

Develop a SaaS application on GAE or Force.com or Azure

Group F1 (Windows Azure Team)

Deepak Pancras (dxp124330)
Maneesh Abraham (mea130130)
Archit Trivedi (amt130330)
Naimish Patel (njp130330)
Santosh Vivekanandan (sxv135530)

INDEX

1. Project Definition.....	2
2. Exploration of PaaS Platforms	
a. Introduction.....	4
b. Access Control.....	7
c. Web Application Development.....	9
d. Multi-tenancy.....	10
e. Robocode Compilation Feasibility.....	14
f. Robocode Implementation Feasibility.....	15
g. Comparison Summary.....	16
3. Project Implementation	
a. Windows Azure Features.....	17
b. Overview of Software to be Deployed: Robocode.....	18
c. Database Schema.....	19
d. System Architecture.....	23
e. Robocode Comparison Implementation.....	25
f. Access Control Implementation.....	27
g. Compilation of Robocode Robots on Azure Cloud.....	33
h. Run Robocode Battle.....	34
i. Robot Ranking.....	36
j. Installation Summary.....	36
k. Screenshots.....	37
l. References.....	45

1. Project Definition

Project Title: Develop a SaaS application on GAE or Force.com or Azure.

Goal: Port the standalone Robocode application to the Windows Azure cloud.

TA: Nidhi Solanki [Nidhi.Solanki@utdallas.edu]

Study the PaaS platforms:

1. GAE
2. Force.com
3. Azure

Objectives:

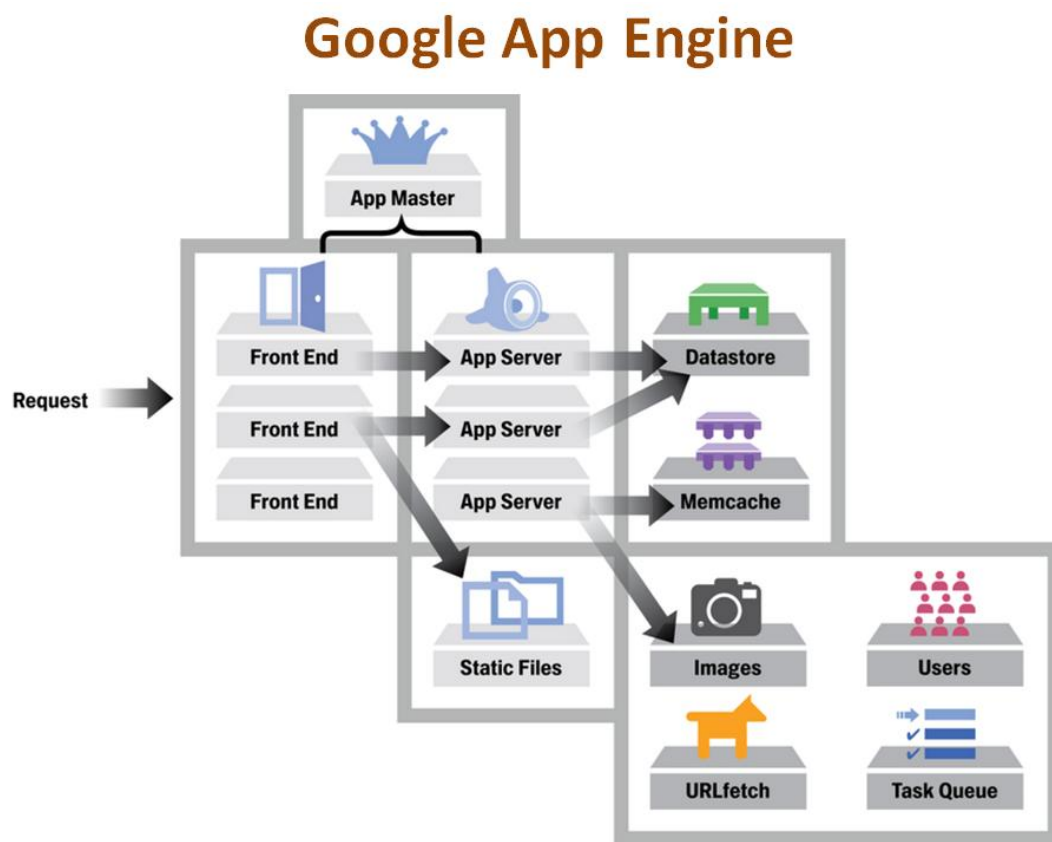
- Access Control : Robots as a Resource
 - Intra-Domain Access Control
 - Inter-Domain Access Control
- Compile Robocode on the cloud
- Robocode Code Comparison
- Run Robocode Battle
 - Play Battle without UI (Executes on the server)
 - Play Battle with UI (Executes battle on the client)

2. Exploration of PaaS Platforms

a) Introduction

GAE

- Is a platform as a service (PaaS) cloud computing platform for developing and hosting web apps in Google data centers.
- Programming languages supported: Python, Java, Go and PHP.
- DataStore – key-value store.
- Google Cloud SQL, a managed MySQL service.

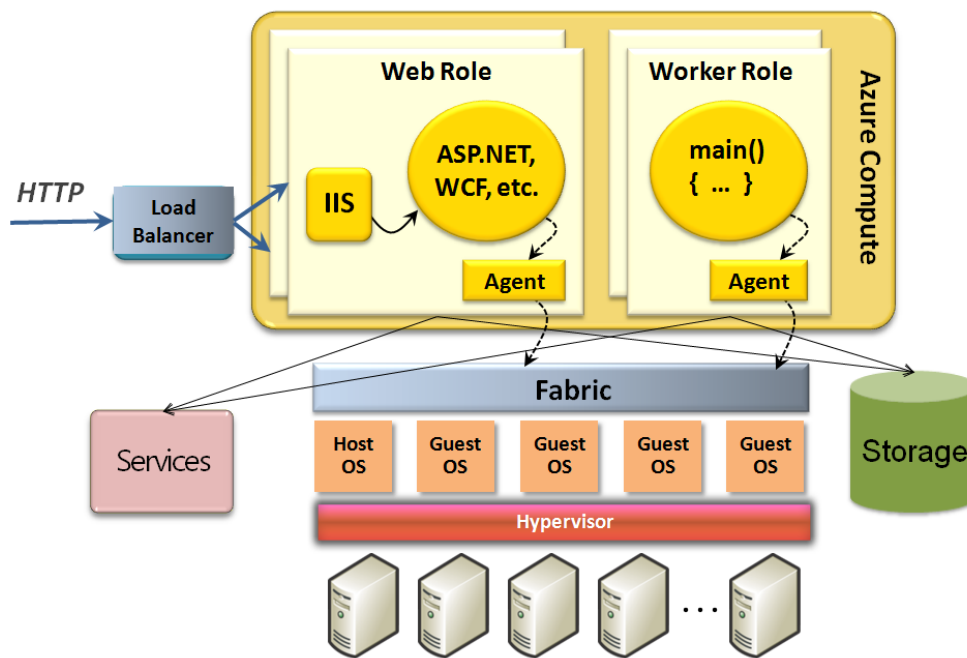


Google App Engine Architecture

Azure

- Provides both PaaS and IaaS services.
- Allows flexibility to use their own data store and web server containers etc.
- Programming languages supported: .NET, JAVA, Ruby, Node.js, PHP, Python.
- Azure SQL Database, Azure NoSQL, queues, tables and blobs.

Windows Azure Compute



Windows Azure Architecture

Force.com

- Force.com has its own programming language, called Apex.
- Apex Code is designed explicitly for expressing business logic and manipulating data.
- Web pages designed using their own language, VisualForce pages.
- In built web containers.
- RDBMS has provided as service and queried through SOQL.



Force.com Architecture

b) Access Control

GAE

Levels of access control:

- Configuring instance level access control
- Configuring MySQL database access control

Role/Resource Based Access Control

- GAE does not provide any default role/resource based access control
- GAE only provides authentication based on using Google Accounts

Roles

- App Engine provides three roles supplying different levels of access to Administration Console features.
- Each progressively stronger role includes all permissions from the previous role.
- But these roles are not used in the applications, these roles are used for managing the applications deployed on GAE.

Azure

Active Directory

- Authenticate and authorize users to gain access to your web applications and services while allowing the features of authentication and authorization to be factored out of your code.

Features of ACS

- Integration with Windows Identity Foundation (WIF).
- Support for OAuth 2.0 (draft 13), WS-Trust, and WS-Federation protocols.
- Out-of-the-box support for Active Directory Federation Services (AD FS) 2.0.
- In azure in each active directory we can we can define domain.
- In each domain we can make groups.
- And each group have many users.

Role Based Access Control

- Windows Azure AD Graph.
- Can add the service principal that represents an application to one of the built-in roles in the following table to grant it either read-write or read-only access to the Windows Azure AD entities in your organization.

