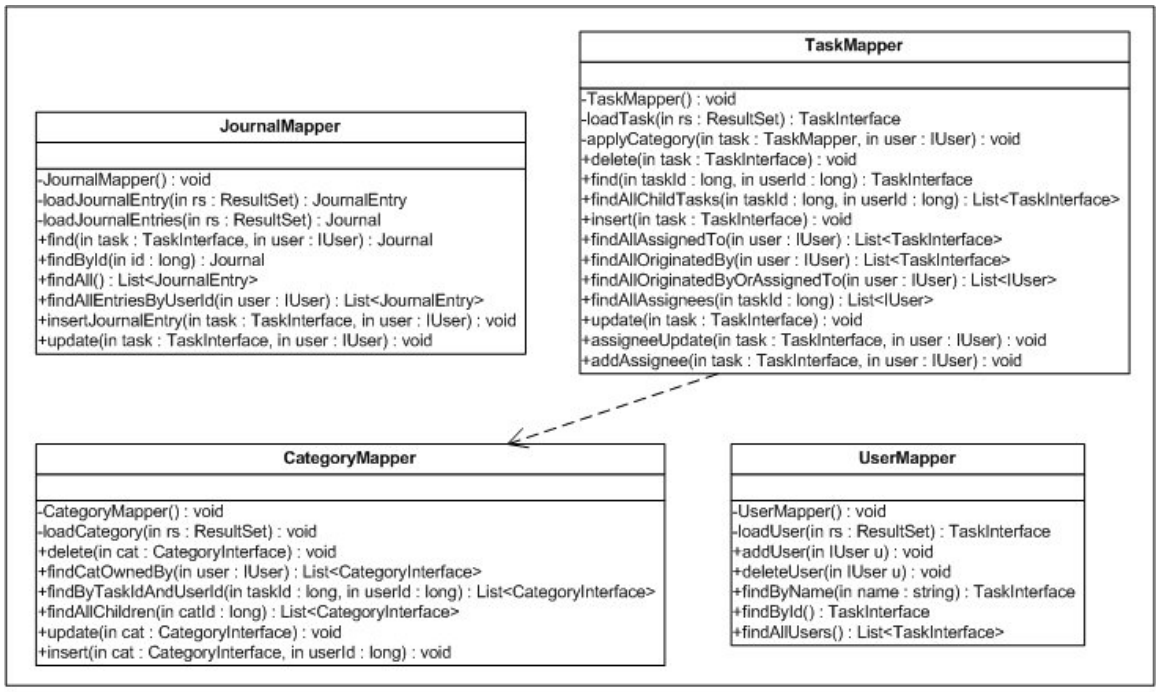


Figure 5. Contents of the Data Mapper package

Class	Description
CategoryMapper	Maps the Category objects to the Category Table Data Gateway database accessor.
JournalMapper	Maps the Journal objects to the Journal Table Data Gateway database accessor.
TaskMapper	Maps the Task objects to the Task Table Data Gateway and UserTask Table Data Gateway database accessors.
UserMapper	Maps the User objects to the User Table Data Gateway database accessor.

Each class in the Data Mapper package depends on TDG classes.

5.2.4 Helper

This package acts as a go-between between the Presentation Layer and Domain Logic Layer by wrapping one or more objects from the Domain Model into a single Helper object to ease the passing of data in the presentation.

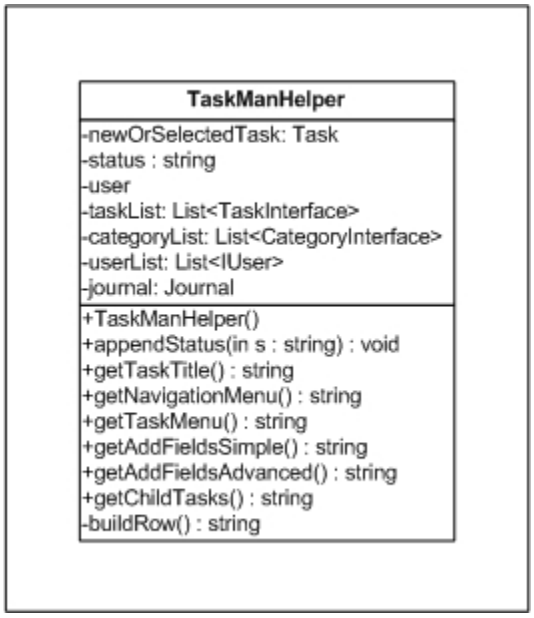


Figure 6. Contents of the Helper package[1]

Class	Description
TaskManHelper	Wraps one or more Task and User objects to be passed between the Presentation and Domain Logic Layer.

