Unraveling ADF Framework

2022 WORKSHOP ON AI AND SIMULATION FOR NATURAL DISASTER MANAGEMENT

JULY 17, 2022

Agenda

- Introduction
- Installation & Execution
- Modules
- Configuration
- Tactics

Introduction

Introduction

- Agent Development Framework (ADF) is a modular agent-development framework that facilitates the development of agents for the Rescue Simulator
- ADF was developed by Shumki Takami (University of Tsukuba, Aichi Institute of Technology, Japan) supervised by Nobuhiro Ito (Aichi Institute of Technology, Japan)
- ADF project was transferred to RoboCup Rescue Agent Simulation ownership on October, 2017
  - Shumki Takami continues sporadically contributing to the project

Introduction

- ADF is structure in two repositories
  - **adf-core-java**
    - Define the ADF modules and provide their default implementation
  - **adf-sample-agent-java**
    - Implement a RoboCup Rescue simulation team and replace some of the modules' default implementation

Installation and Execution

Installation

- Software – Prerequisites
  - Git
  - OpenJDK Java 17
  - Gradle
- Download

```
$ git clone https://github.com/roborescue/adf-sample-agent-java.git
```

- Compile

```
$ cd adf-sample-agent-java
$ ./gradlew clean build
```

Folder Structure

/build	Java classes
/config	configuration files
/docs	documentation
/lib	additional libraries
/logs	execution logs
/precomp_data	precomputation data storage
/src	ADF source-code

Running

- First, run the Rescue Simulator

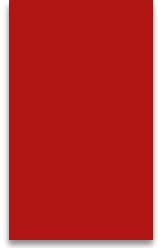
```
$ cd rcrs-server/scripts
$ ./start.sh –m ../maps/kobe/map –c ../maps/kobe/config
```

- Run the ADF Sample Agent

```
$ cd adf-sample-agent-java
$ ./launch.sh -all
```

Hands-on

- Install the Rescue Simulator and ADF Sample Agent
- Execute the ADF Sample Agent on the default Kobe scenario on the Rescue Simulator distribution



Modules

ADF Core Packages

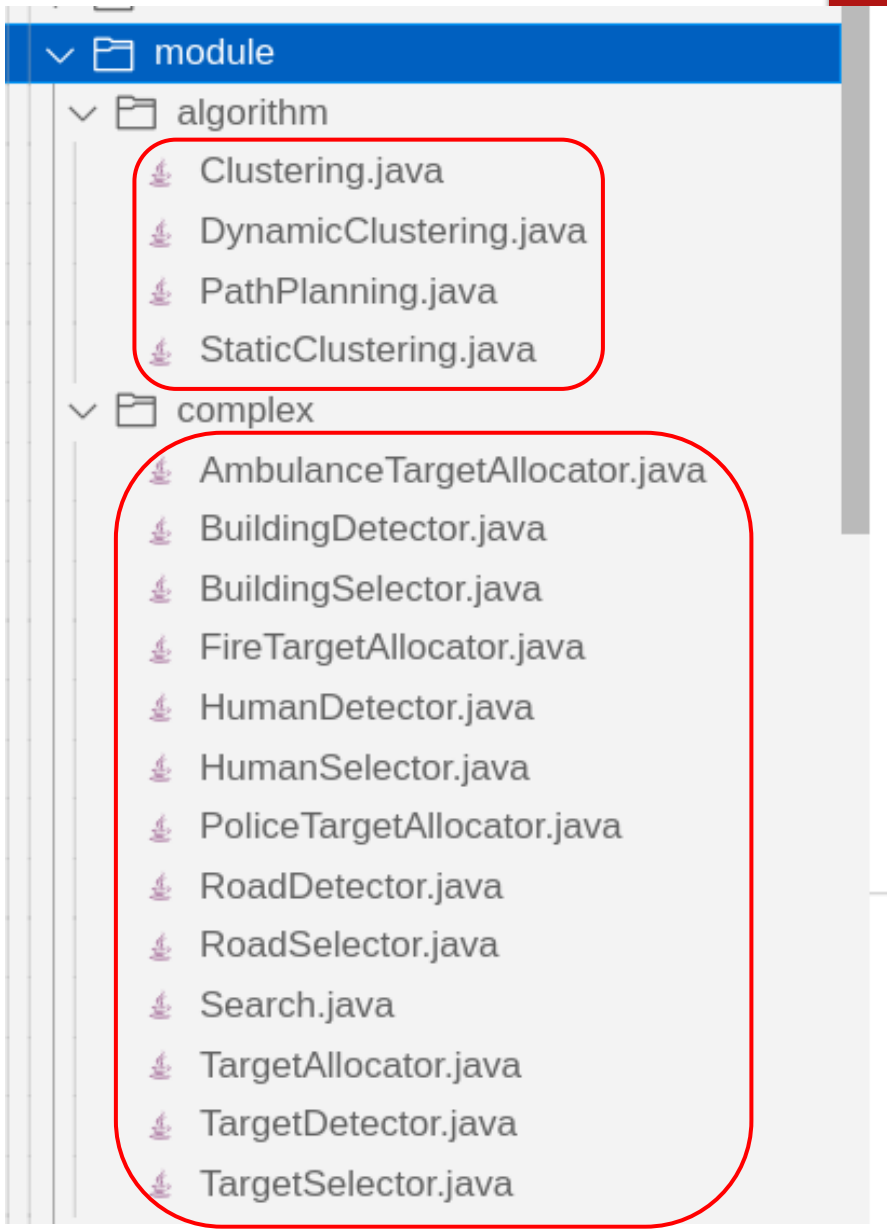
- Split in two major packages
  - **adf.core**
    - Functioning of the ADF
    - Definition of the extensible modules
  - **adf.impl**
    - Default implementation of the extensible modules