Force.com

Force.com provides the following access control mechanism:

- Record Ownership
- Role Hierarchy
- Sharing Rules
- Manual Sharing
- APEX managed sharing

Record Ownership

- Each record is owned by a user or optionally a queue for custom objects, cases and leads.
- The record owner is automatically granted Full Access, allowing them to view, edit, transfer, share, and delete the record.

Role Hierarchy

- The role hierarchy enables users above another user in the hierarchy to have the same level of access to records owned by or shared with users below.
- Consequently, users above a record owner in the role hierarchy are also implicitly granted Full Access to the record, though this behaviour can be disabled for specific custom objects.
- The role hierarchy is not maintained with sharing records. Instead, role hierarchy access is derived at runtime.

Sharing Rules

- Sharing rules are used by administrators to automatically grant users within a given group or role access to records owned by a specific group of users.
- Sharing rules cannot be added to a package and cannot be used to support sharing logic for apps installed from Force.com.
- Sharing rules can be based on record ownership or other criteria.
- Apex can't be used to create criteria-based sharing rules.
- Also, criteria-based sharing cannot be tested using Apex.

Manual Sharing

- User Managed Sharing, also known as Manual Sharing.
- User managed sharing allows the record owner or any user with Full Access to a record to share the record with a user or group of users.
- This is generally done by an end-user, for a single record.
- Only the record owner and users above the owner in the role hierarchy are granted Full Access to the record.

- It is not possible to grant other users Full Access. Users with the “Modify All” object-level permission for the given object or the “Modify All Data” permission can also manually share a record.
- User managed sharing is removed when the record owner changes or when the access granted in the sharing does not grant additional access beyond the object's organization-wide sharing default access level.

APEX Managed Sharing

- Apex managed sharing provides developers with the ability to support an application’s particular sharing requirements programmatically through Apex or the SOAP API.
- This type of sharing is similar to Force.com managed sharing.
- Only users with “Modify All Data” permission can add or change Apex managed sharing on a record.
- Apex managed sharing is maintained across record owner changes.

c) Web Application Development

GAE

- App Engine runtime
- Persistent storage with queries, sorting, and transactions.
- Automatic scaling and load balancing.
- Asynchronous task queues for performing work outside the scope of a request
- Have an in built web server which takes care of handling web application requests

Azure

- Has in-built IIS server to handle web requests
- Also has provision for the user to use their own web servers like tomcat, web sphere etc

Force.com

- Has in built web servers
- Abstracted away from the users cannot configure or tune web servers

d) Multi-Tenancy

- One instance of an application, running on a remote server, serves many client organizations (also known as tenants)
- Types:
 1. Hardware and Resource multi-tenancy
 2. Data multi-tenancy
 3. Application multi-tenancy

Strategies of Multi-Tenancy

- Physical Separation: Ensures that each tenant gets his own dedicated hardware resources.
- Virtualization: Aims at creating application hosting environments that provide logical boundaries between each tenant.
- Allows the application to automatically adjust its behavior differently for different tenants at runtime

GAE

- GAE provides the Namespace API to achieve multitenancy at a data level as of today.
- The Namespace API is available for Datastore, Task Queues and Memcache.
- The Blobstore API does not support Namespace
- Namespace API also supports Google Apps domain, so if your tenants are going to be Google apps users, then you can set their domain name as Namespace for their data.
- API's supported by Namespaces:
Datastore, Memcache, Task queue, Search

Namespace

1. Creating a namespace

- Setting the namespace to the current google apps domain will split data for each google apps user.
`NamespaceManager.set(NamespaceManager.getGoogleAppsNamespace());`
- If the data is split for each user, then user ID can be used as namespace.
`UserServiceFactory.getUserService().getCurrentUser().getUserId();`
- Common data can be stored in common namespace, and namespaces can be switched when needed.

2. Using namespaces with the Datastore

- By default, the datastore uses the current namespace setting in the namespace manager for datastore requests. The API applies this current namespace to Key or Query objects when they are created.
- Key and Query objects are stored in serialized forms.
- Query and key objects inherit the current namespace when constructed.
- There is no Java API to explicitly set the namespace of a key or query.

3. Using namespaces with Memcache

- By default, memcache uses the current namespace from the namespace manager for memcache requests.
- If you explicitly set a namespace in the memcache, it will ignore the current settings from the namespace manager.
- `getMemcacheService(Namespace)`: used for explicitly setting a namespace

4. Using Namespaces with Task Queue

- Push queues use the current namespace as set in the namespace manager at the time task was created.
- Pull queues do not support namespace functionality.
- Task names are shared across all namespaces i.e all task names should be unique, even if they are from different namespaces.

5. Using Namespaces with Search

- `SearchService` takes the current namespace
`SearchService searchService = SearchServiceFactory.getSearchService();`
- Explicitly configuring a namespace is done by `SearchServiceConfig`.
`SearchServiceConfig config =`
`SearchServiceConfig.newBuilder().setNamespace("anotherSpace").build();`

Force.com

- The well-defined metadata-driven architecture ensures a clear separation of the compiled runtime database engine (kernel), tenant data, and the metadata that describes each application thereby making it possible to independently update the system kernel and tenant-specific applications and schemas, with virtually no risk of one affecting the others.
- Every logical database object that Force.com exposes is internally managed using metadata.
- Objects, tables, fields, stored procedures, and database triggers are all abstract constructs that exist merely as metadata in Universal Data Dictionary.

- When you create application schemas, the UDD keeps track of metadata concerning the objects, their fields, their relationships, and other object attributes.
- When you need to modify or customize something about the application schema, like modify an existing field in an object, all that's required is a simple non-blocking update to the corresponding metadata
- Multitenant Metadata:
 - ❖ Two internal tables are used to manage the metadata corresponding to a tenant's schema object: MT_Object and MT_Fields
 - ❖ MT_Object: stores metadata about the tables defined by an organization for an application. It includes object id, the organization id and the object name
 - ❖ MT_Fields: stores metadata about the fields defined for an object; also includes unique identifier for a field, field name, organization id and the object id
- Multitenant Data:
 - ❖ MT_Data system table: Stores the data that can be accessed by an application
 - ❖ It maps to all organization specific tables and fields which are declared in MT_Objects and MT_fields respectively
 - ❖ Each row includes global unique identifier(GUID), object id, organization id.
 - ❖ Each row also has a Name field that stores a "natural name" for corresponding records; for example, an Account record might use "Account Name," a Case record might use "Case Number," and so on.
 - ❖ Flex columns, otherwise known as slots, store application data that maps to the tables and fields declared in MT_Objects and MT_Fields, respectively
 - ❖ MT_Data also contains other columns for auditing data(which user created a row, when was the row created and so on.)
 - ❖ MT_Data also contains an IsDeleted column to indicate when a row has been deleted.
- Multitenant Indexes:
 - ❖ Force.com automatically indexes various types of fields to deliver scalable performance
 - ❖ Not practical to create native database indexes for the flex columns of MT_Data because Force.com uses a single flex column to store the data of many fields with varying structured datatypes.
 - ❖ Force.com manages an index of MT_Data by synchronously copying field data marked for indexing to an appropriate column in an MT_Indexes pivot table.
 - ❖ MT_Indexes has strongly typed, indexed columns such as StringValue, NumValue, and DateValue used to locate field data of the corresponding datatype

- ❖ MT_Unique_Indexes pivot table is used to support uniqueness for custom fields(Usually used when an application attempts to insert a duplicate value into a field that requires uniqueness, or an administrator attempts to enforce uniqueness on an existing field that contains duplicate values)
- Multitenant Relationships:
 - ❖ When an organization declares an object's field with a relationship type, Force.com maps the field to a value field in MT_Data, and then uses this field to store the Object ID of a related object.
 - ❖ MT_Relationships pivot table is used to optimize join operations.
 - ❖ MT_Relationships system table has two underlying database unique composite indexes that allow for efficient object traversals in either direction, as necessary.
- Multitenant Bulk Operations:
 - ❖ Bulk jobs are a great way to improve efficiency, but are more likely to create transactional errors.
 - ❖ When a bulk job is executed a Roll-Back save point is created, in order to handle the incomplete transactions.
- Multitenant Isolation and Limits:
 - ❖ Multitenancy isolation and governance is important when working with a cloud-based platform.
 - ❖ To prevent one client to consume an overbearing amount of resources, while other clients are starved for resources, governor limits have been implemented

Azure

- Have partitioning schemes which can be used with Windows Azure SQL Database, or with a database such as SQL Server or MySQL hosted in a Windows Azure VM
- Example schemes
 1. One server per tenant
 2. Multiple tenants per database
- Web and Worker roles can also be multitenant
- In all multi-tenant applications the design should make sure that tenants can access their own data alone. To achieve proper isolation, we should not to reveal any storage account keys, and all queries access and return the correct tenant's data.
- For all the data storage mechanisms, if the tenant provides the subscription this makes clear that the tenant owns, and is responsible for, the data stored in any storage account or database in the subscription.
- Shared Access Signatures:

- ❖ Both Windows Azure table storage and Windows Azure blob storage support shared access signatures as a mechanism for controlling access to data. You can use shared access signatures to ensure isolation of tenant data

Advantages of Multi-Tenancy

- Application behavior can be customized; no need to modify application code; can be achieved by modifying application metadata that is invoked on a user-by-user basis
- Each customer can use the metadata to configure the way the application appears and behaves for its users.
- Scalability, Cost and Effort Saving
- Strong separation between app logic and its datasets

e) Robocode Compilation Feasibility

- Robots are compiled on the Azure cloud platform
- Robocode robots are written in Java
- We can compile java code using Java Compiler API in JDK7
- Also we need access to the robocode core libraries to compile robots

GAE

- Google App Engine has Java support, but they do not have all the Java APIs whitelisted.
- But Java Compiler APIs are disabled. So we cannot compile new robots online using the developed app.
- Can load the robocode core libraries as a web application and use them.