5.2.5 Domain Model

This package provides an object model of the domain in which the Task Manager operates. The model incorporates the attributes and behaviour needed into the appropriate objects.

[1] The getters and setters for each attribute of are omitted from the diagram.

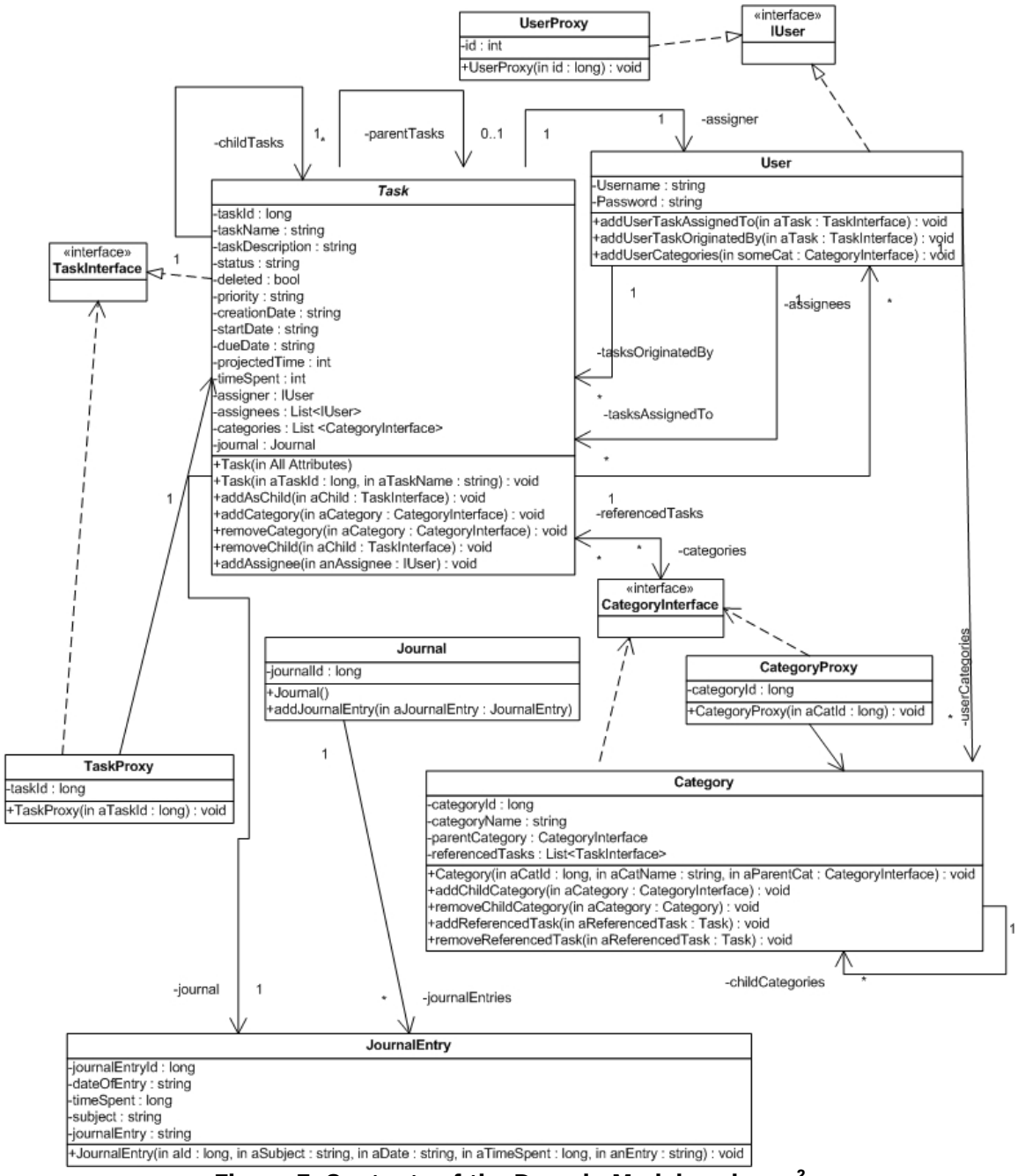


Figure 7. Contents of the Domain Model package.[2]

Class	Description
Category	Models the attributes and behaviour of a Category.
Journal	Models the attributes and behaviour of a Journal. A Journal object contains one or more Journal

[2] The getters and setters for each attribute of each class are omitted.

