Architectural Design Document

Group 4, Inc.

Team Members: Corentin (Corey) Rejaud
Elliot Linder
Joseph (Joey) Kotzker
Deepak Nalla

Introduction

The technological answer to the challenges of distributed programming is modularization." A modular application, in contrast to one monolithic chunk of tightly-coupled code in which every unit may interface directly with any other, is composed of smaller, separated chunks of code that are well isolated. Those chunks can then be developed by separate teams with their own life cycles and their own schedules"(1).

However, to build large programs out of modules effectively, we need to be able to write code modules that we can convince ourselves are correct in isolation from the rest of the program. Rather than have to think about every other part of the program when developing a code module, we need to be able to use local reasoning: that is, reasoning about just the module and the contract it needs to satisfy with respect to the rest of the program. If everyone has done their job, separately developed code modules can be plugged together to form a working program without every developer needing to understand everything done by every other developer in the team. This is the idea of modular programming.(2)

Our overall goal therefore is to make this program as modular as possible by breaking it down into smaller components which each performs basic functions and are called modules. We hope this will allow us to perform local changes instantaneously and easily without worrying about changes to the whole stack and the program.

References:

1) Netbeans Modular Programming
   -https://netbeans.org/project_downloads/usersguide/rcp-book-ch2.pdf
2) Cornell University Lecture Notes -
   http://www.cs.cornell.edu/courses/cs3110/2011sp/lectures/lec07-modules/modules.htm

An Overview of Our Modules-

Modules

- Workflow file interpreter (uploading workflow into the workflow system)
    - Correct file checker - This sub-module checks to see if there is a legal file type attached; one that is supported. It calls on a function check_file(File file) which takes in a file format and checks with nested-if-then statements to check for comparison. We can parse data that is only in JSON , or XML format. An advanced feature is to include Protobuf file parsing.
    - Read proto file or JSON file for data including metadata for the workflow - An additional submodule to read and parse metadata that will be used to store state information
    - GUI Generator from - A Java file that will instantiate FXML objects for the User depending on their input, and selection.

  Workflow State -
    - Assign State - Assign workflow state information
    - Create StateMachine - Create StateMachine of workflow
    - Archive Workflow - You can select workflows to archive
    - Remove Workflow - You can remove or trash workflows that you select
    - DoWorkflow - Interface that gets invoked when Workflow Template is created, and workflow gets initialized

- Action system
    - Send notifications/reminders if certain period of time expires
    - Alert users of the current step of the process when it starts
    - Keep track of the workflow process and state information using iterators
- Users/groups
    - Creating a new user
        - Changing username/personal information
        - Changing user Type/privileges

      Adding/Assigning Role - Add User and Assign Role of ‘Admin’ or ‘End User’
      Removing User/Role - Remove User role of ‘Admin’, or ‘End User’
      Assigning Group - Create group with multiple members
      Setting Domain - Set the domain information of the User/Group

Packages

Package	Description
com.group4inc.wims.actions	
com.group4inc.wims.idm	
com.group4inc.wims.workflow	

Package com.group4inc.wims.actions

Class Summary

Class	Description
Email	This utility class contains the code that sends emails out to the user.
Notification	This class defines the Notification object that is created by the Workflow to handle notifications.
WorkflowDone	Actions to perform when a workflow has completed.

com.group4inc.wims.actions

Class Email

java.lang.Object
  com.group4inc.wims.actions.Email

```
public class Email
extends java.lang.Object
```

This utility class contains the code that sends emails out to the user. Primarily used by the Notification class.

Author:

Elliot Linder (eml160)

See Also:

Notification

Constructor Summary

Constructors

Constructor and Description
`Email()`

Method Summary

All Methods **Static Methods** **Concrete Methods**

Modifier and Type	Method and Description
static boolean	**sendInitialEmail**(java.lang.String emailAddress, java.lang.String subject, java.lang.String body) Sends an "initial" email to the user, i.e.
static boolean	**sendReminderEmail**(java.lang.String emailAddress, java.lang.String subject, java.lang.String body) Sends a "reminder" email to the user, i.e.

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

Email

```
public Email()
```

Method Detail

sendInitialEmail

```
public static boolean sendInitialEmail(java.lang.String emailAddress,
                                       java.lang.String subject,
                                       java.lang.String body)
```

Sends an "initial" email to the user, i.e. an email being sent for the first time.

Parameters:

`emailAddress - the email address to send the email to`

`subject - the subject of the email`

`body - the body of the email`

Returns:

`If the operation completed successfully. TRUE if completed successfully and FALSE is there were errors.`

sendReminderEmail

```
public static boolean sendReminderEmail(java.lang.String emailAddress,
                                        java.lang.String subject,
                                        java.lang.String body)
```

Sends a "reminder" email to the user, i.e. an email being sent for the second+ time. Automatically appends a "REMINDER" tag to the front of the subject.

