MS/CS Project Proposal
Web-based Distributed EJB BugTracker
An integration of JSP/Java-servlets, EJB, application server, and relational database

Student: San Htun Aung
sha5239@cs.rit.edu
December 2, 2002

[1] Abstract:

J2EE — Java 2 Platform, Enterprise Edition based application servers are increasingly becoming the engine rooms of web enabled distributed applications. They provide scalable, high-performance Java infrastructure for processing many simultaneous requests from users for Internet application services. They also offer a consistent design and deployment model, and various software components that aim to the development of complex internet-based applications.

In this project, I will gain first-hand knowledge of using two open source servers: Apache Tomcat and JBoss 3 Application Server – a standards-compliant, J2EE application server.

Second, focus on implementing the following new Enterprise JavaBeans technologies *(EJB 2.0 specification)* – that was released on August 2001.

1. A new Container-Managed Persistence – *CMP 2.0* – model was the first major object oriented specification that recognized the need to use objects and relational database together.
2. An Enterprise JavaBeans query language – *EJB QL*. EJB QL is used to define finder queries and select methods of an entity bean with container-managed persistence.
3. A new EJB's Local/LocalHome Interface as a high-performing way to access an enterprise bean.
4. Java Naming and Directory Interface – *JNDI* is used to obtain remote references to an EJB's home interface, a factory interface for creating, finding and removing EJBs

Third, I will be implementing Java Front-End technology – web client application: that is, Java-Servlets and JSP on the Tomcat that tie to the EJBs component, the middle tier.

Finally, I will be implementing end-to-end project – which includes an integration and deployment of – "4-tier EJB BugTracker" architecture. This experimental project will be build for software managers, engineers and quality assurances that keep track of software bugs for multiple projects with independent user login. It has features such as creating users login and password, projects/bugs tracking with various project versions and project components, and its detailed histories.

[2] Architectural overview of the project:

This project will be implemented according to the following EJB Architecture.

1. An internet browser as (html/web client)
    - Microsoft Explorer
2. A Java WebServer
    - Latest open source web server -- Tomcat 4.0.6 {9}
        i. Tomcat is a commercial-quality web server solution based on the Java platform that supports the Servlet and JSP specifications.
3. An Application Server
    - Latest application server -- JBoss 3.0.4 {8}
        i. JBoss application server provides an EJB container.
        ii. The container, which EJB components live, supplies middleware services to the client.
4. Any Relational Database server {8}
    - JBoss already comes with an Object-Relational database (hypersonic database) that is used as backend storage for this project.
5. Other auxiliary systems like the Java Naming and Directory Interface (JNDI)

Figure-1 shows the basic structure of a J2EE application: a typical designed for creating server-side scalable, transactional, multi-user secure enterprise-level web applications. It provides a consistent component architecture framework for creating distributed n-tier middleware.

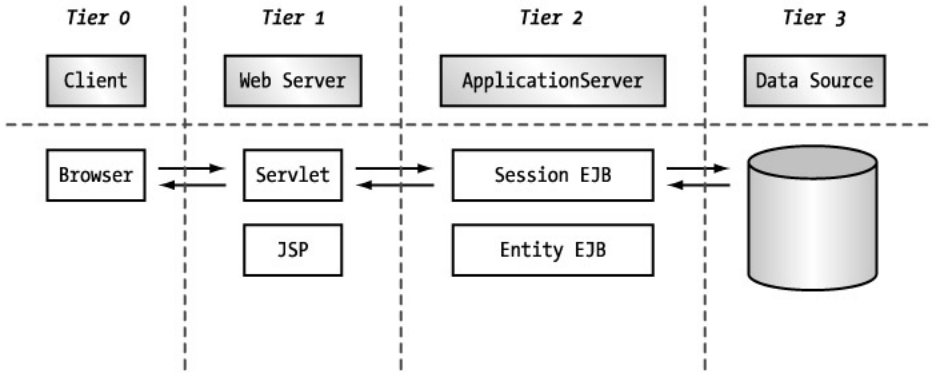


Figure 1: Principal structure of a J2EE application, including the Web server, application server, and data source. Used technologies are displayed below the dashed line; normal application designations are displayed above it {1}.

A brief description of Application Server (JBoss 3.0):
An application server provides an EJB container. The container, which EJB components live, supplies middleware services to the EJB components and manages them. Examples of EJB containers are BEA's WebLogic, Sun-Netscape-AOL alliance's iPlanet, IBM's WebSphere, Inprise's Inprise Application Server, and many others. JBoss is one of the open source application servers.

JBoss is an applications server that provides an EJB container along with other low-level services such as distributed transaction management, security, resource management, persistent management, remote accessibility, multi-client support and location transparency to an enterprise system. *Figure 2* illustrates JBoss application server that is located between a Web server and backend database system. Typically, accessing EJB components that are deployed to the JBoss application server is performed via a Web server such as Apache Tomcat from a browser.