	Entries and is associated to one particular Task.
JournalEntry	Models the attributes and behaviour of a Journal Entry. A Journal Entry is contained within a Journal and has information concerning the progress of a Task.
Task	Models the attributes and behaviour of a Task.
IUser	Provides the interface for the User object model.
User	Models the attributes and behaviour of a User.
UserProxy	Models the attributes and behaviour of a temporary placeholder for a real User object. This is needed for Lazy Loading.

5.2.6 TDG

This package provides access to the database.

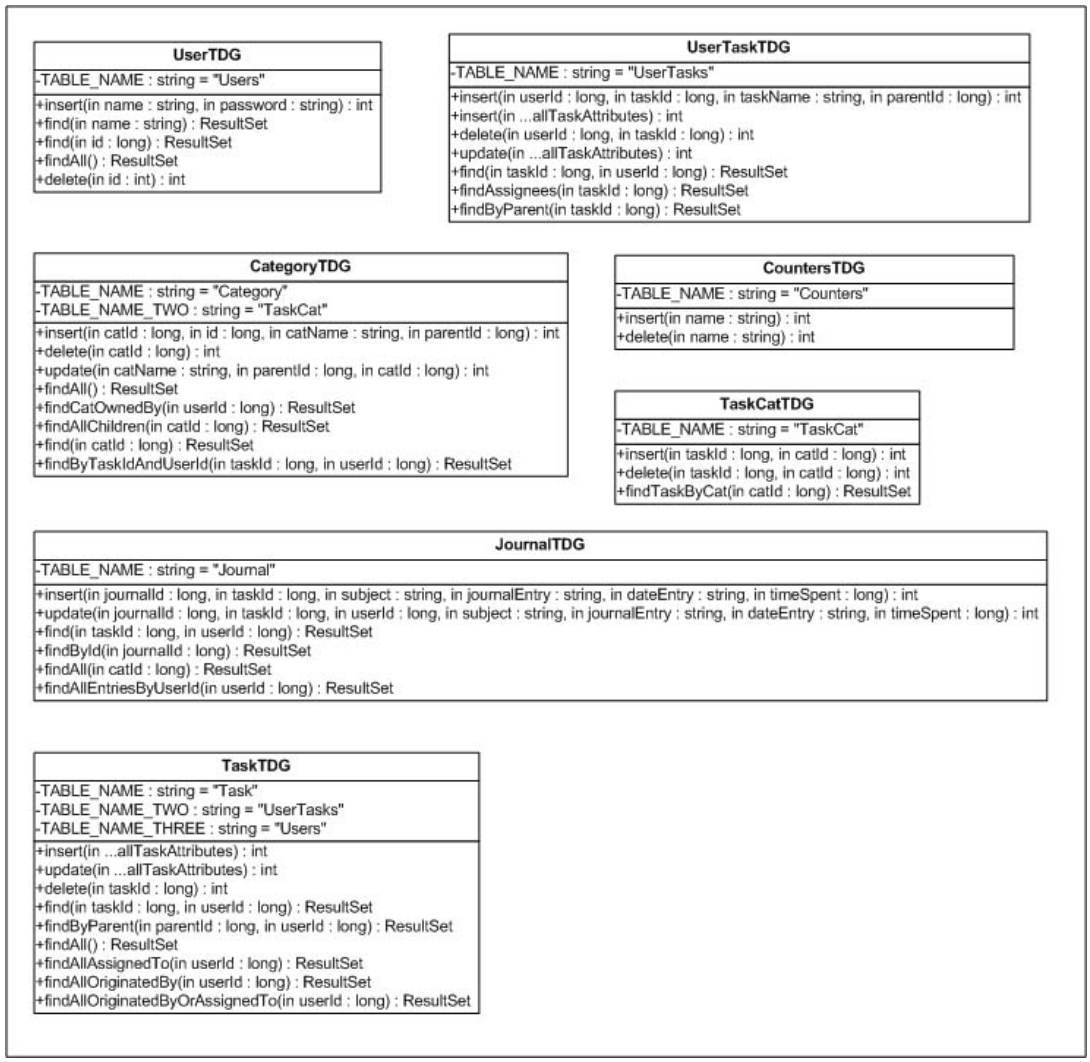


Figure 8. Contents of the TDG package

Class	Description
CategoryTDG	Provides access to the Category table.
JournalTDG	Provides access to the Journal table.
TaskCatTDG	Provides access to the TaskCat table. The TaskCat table establishes the relationship between a Task and a Category (a Task can have Categories associated with it)
TaskTDG	Provides access to the Task table.
UserTaskTDG	Provides access to the UserTask table. The UserTask table establishes the relationship between a User and a Task.
UserTDG	Provides access to the Users table.

