IDK5151 kohtumine kaugõppijatega 26.04.2013

Prof Kuldar Taveter,
Tallinna Tehnikaülikool

Hindamine

- Analüüsimudelite kaitsmine: 10%
- Disainimudelite kaitsmine: 10%
- Kirjalik aruanne, umbes 2000 sõna koos lisadega, joonistega, tabelitega ja lähtekoodi näidetega: 20%
- Töö lõpukaitsmine: 10%
- Kahetunnine kirjalik eksam: 50%

Ajakava

- Esimene kohtumine 01.02.2013: Sissejuhatus
- **15.02.2013: Tähtaeg tiimide moodustamiseks ja teemade valikuks**
- Teine kohtumine 12.04.2013: Analüüsimudelite kaitsmine
- Kolmas kohtumine 26.04.2013: Disainimudelite kaitsmine
- **08.05.2013: Tähtaeg miniprojekti esitamiseks**
- Neljas kohtumine 10.05.2010: miniprojekti lõpukaitsmine

Miniprojektid

- Nõudmised:
  - Mudelid korralikult vormistatud seletuskirjaga (k.a. sissejuhatus)
  - Esitamise tähtaeg: **8. mai**
- Kaitsmine **10. mail kell 19.30-21.00**:
  - Ettekanne igalt tiimilt umbes 10 minutit 2-4 slaidiga
  - Realisatsiooni demo
  - Küsimused tiimi liikmetele
- Hindamine:
  - Lahenduse hajutatud iseloom
  - AOM-i vaatepunktide raamistiku 6 ülemise lahtri kaetus mudelitega
  - Mudelite põhjendused (miks on vaja üht või teist liiki mudeleid?)
  - Mudelite seostatus
  - Mudelite korrektsus
  - Realisatsioon