Windows Azure

- Windows Azure allows uploading the complete Java Development Kit to the Azure platform.
- So, there won't be any restrictions to use JAVA COMPILER APIs for compile the robots online.

Force.com

- They do not provide any jdk or compilation unit for compiling java classes
- The only way to implement any thing in java is to host the java application as a web service and access it from force.com

- Also, Force.com has integrations with Heroku cloud platform which provides a java compilation unit

f) Robocode Implementation Feasibility

- Since Robocode is implemented as a standalone application, the best bet to move it to cloud is to create a web application with serves as an UI for the robocode application and call the robocode functions in the Robocode library from the UI.

Google App Engine

- GAE has a complete servlet life cycle implementation suing the Jetty servlet container. We can create traditional servlet based web applications

Windows Azure

- Windows Azure allows us to use our own servlet containers to develop web applications.
- For example, one can use Apache Tomcat to develop my Java based web applications

Force.com

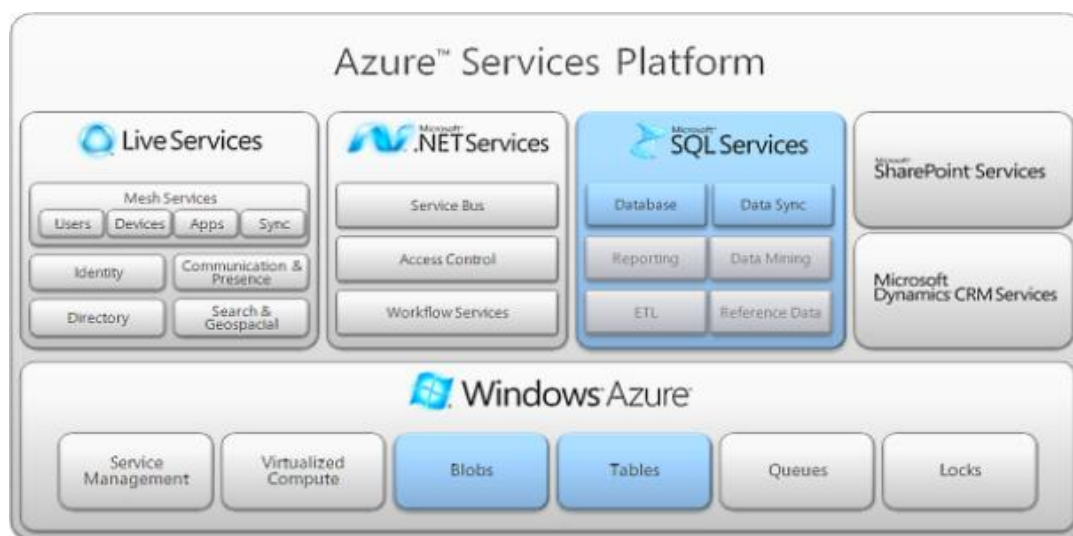
- Force.com has their proprietary containers and servlet implementations.
- We have to use APEX and VisualForce pages to create the web applications.

g) Comparision Summary

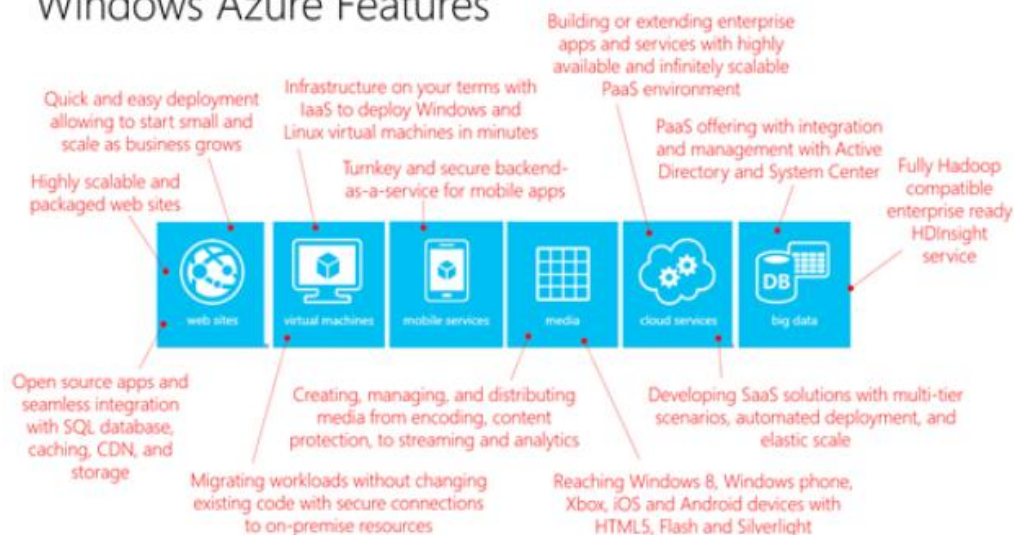
Features	Google App Engine	Windows Azure	Force.com
App Engine Run Time	JAVA, Python, PHP, GO	.NET, JAVA, Ruby, Node.js, PHP, Python	Proprietary Run Time (APEX & VisualForce)
Database	Cloud SQL, Datastore, Blob store	Azure SQL Database, Windows Azure NoSQL Queues, Tables and blobs	RDBMS
Query Language	SQL, NoSQL	SQL, NoSQL	SOQL (Salesforce Object Query Language)
Cost	Limited Free Access	30 days Full Access	30 days Full Access
ACL	No RBAC. Only Google account authentication.	Supports RBAC using Active Directory and AD Graph API	Supports RBAC, Record Ownership, Sharing Rules
Multi-tenancy	YES Namespace API	YES	YES

a) Features of Windows Azure

- Windows Azure is a cloud computing platform and infrastructure, created by Microsoft, for building, deploying and managing applications and services through a global network of Microsoft-managed datacenters.
- It provides both PaaS and IaaS services and supports many different programming languages, tools and frameworks, including both Microsoft-specific and third-party software and systems.
- Windows Azure was released on February 1, 2010.



Windows Azure Features

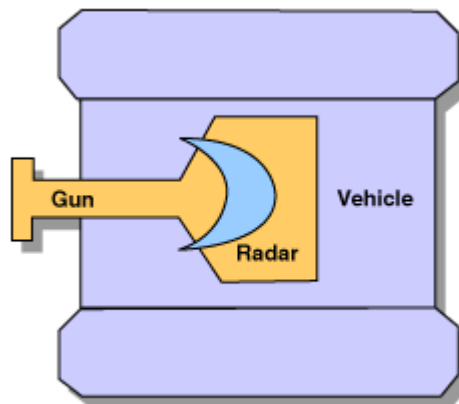


b) Overview of Software to be Deployed: Robocode



- Competitors write software that controls a miniature tank that fights other identically-built (but differently programmed) tanks in a playing field.
- Robots can move, shoot at each other, scan for each other, and hit the walls or other robots if misprogrammed.
- Though the idea of the game is simple, the strategy needed to win is not. Robots can have thousands of lines in their code dedicated to strategy.
- Robocode is an open source educational game started by Mathew Nelson. Contributions are being made by people including Flemming N. Larsen and Pavel Šavara who work on Robocode to keep it current and fix the bugs.

