Goal-Directed Interactions in Artifact-Based MAS:
Jadex Agents playing in CARTAGO Environments

Michele Piunti, Alessandro Ricci
ALMA MATER STUDIORUM – Università di Bologna, DEIS
{michele.piunti,a.ricci}@unibo.it
Lars Braubach, Alexander Pokahr
Distributed and Information Systems – University of Hamburg
{braubach,pokahr}@informatik.uni-hamburg.de

WI-IAT 2008, Sydney
1 Agents and Artifacts

2 Jadex CARTAGO Integration Model

3 The booktrading example in A&A

4 Conclusions
Outline

1. Agents and Artifacts
2. Jadex CARTAGO Integration Model
3. The booktrading example in A&A
4. Conclusions
Agents and Environment

- Cognitive agents / Environment Interactions
- Environment often built as monolithic components or middleware
- Interactions based on low level mechanisms and language constructs (i.e. forming agents to objects)

![Diagram of MAS and Environment Interactions]

Example: JAVA PLATFORM AGENTS MAS
Agents and Environment

- To model interactions we need a clear understanding of agent *basic activities*
- Action (to change the world) and perception (to get information)
- We need to define the *rules of encounter*
Agents and Artifacts

- Environment can be divided in different workspaces
- Self-contained computational units can be defined based on their functionalities (i.e. blackboard, web services, database, coordination tools)
- Persistent entities, conforming to the action/perception laws defined in the environment
Agents and Artifacts

- Artifact as a first class abstraction, dual to ones used by agents
- As agents, artifacts are entities existing (persisting) in the same work environment
- Differently from agents, artifacts don’t encapsulate their control, have interfaces (to be **used**), and conforms to the interaction dynamic defined in the working environment
Agents

- Autonomous, goal-oriented, situated, social, pro-active entities
- Create and co-use artifacts for supporting their activities
  - i.e., artifact mediated interactions

Artifacts

- Non-autonomous, automatic/reactive, function-oriented entities
  - Conceived by MAS designers to provide serviceable operations, preprocessed information and facilities in order to ease agent work
  - Controllable and observable by agents
- Modelling facilities, tools and resources exploitable by agents

Workspaces

- Logically grouping agents and artifacts
- Defining the topology of the computational environment and the physical dynamics of interactions (rules of encounter)
Agents
- Autonomous, goal-oriented, situated, social, pro-active entities
- Create and co-use artifacts for supporting their activities
  - i.e., artifact mediated interactions

Artifacts
- Non-autonomous, automatic/reactive, function-oriented entities
  - Conceived by MAS designers to provide serviceable operations, preprocessed information and facilities in order to ease agent work
  - Controllable and observable by agents
- Modelling facilities, tools and resources exploitable by agents

Workspaces
- Logically grouping agents and artifacts
- Defining the topology of the computational environment and the physical dynamics of interactions (rules of encounter)
Agents

- Autonomous, goal-oriented, situated, social, pro-active entities
- Create and co-use artifacts for supporting their activities
  - i.e., artifact mediated interactions

Artifacts

- Non-autonomous, automatic/reactive, function-oriented entities
  - Conceived by MAS designers to provide serviceable operations, preprocessed information and facilities in order to ease agent work
  - Controllable and observable by agents
- Modelling facilities, tools and resources exploitable by agents

Workspaces

- Logically grouping agents and artifacts
- Defining the topology of the computational environment and the physical dynamics of interactions (rules of encounter)
Artificial Computational Model

- **Artifact**
- **Usage Interface**
- **Observed Events Generation**
- **Observed Properties**
- **Link Interface**

- **Agents and Artifacts**
  - **OpControlName(Params)**
  - **Value**
  - **Manual Observables**
  - **Events Generation**
  - **<EvName,Params>**
  - **Operation X**
  - **Operation Y**
  - **ObsPropName**
  - **Value**

- **Graphical Representation**
  - Nodes: **ObsPropName**, **Value**
  - Connections: **Link Interface**, **Operation X**, **Operation Y**

**References**
- Piunti et al. (2008) Goal-Directed Interactions in Artifact-Based MS
- WI-IAT 2008, Sydney
Outline

1. Agents and Artifacts
2. Jadex CARTAGO Integration Model
3. The booktrading example in A&A
4. Conclusions
Jadex agents playing in CARTAGO Working Environments

- **CARTAGO** allows to build environments with artifacts conceived for easing agents work.
- **Jadex** is a well established agent platform to build MAS based on the BDI model of agency:
  - Jadex agents can be designed based on concepts and notions mimicking human reasoning (Beliefs, Desires, Intentions).
  - An internal engine allows agents to select, at any given moment, the course of action based on internal state, perceived events, and adopted goals.
- **C4Jadex** is an integration technology enabling Jadex agents to operate in CARTAGO working environment.
Goal Directedness: Jadex uses an explicit representation of agent Goals and defines rules to deliberate according to agent internal state [Braubach et al., 2004]