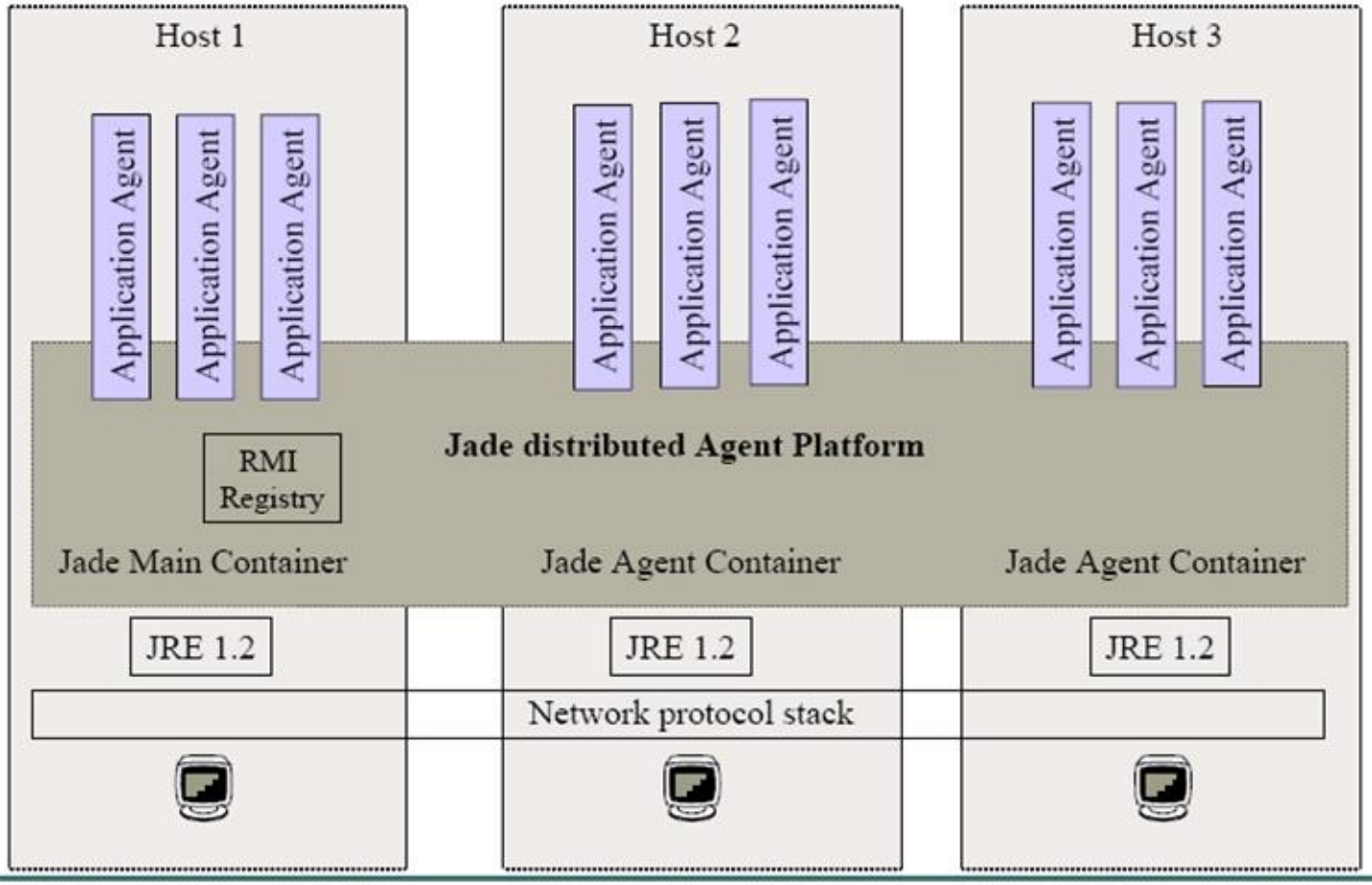
Kirjandus

- **Sterling, L. & Taveter, K. (2009). *The art of agent-oriented modeling*. MIT Press.**
- Wooldridge, M. (2009). *Introduction to multi-agent systems, 2nd Edition*. Addison-Wesley.
- d'Inverno, M. & Luck, M. (2001). *Understanding agent systems*. Springer-Verlag.
- Padgham, L. & Winikoff, M. (2004). *Developing intelligent agent systems: A practical guide*. John Wiley and Sons.
- Bellifemine, F., Caire, G, & Greenwood, D. (2005). *Developing multi-agent systems with JADE*. John Wiley & Sons.
- Bordini, R. H., Hübner, J. F., & Wooldridge, M. (2007). *Programming multi-agent systems in AgentSpeak using Jason*. John Wiley & Sons.

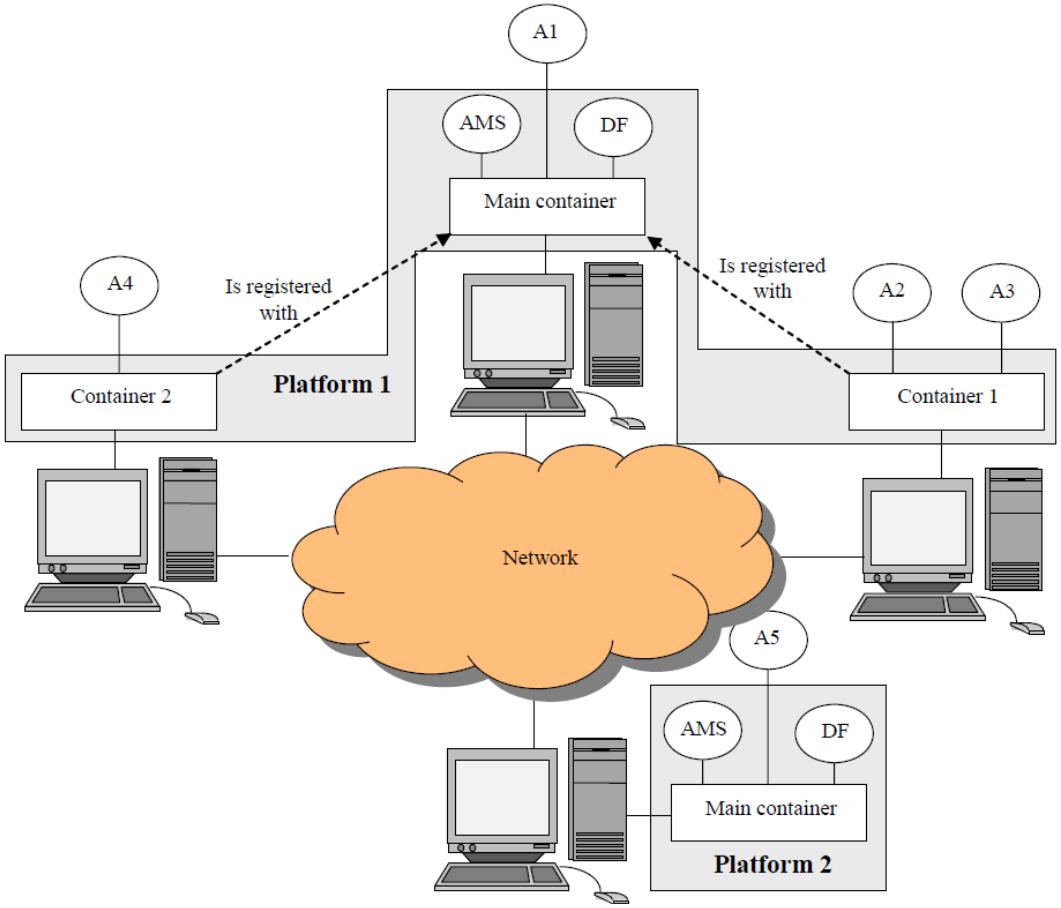
JADE (Java Agent Development Environment)