The Anatomy of a Robocode



- Note that the robot has a rotating gun, and on top of the gun is a rotating radar.
- The robot vehicle, the gun, and the radar can all rotate independently: at any moment in time, the robot's vehicle, the gun, and radar can be turned in different directions.
- By default, these items are aligned, facing the direction of the vehicle movement.

c) Database Schema

- **Table Name : users**

users		
Column Name	Datatype	
user_id	INT AUTOINC	
user_name	NVARCHAR	
user_pwd	NVARCHAR	
user_domain_id	INT	
user_role_id	INT/ENUM	
user_ranking	INT	
created_date	DATETIME	
updated_date	DATETIME	

- **Table Name : robots**

robots	
Column Name	Datatype
robot_id	INT AUTOINC
robot_name	NVARCHAR
robot_desc	NVARCHAR
robot_code	TEXT
robot_ranking	INT
created_by	INT
created_date	DATETIME
updated_date	DATETIME

- **Table Name : roles**

roles	
Column Name	Datatype
role_id	INT AUTOINC
role_name	NVARCHAR

- **Table Name : rights**

rights	
Column Name	Datatype
right_id	INT AUTOINC
right_name	NVARCHAR

- **Table Name : inter_domain_mapping**

Column Name	Datatype
idm_id	INT AUTOINC
idm_domain_id	INT
idm_role_id	INT
idm_other_domain_id	INT
idm_other_domain_role_id	INT
created_date	DATETIME
updated_date	DATETIME

- **Table Name : robot_updates**

Column Name	Datatype
ru_id	INT AUTOINC
ru_robot_id	INT
ru_user_id	INT
ru_domain_id	INT
created_date	DATETIME
updated_date	DATETIME

- **Table Name : access_rights**

Column Name	Datatype
ar_id	INT AUTOINC
ar_robot_id	INT
ar_domain_id	INT
ar_role_id	INT
ar_right_id	INT
created_date	DATETIME
updated_date	DATETIME

- **Table Name : moss_result**

Column Name	Datatype
mr_id	INT AUTOINC
compare_robot_id_1	INT
compare_domain_id_1	INT
compare_robot_id_2	INT
compare_domain_id_2	INT
percentage	FLOAT



d) System Architecture

Design of the Application

- We have created a web application so it involves servlet calls from the web. We have designed our application carefully to remove the high coupling and making it low cohesive.
- Separated the different layers of our application:
 - UI
 - Servlets
 - Service
 - Repositories
- Most of the UI interactions are captured as a request from the web to the servlet
- From the servlet we call different services like LoginService.java, CreateRobotService.java to perform the business logic of the particular use case
- From the services we call the repository layer classes like LoginRepository.java, CreateRobotRepository.java which makes the actual database queries interacting with the database.

Architecture

- Windows Azure is our deployment platform
- Created a dynamic web application in java which uses Apache Tomcat web container as the server which intercepts all the web calls for the application and routes it to the specific servlets based on the configuration given in the web.xml in our application
- The whole application is packaged with the tomcat server and java runtime jdk which will be deployed on the azure instance as a service

Azure has no inbuilt java runtime or tomcat web container thats why when we package our application we also add the java runtime and tomcat server and deploy it on the azure instance

Hibernate

We are using Hibernate Object Relational Mapping library to map our Java Models to the Tables in the databases. Also, hibernate query language (HQL) is used for querying the database tables.

Sample:

For example: For Login table

- Create hibernate.cfg.xml configuration file
- Create Login.hbm.xml mapping file
- Create Login.java source file
- Create LoginRepository.java source file which is used to query the database
- In the LoginRepository.java, we create a query using hibernate session and get the results from the Login table

Steps used in our application for hibernate integration

- Download the hibernate library from <http://hibernate.org/orm/downloads/>
- Add the downloaded hibernate jars to our application build path

- Create hibernate configuration file which will be used by the hibernate to make connection to the database
- In our proposed application we assumed we will use only two domains so we need to have two separate databases.
- On Azure cloud we have created two database instances and you can get the connection details for that particular database
- So we will be having two hibernate configuration file for two database hibernate1.cfg.xml and hibernate2.cfg.xml
- We have bean classes in the package com.utd.robocode.dto which will be directly mapped to the tables in the database. But we need to have hbm files for this mapping so that hibernate can map these bean classes to the tables
- We create *.hbm.xml files which contain details on how the bean classes properties are mapped to the table attributes

Use Case Of Hibernate



- To query Login table
- Use the DataStoreUtils.buildSessionFactory() which will return a singleton hibernate session reference to the database which we want to connect to.
- We then use the hibernate session to create a query which will return result as an object of the bean class User as mapped by hibernate

e) Robocode Comparison Implementation

One of the important aspect or need in the project is Code Comparison. In it we would be comparing codes between two different Robots. MOSS tool can be used to do the Robot's code comparison. This MOSS tool can be used only through PERL scripts.

What is MOSS?

- Moss (for a Measure Of Software Similarity) is an automatic system for determining the similarity of programs. To date, the main application of Moss has been in detecting plagiarism in programming classes. Since its development in 1994, Moss has been very effective in this role. The algorithm behind moss is a significant improvement over other cheating detection algorithms.

Sample Perl Script	 moss.pl
Sample RobotCodeBaseFile	 RobotCodeBaseCode.java

Which are the languages supported by MOSS?

- **Moss can currently analyze code written in the following languages:** C++, Java, C#, Python, Visual Basic, Javascript, FORTRAN, ML, Haskell, Lisp, Scheme, Pascal, Modula2, Ada, Perl, TCL, Matlab, VHDL, Verilog, Spice, MIPS assembly, a8086 assembly, a8086 assembly, MIPS assembly, HCL2.

Which are the requirements for using MOSS tool?

- A server side programming language
- MOSS Account

What is the main purpose of using MOSS tool?

- Functional dependency analysis

What is use of MOSS tool in Robocode as a Service?

- In this project User have to make Robots by writing out Java code for the same. Better the code makes brighter chances for that Robot to win the battle.
- We are also maintaining Access Control Mechanism where one user has Read, Write, Read + Update, Read + Update + Delete right where he can see Code of someone else's Robot and either use same or else change certain part of it or may be come up with his own original code.
- So, we have kept Analyze Robot portion wherein we check dependencies of a particular Robot with all the other Robots.

- In this project we have kept some code as base code which can be syntactically same for each and every Robot and then check the similarity of code in form of percentage that would give an idea regarding dependency of that Robot on others and vice versa.

➤ **How it works in our application:**

When the user is doing any of this operations-

- Create a Robot
- Edit a Robot

➤ **Steps:**

- When you perform any of the above said operations, the MossRobotService.java thread will be invoked
- This thread will check the Robot which you are creating or editing against all the other robot which we have created in the database previously for any functional dependencies using the MOSS tool
- To use the MOSS tool we need to have PERL run time installed in the server where our application is hosted
- And we also have a specific PERL script provided by the MOSS which we have to use to run the code comparison on the MOSS
- The MOSS tool will return an web URL which we have to again call to get the results of the code comparison

Programming Details

- In the MossRobotService.java thread, we first get the robot which we are going to save and save it as a local file.
- Also get all the other existing robots from the database and save it as a local file.
- Now get Java Runtime Exec and call the above moss perl script along with the RobotCodeBaseFile.java with the saved local files.
- The MOSS server will return the results which we will update in the moss_results table
- And delete all the temporary local files created.

Installation on the Azure Server

- Azure does not provide an in built PERL runtime
- Download the PERL installation from the following URL - <http://www.activestate.com/activeperl/downloads>
- Run the MSI installer
- Have to install PERL runtime in the following path in the Azure Instance -
 - C:\ProgFiles\Perl64\
- Copy the MOSS perl script, the base robot code file to the following folder -
 - C:\ProgFiles\apps\

f) Access Control Implementation

OVERVIEW

The Application provides access rights to be mentioned for every robot for every role in that particular domain. The application implements this by Access Control List.

ROLES

We maintain a role hierarchy in each of the domain. The domain information is hidden from the other domain. In our project, the Domain 1 has four roles and the Domain 2 has five roles. The Creator role is a role that can only create robots (resources), this role will have access to its own robot but not to other robots.

<u>Role Id</u>	<u>Domain 1</u>	<u>Domain 2</u>
1	Creator	Creator
2	Viewer	Viewer
3	Developer	Developer
4	Manager	Contributor
5		Manager

ACCESS RIGHTS (OR) PERMISSIONS

The project provides 4 permissions that can be assigned to a role in a particular domain for each robot created in that domain. While assigning rights to the robot for all the roles in that domain, we make sure we preserve the hierarchy of the domain by enabling the user to only choose the rights in an increasing order from viewer to manager. We do not associate the creator role with the access control as mentioned above.

<u>Right Id</u>	<u>Rights</u>
1	No Access
2	Read only
3	Read + Update
4	Read + Update + Delete

ASSIGNING OR SETTING ACCESS RIGHTS

The process of assigning access rights for a particular role of that domain is done as soon as a robot is created. These rights do not change; it stays the same for that particular robot throughout its presence in the system.

PROVIDING ACCESS

Providing access can be classified into two ways. One case is if the User is in the same domain as the Robot. Second case is when the User is in a different domain.

