SAN FRANCISCO STATE UNIVERSITY

FLORIDA ATLANTIC UNIVERSITY

SOFTWARE ENGINEERING

CSC 640/848, CEN 4010

FALL 2008

INFINITYNET:
AUTOMATIC COLLABORATION METRICS

REQUIREMENTS SPECIFICATION MILESTONE 1

Authors

Group 8

Marcello de Sales (msales@sfsu.edu)
Andres Ardila (aardila1@fau.edu)
Brett Fisher (bfisher@sfsu.edu)
Gurdeep Singh (gsingh@sfsu.edu)
Marilyne Mendolla (mmendoll@fau.edu)
Mateo Mejia (mmejia11@fau.edu)

Advisors

Dr. Dragutin Petkovic (SFSU), Gary Thompson (SFSU),
Dr. Shihong Huang (FAU)

San Francisco, CA
Boca Raton, FL

Document ID

Date	09/27/2008
Responsible	Marcello de Sales
Document ID	INFINITYNET_RS_20080927_02
Location	https://ppm-8.dev.java.net/source/browse/ppm-8/trunk/www/milestones/1/INFINITYNET_M1_REQSPEC.doc

Revision History

Date	Version	Authors	Description
2008-09-28	0.1	Marcello de Sales	Template Creation
2008-09-28	0.2	Marcello de Sales	Modifying template according to specification (See Requirements documentation under Milestone 1 folder at the Directories and Files PPM-8 Java.net)
2008-10-04	0.3	Andres Ardila and Marcello de Sales	Adding information on sections 2, 3, 4, 5, 6.

Table of Contents

EXECUTIVE SUMMARY

Software Engineering is one of the fastest growing fields today's global economy. This rapidly-evolving industry requires educational institutions to adopt innovative changes in their curriculums in order to provide students with the necessary competitive skills and experience required in today's job market. Precisely due to this globalization, more and more companies are developing software in virtual, distributed environments that transcend national boundaries. As a consequence, educational institutions have redesigned their programs to simulate team software development in a controlled environment. This evolution in the pedagogy of Software Engineering, however, presents new challenges to instructors regarding student evaluation. It is therefore of paramount importance to provide instructors with the proper tools to assess team and individual progress, and to assess individual student participation within those teams in these new virtual classroom environments.

In light of these challenges, InfinityNet will provide cutting-edge and unique tools to instructors who choose the Java.net development platform as part of a Software Engineering class curriculum. InfinityNet will allow instructors to monitor the participation of individual students in globally and/or locally distributed Java.net group projects as well as the progress of the team as a whole. InfinityNet will be an efficient, automated, and user-friendly web application that collects and reports on student participation metrics for such class settings. The application allows an instructor or set of instructors in different locations (i.e. concurrent Software Engineering courses taking place in different universities) to automatically collect participation data for a set of Java.net projects and report on its members' contribution to the project at all stages of development. Because InfinityNet understands the need for intuitive, easy-to-use software in academia, the initial set-up will automatically generate the metrics categories available for the given projects, whilst allowing instructors to retain full control to modify these settings at any point throughout the semester.