Each class in the TDG package is independent from each other, but each is relied upon by the appropriate class in the Data Mapper package.

5.3 Use-Case Realizations

This section will describe the realization of the architecturally significant use case Add New Task, and how the packages contribute to its functionality.

Use case: Add a New Task

Refer to group8d's Use Case Model – Task Management Use Cases: Add a new Task (page 8).

In this use case, the user interacts directly with components of the Presentation Layer and that action propagates through the Domain Logic and Data Source Layers.

The user interacts with the graphical user interface (TaskManager.jsp), filling the form to add a new task and indicates to the system that he/she wants to add that new task by invoking the AddTask command through the web page.

The AddTask command constructs a new Task object and calls on the TaskMapper to insert it into the database.

The TaskMapper then extracts the information from the Task object and calls on the TaskTDG to insert the appropriate tuple into the database. If there is more than one assignee, then the TaskMapper will call on the TaskTDG to insert a tuple for each.

After successfully inserting the task into the database, the flow of the program returns to the AddTask command which will call upon the Mapper to construct an updated list of the logged in user's tasks, list of other users of the system, and

list of the logged in user's categories and store them in the helper. The command then forwards this helper to the TaskManager.jsp where the page will be refreshed with an updated list of tasks.

If the transaction was not successful, then the command would forward the control to the Error.jsp page.

6. Process View

This Process View will be used to visualize the system's decomposition into threads and processes. The Task Manager will run on Stu, a server provided to us by Concordia University. It is a multi-threaded application, where each user request via an Internet browser spawns a new thread to be handled by a single servlet. The system will ensure a certain level of concurrency control.

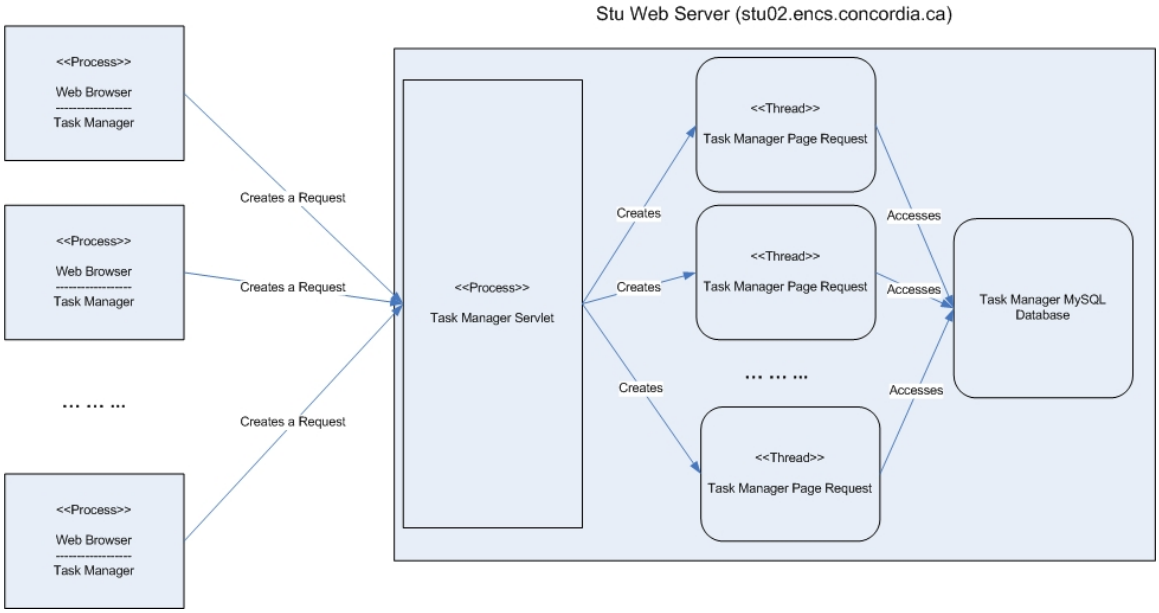


Figure 9. Process View

7. Deployment View

This Deployment View is used to show how the Task Manager application is to be deployed upon hardware and how tasks are distributed over the hardware. There is only one configuration possible for the Task Manager. The system will be deployed on the Stu server and accessed via an Internet browser from any computer that can be connected to the Web.