Parameters:

`emailAddress - the email address to send the email to`

`subject - the subject of the email`

`body - the body of the email`

Returns:

`If the operation completed successfully. TRUE if completed successfully and FALSE is there were errors.`

com.group4inc.wims.actions

Class Notification

java.lang.Object
    com.group4inc.wims.actions.Notification

```
public class Notification
extends java.lang.Object
```

This class defines the Notification object that is created by the Workflow to handle notifications.

A notification is sent to a user when it is their turn to complete a task or if a certain period of time expires (e.g. 4 days after initial notification to complete task).

Author:
Elliot Linder (eml160)

See Also:
Workflow

Constructor Summary

Constructors

Constructor and Description
Notification(java.lang.String name, **User** user, java.lang.Long secToWait, java.lang.String subject, java.lang.String body) Constructor for Notification objects.

Method Summary

All Methods **Instance Methods** **Concrete Methods**

Modifier and Type	Method and Description
java.lang.String	**getBody**() Returns the body of the email.
java.lang.String	**getName**() Returns the name of the notification.
java.lang.Long	**getSecToWait**() Returns the amount of time to wait prior to sending the reminder email.
java.lang.String	**getSubject**() Returns the subject of the email.
User	**getUser**() Returns the User object who needs the notification.
boolean	**sendReminder**() Sends a reminder email to the user after the waiting period defined in secToWait.
void	**setBody**(java.lang.String body) Sets the body of the email.
void	**setName**(java.lang.String name) Sets the name of the notification.
void	**setSecToWait**(java.lang.Long secToWait) Sets the amount of time to wait prior to sending the reminder email.

void	**setSubject**(java.lang.String subject) Sets the subject of the email.
void	**setUser**(**User** user) Sets the User object who needs the notification.

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

Notification

```
public Notification(java.lang.String name,
                    User user,
                    java.lang.Long secToWait,
                    java.lang.String subject,
                    java.lang.String body)
```

Constructor for Notification objects.

Parameters:

name - the name of the Notification to be constructed

user - the User who needs the notification that is being constructed

secToWait - the amount of time to wait in seconds prior to sending a reminder Notification email.

subject - the subject of the Notification email being constructed

body - The body of the notification email being constructed

Method Detail

getName

```
public java.lang.String getName()
```

Returns the name of the notification.

Returns:

the name of the notification

setName

```
public void setName(java.lang.String name)
```

Sets the name of the notification.

Parameters:

name - new name of the notification

getUser

```
public User getUser()
```

Returns the User object who needs the notification.

Returns:

the User object who needs the notification

setUser

```
public void setUser(User user)
```

Sets the User object who needs the notification.

Parameters:

`user - new User object of the notification`

getSecToWait

```
public java.lang.Long getSecToWait()
```

Returns the amount of time to wait prior to sending the reminder email.

Returns:

`the Long of the amount of time to wait`

setSecToWait

```
public void setSecToWait(java.lang.Long secToWait)
```

Sets the amount of time to wait prior to sending the reminder email.

Parameters:

`secToWait - new amount of time to wait`

getSubject

```
public java.lang.String getSubject()
```

Returns the subject of the email.

Returns:

`the String subject line of the email`

setSubject

```
public void setSubject(java.lang.String subject)
```

Sets the subject of the email.

Parameters:

`subject - new String subject line of the email`

getBody

```
public java.lang.String getBody()
```

Returns the body of the email.

Returns:

`the body of the email`

setBody

```
public void setBody(java.lang.String body)
```

Sets the body of the email.

Parameters:

`body - new String body of the email`

sendReminder

```
public boolean sendReminder()
```

Sends a reminder email to the user after the waiting period defined in secToWait.

Returns:

`If the operation completed successfully. TRUE if completed successfully and FALSE is there were errors.`

com.group4inc.wims.actions

Class WorkflowDone

java.lang.Object
    com.group4inc.wims.actions.WorkflowDone

```
public class WorkflowDone
extends java.lang.Object
```

Actions to perform when a workflow has completed.

This class defines actions that can be performed once a workflow has completed.

A workflow can either be archived for historical purposes or be cleared (deleted from memory).

Author:

Elliot Linder (eml160)

See Also:

Workflow

Constructor Summary

Constructors

Constructor and Description
WorkflowDone()

Method Summary

All Methods | **Static Methods** | **Concrete Methods**

Modifier and Type	Method and Description
static boolean	archiveWorkflow(Workflow workflow) Archives the workflow.
static boolean	clearWorkflowData(Workflow workflow) Clears the data from the workflow from the system.

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

WorkflowDone

```
public WorkflowDone()
```

Method Detail

clearWorkflowData

`public static boolean clearWorkflowData(Workflow workflow)`

Clears the data from the workflow from the system.