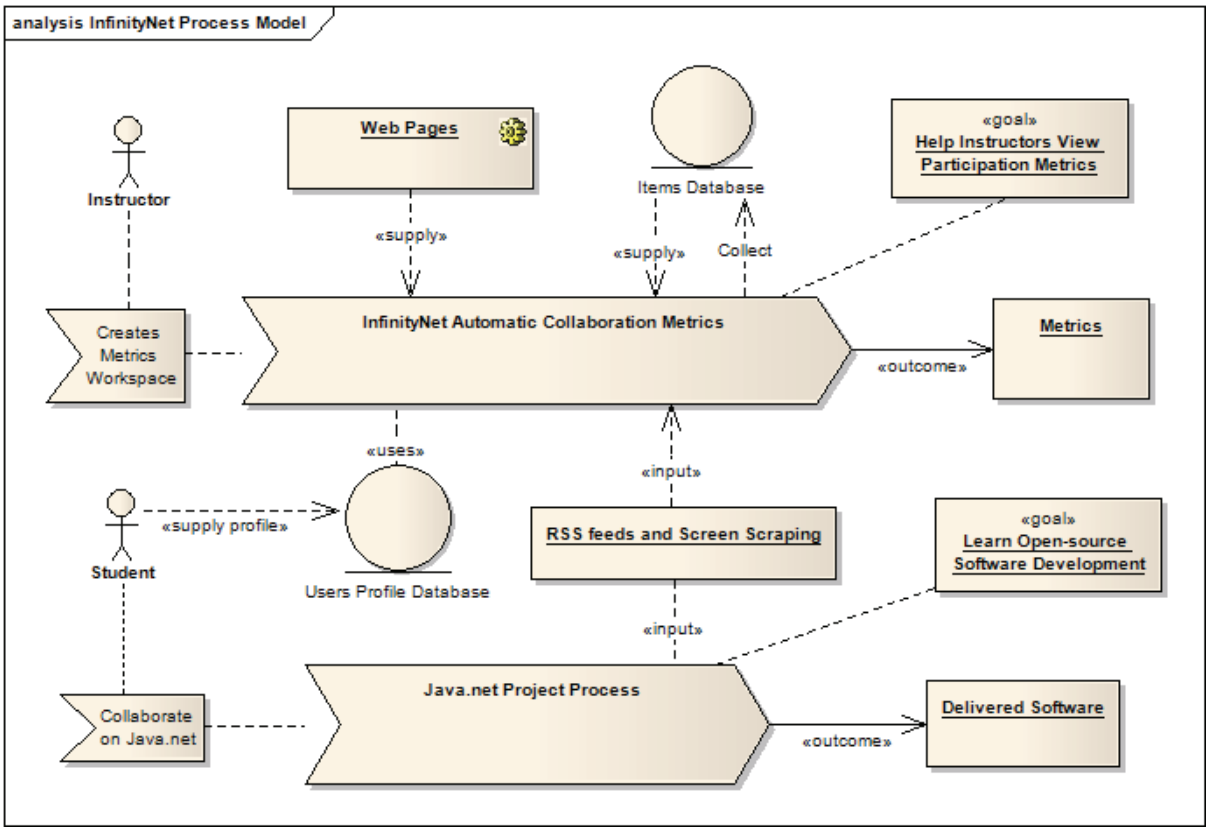
We believe that information is power. That's why our system will provide instructors with the ability to collect and view a wide ranging variety of metrics from Java.net projects. It will also provide instructors with insightful information about trends and common difficulties that may arise in the development and learning processes. Our tool's seamless integration with the Java.net platform, the valuable data it can extract and present (from granular to global), and its minimal maintenance overhead will allow instructors to focus their efforts in the most important aspect of their task: teaching. InfinityNet will thus enable and improve the quality of learning for students by providing instructors with the tools needed to be heavily involved in the progress of the class and its individuals.

1. Business Process

The business process defines the information about the overall project, gathering necessary information about the understanding of the problem to be solved with our system. The business process is described as a high-level understanding of the system requested by Gary Thompson, and the domain vocabulary is described as the team captured both the high-level requirements and in-class information about the system.

As a matter of understanding the idea of the product, the business process model captures the main business activities, focusing on the inputs, outputs, goals and key events that drive the process. The goal of InfinityNet is to help instructors view participation metrics while students are learning software development on Java.net.

We can describe InfinityNet as having two processes: one started by an instructor, and the other by students. Instructors create a configuration Workspace with the list of projects from Java.net and participating members, while students provide their initial profile. As students start collaborating on Java.net and, as a result, generating RSS feeds and additional information, these metrics are collected and stored by InfinityNet. At this point, InfinityNet uses web pages to display these metrics in rich and meaningful ways to the instructor.



2. REQUIREMENTS SPECIFICATIONS

This section describes the requirements gathered from the high-level specification documentation provided by Gary Thompson. Additionally, the wiki page for the PPM project was explored. The *Formal Requirements* catalogues the categorized specification for each component to be developed. On the other hand, the *Non-Functional Requirements* section lists generalized specifications. Finally, the *Use Case Model* groups these requirements and associates them the functionalities they fulfill.

As the reader will notice, the functional and non-functional requirements are specified by means of *User Stories*, following the formal conventions and best practices of the Agile Development community, where the stories remind us of a conversation with the client.

2.1. FORMAL REQUIREMENTS

Metrics Workspace	
ID	**Description**
US002	As an Instructor, I would like to track the Event of Code Repository, so that I know how many commits each student or the group made.
US003	As an Instructor, I would like to track the Event of Tracking System, so that I know how many changes to the Issue Tracker the group made
US004	As an Instructor, I would like to track the Event of Emails sent to the available Mailing Lists, so that I know how many emails the group has sent.
US005	As an Instructor, I would like to track the Event of Posts sent to Discussion Forums, so that I know how many posts the team made.
US006	As an Instructor, I would like to track the Event of Documentation changes, so that I will be able to see how many documents were uploaded.
US007	As an Instructor, I would like to track the Event of Disputes or others on the teams, so that I will know what happened to have a given team changed.
US010	As an Instructor, I would like to be able to monitor events, so that I know the participation of each group.
US011	As an Instructor, I would like to view the Events by groups, so that I know the participation of the groups.
US012	As an Instructor, I would like to analyze Events from the groups by their event category, so that I know the participation of groups.
US014	As an Instructor, I would like to manually add customized group interaction Events (i.e. team disputes), so that I can track different Events.
US015	As an Instructor, I would like to set up a Project Metrics Configuration Workspace so that I can have a set of projects or groups together.
US016	As an Instructor, I would like to create a Report for a Metrics Configuration so that I know the high-level details of the project participation.
US017	As an Instructor, I would like to share a Metrics Configuration so that my peer instructors can analyze the Reports with me.
US018	As an Instructor, I would like to filter the students from my institution on a given Configuration, so that I know which ones are active or not.
US019	As an Instructor, I would like to list the Java.net username of each group member on a given Configuration, so that I can identify each of them.
US022	As an Instructor, I would like to see each student's profile, with their Real Name, Student ID, Group Number, if he/she is a team leader, etc, so that I know who he/she is and what his/her role is.
US023	As an Instructor, I would like to send emails to a given Project alerting about students who haven't completed their profile information.
US024	As an Instructor, I would like to add, remove and modify the Configuration throughout the configuration life-cycle, so that I can delete an old configuration data or edit information.