- Distributed agent platform which can be split among several hosts
- Java Application Programmer's Interface.
- Graphical User Interface to manage several agents from the same Remote Management Agent
- Library of FIPA interaction protocols, such as Contract Net
- Available at http://jade.tilab.com/

JADE Agent Platform



Agents, Containers and Platforms



AMS and DF

- AMS – Agent Management System
  - Providing naming services
- DF – Directory Facilitator
  - Provides Yellow Pages service

Creating JADE agent

```
import jade.core.Agent;
import jade.core.AID;

public class PhysicianAgent extends Agent {

  protected void setup() {
    // Printout a welcome message
    System.out.println("Hello! Physician-agent "+getAID().getName()+" is ready.");
   }
}
```

Running JADE agent

javac -classpath <JADE-classes> PhysicianAgent.java

java -classpath <JADE-classes> jade.Boot -gui physician1:PhysicianAgent

Classpath can/should be set beforehand.

Passing arguments to an agent

```
import jade.core.Agent;
import jade.core.AID;
public class PhysicianAgent extends Agent {
                private String specialty;
  protected void setup() {
    // Printout a welcome message
    System.out.println("Hello! Physician-agent "+getAID().getName()+" is ready.");
          Object[] args = getArguments();
          if (args != null && args.length > 0){
                specialty = (String) args [0];
                System.out.println("My area of specialisation is  "+specialty);
          }  else {
                System.out.println("No specialty has been assigned.");
                doDelete();
          } }
  protected void takeDown(){
                System.out.println("Physician-agent "+getAID().getName()+" terminated");
  } }
```

