Ecole Nationale Supérieure des Mines
SAINT-ETIENNE

Projet :

Multi-Agent Oriented Programming

Réaliser par:

Nicolas Djambazian

Thami Inaflas

Encadrer par :

M.Olivier BOISSIER

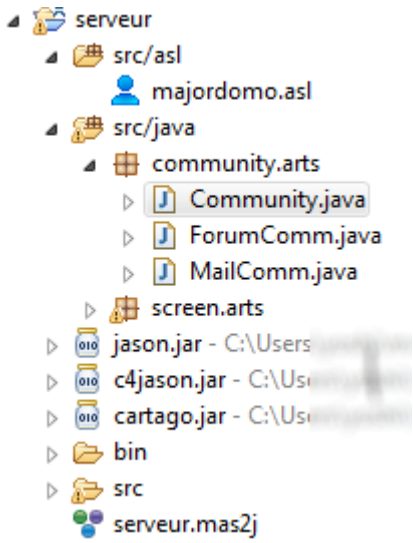
2013-2014

Commentaires sur le projet

Le projet réalisé comporte deux parties :

- Un client
- Un serveur

Le serveur est structuré de la façon suivante :



L'agent majordomo.asl s'occupe de créer l'espace de travail qui sera le serveur contenant les actions que pourront effectuer les assistants (les agents clients) :

```
!setup_and_monitor.

+!setup_and_monitor
    <-  createWorkspace("server");
        joinWorkspace("server",Id);
        !setupArtifacts.
```

Description de l'artifact Community

Les artifacts sont des objets se trouvent dans l'environnement et qu'un agent peut manipuler afin de faire de faire actions.

Community est l'artifact de base sur lequel se baseront 'ForumCommunity' et 'MailCommunity' car il contient certaines actions communes à toute communauté comme la création, le suivi la destruction etc...

Voici un aperçu des fonctions et des composantes de l'artifact Communauté :

`protected LinkedList<AgentId> m_followers ;`	Liste des follower de la communauté selectionnée
`protected AgentId m_owner ;`	Identifiant du créateur de la communauté
`protected String m_type ;`	String contenant le nom du type de communauté '

Init est lancer lors de l'instanciation de l'artifact et permet de remplir les différents paramètres comme le type de la communauté, l'identifiant de l'agent qui l'a créé, la liste des followers (contenant seulement le créateur pour le moment).

```
void init(String type) {
	m_type = type ;
	m_owner =  getCreatorId() ;
	m_followers = new LinkedList<AgentId>() ;
	m_followers.add( m_owner );
}
```

Lorsqu'un utilisateur essaie de rejoindre une communauté son identifiant est récupéré à l'aide de getOpUserId puis le reste des utilisateurs reçoivent un signal les informant de cet évènement.

```
@OPERATION void follow() {
	AgentId src = this.getOpUserId() ;
	if(!m_followers.contains(src))
		m_followers.add( src );
	signal( "arti", "newFollower", src );

}
```

Permet à un utilisateur de ce désabonner d'une communauté qu'il suit.

```
@OPERATION void stopfollow() {
	AgentId src = this.getOpUserId() ;

	if (src == m_owner)
		deleteComm();
	else
		m_followers.remove(src);
}
```

Informe les follower d'une communauté que celle-ci sera supprimée

```
@OPERATION void deleteComm() {

	AgentId src = this.getOpUserId() ;

	if (src == m_owner){
		m_followers.clear();
	}

	signal( "arti", "willBeDeleted" );
}
```

Permet de récupérer le type de la communauté qui pourra être utilisé par un agent afin de savoir qu'elle type d'action il peut effectuer

```
@OPERATION void getType( OpFeedbackParam<String> type ) {
	type.set(m_type);
}
```

Permet d'envoyer un message public à partir d'un agent vers tous ceux qui ont un focus sur la communauté.

```
protected void sendPublicMessage( AgentId from ) {
	signal("arti", "newMessage", from);
}
```

Permet d'envoyer un message privée à un autre agent

```
protected void sendPrivateMessage( AgentId from, AgentId to ) {
	signal(to, "arti", "newMessage", from);
}
```

Permet d'envoyer un message privé à un agent identifié par un nom

```
protected void sendPrivateMessage( AgentId from, String to ) {

	AgentId toId = findFollowerByName(to );
	signal(toId, "arti", "newMessage", from);
}
```

Permet de retourne l'identifiant d'un agent qui suit cette communauté en prenant comme paramètre le nom de cet agent.

```
protected AgentId findFollowerByName( String name ) {

	ListIterator<AgentId> ite = m_followers.listIterator(0);
	while( ite.hasNext() ) {
		AgentId currname = ite.next();

		if( currname.getAgentName() == name )
			return currname ;
	}
	return null ;
}
```

Les deux artifacts MailComm et ForumComm se basent ensuite sur Community pour implémenter le reste des fonctions comme celles de l'affichage de l'ensemble des messages dans un forum etc . . .

Ces dernières permettent aux agents (côté client) d'exécuter les tâches liées à l'exploitation des communautés, depuis leur création jusqu'à leur destruction.