US025	As an Instructor, I would like to view Reports in a tabular format, so that I can see each event category.
US026	As an Instructor, I would like to view Reports in a graphical format, so that I can the big picture of the data
US027	As an Instructor, I would like to export Reports to flat text files in delimited format, so that I can use spreadsheet software (i.e. OpenOffice Spreadsheet, Microsoft Excel, Good Docs Spreadsheets, etc.) to visualize or perform operations on the data.
US031	As an Instructor, I would like to be able to read a Report comparing all the projects in a Configuration in a tabular way, so that I can compare each of them.
US032	As an Instructor, I would like to read a Report for a single project in a tabular format for all the categories, so that I know the participation figures on each of the categories.
US033	As an Instructor, I would like to read the Report for a single Event category for a given Configuration, so that I can compare the groups on that category.
US034	As an Instructor, I would like to read the Report comparing all the projects in a Configuration in a graphical format, so that I have the overall picture of the groups on each event category.
US035	As an Instructor, I would like to read the Report for a single project in a graphical format, so that I have an idea of the big picture of the project participation on each event category.
US036	As an Instructor, I would like to read the Report on a single event category in graphs for the groups on a Configuration, so that I have the idea of the big picture of each for the event category.
US043	As an Instructor, I would like to browse for projects in a given Configuration, so I can easily see the list of the groups in a given Configuration.
US044	As an Instructor, I would like to browse for Event categories in a given Configuration, so that I know which events I am tracking in a given Configuration
US045	As an Instructor, I would like to search for projects in a configuration, so that I don't need to memorize names.
US046	As an Instructor, I would like to search for categories in a Configuration.
US050	As an Instructor, I would like to add the list of team members to each of the projects on a Workspace Configuration, so that I will be able to know where students will be allocated.
US051	As an Instructor, I would like to import the list of team members from a delimited file into a configuration, so that I can easily set up a Configuration.
US053	As an Instructor, I would like to see the top 5 Projects based on participation by the sum on categories or each of them.
US058	As an Instructor, I would like to pause or make inactive the Metrics Workspace, so that no event items are collected when the project has finished.

User Management	
ID	**Description**
US020	As a user, I would like to identify myself into the system by using my Java.net username and password, so that I don't need to create more user accounts
US021	As an Instructor, I would like to see the real name for each of the students on the reports, so that I don't have to memorize the Java.net username for them.
US039	As a user, I would like to login into the system with my username and password created in the registration.
US052	As an Instructor, I would like to view my account detail, so that I can edit it as needed.
US054	As an Instructor, I would like to have the student to associate his username from Java.net with his Real Name, Student ID, School he/she belong to, the group he/she belong to, so that I will know who he/she is.
US055	As a user, I would like to retrieve my lost/forgotten password by email or web browser, so that I don't need to email the webmaster/support.
US59	As a user, I would like to change my profile information.
US060	As a user, I would like to receive an email after registering on the website, so that I will confirm I am registered into the system.

Personal Agent	
ID	**Description**
US047	As an Instructor, I would like a Personal Agent to use my Java.net username and password to parse and extract the RSS feeds from each project's events from Java.net so that I can view the consolidated data as metrics.
US048	As an Instructor, I would like to see the students' login activities such as number of visits to Java.net and the last time he logged in, so that I know how his/her participation through the system
US049	As an Instructor, I would like my personal agent to projects automatically go to the parent project and load the project names for a new configuration, so that I don't need to enter the names of the groups for the term manually.
US057	As an Instructor, I would like my personal agent to verify the summary of the RSS posts on a given project has been changed, so that I don't spend time waiting for all posts to be collected.