Running JADE agent with arguments

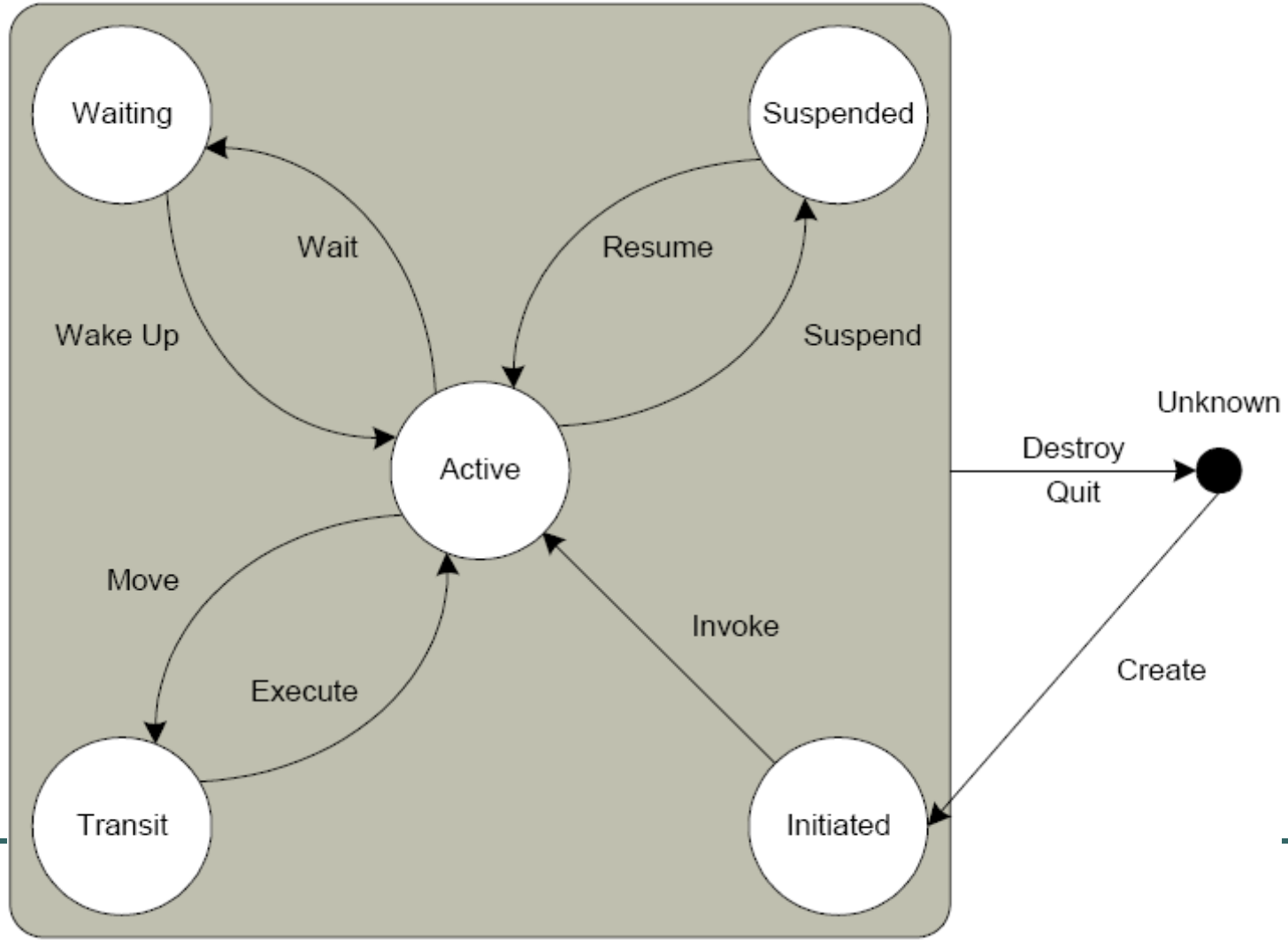
javac PhysicianAgent.java

java jade.Boot -gui physician1:PhysicianAgent(Cardiologist)

Terminating JADE agent

```
import jade.core.Agent;
import jade.core.AID;
public class PhysicianAgent extends Agent {
                private String specialty;
  protected void setup() {
    // Printout a welcome message
    System.out.println("Hello! Physician-agent "+getAID().getName()+" is ready.");
          Object[] args = getArguments();
          if (args != null && args.length > 0){
                specialty = (String) args [0];
                System.out.println("My area of specialisation is  "+specialty);
          }  else {
                System.out.println("No specialty has been assigned.");
                doDelete();
          } }
  protected void takeDown(){
                System.out.println("Physician-agent "+getAID().getName()+" terminated");
  } }
```