1. Same Domain

Since the rights are assigned to various roles in that domain for each and every robot. The User gets the access right as per the permission set to User's role by the owner of the robot.

2. Cross Domain

When accessing the robots of the other domain we may encounter some issues as there may not be a particular kind of role in the robot's domain. For this issue, we make use of the cross domain role mapping. This Mapping is done by the administrator who maintains the two domains. So whenever a user wants to access the robot in another domain, there are two steps that are followed.

1. User's role gets mapped to an already existing role (intermediate) in the robot's domain.
2. The User gets the access rights which are specified to the intermediate role in the robot's domain.

The access to a resource is provided keeping in mind the various users that have contributed (updated) to that resource. The Contributors can be from any domain. Other domain Users could have contributed to the resource only if they have access to update that resource, after the cross domain mapping is done. We maintain the information regarding provenance issue separately.

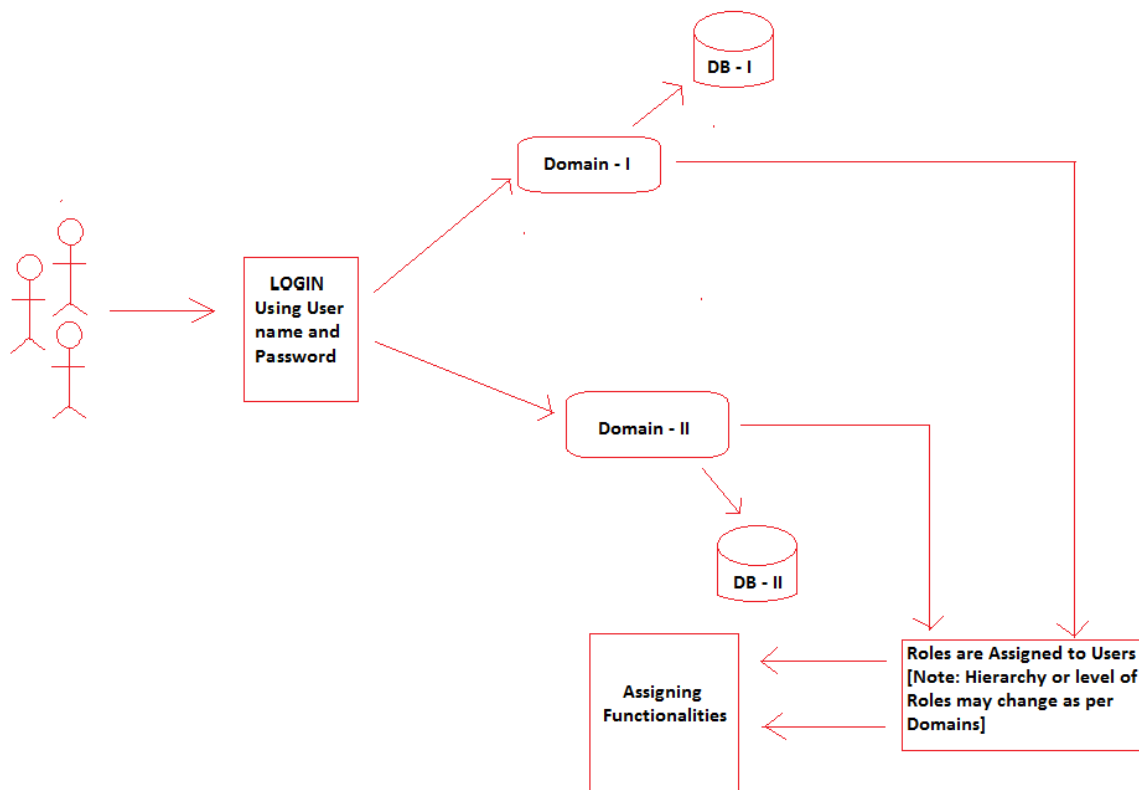
The **Algorithm** for providing access is as follows:

```
For Each Robot
{
    If User is the owner of the robot, give full permission for that resource
    Else
    {
        Check the users that have worked on the robot.
        For the contributors of the robot.
        {
            If Contributors Domain is the not the same as Robot's Domain.
            {
                Map the user's role to the role in the Contributors Domain.
                Then Map that role on to the Robot's Domain. Set User_Role
                from the value attained.
            }
            Check Access Rights table in the Robot's Domain and set access.
        }
        Set Accessright for the user for that particular robot as the minimum
        of access_rights.
    }
}
```

Rules for Implementation of Access Control:

- Robots are resources, access to which needs to be controlled.
- Different organizations are collaborating on developing a resource (In this case, a robocode robot).
- Roles hierarchy of different organizations can be different.
- Set Access Rights for different roles Per Resource.
- Map roles of one organization (domain) to another organization to allow cross domain access of resources

Below Diagram exhibits how Starting Flow in Access Control would look like,



Multiple Domains

[Note: 'Creator' role allows only create permission.]

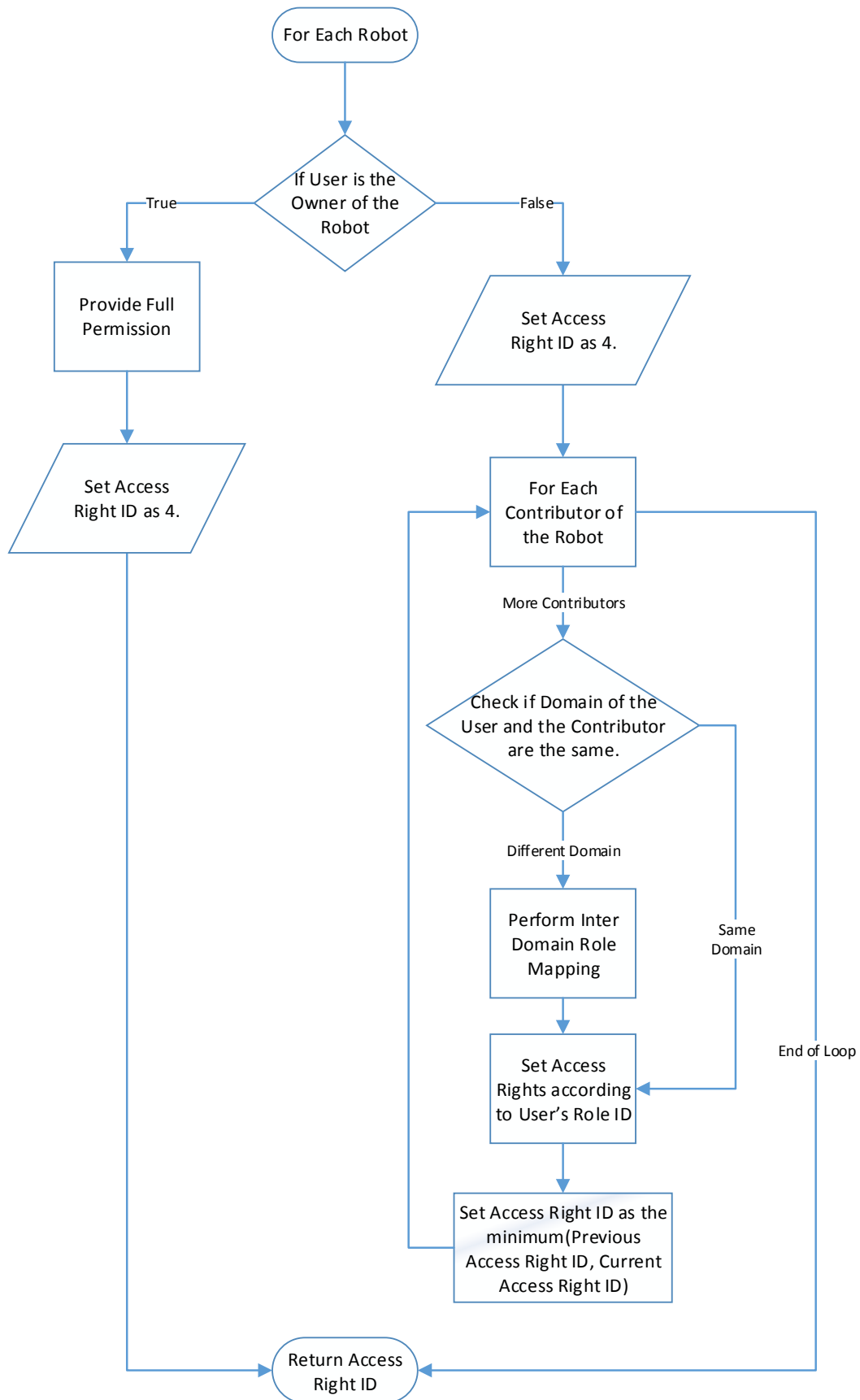
[Note: Rights are mapped to Roles on per Robot basis to achieve Access Control on a specific Resource.]