2.2. Non-Functional Requirements

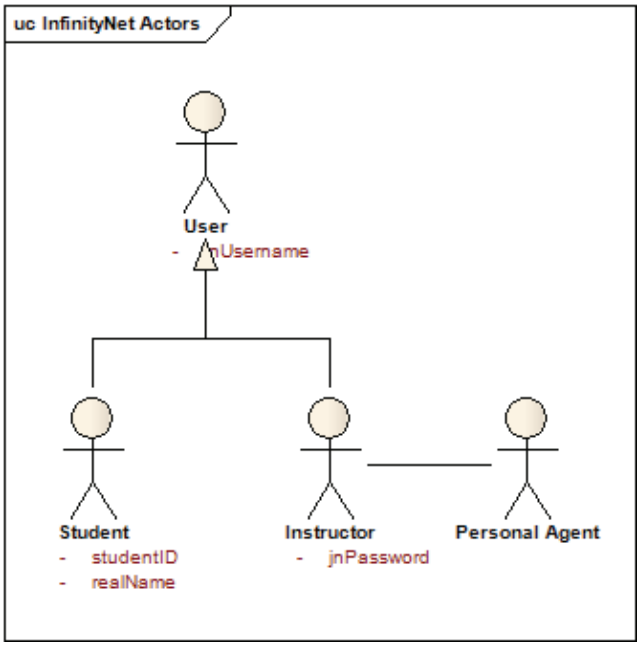
General Application	
ID	**Description**
US001	As an Instructor, I would like to access to the system using a web browser, so that I don't need to open/install any additional software.
US008	As an Instructor, I would like the system to browse each project's website to collect events' data automatically.
US009	As an Instructor, I would like the system to store the RSS feeds collected on a database, so that I can search and browse the data on the web.
US013	As an Instructor, I would like to have the tool working on the Java.net collaborative platform, so that I can view the participation of the members on a set of projects.
US028	As an Instructor, I would like to use the system without any additional training.
US029	As an Instructor, I would like to use the system without any knowledge of web application development or Database systems, such as writing SQL.
US030	As a User, I would like to take advantage of high-speed Internet connection (DSL or cable), so that I can view rich, meaningful information quickly.
US038	As a User, I would like to know that my username and password are encrypted into the database, so that I'm sure they are secure.
US040	As a User, I would like to view an online help section in the website, so that I can simply send email to the webmaster whenever needed.
US041	As a User, I would like to have the system respond to my input in less than 3 seconds.
US042	As a User, I would like to navigate through the system under a secure connection and see the lock icon on my browser, so that I can feel secure providing personal information
US061	As an Instructor, I would like to have the students authenticated on Java.net, so that I am certain that the students are the real person.

2.3. Use Case Diagrams

This section describes the use case diagrams for InfinityNet. It represents the fulfillment of each of the User Stories into the components as a given functionality. First, each actor is described and shown in the *Actors* UML diagram, and the following sections describe the use cases for each component listed in the *Requirements* section: Users Management is responsible for the management of users and their profiles; Metrics Workspace is the area used by the instructors to create the metrics configuration, and finally, the Personal Agent API is the set of functions that the instructor's Personal Agent will perform on his/her behalf.

2.3.1. ACTORS

User	This is the regular user of the website. It can make login using username and password.
Instructor	Instructors are the Java.net users who create the metrics. They will be providing Java.net username and password to be able to navigate through the private projects.
Student	Students will just fill out forms about their personal information, institution and group.
Personal Agent	The instructor's personal agent is in the form of a web crawler that represents the instructor and virtually navigates Java.net to collect information.



2.3.2. USER MANAGEMENT

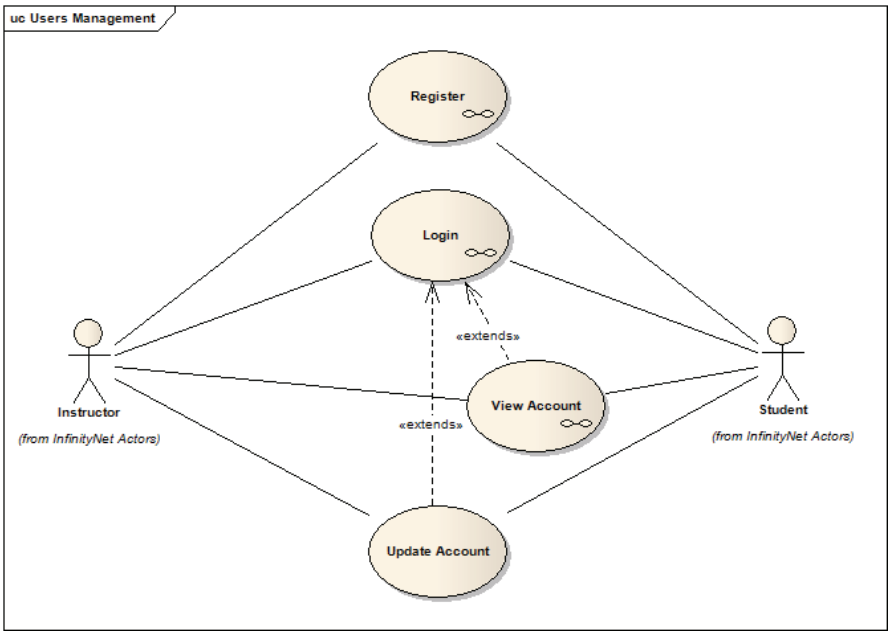
The User Management component expresses to how the main actors of InfinityNet will use the system to create an account, register into the system and modify their information. This component is the first section of InfinityNet available to the customers.