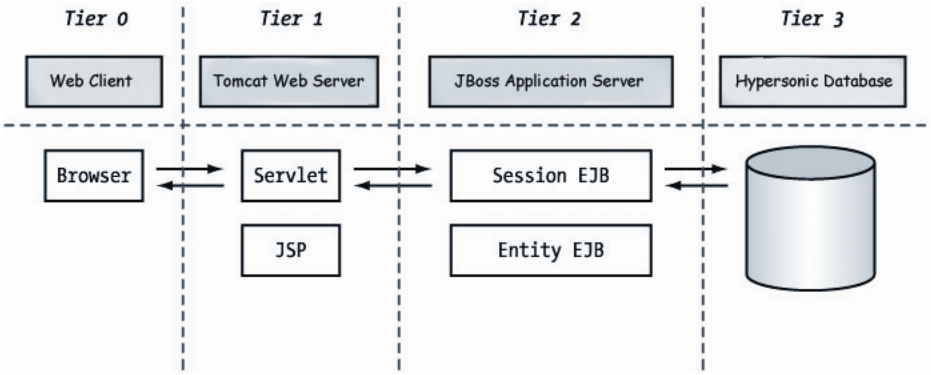


Figure 2, JBoss: Java Middleware{8}

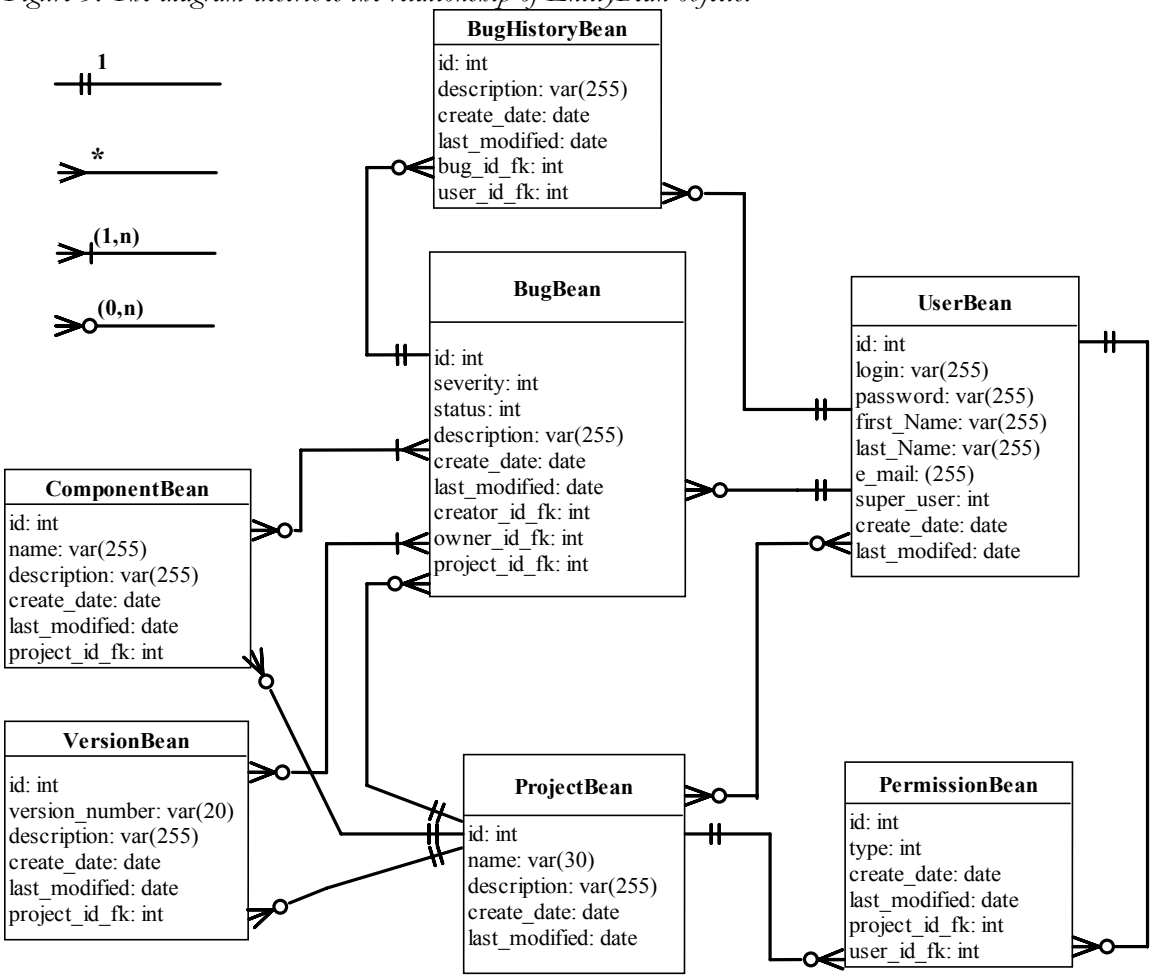
The JBoss application server is an open source Java implementation of the J2EE specification. The JBoss community (over 1000 developers world wide) is working to deliver the full range of J2EE tools as the premier Enterprise Java application server for the Java 2 Enterprise Edition platform. The JBoss application server is delivered under a public LGPL license. Modularity developed from the ground up; the JBoss application server is completely implemented using component-based plug-ins.