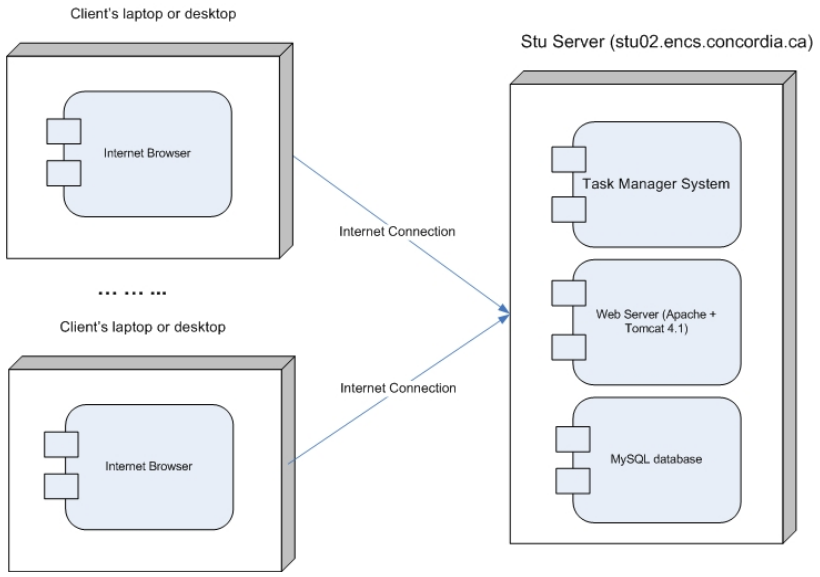


Figure 10. Deployment View

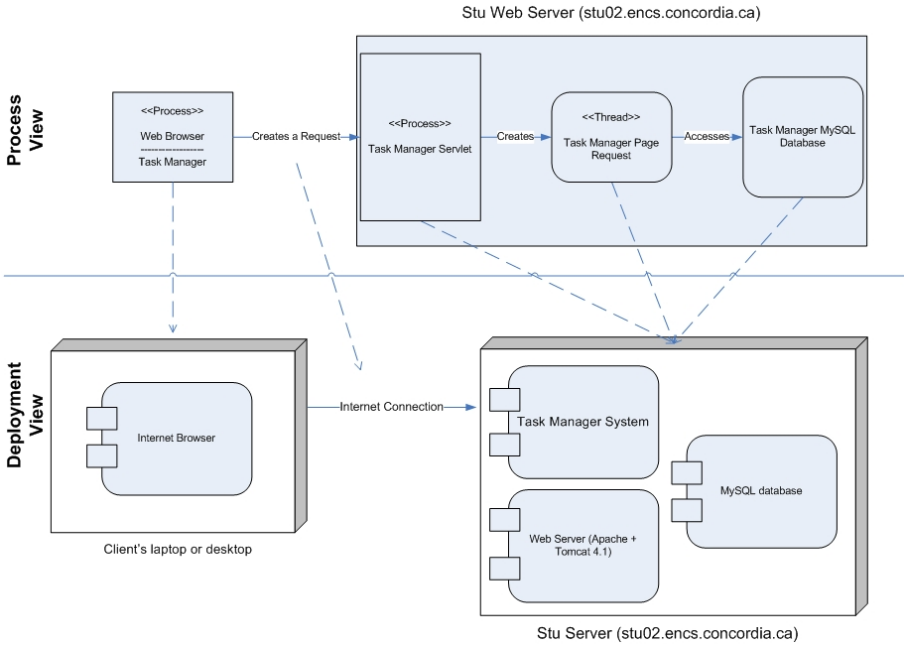


Figure 11. Mapping of the Process View to the Deployment View

8. Implementation View

This Implementation View describes the overall structure of the implementation of the Task Manager.

8.1 Client side

On the client side of the system, only the Mozilla web browsers with JavaScript enabled and supporting CSS will be supported.