UC001	Register
Description	An Instructor or Student registers into the system.
Requirement Traceability	US020, US037, US038, US054, US001, US028, US038

UC002	Login
Description	An Instructor or Student logs in into the system.
Requirement Traceability	US020, US039, US001, US060, US001, US028

UC003	View Account
Description	An Instructor or Student views his/her user information.
Dependency	UC002
Requirement Traceability	US052, US001, US028

UC004	Update Account
Description	An Instructor or Student updates his/her account information.
Dependency	UC002
Requirement Traceability	US021, US059, US001, US028



2.3.3. METRICS WORKSPACE

Metrics Workspace component is responsible for the metrics management by the instructors. It includes regular activities such as create new Metrics Workspace, view metrics for a given Workspace, view metrics for a given project in a Workspace, and export the metrics Workspace.

UC005	Create Metrics Workspace
Description	The instructor creates a new metrics workspace for a given set of projects already registered on Java.net, identifying it by a name (term name).
Dependency	UC002
Requirement Traceability	US015

UC006	View Workspace Metrics
Description	The instructor views the metrics for the set of projects of a given workspace by choosing the name chosen.
Dependency	UC002
Requirement Traceability	US002, US003, US004, US005, US006, US007, US010, US011, US012, US025, US026, US031, US032, US033, US034, US035, US036, US008, US009, US013, US047, US057

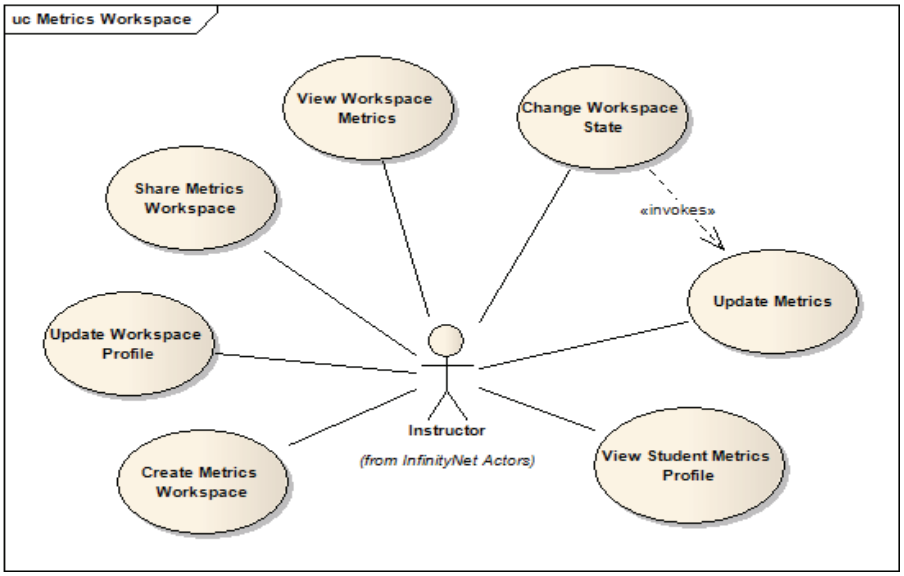
UC007	Update Workspace Profile
Description	The instructor updates any information on a given workspace such adding or removing projects on the workspace, renaming the workspace, etc.
Dependency	UC006
Requirement Traceability	US014, US024

UC008	Share Metrics Workspace
Description	The instructor shares the metrics workspace with another instructor by sending him/her an invitation.
Dependency	UC006
Requirement Traceability	US017

UC006	View Student Metrics Profile
Description	The instructor views the metrics from a given student in a given workspace. This includes his real name and the number of participation on all the event categories.
Dependency	UC006
Requirement Traceability	US022, US21, US054

UC007	Update Metrics
Description	The instructor updates information on a given workspace.
Dependency	UC006
Requirement Traceability	US007, US014, US050, US051

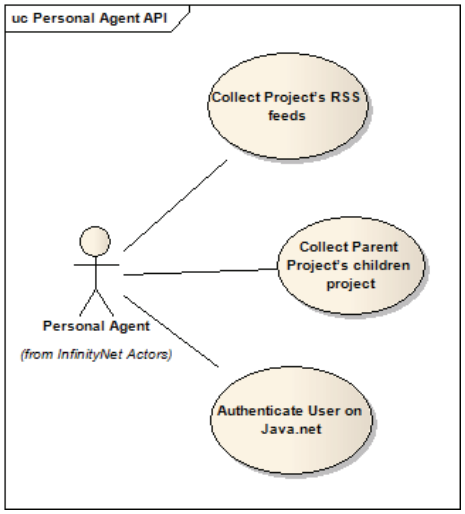
UC007	Change Workspace State
Description	The instructor changes the workspace state.
Dependency	UC006
Requirement Traceability	US058



2.3.4. Personal Agent API

The personal agent API will be responsible for the collection of data from Java.net. The personal agent will represent the Instructor on the web, and will be navigating Java.net him/her, using his/her personal username and password.

For most of the functionalities that depends on Java.net, personal agents can be used to ensure that the data is valid. For example, if different students can hack the system by registering their username as someone else. In order to prevent such malicious activities, the personal agent will always verify the Java.net related information.