Returns:

`If the operation completed successfully. TRUE if completed successfully and FALSE is there were errors.`

archiveWorkflow

`public static boolean archiveWorkflow(Workflow workflow)`

Archives the workflow.

Returns:

`If the operation completed successfully. TRUE if completed successfully and FALSE is there were errors.`

Package com.group4inc.wims.idm

Class Summary

Class	Description
Admin	
Domain	Domain Object.
EndUser	
Group	Group Object.
IdMSerDB	Object to act as a database for the Identiy Management portion of the program.
User	User Object.

com.group4inc.wims.idm

Class Admin

java.lang.Object
  com.group4inc.wims.idm.User
    com.group4inc.wims.idm.Admin

```
public class Admin
extends User
```

Constructor Summary

Constructors

Constructor and Description
`Admin(java.lang.String name, java.lang.String email, java.lang.String username, java.lang.String password, java.lang.String initdomain)`

Method Summary

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method and Description
`java.util.List<WorkflowTemplate>`	**`getWorkflowTemplates`**`()` Get the list of workflow templates
`java.lang.String`	**`isWorkflowLanguageValid`**`(JSONObject template)` Check if the workflow language is in a valid format
`void`	**`uploadWorkflow`**`(JSONObject template)` Upload a workflow template using the workflow langauge
`void`	**`viewPendingUserInvites`**`()` View the list of pending invites from users that want to join the admin's domain

Methods inherited from class com.group4inc.wims.idm.User

`addDomain, changeEmail, changeName, changePassword, getDomain, getEmail, getName, getPassword, getUsername, removeRole`

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

Admin

```
public Admin(java.lang.String name,
             java.lang.String email,
             java.lang.String username,
             java.lang.String password,
             java.lang.String initdomain)
```

Method Detail

getWorkflowTemplates

```
public java.util.List<WorkflowTemplate> getWorkflowTemplates()
```

Get the list of workflow templates

Returns:

list of workflow templates

viewPendingUserInvites

```
public void viewPendingUserInvites()
```

View the list of pending invites from users that want to join the admin's domain

uploadWorkflow

```
public void uploadWorkflow(JSONObject template)
```

Upload a workflow template using the workflow langauge

Parameters:

template - - the workflow language

isWorkflowLanguageValid

```
public java.lang.String isWorkflowLanguageValid(JSONObject template)
```

Check if the workflow language is in a valid format

Parameters:

template - - the workflow language

Returns:

empty if valid, otherwise gives errors

com.group4inc.wims.idm

Class Domain

java.lang.Object
  com.group4inc.wims.idm.Domain

---

public class **Domain**
extends java.lang.Object

Domain Object.

This class defines Roles in the context of WIMS.

Roles are a way to give users permissions. A role can be assigned to a user in two different ways; one by direct assignment to the User object and the other by group assignment. For example, if a user is a member of the "overriders" group then they and all other members of the group could be assigned the "all_overriders" role in one assignment instead of individual assignment.

Author:

Elliot Linder (eml160)

See Also:

User, Group

Constructor Summary

Constructors

Constructor and Description
Domain(java.lang.String name) Constructor for Role objects.

Method Summary

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method and Description
java.lang.String	**getName**() Returns the Role's name.
java.util.ArrayList<User>	**getUsersInDomain**() Returns an ArrayList of **individual** Users with the role.

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

Domain

public Domain(java.lang.String name)

Constructor for Role objects.

Parameters:

name - the name of the Role to be constructed

Method Detail

getName

public java.lang.String getName()

Returns the Role's name.

Returns:

the name of the Role

getUsersInDomain

public java.util.ArrayList<User> getUsersInDomain()

Returns an ArrayList of **individual** Users with the role.

Returns:

the ArrayList of Users with the role.

com.group4inc.wims.idm

Class EndUser

java.lang.Object
  com.group4inc.wims.idm.User
    com.group4inc.wims.idm.EndUser

```
public class EndUser
extends User
```

Constructor Summary

Constructors

Constructor and Description
`EndUser(java.lang.String name, java.lang.String email, java.lang.String username, java.lang.String password, java.lang.String initdomain)`

Method Summary

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method and Description
`java.util.List<WorkflowInstance>`	`getActiveWorkflows()` Get the list of all active workflows pertaining to this user

Methods inherited from class com.group4inc.wims.idm.User

`addDomain, changeEmail, changeName, changePassword, getDomain, getEmail, getName, getPassword, getUsername, removeRole`

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

EndUser

```
public EndUser(java.lang.String name,
               java.lang.String email,
               java.lang.String username,
               java.lang.String password,
               java.lang.String initdomain)
```

Method Detail

getActiveWorkflows

```
public java.util.List<WorkflowInstance> getActiveWorkflows()
```

Get the list of all active workflows pertaining to this user

Returns:

list of active workflows

com.group4inc.wims.idm

Class Group

java.lang.Object
  com.group4inc.wims.idm.Group

```
public class Group
extends java.lang.Object
```

Group Object.