- Goals can be mapped on artifact operation outcomes
- Goal Types to shape different interactions (query, maintain, perform, achieve)
Rationale of Jadex CARTAGO Integration: Events

**Events**  Jadex Internal Events can be customized and mapped on Artifact Events

- Used to send relevant signal to the reasoning engine (i.e. deliberate, trigger plans)
Rationale of Jadex CARTAGO Integration: Capabilities

**Capability**  Jadex notion of capability consists of reusable BDI elements as Beliefs, Plans, Events and Goals  
[Braubach et al., 2005, Busetta et al., 2000]
- Defines visibilities on those elements and exposes an interface for the user of the capability
- The agent can transparently use capability main functionalities, and delegating low level details
Basic Actions for interacting in CARTAGO Working Environments

(1) joinWorkspace(+Workspace [,Node])
(2) quitWorkspace
(3) makeArtifact(+Artifact,+ArtifactType [,ArtifactConfig])
(4) lookupArtifact(+ArtifactDesc,?Artifact)
(5) disposeArtifact(+Artifact)
(6) use(+Artifact,+UIControl([Params]) [,Sensor] [,Timeout] [,Filter])
(7) sense(+Sensor,?PerceivedEvent [,Filter] [,Timeout])
(8) focus(+Artifact [,Sensor] [,Filter])
(9) stopFocussing(+Artifact)
(10) observeProperty(+Artifact,+Property,?PropertyValue)

Table: Jadex Goals for managing workspaces (1–2), creating, disposing and looking up artifacts (3–5), using artifacts (6–7), and observing artifacts (8–10). Syntax is expressed in a logic-like notation, where italicised items in square brackets are optional [Ricci et al., 2008].
Agents simply need to import the elements of the Cartago Bridge capability defined in the Goal-Directed Interactions in Artifact-Based MAS.
Outline

1. Agents and Artifacts
2. Jadex CARTAGO Integration Model
3. The booktrading example in A&A
4. Conclusions
Initiator takes the role of manager which wishes to have some task performed by Participants
- By optimising a function characterizing the task (i.e., price, time).
- For a given task, any number of the Participants may respond with a proposal; the rest must refuse.
- Negotiations then continue with the Participants that proposed.

[Braubach and Pokahr, 2007]
Booktrading example using Agents and Artifacts
Benefits

Artifacts ease agents computational loads:

- cfp and proposals can be sorted based on specified criteria (i.e. price, time)
- signals can be addressed to agents based on specific situations
- artifacts functioning can be changed: they provide a *loci of control* for regulating the system dynamics
- artifacts can store additional information (i.e. *reputation*) and make it to persist even beyond agents presence

No need to share ontologies and protocols among heterogeneous agents:

- If required, artifact can translate information
- Artifact function, in this case, is also to locate agents and define their roles
Benefits

Artifacts ease agents computational loads:

- cfp and proposals can be sorted based on specified criteria (i.e. price, time)
- signals can be addressed to agents based on specific situations
- artifacts functioning can be changed: they provide a *loci of control* for regulating the system dynamics
- artifacts can store additional information (i.e. *reputation*) and make it to persist even beyond agents presence

No need to share ontologies and protocols among heterogeneous agents:

- If required, artifact can translate information
- Artifact function, in this case, is also to locate agents and define their roles
From an agent viewpoint, artifacts embeds:

1. **Operational (goal directed) Function**
   - Goals can be achieved by the mean of operations which have been defined within artifacts control interface.

2. **Doxastic (epistemic) Function**
   - Artifact representational contents can be used as an external informational structure (i.e. intelligent use of space)
Operational Function

- Agent’s Goals are in Operation Outcomes:

**OBSERVABLE PROPERTIES:**
- sellers: list<seller>
- seller_reputation: list<double>

**USAGE INTERFACE:**
- MakeCfp(Object: cfp) : [ op_exec_completed ]
- RetrieveProp() : [ proposals(list<item>: props), op_exec_completed ]
- BuyBook(item:Item) : [ op_exec_completed ]
- GiveFeedback(double: rep) : [ op_exec_completed ]
Doxastic Function