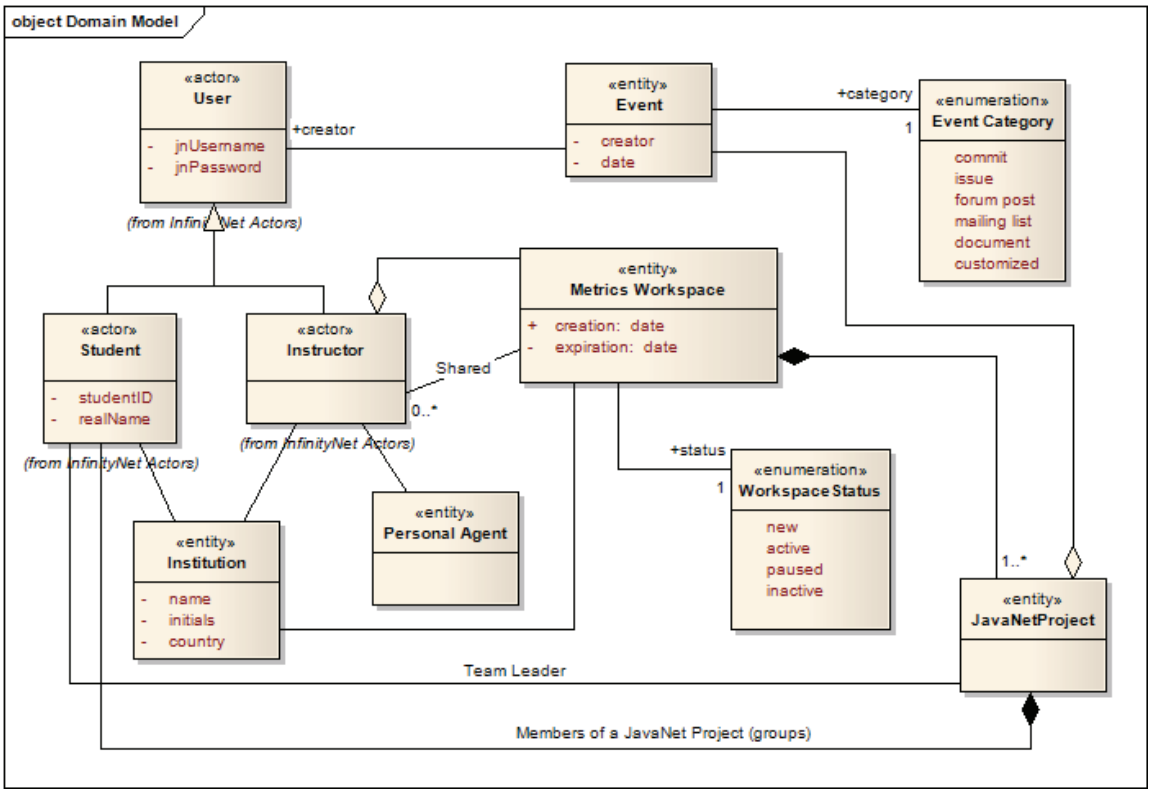
UC008	Collect Project's RSS feeds
Description	The instructor's personal agent logs into a given project and collects RSS data feeds from all the available events such as mailing lists and forums.
Requirement Traceability	US002, US003, US004, US005, US006, US007, US008, US010, US047, US057

UC009	Collect Parent Project's children project
Description	While creating a new Workspace Profile, the instructor's personal agent logs into the given Java.net parent project and collects the names of all children projects.
Requirement Traceability	US015, US049

UC010	Authenticate User on Java.net
Description	While creating a new user on the system, the personal agent can use the Java.net username and password from the user to authenticate him/her on Java.net.
Requirement Traceability	US061

2.4. Domain Vocabulary

The Domain Model defines a consistent, common vocabulary across a project. After analyzing the requirements and use cases, the entities and their relationships are defined below. The domain vocabulary can be thought of as the first abstraction of the high-level class diagram.



User	Derived from the Actors, a user has the username and password from Java.Net. It will be used by the specialized users Instructor and Student for identity verification.
Student	Students play a secondary role on InfinityNet where he/she just fills out the participation form with his/her personal information. Also, he can be identified as a team leader in any Java.net project and is always tied to a given institution.
Instructor	Instructors are aggregated by their Metrics Workspaces for different terms on a given institution. They also can participate on the ownership of a given Workspace.
Personal Agent	The instructor's personal agent is the internet crawler that uses the professor's username and password to navigate through Java.net on his/her behalf in order to get metrics information and find out more about the projects.
Metrics Workspace	The Metrics Workspace is composed by a set of Java.net projects names and it can assume different states during the metrics workspace life-cycle.
Java.net Project	The Java.net project space name that will be tracked. It is composed of a set of Java.net usernames in the form of the student names.
RSS Event	Each item on a Java.net project RSS feed is called an RSS Event. Each event is part an event category such as the commit message, an issue, and a topic posted in a discussion forum or an email sent to a given mailing list. Finally, it contains a reference to the creator of the item, who can be a student or a professor. This information is associated with the Java.net username used in the RSS 'creator' tag.
Event Category	This is an enumeration of the types of events, or from which category the RSS item originated. It will be associated with each event and will also include the customized values of events such as disputes on the team.
Workspace Status	It is associated with the Metrics Workspace. Active is defined by default when the instructor creates the workspace environment. In that way, the instructor can tell his Personal Agent to collect metrics online.