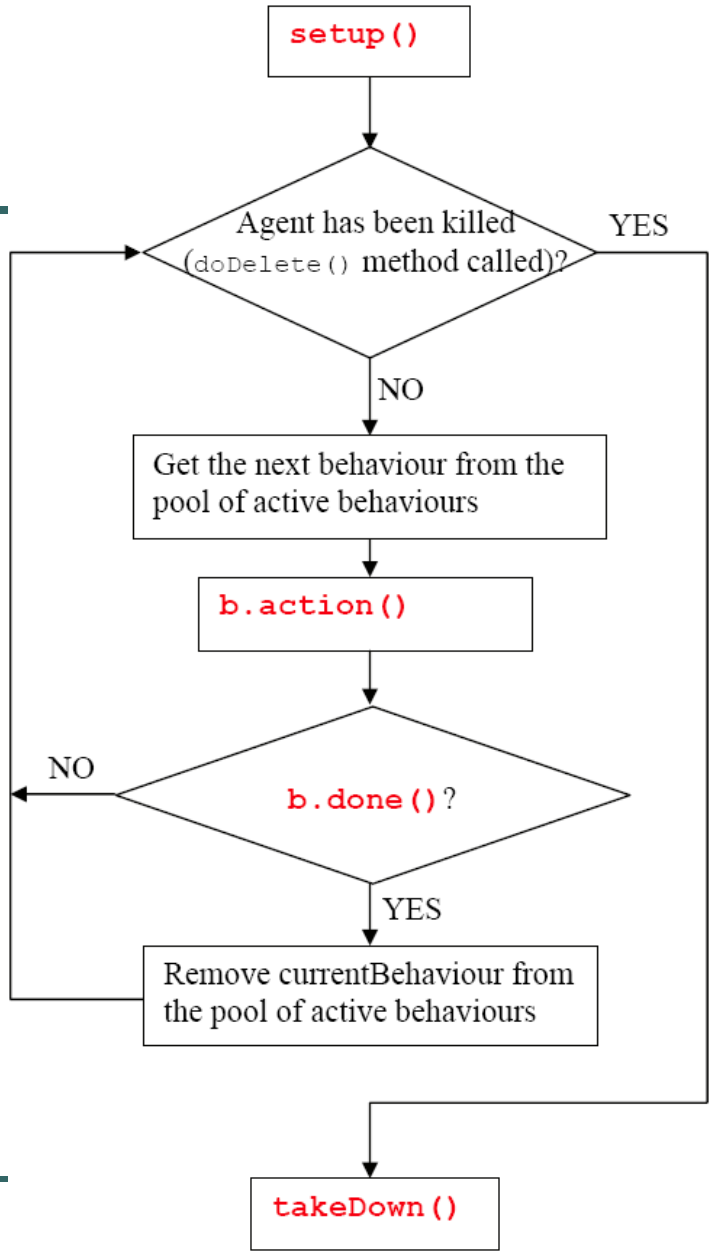
Agent life cycle



Concurrent tasks

- An agent must be able to carry out several concurrent tasks in response to different external events
- Every JADE agent is composed of a single execution thread
- Concurrent tasks are modelled and can be implemented as instances of jade.core.behaviours.Behaviour

Agent thread



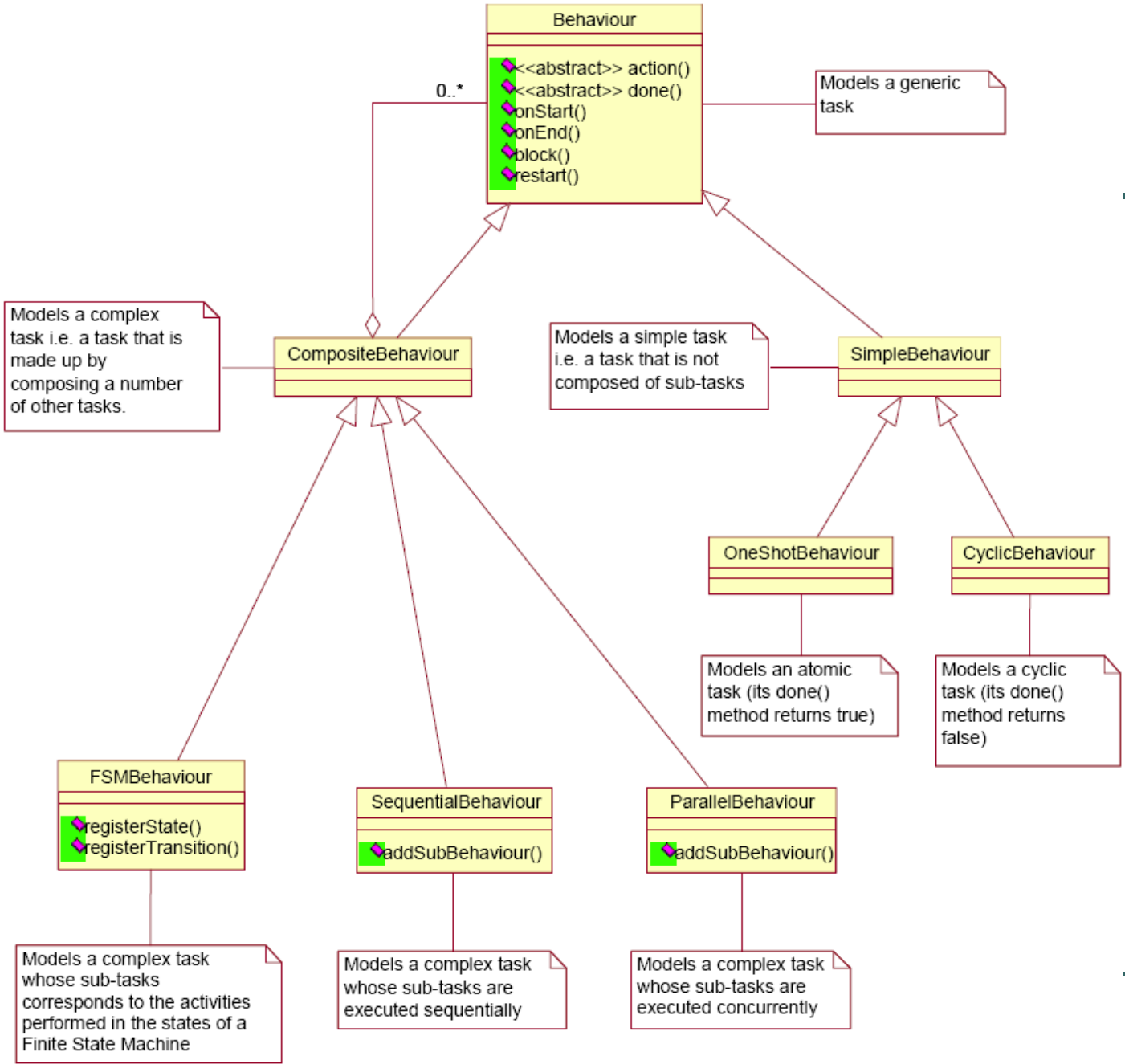
- Initializations
- Addition of initial behaviours