For the database backend, JBoss already comes with an Object-Relational database *(hypersonic database)* that is used as backend storage for small applications. In this BugTracker project, I will be using newly release versions of JBoss integrate with Tomcat Web server and Hypersonic backend.

[3] Database and Enterprise JavaBean Design specification

(A) Entity-Relationship Diagram:

Figure 3: The diagram describes the relationship of EntityBean objects.



(B) Data Dictionary:

TableName: bugbean

Column Name	Data Type	Description
Id	INT	*Primary Key*
Severity	INT	*1 – Critical* *2 – Major* *3 – Minor* *4 – Enhancement*
Status	INT	*1 – Assigned* *2 – Resolved* *3 – Closed*
Description	VARCHAR(255)	
Create_date	DATETIME	

Last_Modified	DATETIME	
Creator_id_fk	INT	*FK, Point to UserBean*
Owner_id_fk	INT	*FK, Point to UserBean*
Project_id_fk	INT	*FK, Point to ProjectBean*

```
create table bugbean (
  id                INT               PRIMARY KEY,
  severity          INT,
  status            INT,
  description       VARCHAR(255),
  create_date       DATETIME,
  last_modified     DATETIME,
  creator_id_fk     INT,
  owner_id_fk       INT,
  project_id_fk     INT
);
```

TableName: bughistorybean

Column Name	Data Type	Description
Id		*Primary Key*
Description		
Create_date		
Last_modified		
Bug_id_fk		*FK, Point to BugBean*
User_id_fk		*FK, Point to UserBean*

```
create table bughistorybean (
  id                INT               PRIMARY KEY,
  description       TEXT,
  create_date       DATETIME,
  last_modified     DATETIME,
  bug_id            INT,
  user_id           INT
);
```

TableName: permissionbean

Column Name	Data Type	Description
Id		*Primary Key*
Type		*1: Project administration* *2: Create bugs* *3: Edit bugs* *4: Close bugs* *5: Allow assign bugs*
Create_date		
Last_modified		
Project_id_fk		*FK, Reference to project id*
User_id_fk		*FK, Reference to user id*

```
create table permissionbean (
  id                INT          PRIMARY KEY,
  type              INT,
  create_date       DATETIME,
  last_modified     DATETIME,
  project_id_fk     INT,
  user_id_fk        INT
);
```

TableName: projectbean

Column Name	Data Type	Description
Id		*Primary Key*
Name		*Description of the project*
Description		
Creat_date		
Last_modified		

```
create table projectbean (
  id                INT          PRIMARY KEY,
  name              VARCHAR(255),
  description       VARCHAR(255),
  create_date       DATETIME,
  last_modified     DATETIME
);
```

TableName: userbean

Column Name	Data Type	Description
Id		*Primary key*
Login		*User's login ID*
Password		*User's login password*
First_name		*First name*
Last_name		*Last name*
Email		*Email address*
Super_user		*1 is super user – 0 is not super user*
Create_date		
Last_modified		

```
create table userbean (
  id                INT                PRIMARY KEY,
  login             VARCHAR(255),
  password          VARCHAR(255),
  first_name        VARCHAR(255),
  last_name         VARCHAR(255),
  email             VARCHAR(255),
  super_user        Boolean,
  create_date       DATETIME,
  last_modified     DATETIME,
);
```

TableName: versionbean

Column Name	Data Type	Description
Id	INT	*Primary key*
Version_number	VARCHAR(255)	*Project's version: 1, 2 .. n etc*
Description	VARCHAR(255)	
Create_date	DATETIME	
Last_modified	DATETIME	
Project_id_fk	INT	*FK*

```
create table versionbean (
  id                INT           PRIMARY KEY,
  version_number    VARCHAR(255),
  description       VARCHAR(255),
  create_date       DATETIME,
  last_modified     DATETIME,
  project_id_fk     INT
);
```

TableName: componentbean

Column Name	Data Type	Description
Id	INT	*Primary Key*
Name	VARCHAR(255)	*Describe any project's component name such as Web, JavaClient, and Server component*
Description	VARCHAR(255)	
Create_date	DATETIME	
Last_Modified	DATETIME	
Project_id_fk	INT	*FK*

```
create table componentbean (
  id                INT                 PRIMARY KEY,
  name              VARCHAR(255),
  description       VARCHAR(255),
  create_date       DATETIME,
  last_modified     DATETIME,
  project_id_fk     INT
);
```

[4] A functional specification (bug tracking scenario) of the project:

(Note: Following web front-end design is just a scenario. Enhancement will done at final implementation.)