3. User Interfaces — Mock-ups

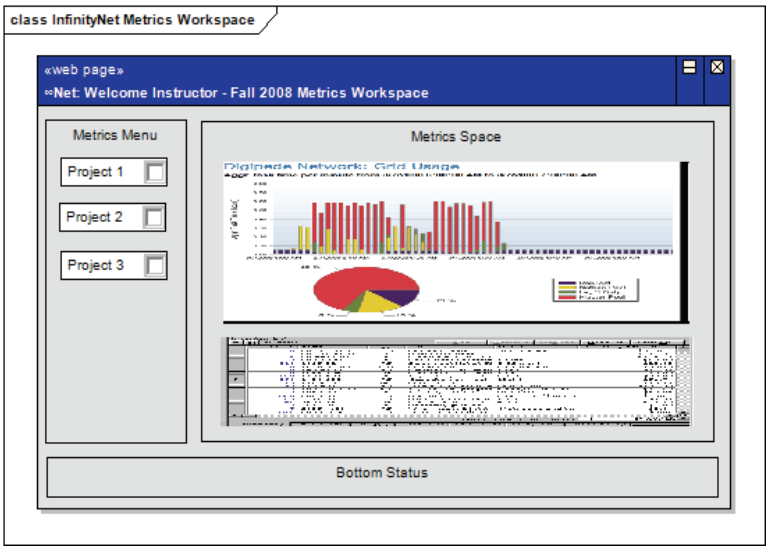
As user interfaces goes, InfinityNet will provide the regular registration forms and login forms as any other web application, containing their own set of required field for the type of user: instructor or students. However, the most important aspect of the application is the instructor's metrics workspace.

3.1. Metrics Workspace

The Metrics workspace consists of the top bar, a left-side navigation menu and the main area in the center of the screen.

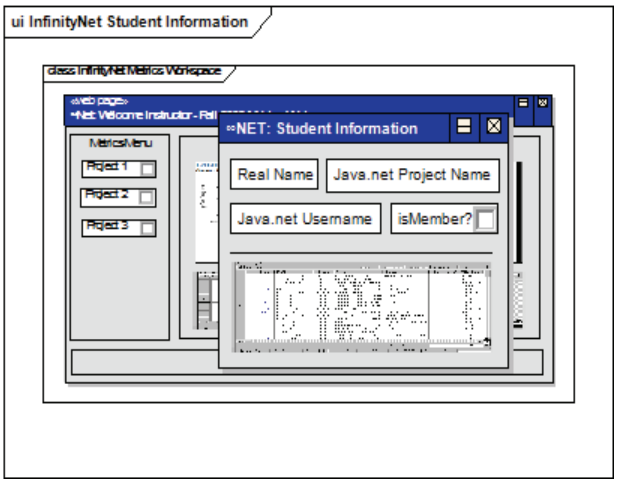
The top bar contains the instructor's name, the name of the current workspace being viewed and the name of other instructors with whom the Workspace is shared.

The left-side navigation menu lists projects contained in the current Metrics Workspace, as well as general related functional items such as edit list, download events data file, search student, etc.



Finally, the center are contains the metrics in tabular and the graphical representation, from the aggregated events data for the general set of projects. When one of the projects is clicked from the left-side navigation, the tabular data and graphical representation are changed accordingly.

3.2. Student Information for Instructors



When the instructor clicks on a given student, he/she is shown the students' participation metrics in the current workspace metrics opened. Note that the information displayed shows his participation in the project only. It shows his list of team members and which one is the team leader. Also, it shows statistical information about the overall performance in the current project on the workspace for all the configured event categories such as commits to repository, participation on issues and posts on discussion forums.