This class defines Groups in the context of WIMS.

Groups are used in WIMS to easily manage multiple User objects. For example, an "overriders" group can be defined for those users who can override steps in the workflow process.

Author:

Elliot Linder (eml160)

See Also:

User, Domain

Constructor Summary

Constructors

Constructor and Description
`Group(java.lang.String name)` Constructor for Group objects.
`Group(java.lang.String name, java.lang.String initmem)` Constructor for Group objects.

Method Summary

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method and Description
`void`	`addMember(User user)` Adds a user as a member of the Group.
`void`	`addRole(Domain role)` Adds a role to the Group.
`java.util.ArrayList<User>`	`getMembers()` Returns an ArrayList of Users containing the Group's members.
`java.lang.String`	`getName()` Returns the Group's name.
`void`	`removeMember(User user)` Removes a User member from the group.
`void`	`removeRole(Domain role)` Removes a role from the Group.

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

Group

public Group(java.lang.String name)

Constructor for Group objects. This also instantiates empty members and roles ArrayLists.

Parameters:

name - the name of the Group to be constructed

Group

public Group(java.lang.String name,
java.lang.String initmem)

Constructor for Group objects. This also instantiates the members ArrayList with a member. An empty roles ArrayList is instantiated.

Parameters:

name - the name of the Group to be constructed

initmem - the username of the User to be added to the Group

Method Detail

getName

public java.lang.String getName()

Returns the Group's name.

Returns:

the name of the Group

getMembers

public java.util.ArrayList<User> getMembers()

Returns an ArrayList of Users containing the Group's members.

Returns:

the ArrayList of User's in the Group.

addMember

public void addMember(User user)

Adds a user as a member of the Group.

Parameters:

user - The User object to be added to the Group.

See Also:

User

removeMember

```
public void removeMember(User user)
```

Removes a User member from the group.

Parameters:

`user` - The User object to be removed from the Group.

See Also:

`User`

addRole

```
public void addRole(Domain role)
```

Adds a role to the Group.

Parameters:

`role` - The Role object to be added to the Group.

See Also:

`Domain`

removeRole

```
public void removeRole(Domain role)
```

Removes a role from the Group.

Parameters:

`role` - The Role object to be removed from the Group.

See Also:

`Domain`

com.group4inc.wims.idm

Class IdMSerDB

java.lang.Object
    com.group4inc.wims.idm.IdMSerDB

```
public class IdMSerDB
extends java.lang.Object
```

Object to act as a database for the Identiy Management portion of the program.

This class acts as a database, containing multiple ArrayList objects, one for all User objects, another for all Group objects, and the final for all Role objects.

Author:
Elliot Linder (eml160)

See Also:
User, Domain, Group

Constructor Summary

Constructors

Constructor and Description
IdMSerDB()

Method Summary

All Methods | **Static Methods** | **Concrete Methods**

Modifier and Type	Method and Description
static void	addDomainToDomainDB(Domain domain) Adds a Role object to the RoleDB (ArrayList of Role objects).
static void	addGroupToGroupDB(Group group) Adds a Group object to the GroupDB (ArrayList of Group objects).
static void	addUserToUserDB(User user) Adds a User object to the UserDB (ArrayList of User objects).
static Domain	getDomainByName(java.lang.String domainname) Returns a User object after searching by the User's username property.
static java.util.ArrayList<Domain>	getDomainDB() Returns the RoleDB ArrayList (ArrayList of all Role objects).
static java.util.ArrayList<Group>	getGroupDB() Returns the GroupDB ArrayList (ArrayList of all Group objects).
static User	getUserByUsername(java.lang.String username) Returns a User object after searching by the User's username property.
static java.util.ArrayList<User>	getUserDB() Returns the UserDB ArrayList (ArrayList of all User objects).
static void	removeGroupFromGroupDB(Group group) Removes a Group object from the GroupDB (ArrayList of Group objects).

static void	**removeRoleFromRoleDB**(Domain role) Removes a Role object from the Role (ArrayList of Role objects).
static void	**removeUserFromUserDB**(User user) Removes a User object from the UserDB (ArrayList of User objects).

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

IdMSerDB

```
public IdMSerDB()
```

Method Detail

addUserToUserDB

```
public static void addUserToUserDB(User user)
```

Adds a User object to the UserDB (ArrayList of User objects).

Parameters:

user - The User object to be added to the UserDB

See Also:

User

removeUserFromUserDB

```
public static void removeUserFromUserDB(User user)
```

Removes a User object from the UserDB (ArrayList of User objects).

Parameters:

user - The User object to be removed from the UserDB

See Also:

User

getUserByUsername

```
public static User getUserByUsername(java.lang.String username)
```

Returns a User object after searching by the User's username property. Will return NULL if no match is found, case-sensitive search!

Parameters:

username - The username of the User object to be retrieved.