1. Main Page (List of user's open bugs)
2. Project administration
3. Bug Tracking
    - Creating a new bug
    - Editing a bug
    - Listing bugs
    - Searching for bugs
4. User administration
5. Logout

(1) Main Page (Top Menu Bar)

EJB BugTracker
by San Aung

Main Page | Project Administration | Bug Tracking | User Admin | Logout

Listing of open bugs!

(2) Project administration -- multiple projects administration views:

Once users click at the heading [Project Administration Menu], it leads users into [Multiple Project Views] where users see all the projects as *Figure-4*. From this multiple project view, users can do the following two scenarios:

1. Creating a **N**ew project (By clicking [N])
2. **E**diting/updating existing project properties and related project information (by clicking [E])

(*Figure 4*: Multiple Project Administration)

[N]

	Name	Description	Created	Bugs
[E]	Manager Created CocaCola Project1		11/07/2002	1
[E]	Developer Create Pepsi Project2		11/07/2002	2
[E]	QA created internal project		11/07/2002	3

When clicking for "Creating a **N**ew project", it goes and "Create New Project" page and key in the necessary information such as: PROJECT-NAME and PROJECT-DESCRIPTION and return back to the "Project Administration" page.
When clicking for "Editing existing project", it goes to "Edit Project" page where user either can further defined "Version" or "Component" of that specific project.

(3) Bug Tracking -- (By Project List)
Once user click at the [Bug Tracking Menu], it will first list all the projects with the following summary reports for all of its existing projects.

(*Figure 5*: Bug Tracking)

	Project Name	Open Bug	Resolved Bug	Total Bug
[L][N][S]	Manager Created CocaCola Project1	2	2	4
[L][N][S]	Developer Create Pepsi Project2	1	1	2
[L][N][S]	QA created internal project	0	0	0

From above Bug Tracking page, use can click for **L**ist ([L]), create **N**ew ([N]) and **S**earch ([S]) for a specific bug related information.

When users click (L), it goes to "*All of the Bug **L**isting* for that specific project".
When users click for (N), it goes to "Create **N**ew bug page"
When users click for (S), it goes to "Bug **S**earch page"

Editing an existing bug:
If user wants *editing* an existing bug for a specific project, users must fist go to the bug Listing page by clicking at the [Bug Tracking Menu] screen than follow with *"bug listing page"* by clicking at the icon (L) just beside the project you want a list of bugs for. Once users have a list of bugs, you can select the edit icon (E) beside the bug you wish to edit.

(4) Users Administration Screen:

(*Figure 6*: User Administration)

Create new user: N

Login	Name	Email	Admin	Last Mod
E Admin	Super User	admin@hotmail.com	Yes	11/07/2002
E Manager	San	sanaung@hotmail.com	Yes	11/12/2002
E Developer	John	john@hotmail.com	No	11/12/2002

[5] Deliverables

1. Implemented codes.
2. User manual.
3. Testing document.
4. Project Presentation (Expected by the end of winter 2002 quarter)

[6] Annotated references

1. Lennart Jorelid, "J2EE FrontEnd Technologies: A Programmer's Guide to Servlets, JavaServer Pages, and JavaBeans", December 15, 2001.
2. Ed Roman, Scott W. Ambler, Tyler Jewell, Floyd Marinescu, "Mastering Enterprise Java Beans Second Edition", December 14, 2001.
3. Monson-Haefel, Richard, "Enterprise JavaBeans, 3rd Edition", September 2001.
4. Scott Stark, Marck Fleury, "JBoss Administration and Developmenty," The JBoss Group, January 2002.
5. Jim Farley, William Grawford & David Flanagan, "Java Enterprise in a Nutshell, Second Edition", April 2002.
6. Power Vision, "Building Data-Centric n-Tier Enterprise Systems,"Technical White Paper, July 2001.
7. J2EE Community. "http://www.TheServerSide.com"
8. JBoss Application Server, "http://www.jboss.org"
9. Jakarta Tomcat. "http://jakarta.apache.org/tomcat/index.html"
10. Sun Microsystems, "Enterprise JavaBeans Specification v2.0,"August 2001. http://java.sun.com/j2ee/
11. The J2ee tutorial: http://java.sun.com/j2ee/tutorial/