4. High-level System Architecture

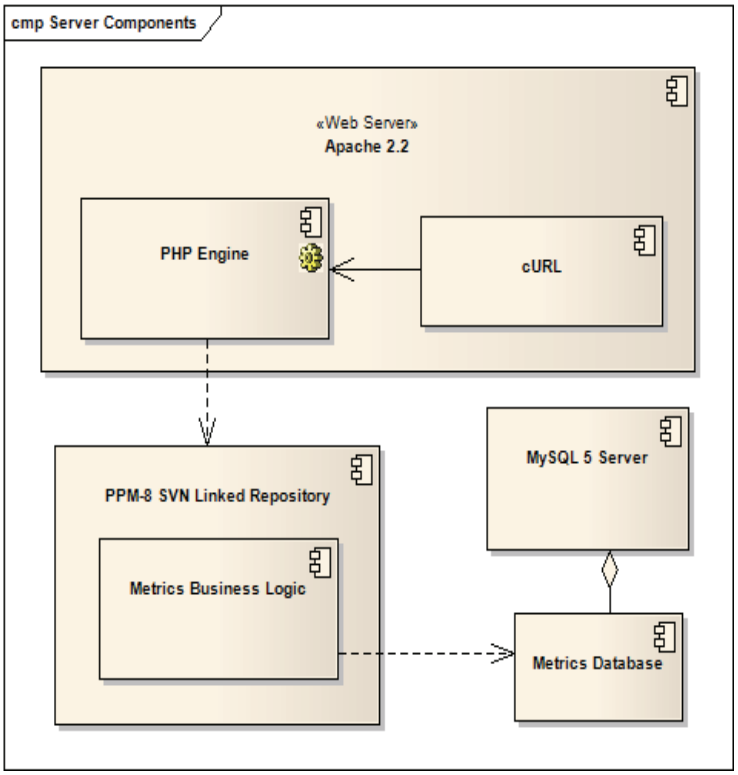
4.1. Web Server and Web technology

The font-end of the InfinityNet implementation consists of an extensive User Interface, constructed mainly in PHP and perhaps Ajax. InfinityNet will be redesigning and expanding on the core functionality of the StatsWidgets Project, and in doing so, will work to bring this functionality up-to-date with current technology by implementing a well-constructed, web-based user interface along with its back-end crawler. In order to offer most of the features planned, the server will feature an Apache Web Server, along with additional PHP modules. The most important additional module is called cURL, which can be used as an HTTP client.

As for the PHP source-code, it will be deployed to the SFSU T-2000 server using Subversion infrastructure during the development process. In this way, a given revision **must** be tested and selected to be deployed. This revision includes the core components for the Metrics Business logic, with the intent to make use of additional PHP frameworks.

4.2. Database Server

In order to overcome the short-comings of RSS feeds where older data drops as new data is acquired, the back-bone of InfinityNet system will contain a MySQL database for information storage. In this way, the metrics data will be categorized in a meaningful way to be later retrieved. The MySQL server consists of the version 5, and will use advanced stored procedures and triggers during implementation.



At the end of a given semester, or upon project completion, the data that was collected and stored will not be discarded. This will allow instructors not only to maintain access over the project life-cycle and eventual project completion, but into the future, where analysis and comparisons can be made between different Workspace Configurations in order to infer trends or the effects of changes in the structure of the course by student participation and progress in the projects.

5. COMPETITIVE ANALYSIS

Despite its potential wide-ranging applications, research on the automation of collecting participation metrics of Java.net projects by the community at large has yielded little significant results. In fact, GlassFish, one of the largest and most widely used projects on Java.net[1] has no automated participation metrics system[2]. While StatsWidgets provides the limited functionality of outputting delimited text files of a given RSS feed (or a set of them) from public and/or private projects, this implementation provides little usability for the intended audience of our system because it would require instructors to set up, configure and maintain a database to handle the output files generated by StatsWidgets. Since the requirements explicitly state that users should have no such prior knowledge of database systems, StatsWidgets proves inadequate in this regard. In addition, not all metrics generated by a project are traceable using RSS, and so the need for a tool that will collect metrics on these non-RSS participation metrics is essential. We aim to automate not only the retrieval of the RSS feeds, but the storage in a database that will enable presentation in rich and meaningful ways to the instructor.

Notwithstanding StatsWidgets' differing intended audience, the similarity of its core functionality makes it a competitor to our system. Following is a side-by-side architectural and functional comparison of the two systems.

Competitive Analysis – Financial and Technical					
Product	**Open-Source**	**Cost**	**Platform**	**Persistence**	**Modules**
∞Net	Yes	Free	PHP 5.2	MySQL	PHP, cURL
StatsWidgets	Yes	Free	JDK 2	Text Files	Kosuke Scraper

Competitive Analysis – Functional		
Functionality	**∞Net**	**StatsWidgets**
Provides Metrics for Public projects	✓	✓
Provides Metrics for Private Projects	✓	✓
Store Metrics in a Database System	✓	
Provides Web Interface for Users	✓	
Provides Tabular Data to Users	✓	✓
Provides Graphical Data to Users	✓	
Export Delimited Data	✓	✓
Saves Configuration	✓	

[1] http://community.java.net/projects/top.csp

[2] http://wiki.glassfish.java.net/Wiki.jsp?page=CommunityStatistics

6. RESOURCES ALLOCATION

Since the team will be following an Agile/Scrum process, the team will be cross-functional, with all the members being developers, and the team leaders as product owner. More detail is presented in the following table, as well as the roles each of us will be playing in the team.

Team Member	Role
Marcello de Sales	Team Leader, Scrum Master, Product Owner, Cross-functional Developer
Andres Ardila	Product Owner, Cross-functional Developer
Brett Fisher	Cross-functional Developer
Gurdeep Singh	Cross-functional Developer
Marilyne Mendolla	Cross-functional Developer
Mateo Mejia	Cross-functional Developer