MY ROBOTS

From my robots page user can access robots which are created by him. It shows name and description of robots. There are contributor, view, edit and delete option for each robot. From contributor we can see who the other users who have modified that robot are. From View Robots we can view our robocode, from Edit Robot we can edit our robocode and can also modify access-rights and using Delete Robot option we can delete that robot from database.

MY ROBOTS - CONTRIBUTORS

This page will come up when we click on contributor link on my robots page. On this page it will display name of contributors, their domain id and updated date for that particular robot. Always 1st contributor of robot would be its creator.



g) Compilation of Robocode Robots on Azure Cloud

- Compilation is done using **javax.tools.JavaCompiler** in JSP.
- Robots are placed and compiled in the directory of that particular user in the directory robots of Robocode installation (Desktop Application) installed on the Azure Cloud Instance as these files are required to execute battle on the server side as well as for packaging the robots into JAR file for executing battle on the client side.
- Compilation requires **robocode.jar** to be specified in the classpath of the compiler call.
- The following is the code used for compilation-

```
JavaCompiler compiler = ToolProvider.getSystemJavaCompiler();
System.out.println(compiler);
String contextrealpath = request.getRealPath("/");
List<String> optionList = new ArrayList<String>();
optionList.addAll(Arrays.asList("-cp", contextrealpath+"/WEB-INF/lib/robocode.jar"));
DiagnosticCollector<JavaFileObject> diagnostics = new
DiagnosticCollector<JavaFileObject>();
StandardJavaFileManager fileManager =
compiler.getStandardFileManager(diagnostics, null, null);
Iterable<? extends JavaFileObject> compilationUnits =
fileManager.getJavaFileObjectsFromStrings(Arrays.asList(robotFileFullPathStr));
JavaCompiler.CompilationTask task = compiler.getTask(null, fileManager,
diagnostics, optionList, null, compilationUnits);
boolean success = task.call();
fileManager.close();
out.println(success);
```
- This compilation code will work across any platform – Windows server / Linux server as long as JDK installation is configured properly and has been tested on **JDK 1.7**.
- In our application, an Ajax call is made to our compiler code and the entire robot code is sent in the Ajax request to the compiler and the compiler results are returned to the user in response.
- For successful compilation, here are some points that need to be taken care of-
 - The **classname** of the robot should be same as the **Robot Name** (which will be used as the Robot Filename).
 - Robots created by a particular user are saved in the directory of that particular user on the Azure Cloud and hence the **package name** in the robot code must be the same as the **username** of that user.

h) Run Robocode Battle

a. In our application, we have two ways that a user can play the Robocode Battle-

- i. **Play Battle without UI (Executes Battle on the server side)**
- ii. **Play Battle with UI (Executes Battle on the client side)**

b. Play Battle without UI

- i. Executes Battle on the server side by executing java code from a JSP page. The JSP page makes a call to the Windows CMD command line command to execute the battle. This is achieved using **Runtime.getRuntime()** in Java.
- ii. Robocode (Desktop Application) needs to be installed on the following path on Azure – **C:\ProgFiles\robocode**
- iii. Two files are created in the process-

1. Battle File: Contains the parameters for the battle defined in it.

#Battle Properties

robocode.battleField.width=800

robocode.battleField.height=600

robocode.battle.numRounds=10

robocode.battle.gunCoolingRate=0.1

robocode.battle.rules.inactivityTime=450

robocode.battle.hideEnemyNames=true

robocode.battle.selectedRobots=developer1.devrobo1*,developer1.robobo*,developer1.terminator*

2. Results File: Contains the results of the battle after execution.

- iv. The Results file is read and the results are returned to the user to view the results of the battle which contains the scores of the battle.
- v. The Results file is parsed to determine the winner to update Robot Ranking after execution of the battle.

c. Play Battle with UI

- i. Executes Battle on the client side by executing java code from a Java Applet.
 - Robocode is a desktop application developed in Java using **Swing**. Swing can be ported to Applet using **JApplet**.
 - The Applet is run by calling a JAR file from the server and the Web Browser downloads and executes the JAR file on the Web Browser.
 - The client computer must have Java Runtime Environment (**JRE**) installed to be able to run and view the Battle.
- ii. The Applet takes selected robots as a parameter is set in the <param> tag which is retrieved from the URL.
- iii. For the process, the server needs to perform 4 Steps-
 1. **Package** the **robots** by copying the robots into the packaging folder at runtime into the JAR file as new robots would keep getting added/updated
 2. Create the JAR file at runtime which contains the robots, robocode classes and resources folder (which contains the images, sounds,etc. for the UI) using **jar cf** command.
 - a. The JAR file is packaged on the following path – **<context path> /battle_applet/packaging/**
 - b. The **unsigned JAR** file will be generated in the following path - **<context path> /battle_applet/packaging/battleapplet.jar**
 - c. The signed JAR file will be generated in the following path - **<context path> /battle_applet/signedbattleapplet.jar**
 3. Generate keystore using **keytool** command for digitally signing the JAR file.
 - a. The keystore is generated on the following path – **C:\ProgFiles\keystore**
 4. Digitally sign the JAR file using **jarsigner** command.

We faced issues sometimes that the jar packaging command doesn't release its handle on the newly created jar file in time so that the next command to digitally sign it may have access to the file.

- iv. An Ajax call is made to the server at the end of the battle to send battle results to the server and update Robot Ranking.

i) Robot Ranking

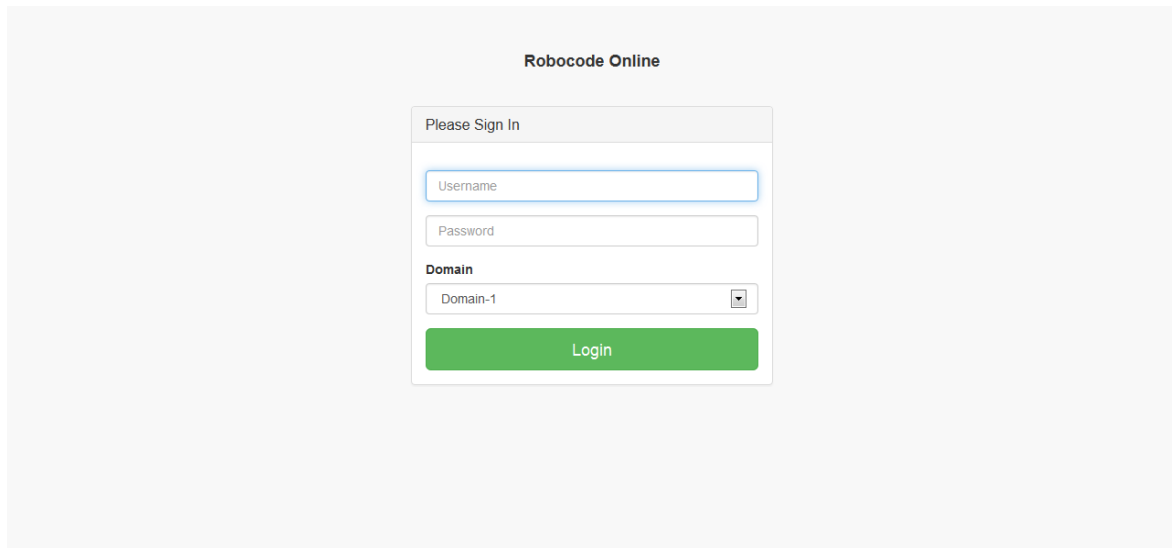
- d. Robot Ranking is updated after each battle is played.
- e. Our ranking algorithm, calculates ranking in the following manner:
 - i. More the number of robots, a robot wins against, the more points it will earn.
 - ii. Eg. If the robot won against 3 robots, it earns 3 points, whereas if the robot wins against 10 robots, it earns 10 points.
 - iii. The more a robot plays a battle, the more points it will earn.
 - iv. All participants of the battle get points in the decrementing order from the winner.

j) All Installation Steps are summarized as below-

- f. **Robocode** - Install Robocode (Desktop Application) on the following path – C:\ProgFiles\robocode
- g. **Keystore Directory** - Create a directory for keystore – C:\ProgFiles\keystore
- h. **Perl Runtime** - Install Perl Runtime on the following path – C:\ProgFiles\Perl64
- i. **MOSS Directory** - Create a directory for MOSS – C:\ProgFiles\apps
- j. **Tomcat** - Tomcat is automatically deployed when the application is deployed using Eclipse with Azure plugin (Tested on Eclipse Kepler).
- k. **JDK 1.7** - JDK 1.7 is required and is automatically deployed when the application is deployed using Eclipse with Azure plugin (Tested on Eclipse Kepler).