Highlighted in red the methods that programmers have to implement

- Agent "life" (execution of behaviours)

- Clean-up operations

Hierarchy of behaviours



Defining JADE agents

```
package DigitalPet;
import jade.core.*;

public class Tamagotchi extends Agent {

    // Put agent initializations here
    protected void setup() {
        // Adding behaviours
        addBehaviour(new MessageHandler (this));
        …
    }

    // If needed, put agent clean-up operations here
    protected void takeDown() {
        System.out.println("Tamagotchi "+getAID().getName()+" terminating.");
        …
    }
}
```

Defining behaviours

```
package DigitalPet;

import jade.core.*;
import jade.core.behaviours.*;
import jade.lang.acl.*;

public class MyOneShotBehaviour extends OneShotBehaviour {
        public void action() {
            // perform operation X
        }
}
public class MyCyclicBehaviour extends CyclicBehaviour {
        public void action() {
            // perform operation Y
        }
}
```

Sending messages

```
ACLMessage msg = new ACLMessage(ACLMessage.INFORM);
msg.addReceiver(new AID(“tama1”, false);
msg.setLanguage(“English”);
msg.setOntology(“Weather-forecast-ontology”);
msg.setContent(“Today it’s raining”);
myAgent.send(msg);

// Message carrying a request for offer
ACLMessage cfp = new ACLMessage(ACLMessage.CFP);
for (int i = 0; i < sellerAgents.lenght; ++i) {
    cfp.addReceiver(sellerAgents[i]);
}
cfp.setContent(targetBookTitle);
myAgent.send(cfp);
```

Receiving messages

```
public void action() {
    ACLMessage msg = myAgent.receive();
    if (msg != null) {
        // Message received. Process it
        ...
    }
    else {
        block();
    }
}
```

Setting classpath

- Please include in the classpath the following library files:
  - …\jade\lib\jade.jar
  - …\jade\lib\jadeTools.jar
  - …\jade\lib\http.jar
  - …\jade\lib\iiop.jar
- Please include in the classpath the location(s) of your Java class files

Compiling and running JADE agents

javac Tamagotchi.java Behaviours.java

…

java jade.Boot –gui -platform

java jade.Boot –container tama1:DigitalPet.Tamagotchi

Please consult API!

- http://jade.tilab.com/doc/api/index.html