Modularity

- General algorithms
  - Path Planning and Clustering
- Agents' operational algorithms
  - Search, Detector, Target Allocator and Selector
- Agents' tactical algorithms
  - Platoon and Center tactics
- Implementation of these modules requires extending ADF Core module classes

Types of Modules

- **adf.core.component.module.algorithm**
  - PathPlanning
  - DynamicClustering
  - StaticClustering
- **adf.core.component.module.complex**
  - Detector
  - Selector
  - Target Allocator



Default Implementation

- ADF Core implementations
  - adf.impl.module.algorithm
  - adf.impl.module.complex
  - adf.impl.module.comm
- Used if a module is not specified in the module configuration
  - No need to implement all modules
  - Avoid any error if the configuration is not properly done

robocuprescue > workspace > adf-core-java > src > main > java > adf > impl > module > algorithm >

```
32  import rescuecore2.worldmodel.EntityID;
33
34  public class KMeansClustering extends StaticClustering {
35
36      private static final String KEY_CLUSTER_SIZE = "clustering.size";
37      private static final String KEY_CLUSTER_CENTER = "clustering.centers";
38      private static final String KEY_CLUSTER_ENTITY = "clustering.entities.";
39      private static final String KEY_ASSIGN_AGENT = "clustering.assign";
40
41      private int repeatPrecompute;
42      private int repeatPreparate;
43
44      private Collection<StandardEntity> entities;
45
46      private List<StandardEntity> centerList;
47      private List<EntityID> centerIDs;
48      private Map<Integer, List<StandardEntity>> clusterEntitiesList;
49      private List<List<EntityID>> clusterEntityIDsList;
50
51      private int clusterSize;
52
53      private boolean assignAgentsFlag;
54
55      private Map<EntityID, Set<EntityID>> shortestPathGraph;
56
57      public KMeansClustering(AgentInfo ai, WorldInfo wi, ScenarioInfo si,
        ModuleManager moduleManager, DevelopData developData) {
58          super(ai, wi, si, moduleManager, developData);
59          this.repeatPrecompute = developData.getInteger(
```

Static Clustering

- Extend the `StaticClustering` abstract class
- Implement methods
  - `getClusterNumber()`
  - `getClusterIndex(StandardEntity entity)`
  - `getClusterEntity(Entity id)`
  - `getClusterEntities(int index)`
  - `getClusterEntityIDs(int index)`
- Custom implementation will reside in the team's project (e.g., `adf-sample-agent-java`)

robocuprescue > workspace > adf-core-java > src > main > java > adf > core > component > module

```java
package adf.core.component.module.algorithm;

import adf.core.agent.develop.DevelopData;
import adf.core.agent.info.AgentInfo;
import adf.core.agent.info.ScenarioInfo;
import adf.core.agent.info.WorldInfo;
import adf.core.agent.module.ModuleManager;

public abstract class StaticClustering extends Clustering {

    public StaticClustering(AgentInfo ai, WorldInfo wi, ScenarioInfo si,
    ModuleManager moduleManager, DevelopData developData) {
        super(ai, wi, si, moduleManager, developData);
    }
}
```