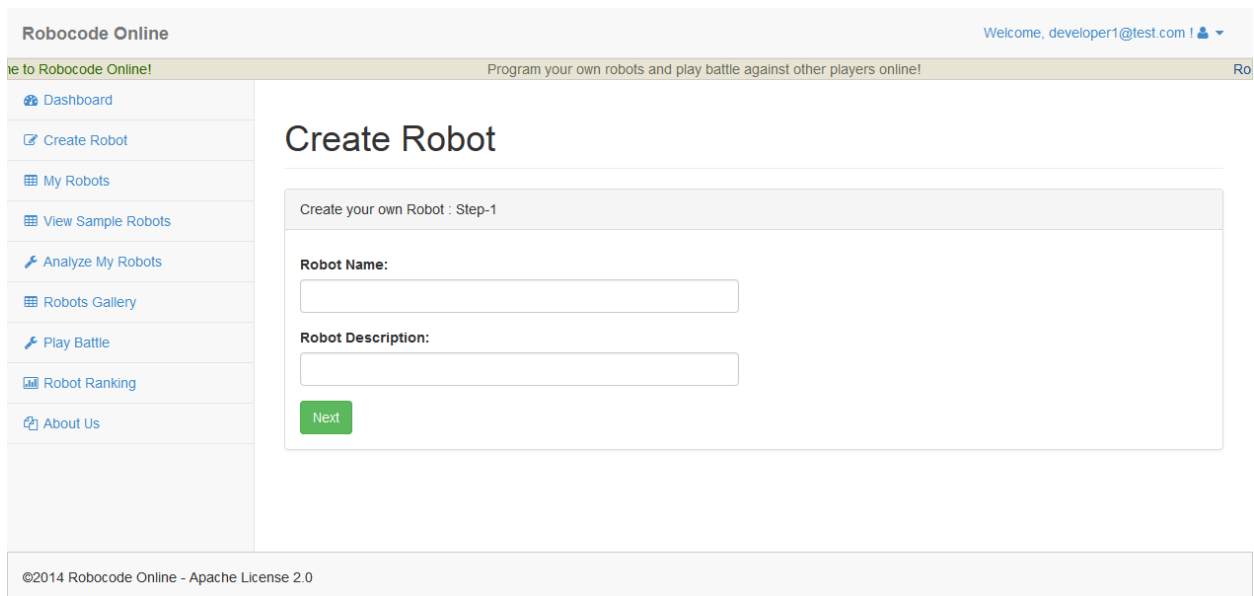
k) Screenshots

○ Login Page



The screenshot shows the Robocode Online login page. It features a central form titled "Please Sign In" with the following fields: a text input for "Username", a text input for "Password", and a dropdown menu for "Domain" currently showing "Domain-1". A green "Login" button is positioned at the bottom of the form. The page header includes the text "Robocode Online".

○ Create Robot : Step-1



The screenshot displays the Robocode Online "Create Robot : Step-1" page. On the left is a sidebar menu with links: Dashboard, Create Robot, My Robots, View Sample Robots, Analyze My Robots, Robots Gallery, Play Battle, Robot Ranking, and About Us. The main content area is titled "Create Robot" and contains a form with the heading "Create your own Robot : Step-1". The form includes a "Robot Name:" label and a text input field, a "Robot Description:" label and a text input field, and a green "Next" button at the bottom. The page header shows "Robocode Online" and a welcome message "Welcome, developer1@test.com!". The footer contains the copyright notice "©2014 Robocode Online - Apache License 2.0".

○ Create Robot : Step-2

Robocode Online

Welcome, developer1@test.com !

Welcome to Robocode Online!

Dashboard

Create Robot

My Robots

View Sample Robots

Analyze My Robots

Robots Gallery

Play Battle

Robot Ranking

About Us

Create Robot

Create your own Robot : Step-2

```
1 package developer1;
2 import robocode.*;
3 //import java.awt.Color;
4
5 // API help : http://robocode.sourceforge.net/docs/robocode/robocode/Robot.html
6
7 /**
8  * Test1 - a robot by (your name here)
9  */
10 public class test extends Robot
11 {
12     /**
13      * run: Test1's default behavior
14      */
15     public void run() {
16         // Initialization of the robot should be put here
17
18         // After trying out your robot, try uncommenting the import at the top,
19         // and the next line:
20
21         // setColors(Color.red,Color.blue,Color.green); // body,gun,radar
22
23         // Robot main loop
24         while(true) {
25             // Replace the next 4 lines with any behavior you would like
26             ahead(100);
27             turnGunRight(360);
28             back(100);
29             turnGunRight(360);
30         }
31     }
32
33     /**
34      * onScannedRobot: what to do when you see another robot
35      */
36     public void onScannedRobot(ScannedRobotEvent e) {
37         // Replace the next line with any behavior you would like
38         // e.setDistance(100);
39     }
40 }
```

Compile

Next

©2014 Robocode Online - Apache License 2.0

○ Create Robot : Step-3

Robocode Online

Welcome, developer1@test.com !

Welcome to Robocode Online! Program your own robots and play

Dashboard

Create Robot

My Robots

View Sample Robots

Analyze My Robots

Robots Gallery

Play Battle

Robot Ranking

About Us

Create Robot

Create your own Robot : Step-3

Assign Rights Per Robot	
VIEWER	Read
DEVELOPER	Read + Update
MANAGER	Read + Update + Delete

Save

©2014 Robocode Online - Apache License 2.0

○ My Robots

Robocode Online

Welcome, developer1@test.com !

Welcome to Robocode Online!

Program

Dashboard
Create Robot
My Robots
View Sample Robots
Analyze My Robots
Robots Gallery
Play Battle
Robot Ranking
About Us

My Robots

List of My Robots

Created Successfully

Robot Name	Description				
devrobo1	basic robo dev				
rambo	rambo				
don	don				
devrobo2	my second robo				
terminator	terminator				
testing_santhosh	test for demo				
test	test				

©2014 Robocode Online - Apache License 2.0

○ View Robot

Robocode Online

Welcome, developer1@test.com !

Welcome to Robocode Online!

Program your own robots a

Dashboard
Create Robot
My Robots
View Sample Robots
Analyze My Robots
Robots Gallery
Play Battle
Robot Ranking
About Us

View Robot

View Robot Code

```

1 package developer1;
2 import robocode.*;
3 //import java.awt.Color;
4
5 // API help : http://robocode.sourceforge.net/docs/robocode/robocode/Robot.html
6
7 /**
8  * Test1 - a robot by (your name here)
9  */
10 public class devrobo1 extends Robot
11 {
12     /**
13      * run: Test1's default behavior
14      */
15     public void run() {
16         // Initialization of the robot should be put here
17
18         // After trying out your robot, try uncommenting the import at the top,
19         // and the next line:
20
21         // setColors(Color.red,Color.blue,Color.green); // body,gun,radar
22
23         // Robot main loop
24         while(true) {
25             // Replace the next 4 lines with any behavior you would like
26             ahead(100);
27             turnGunRight(360);
28             back(100);
29             turnGunRight(360);
30         }
31     }
32
33     /**
34      * onScannedRobot: What to do when you see another robot
35      */
36     public void onScannedRobot(ScannedRobotEvent e) {
37         // Replace the next line with any behavior you would like
38         // e.g.
39         // turnGunRight(180);
40     }
41 }

```

Back

©2014 Robocode Online - Apache License 2.0

○ Edit Robot

Robocode Online

Welcome, developer1@test.com !

Dashboard

Create Robot

My Robots

View Sample Robots

Analyze My Robots

Robots Gallery

Play Battle

Robot Ranking

About Us

Welcome to Robocode Online!

Edit Robot

Edit Robot and Compile it

Compile

```
1 package developer1;
2 import robocode.*;
3 //import java.awt.Color;
4
5 // API help : http://robocode.sourceforge.net/docs/robocode/robocode/Robot.html
6
7 /**
8  * Test1 - a robot by (your name here)
9  */
10 public class devrobo1 extends Robot
11 {
12     /**
13      * run: Test1's default behavior
14      */
15     public void run() {
16         // Initialization of the robot should be put here
17
18         // After trying out your robot, try uncommenting the import at the top,
19         // and the next line:
20
21         // setColors(Color.red,Color.blue,Color.green); // body,gun,radar
22
23         // Robot main loop
24         while(true) {
25             // Replace the next 4 lines with any behavior you would like
26             ahead(100);
27             turnGunRight(360);
28             back(100);
29             turnGunRight(360);
30         }
31     }
32
33     /**
34      * onScannedRobot: What to do when you see another robot
35      */
36     public void onScannedRobot(ScannedRobotEvent e) {
37         // Replace the next line with any behavior you would like
38         // e.g.
39         // turnGunRight(180);
40     }
41 }
```

Save

©2014 Robocode Online - Apache License 2.0

○ My Robot Contributors

Robocode Online

Welcome, developer1@test.com !

Program your own robots and play battle against other players online!

Robocode - Build the Best, Destroy the Rest!