Returns:

the User object which has the username that was being searched.

See Also:

User

getDomainByName

`public static Domain getDomainByName(java.lang.String domainname)`

Returns a User object after searching by the User's username property. Will return NULL if no match is found, case-sensitive search!

Parameters:
`username - The username of the User object to be retrieved.`

Returns:
`the User object which has the username that was being searched.`

See Also:
`User`

addGroupToGroupDB

`public static void addGroupToGroupDB(Group group)`

Adds a Group object to the GroupDB (ArrayList of Group objects).

Parameters:
`group - The Group object to be added to the GroupDB.`

See Also:
`Group`

removeGroupFromGroupDB

`public static void removeGroupFromGroupDB(Group group)`

Removes a Group object from the GroupDB (ArrayList of Group objects).

Parameters:
`group - The Group object to be removed from the GroupDB`

See Also:
`Group`

addDomainToDomainDB

`public static void addDomainToDomainDB(Domain domain)`

Adds a Role object to the RoleDB (ArrayList of Role objects).

Parameters:
`role - The Role object to be added to the RoleDB`

See Also:
`Domain`

removeRoleFromRoleDB

`public static void removeRoleFromRoleDB(Domain role)`

Removes a Role object from the Role (ArrayList of Role objects).

Parameters:
`role - The Role object to be removed from the RoleDB`

See Also:
`Domain`

getUserDB

```
public static java.util.ArrayList<User> getUserDB()
```

Returns the UserDB ArrayList (ArrayList of all User objects).

Returns:

An ArrayList of all Users created.

See Also:

User

getGroupDB

```
public static java.util.ArrayList<Group> getGroupDB()
```

Returns the GroupDB ArrayList (ArrayList of all Group objects).

Returns:

An ArrayList of all Groups created.

See Also:

Group

getDomainDB

```
public static java.util.ArrayList<Domain> getDomainDB()
```

Returns the RoleDB ArrayList (ArrayList of all Role objects).

Returns:

An ArrayList of all Roles created.

See Also:

Domain

com.group4inc.wims.idm

Class User

java.lang.Object
  com.group4inc.wims.idm.User

Direct Known Subclasses:

`Admin, EndUser`

```
public class User
extends java.lang.Object
```

User Object.

This class defines Users in the context of WIMS.

Individual users are required in WIMS for accessing the system to perform tasks.

Author:

`Elliot Linder (eml160)`

Constructor Summary

Constructors

Constructor and Description
`User(java.lang.String name, java.lang.String email, java.lang.String username, java.lang.String password, java.lang.String initdomain)` Constructor for User objects.

Method Summary

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method and Description
`void`	`addDomain(Domain domain)` Adds a user to a Domain.
`void`	`changeEmail(java.lang.String email)` Changes the user's email address
`void`	`changeName(java.lang.String name)` Changes the user's name (not to be confused with the username).
`void`	`changePassword(java.lang.String password)` Changes the user's password.
`java.util.ArrayList<Domain>`	`getDomain()` Returns the User's Domain membership.
`java.lang.String`	`getEmail()` Returns the user's email address.
`java.lang.String`	`getName()` Returns the user's name (not to be confused with the username).
`java.lang.String`	`getPassword()` Returns the user's password.

java.lang.String	**getUsername**() Returns the user's username (not to be confused with the name).
void	**removeRole**(Domain domain) Removes a User from a Domain.

Methods inherited from class java.lang.Object

equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

User

```
public User(java.lang.String name,
            java.lang.String email,
            java.lang.String username,
            java.lang.String password,
            java.lang.String initdomain)
```

Constructor for User objects.

Parameters:

name - the name of the User to be constructed

email - the email address of the User to be constructed

username - the username of the User tot be constructed

password - the password of the User to be constructed

initdomain - the initial domain that the user is a part of

Method Detail

getName

```
public java.lang.String getName()
```

Returns the user's name (not to be confused with the username).

Returns:

the name of the User

changeName

```
public void changeName(java.lang.String name)
```

Changes the user's name (not to be confused with the username).

Parameters:

name - the new name of the User

getEmail

```
public java.lang.String getEmail()
```

Returns the user's email address.

Returns:

the email of the User

changeEmail

```
public void changeEmail(java.lang.String email)
```

Changes the user's email address

Parameters:

`email - the new email of the User`

getUsername

```
public java.lang.String getUsername()
```

Returns the user's username (not to be confused with the name).

Returns:

`the username of the User`

getPassword

```
public java.lang.String getPassword()
```

Returns the user's password.

Returns:

`the password of the User`

changePassword

```
public void changePassword(java.lang.String password)
```

Changes the user's password.

Parameters:

`password - the new password to be set for the User`

addDomain

```
public void addDomain(Domain domain)
```

Adds a user to a Domain.