StaticClustering Abstract Class

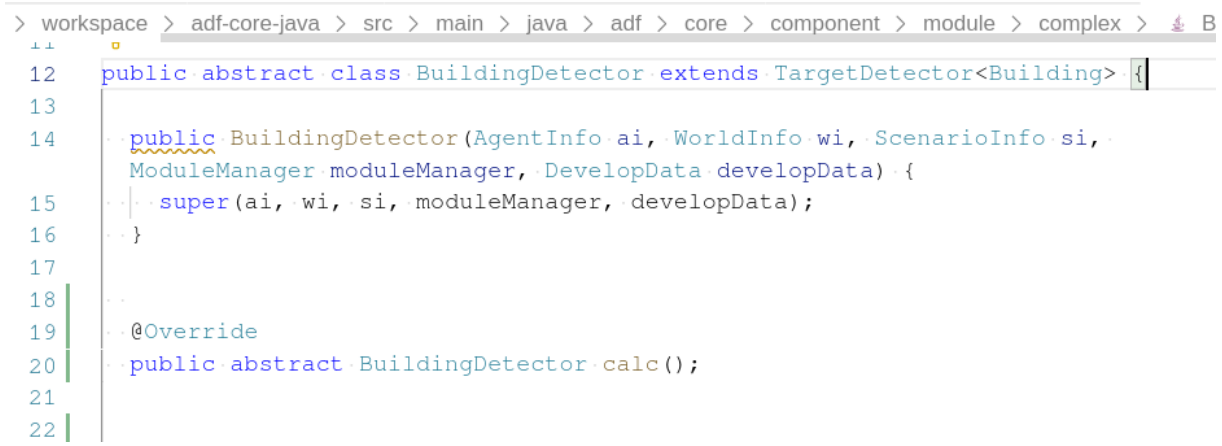
robocuprescue > workspace > adf-core-java > src > main > java > adf > core > component > module > a

```java
public abstract class Clustering extends AbstractModule {

    public Clustering(AgentInfo ai, WorldInfo wi, ScenarioInfo si, ModuleManager
    moduleManager, DevelopData developData) {
        super(ai, wi, si, moduleManager, developData);
    }


    public abstract int getClusterNumber();

    public abstract int getClusterIndex(StandardEntity entity);

    public abstract int getClusterIndex(EntityID id);

    public abstract Collection<StandardEntity> getClusterEntities(int index);

    public abstract Collection<EntityID> getClusterEntityIDs(int index);
```

Clustering Abstract Class

BuildDetector

- Extend the `BuildDetector` abstract class
- Implement methods
  - `calc()`
- Custom implementation will reside in the team's project (e.g., `adf-sample-agent-java`)



DefaultBuildDetector

- Extend the `BuildDetector` abstract class
- Default clustering module
  - Defined in `DefaultBuildingDetector.Clustering`
  - `adf.impl.module.algorithm.KMeansClustering`
- `calc()` implementation
  - Look for a building in the assigned cluster
  - If not found, look for a building anywhere in the world

orkspace > adf-core-java > src > main > java > adf > impl > module > complex > DefaultBuildingDetector.ja

```
24  public class DefaultBuildingDetector extends BuildingDetector {
25
26      private EntityID result;
27
28      private Clustering clustering;
29
30      public DefaultBuildingDetector(AgentInfo ai, WorldInfo wi, ScenarioInfo si,
        ModuleManager moduleManager, DevelopData developData) {
31          super(ai, wi, si, moduleManager, developData);
32          switch (si.getMode()) {
33              case PRECOMPUTATION_PHASE:
34              case PRECOMPUTED:
35              case NON_PRECOMPUTE:
36                  this.clustering = moduleManager.getModule(
37                          "DefaultBuildingDetector.Clustering",
38                          "adf.impl.module.algorithm.KMeansClustering");
39                  break;
40          }
41          registerModule(this.clustering);
42      }
43
44
45      @Override
46      public BuildingDetector calc() {
47          this.result = this.calcTargetInCluster();
48          if (this.result == null) {
49              this.result = this.calcTargetInWorld();
50          }
51          return this;
```

SampleBuildDetector

- Extend the `BuildDetector` abstract class
- Default clustering module
  - Defined in `DefaultBuildingDetector.Clustering`
  - `adf.impl.module.algorithm.KMeansClustering`
- `calc()` implementation
  - Look for a target building that is on fire

adf-sample-agent-java > src > main > java > sample_team > module > complex > SampleBuildingDetector.j

```
24  public class SampleBuildingDetector extends BuildingDetector {
25
26      private EntityID result;
27      private Clustering clustering;
28      private Logger logger;
29
30      public SampleBuildingDetector(AgentInfo ai, WorldInfo wi, ScenarioInfo si,
        ModuleManager moduleManager, DevelopData developData) {
31          super(ai, wi, si, moduleManager, developData);
32          logger = DefaultLogger.getLogger(agentInfo.me());
33          this.clustering = moduleManager.getModule(
34                  "SampleBuildingDetector.Clustering",
35                  "adf.impl.module.algorithm.KMeansClustering");
36          registerModule(this.clustering);
37      }
38
39
40      @Override
41      public BuildingDetector calc() {
42          this.result = this.calcTarget();
43          return this;
44      }
45
```