Dashboard

Create Robot

My Robots

View Sample Robots

Analyze My Robots

Robots Gallery

Play Battle

Robot Ranking

About Us

My Robots - Contributors

Contributors for Robot - devrobo1

Name	Domain id	Updated Date
developer1@test.com	1	2014-05-06 16:48:43.083
developer1@test.com	1	2014-05-08 19:57:03.907

©2014 Robocode Online - Apache License 2.0

○ View Sample Robots

Robocode Online

Welcome, developer1@test.com !

Robocode Online! Program your own robots and play battle against other players online!

Dashboard

Create Robot

My Robots

View Sample Robots

Analyze My Robots

Robots Gallery

Play Battle

Robot Ranking

About Us

View Sample Robots

Crazy

```
1 package sample;
2
3 import robocode.*;
4 import java.awt.*;
5
6
7
8 /**
9  * Crazy - a sample robot by Mathew Nelson.
10  *
11  * This robot moves around in a crazy pattern.
12  *
13  * @author Mathew A. Nelson (original)
14  * @author Flemming N. Larsen (contributor)
15  */
16 public class Crazy extends AdvancedRobot {
17     boolean movingForward;
18
19     /**
20      * run: Crazy's main run function
21      */
22     public void run() {
23         // Set colors
24         setBodyColor(new Color(0, 200, 0));
25         setGunColor(new Color(0, 150, 50));
26         setRadarColor(new Color(0, 100, 100));
27         setBulletColor(new Color(255, 255, 100));
28         setScanColor(new Color(255, 200, 200));
29
30         // Loop forever
31         while (true) {
32             // Tell the game we will want to move ahead 40000 -- some large number
33             setAhead(40000);
34             movingForward = true;
35             // Tell the game we will want to turn right 90
36             setTurnRight(90);
37             // At this point, we have indicated to the game that "when we do something",
38             // ...
```

Fire

Interactive

Tracker

VelociRobot

Walls

©2014 Robocode Online - Apache License 2.0

○ Analyze My Robots

Robocode Online

Welcome, developer1@test.com !

Welcome to Robocode Online! Program your own robots and play battle against other players online!

Dashboard

Create Robot

My Robots

View Sample Robots

Analyze My Robots

Robots Gallery

Play Battle

Robot Ranking

About Us

Analyze My Robots

Compare your robot with other robots in your domain

Select Robot: rambo


rambo

Robot Id	Robot Name	Dependency
2038	devrobo1	64.0%
2048	TestMoss	64.0%
2050	Test_Manager	64.0%
2052	don	64.0%
2053	devrobo2	64.0%
2054	terminator	64.0%
2055	OnCLoud	64.0%

©2014 Robocode Online - Apache License 2.0

○ Robots Gallery

Robocode Online

Welcome, developer1@test.com ! 

Dashboard

Create Robot

My Robots

View Sample Robots

Analyze My Robots

Robots Gallery

Play Battle

Robot Ranking

About Us

Welcome to Robocode Online!





































User Profile

Logout




Robots Gallery

Cross Domain List of Robots

Domain-1

Robot Name			
devrobo1			
TestMoss			
Test_Manager			
rambo			
don			
devrobo2			
terminator			
OnCLoud			
testing_santhosh			
santhosh			
test			


Domain-2

Robot Name			
No Data Available			

©2014 Robocode Online - Apache License 2.0

○ Play Battle

Robocode Online

Welcome, developer1@test.com ! 

Dashboard

Create Robot

My Robots

View Sample Robots

Analyze My Robots

Robots Gallery

Play Battle

Robot Ranking

About Us

Welcome to Robocode Online!

Play Battle

Select Robots and Play Battle

My Domain Robots

<input type="checkbox"/>	Robot Name	Developed By
<input type="checkbox"/>	devrobo1	developer1@test.com
<input type="checkbox"/>	TestMoss	manager1@test.com
<input type="checkbox"/>	Test_Manager	manager1@test.com
<input type="checkbox"/>	rambo	developer1@test.com
<input type="checkbox"/>	don	developer1@test.com
<input type="checkbox"/>	devrobo2	developer1@test.com
<input type="checkbox"/>	terminator	developer1@test.com
<input type="checkbox"/>	OnCLoud	manager1@test.com
<input type="checkbox"/>	testing_santhosh	developer1@test.com
<input type="checkbox"/>	santhosh	manager1@test.com
<input type="checkbox"/>	test	developer1@test.com

Start Battle without UI

Start Battle with UI

©2014 Robocode Online - Apache License 2.0

○ Play Battle without UI

©2014 Robocode Online - Apache License 2.0

○ Play Battle without UI – Results

©2014 Robocode Online - Apache License 2.0

○ Play Battle with UI

Robocode Online

Welcome, developer1@test.com ! 📢 ▼

gram your own robots and play battle against other players online!

Robocode - Build the Best, Destroy the Rest!

Dashboard

Create Robot

My Robots

View Sample Robots

Analyze My Robots

Robots Gallery

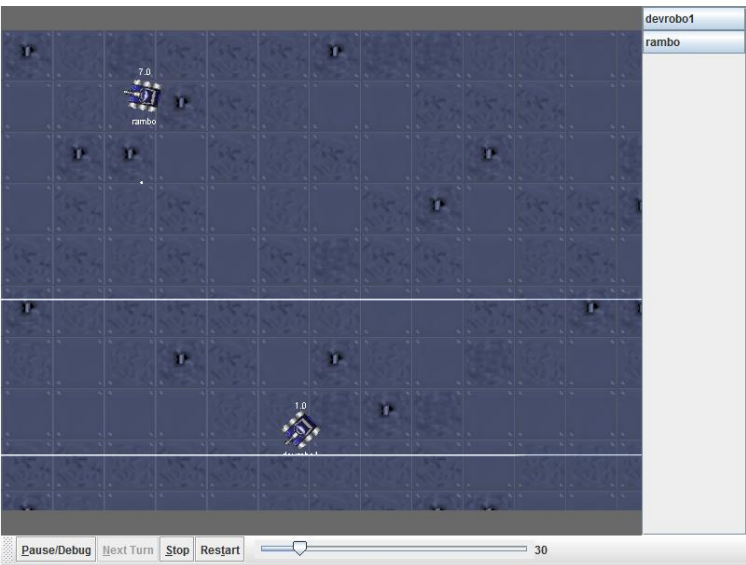
Play Battle

Robot Ranking

About Us

Play Battle with UI

Play Battle with UI on your browser



The screenshot shows the Robocode Battle UI. It features a 20x20 grid battlefield with a dark blue background and a light blue grid. Two robots are visible: 'devrobo1' (a blue robot) and 'rambo' (a red robot). The robot 'rambo' is positioned at approximately (10, 10) and has a health value of 7.0. The robot 'devrobo1' is positioned at approximately (10, 10) and has a health value of 1.0. The UI includes a control bar at the bottom with buttons for 'Pause/Debug', 'Next Turn', 'Stop', and 'Restart', and a progress slider set to 30. A console window at the bottom displays the following text:

```
window Loaded
Robocode Battle
Preparing battle....
className developer1.devrobo1
security true
developer1.devrobo1
robot class false
false class loader
className developer1.rambo
security true
developer1.rambo
robot class false
false class loader
-----
Let the games begin!
Round 1 cleaning up.
=====60.0
=====0.0
-----
Let the games begin!
```

○ Robot Ranking

Robocode Online

Welcome, developer1@test.com ! 👤 ▼

ie! Robocode - Build the Best, Destroy the Rest!

Dashboard

Create Robot

My Robots

View Sample Robots

Analyze My Robots

Robots Gallery

Play Battle

Robot Ranking

About Us

Robot Ranking

Robot Ranking for My Domain

My Domain Robots

Rank	Robot Name	Developed By	Robot Points
1	rambo	developer1	30
2	devrobo1	developer1	27
3	don	developer1	7
4	terminator	developer1	6
5	devrobo2	developer1	3

©2014 Robocode Online - Apache License 2.0

I) References

- GAE Overview: <https://developers.google.com/appengine/docs/whatisgoogleappengine>
- Force.com overview:
http://wiki.developerforce.com/page/Force_Platform_Fundamentals
- Introducing Windows Azure by Mitch Tulloch with Windows Azure Team.
- <http://msdn.microsoft.com/en-us/library/windowsazure/hh690944%28VS.103%29.as>
- http://en.wikipedia.org/wiki/Access_control
- http://robowiki.net/wiki/Robocode/Old_Developers_Guide_for_building_Robocode
- <http://www.windowsazure.com/en-us/solutions/identity>
- http://robowiki.net/wiki/Robocode/Old_Developers_Guide_for_building_Robocode
- <http://cloud-computing.findthebest.com/compare/17-19/Microsoft-vs-salesforce-com>
- <http://msdn.microsoft.com/en-us/library/windowsazure/hh278947.aspx>