Parameters:

`domain - The Domain object that the User is to be added to.`

See Also:

`Domain`

removeRole

```
public void removeRole(Domain domain)
```

Removes a User from a Domain.

Parameters:

`domain - The Domain object that the User is being removed from.`

See Also:

`Domain`

getDomain

```
public java.util.ArrayList<Domain> getDomain()
```

Returns the User's Domain membership.

Returns:

an ArrayList of Domains that the user has been added to.

See Also:

Domain

Package com.group4inc.wims.workflow

Class Summary

Class	Description
WorkflowInstance	This class represents an instance of a workflow from a workflow template.
WorkflowState	This class represents a state inside a workflow state machine.
WorkflowStateMachine	This class is the state machine for a workflow.
WorkflowTemplate	This class represents a workflow template.

com.group4inc.wims.workflow

Class WorkflowInstance

java.lang.Object
  com.group4inc.wims.workflow.WorkflowInstance

All Implemented Interfaces:

`java.io.Serializable`

```
public class WorkflowInstance
extends java.lang.Object
implements java.io.Serializable
```

This class represents an instance of a workflow from a workflow template. A workflow instance will have current metadata, a state machine to follow, and the list of current states.

Author:

crejaud

See Also:

Serialized Form

Constructor Summary

Constructors

Constructor and Description
`WorkflowInstance(WorkflowStateMachine fsm)` The constructor for a workflow instance.

Method Summary

All Methods | **Instance Methods** | **Concrete Methods**

Modifier and Type	Method and Description
void	`overrideState()` Override a step in the workflow instance.

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

WorkflowInstance

```
public WorkflowInstance(WorkflowStateMachine fsm)
```

The constructor for a workflow instance. Will set the current states to the first state and initialize metadata.

Parameters:

fsm - - The workflow state machine

Method Detail

overrideState

```
public void overrideState()
```

Override a step in the workflow instance. Should only be accessible from the owner of a workflow instance.

com.group4inc.wims.workflow

Class WorkflowState

java.lang.Object
  com.group4inc.wims.workflow.WorkflowState

All Implemented Interfaces:

`java.io.Serializable`

```
public class WorkflowState
extends java.lang.Object
implements java.io.Serializable
```

This class represents a state inside a workflow state machine. Each workflow state will keep track of scenes for each role in the workflow.

Author:

crejaud

See Also:

Serialized Form

Constructor Summary

Constructors

Constructor and Description
`WorkflowState(JSONObject template)` The constructor for a workflow state inside a workflow state machine

Method Summary

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

WorkflowState

```
public WorkflowState(JSONObject template)
```

The constructor for a workflow state inside a workflow state machine

Parameters:

`template - - the json object that the state should follow`

com.group4inc.wims.workflow

Class WorkflowStateMachine

java.lang.Object
    com.group4inc.wims.workflow.WorkflowStateMachine

All Implemented Interfaces:

`java.io.Serializable`

```
public class WorkflowStateMachine
extends java.lang.Object
implements java.io.Serializable
```

This class is the state machine for a workflow. It holds a list of workflow states.

Author:

`creja_000`

See Also:

`Serialized Form`

Constructor Summary

Constructors

Constructor and Description
`WorkflowStateMachine(JSONObject template)` The constructor for the workflow state machine

Method Summary

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

WorkflowStateMachine

`public WorkflowStateMachine(JSONObject template)`

The constructor for the workflow state machine

Parameters:

`template - - the json template from the workflow programmer`

com.group4inc.wims.workflow

Class WorkflowTemplate

java.lang.Object
  com.group4inc.wims.workflow.WorkflowTemplate

All Implemented Interfaces:

`java.io.Serializable`

```
public class WorkflowTemplate
extends java.lang.Object
implements java.io.Serializable
```

This class represents a workflow template. A domain has many workflow templates, however a workflow template does not need to know about a domain. This class has quick a bit of information. This class knows about the connection of usernames to roles. It also knows whi instances as well as which users are owners according to their role. This class also builds the workflow state machine to be passed to workflow instances.

Author:

crejaud

See Also:

Serialized Form

Constructor Summary

Constructors

Constructor and Description
`WorkflowTemplate(JSONObject template)` The constructor for a workflow template.

Method Summary

All Methods | Instance Methods | Concrete Methods

Modifier and Type	Method and Description
void	`addUserToRole(java.lang.String username, java.lang.String role)` This will add a username to a specific role in case it was not specified in the initial creation of the workflow template.
void	`instantiate()` This will create a workflow instance.
void	`removeUserFromTemplate(java.lang.String username)` This will remove a username from whatever role it belongs to in this workflow template.

Methods inherited from class java.lang.Object

`equals, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Constructor Detail

WorkflowTemplate

```
public WorkflowTemplate(JSONObject template)
```

The constructor for a workflow template. Will generate the finite state machine and fill all maps pertaining to the information in the json file.