8.2 Server side

In terms of third party packages, Apache + Tomcat 4.1 shall be used as web server and MySQL shall be selected as the database system.

The implementation shall be done using Java Servlet and Java Server Pages technology, in addition to using web programming languages for the interface, such as HTML, JavaScript, and Cascading Style Sheets (CSS). Various Java libraries will also be used, as needed, namely “java.util” and “java.sql”.

All three layers mentioned in the Logical View shall be represented as a different folder in the source tree and shall make use of a package that was provided to us by the teacher’s assistant called “org.dsrg” which contains miscellaneous complementary functionalities for our system, such as a Unique ID Factory. Each layer will have subfolders representing the different Enterprise Application pattern packages mentioned in the Logical View.

.java and .class files shall be kept in separate directories.

9. Data View

The database provides persistence for all data input into the Task Manager. The schema in which the relevant data is represented is as illustrated in the E/R diagram below.

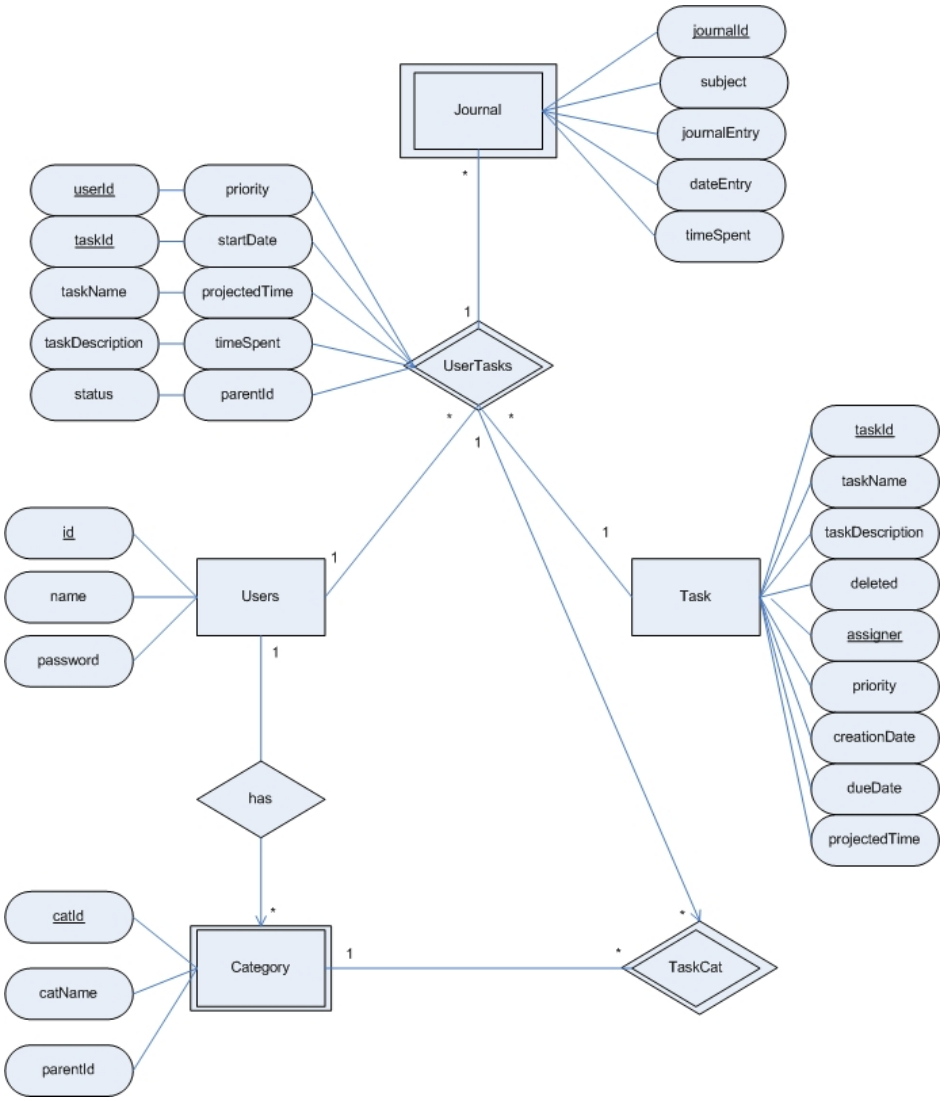


Figure12. E/R Diagram for the Task Manager's database