- From an agent viewpoint, artifacts are informational units, exploitable in a situated way:
- Strategic information as Goal Supporting Beliefs can be read on Artifact Observable Properties

```xml
SELLER

1 focus("sellerboard")

 serviços

SELLERBOARD

OBSERVABLE PROPERTIES:
requests: HashMap<buyer, title>

USAGE INTERFACE:
Register() : 
  [ op_exec_completed ]

MakeProposal(String buyer, Object cfp) : 
  ( requests.containsKey(buyer) ) 
  [ op_exec_completed ]

DeliverBook(item:Item) :
  [ op_exec_completed ]

<performgoal name="focus">
  <parameter name="artifact_name">
</performgoal>
```
Doxastic Function

- Agent’s Beliefs are updated on the basis of Artifact observable state:
  - External repositories, automatically collecting and providing strategic knowledge
  - Additional memory, even shared between agent groups
  - Observable cues in order to highlight relevant information in environments (situated cognition)

```
<internalevent name="artifact_event">
  1 focus("sellerboard")
  2 property updated

<performgoal name="focus">
  <parameter name="artifact_name"/>
</performgoal>

SELLER AGENT

SELLERBOARD

requests: HashMap<buyer, title>

USAGE INTERFACE:
Register() :
  [ op_exec_completed ]
MakeProposal(String buyer, Object: cfp) :
  / ( requests.containsKey(buyer) )
  [ op_exec_completed ]
DeliverBook(item:Item) :
  [ op_exec_completed ]

OBSERVABLE PROPERTIES:
requests: HashMap<buyer, title>
```
General and Special Purpose Capabilities

- Goals can be achieved by the mean of operations which have been defined within capabilities control interface
- Nested capabilities to provide terminal goals (i.e., at the abstraction level of the domain problem)

**JaDEX**

**BUYER/SELLER AGENT**

**GENERAL PURPOSE CAPABILITY**

- `cartago_bridge capability`
- `joinWorkspace(+Workspace[,Node])`
- `quitWorkspace`
- `makeArtifact(+Artifact,+ArtifactType[,ArtifactConfig])`
- `lookupArtifact(+ArtifactDesc,?Artifact)`
- `disposeArtifact(+Artifact)`
- `sense(+Sensor,?PerceivedEvent[,Filter] [,Timeout] )`
- `focus(+Artifact[,Sensor] [,Filter] )`
- `stopFocussing(+Artifact)`
- `observeProperty(+Artifact,+Property,?PropertyValue)`

**BUYER AGENT**

- `getBookTitle(+title)`

**SPECIAL PURPOSE CAPABILITY**

- `cartago_bridge capability`
  - `joinWorkspace(+Workspace[,Node])`
  - `quitWorkspace`
  - `stopFocussing(+Artifact)`
  - `observeProperty(+Artifact,+Property,?PropertyValue)`

**CARTAGO**

**requests**

- `HashMap`
- `<buyer, title>`

- `Register`
- `MakeProposal`
- `DeliverBook`

**SELLERBOARD**

- `list<seller>`
- `list<double>`

- `MakeCfp`
- `RetrieveProp`
- `BuyBook`
- `GiveFeedback`

**BUYERBOARD**

- `Contract Net Workspace`

- `PropResp`
- `ReceiveBook`
<achievegoal name="purchase_book" recur="true" recurdelay="10000">
  <parameter name="order" class="Order">
    <bindingoptions>$beliefbase.initial_orders</bindingoptions>
  </parameter>
  <unique/>
  <creationcondition>
    $beliefbase.initial_orders!=null &&
    $beliefbase.sellers.length &gt; 0
  </creationcondition>
  <targetcondition>Order.DONE.equals($goal.order.getState())</targetcondition>
  <failurecondition>$beliefbase.time &gt; $goal.order.getDeadline().getTime()</failurecondition>
</achievegoal>

<plan name="purchase_book_action">
  <parameter name="order" class="Order">
    <goalmapping ref="purchase_book.order"/>
  </parameter>
  <parameter name="aName" class="String">
    <value>"initiator"</value>
  </parameter>
  <body>new PurchaseBookPlan()</body>
  <trigger>
    <goal ref="purchase_book"/>
  </trigger>
</plan>
public class PurchaseBookPlan extends CartagoPlan {

    public void body() {

        // 1. RETRIEVE NEGOTIATIONS PARAMS
        Order order = (Order) getParameter("order").getValue();
        String aName = getParameter("aName").getValue().toString();

        String[] sellers = (String[]) getBeliefbase().getBeliefSet("sellers").getFacts();

        // 1. LOOK FOR ARTIFACT aName
        IGoal lookup = createGoal("lookup");
        lookup.getParameter("artifact_name").setValue(aName);
        dispatchSubgoalAndWait(lookup);
        ArtifactId aid = (ArtifactId) lookup.getParameter("aid").getValue();

        // 2. FOCUS on IT
        IGoal focus = createGoal("focus");
        focus.getParameter("artifact_name").setValue(aName);
        dispatchSubgoalAndWait(focus);

        // 2. INITIATE INTERACTION
        IGoal cnp = createGoal("cnp_initiate");
        cnp.getParameter("aName").setValue(aName);
        cnp.getParameter("cfp").setValue(order.getTitle());
        cnp.getParameter("cfp_info").setValue((new Integer(acceptable_price)));
        cnp.getParameterSet("receivers").addValues(sellers);
        dispatchSubgoalAndWait(cnp);
    }
}
Conceiving A&A Systems

PROBLEMS:
- Why agents should use artifacts?
- What is going to change – in the reasoning model – to bring about artifacts?