Parameters:

`template - - the json file from the workflow programmer`

Method Detail

instantiate

```
public void instantiate()
```

This will create a workflow instance. Can only be called from owners.

addUserToRole

```
public void addUserToRole(java.lang.String username,
                          java.lang.String role)
```

This will add a username to a specific role in case it was not specified in the initial creation of the workflow template.

Parameters:

`username - - the enduser's username`

`role - - the new role of the enduser`

removeUserFromTemplate

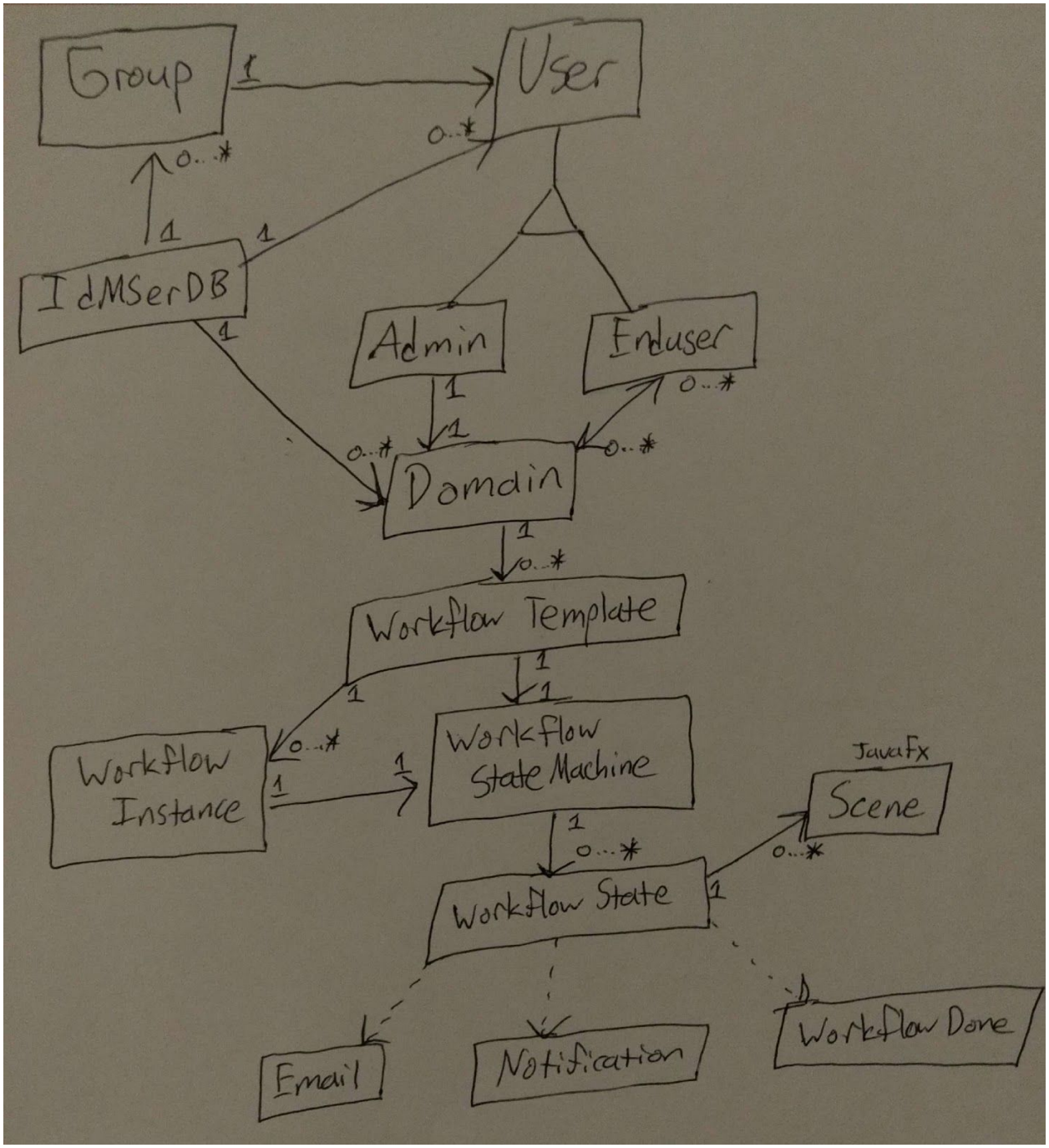
```
public void removeUserFromTemplate(java.lang.String username)
```

This will remove a username from whatever role it belongs to in this workflow template.