Configuration

launch.cfg

- General configuration file
- Define
  - Server to connect
  - Team name
  - Pre-computation
  - Debug mode
  - Develop mode
  - Module configuration file
  - Quantity of agents

```
kernel.host: localhost
kernel.port: 27931
team.name: sample_team
adf.launcher.precompute: 0
adf.debug.flag: 0
adf.develop.flag: 0
adf.develop.filename: config/develop.json
#adf.develop.data:
adf.agent.moduleconfig.filename: config/module.cfg
#adf.agent.moduleconfig.data:
adf.team.platoon.ambulance.count: -1
adf.team.platoon.fire.count: -1
adf.team.platoon.police.count: -1
adf.team.office.ambulance.count: -1
adf.team.office.fire.count: -1
adf.team.office.police.count: -1
```

module.cfg

- Specify the modules' implementation to used when running the team
- Set the team name
- Specified modules are defined inside each implementation
- For example, the Ambulance Team Tactics implementation is defined in the package `adf.core.impl.tactics`

```
Team.Name : sample_agent

## DefaultTacticsAmbulanceTeam

DefaultTacticsAmbulanceTeam.HumanDetector :
sample_team.module.complex.SampleHumanDetector

DefaultTacticsAmbulanceTeam.Search :
sample_team.module.complex.SampleSearch

DefaultTacticsAmbulanceTeam.ExtActionTransport :
adf.impl.extaction.DefaultExtActionTransport

DefaultTacticsAmbulanceTeam.ExtActionMove :
adf.impl.extaction.DefaultExtActionMove

DefaultTacticsAmbulanceTeam.CommandExecutorAmbulance :
adf.impl.centralized.DefaultCommandExecutorAmbulance

DefaultTacticsAmbulanceTeam.CommandExecutorScout :
adf.impl.centralized.DefaultCommandExecutorScout
```

module.cfg

- Use `moduleManager.getModule(…)` to load the required modules
  - Label used to look for the path of the module's implementation in the configuration file
  - Path of the default module's implementation
- Default module's implementation is used in case the label and a module's implementation is found in the configuration file

```
## DefaultTacticsAmbulanceTeam

DefaultTacticsAmbulanceTeam.HumanDetector :
sample_team.module.complex.SampleHumanDetector

DefaultTacticsAmbulanceTeam.Search :
sample_team.module.complex.SampleSearch

DefaultTacticsAmbulanceTeam.ExtActionTransport :
adf.impl.extaction.DefaultExtActionTransport

DefaultTacticsAmbulanceTeam.ExtActionMove :
adf.impl.extaction.DefaultExtActionMove

DefaultTacticsAmbulanceTeam.CommandExecutorAmbulance :
adf.impl.centralized.DefaultCommandExecutorAmbulance

DefaultTacticsAmbulanceTeam.CommandExecutorScout :
adf.impl.centralized.DefaultCommandExecutorScout
```

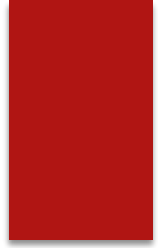
develop.json

- Pass additional information during the development phase of the module
- Enable/Disabled in the `launch.cfg`
- Passed as reference to different module constructors
- Content can be accessed using the methods defined in the `DevelopData`
  - `getInteger(name, defaultValue)`
  - `getDouble(name, defaultValue)`
  - `getString (name, defaultValue)`

```
{
"ActionFireFighting.refill.completed": 10,
"ActionFireFighting.refill.request": 1,
"ActionFireFighting.rest": 100,
"ActionFireRescue.rest": 100,
"ActionExtClear.forcedMove": 3,
"ActionExtClear.rest": 100,
"ActionTransport.rest": 100,
"ActionExtMove.rest": 100,
"SampleBuildingDetector.sendingAvoidTimeClearRequest": 5,
"SampleBuildingDetector.sendingAvoidTimeReceived": 3,
"SampleBuildingDetector.sendingAvoidTimeSent": 5,
"SampleBuildingDetector.moveDistance": 40000,
"SampleHumanDetector.sendingAvoidTimeClearRequest": 5,
"SampleHumanDetector.sendingAvoidTimeReceived": 3,
"SampleHumanDetector.sendingAvoidTimeSent": 5,
"SampleHumanDetector.moveDistance": 40000,
"CommandPickerAmbulance.scoutDistance": 40000,
"CommandPickerFire.scoutDistance": 40000
}
```

Hands-on

- Implement the A* Path Planning algorithm module
  - Implementation code available at https://roborescue.github.io/adf-sample-agent-java/index.html (see Section 4.4.1 and 4.4.2)
- Replace the `SamplePathPlanning` in the ADF Sample Agent



Tactics

Default Tactics

- Define the sequence of actions performed by the agent
- Default implementation
  - **adf.impl.tactics**
    - Priority to find a target
  - **adf.impl.centralized**
    - Centralized decision-making process

Thank You!!