Agents using Artifacts:

An agent $AG$ depends on an Artifact $AR$ for a given goal $G_i$, according to a set of plans $P_{ji}$ if:

1. It has $G_i$ in its set of goals
2. It is not autonomous for $G_i$
   - Lacks at least one of the resources (actions/information) necessary to achieve $G_i$
3. There is a plan $p_{ji} \in P_{ji}$ that achieves $G_i$ where at least one resource (action/information) used in this plan is in $AR$’s set of available operations or $AR$’s observable state
   - Artifact Control Interface or Observable Properties
Conceiving A&A Systems

PROBLEMS:
- Why agents should use artifacts?
- What is going to change – in the reasoning model – to bring about artifacts?

Agents using Artifacts:

An agent $AG$ depends on an Artifact $AR$ for a given goal $G_i$, according to a set of plans $P_{ji}$ if:

1. It has $G_i$ in its set of goals
2. It is not autonomous for $G_i$
   - Lacks at least one of the resources (actions/information) necessary to achieve $G_i$
3. There is a plan $p_{ji} \in P_{ji}$ that achieves $G_i$ where at least one resource (action/information) used in this plan is in $AR$’s set of available operations or $AR$’s observable state
   - Artifact Control Interface or Observable Properties
Outline

1. Agents and Artifacts
2. Jadex CARTAGO Integration Model
3. The booktrading example in A&A
4. Conclusions
Summary

Sophisticated interactions are typical of (open) MAS. Using artifacts working environment allows:

**single agent** To provide external resources to achieve goals and strategic Information:

1. externalise activities required to achieve Goals
2. reason about goals: goal supporting beliefs
3. simplify agent decision making and ease deliberation processes

**multi agent** To provide facilities exploitable in Multi-Agent Societies:

1. Across Agents: organise and make available relevant information as permanent side-effect of artifact use (modification of artifact state)
2. Across Platforms: interoperability, mediated interactions
3. Across Time: hold strategic information persistent over agent presence
4. Across Space: no need for agents mutual presence
Sophisticated interactions are typical of (open) MAS. Using artifacts working environment allows:

**single agent** To provide external resources to achieve goals and strategic Information:

1. externalise activities required to achieve Goals
2. reason about goals: goal supporting beliefs
3. simplify agent decision making and ease deliberation processes

**multi agent** To provide facilities exploitable in Multi-Agent Societies:

1. Across Agents: organise and make available relevant information as permanent side-effect of artifact use (modification of artifact state)
2. Across Platforms: interoperability, mediated interactions
3. Across Time: hold strategic information persistent over agent presence
4. Across Space: no need for agents mutual presence
Sophisticated interactions are typical of (open) MAS. Using artifacts working environment allows:

**single agent** To provide external resources to achieve goals and strategic Information:

1. externalise activities required to achieve Goals
2. reason about goals: goal supporting beliefs
3. simplify agent decision making and ease deliberation processes

**multi agent** To provide facilities exploitable in Multi-Agent Societies:

1. Across Agents: organise and make available relevant information as permanent side-effect of artifact use (modification of artifact state)
2. Across Platforms: interoperability, mediated interactions
3. Across Time: hold strategic information persistent over agent presence
4. Across Space: no need for agents mutual presence
CARTAGO is an open-source technology

- Project HomePage: http://cartago.sourceforge.net
- Cartago and Friends: Jadex, Jason, 2APL, SimpA, JADE

Jadex BDI Agent Framework

- Project Homepage: http://vsis-www.informatik.uni-hamburg.de/projects/jadex/

Agents & Artifacts Theory, Design and Practice

Bibliography


Goal-Directed Interactions in Artifact-Based MAS: Jadex Agents playing in CARTAGO Environments

Michele Piunti, Alessandro Ricci
ALMA MATER STUDIORUM – Università di Bologna, DEIS
{michele.piunti,a.ricci}@unibo.it
Lars Braubach, Alexander Pokahr
Distributed and Information Systems – University of Hamburg
{braubach,pokahr}@informatik.uni-hamburg.de

WI-IAT 2008, Sydney