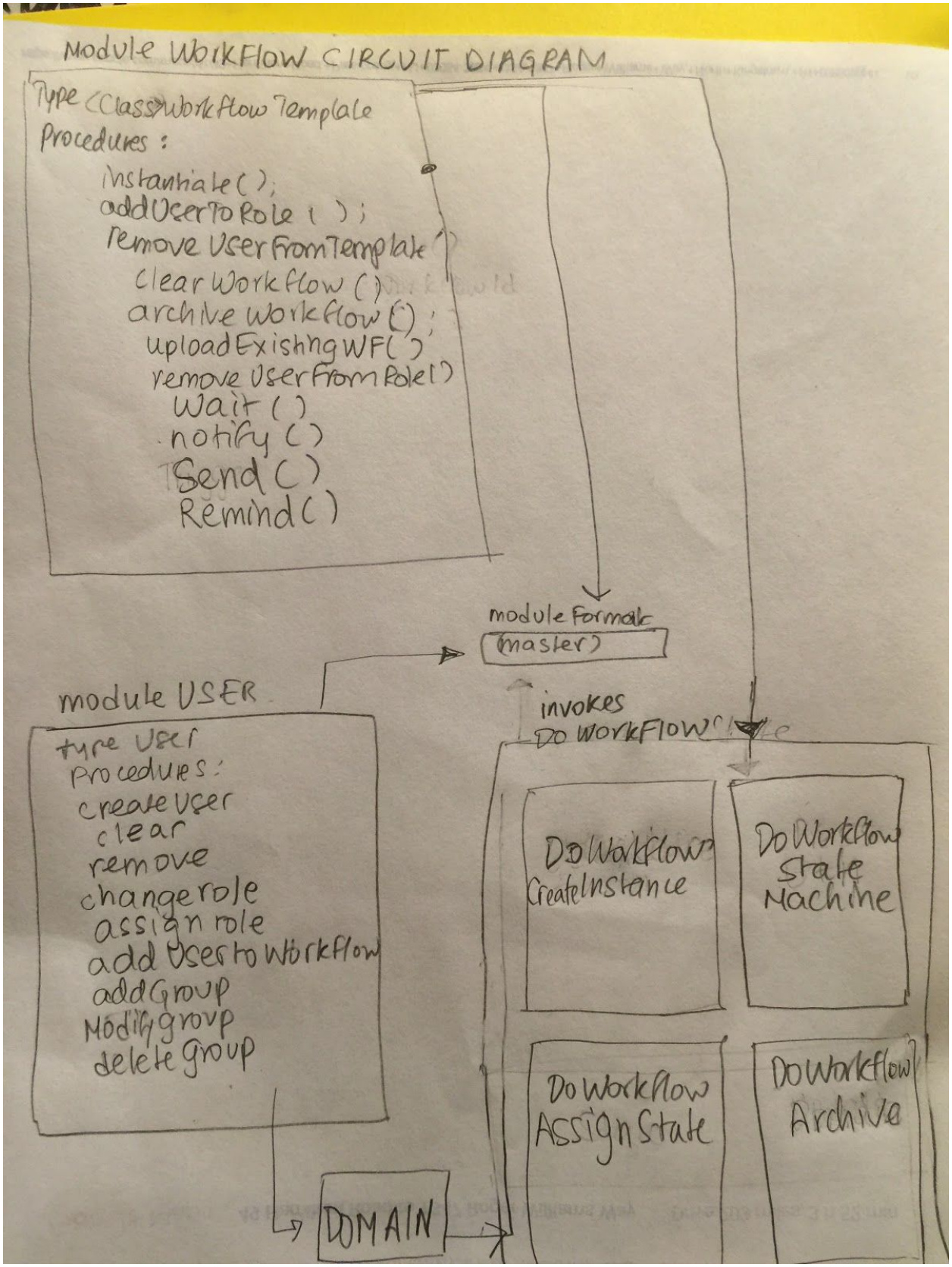
Parameters:

`username - - the enduser's username`

UML Diagram:



The User class is connected in direct relation where it can spawn a different type of User (admin/end user) which then goes in to their specific domain before going into the Workflow modules.


Module Workflow CIRCUIT DIAGRAM
Type (Class) Workflow Template
Procedures:
Instantiate();
addUserToRole();
removeUserFromTemplate()
clearWorkflow()
archiveWorkflow();
uploadExistingWF()
removeUserFromRole()
Wait()
notify()
Send()
Remind()
module Formal
(master)
module USER
type User
Procedures:
createUser
clear
remove
changerole
assign role
add User to Workflow
addGroup
Modify group
delete Group
invokes
DoWorkFlow
DoWorkflow CreateInstance
DoWorkflow State Machine
DoWorkflow AssignState
DoWorkflow Archive
DOMAIN

The Workflow Template module invokes the DoWorkflow chain of the circuit which sets off multiple Workflow routines and their subroutines which allows for the overall modularity of the project. User(s) will be object types that can be iterated and searched for within groups or domains when assigned to a workflow.

Acknowledgments

Packages:

- IDM: Elliot / Corentin
- Workflow: Corentin
- Actions: Elliot
- Introduction: Deepak
- Circuit Diagram: Deepak
- UML Diagram: Corentin
- Professor Borgida for his guidance