

Towards Seamless Agent Middleware

Andrea Omicini – Università di Bologna e Cesena

Giovanni Rimassa – Whitestein Technologies AG



Presentation Outline

- **Middleware and Technology Transfer**
 - Three dimensions affect technology transfer.
 - Why Middleware can help in all three.
- **Seamlessly Situated Agents**
 - Social and Environmental Situation.
 - Linguistic and Non-Linguistic Interaction.
- **The JADE Example**
 - New kernel architecture.
 - Agents + Services = Applications.



Middleware and Models (1)

- The traditional Middleware definition.
 - The *Middleware* lies in the middle between the Operating System and the applications.
 - Emphasis on *vertical* interfaces.
- A key feature of Middleware is *Interoperability*.
 - Applications using the same Middleware can interoperate.
 - This is true of any common platform (e.g. OS file system).
- But, many incompatible middleware systems exist.
 - Applications on middleware A or B can work together.
 - But, A-applications and B-applications cannot!
- The *Enterprise Application Integration* (EAI) task.
 - Emphasis on *horizontal* communication.
 - *Application-to-application* and *middleware-to-middleware*.



Middleware and Models (2)

- Real systems are heterogeneous.
 - Piecemeal growth is a *very troublesome* software evolution path due to lack of *conceptual integrity*.
- Middleware technology is an *integration* technology.
 - Adopting a given middleware should ease *both* new development *and* legacy integration.
 - To achieve integration while limiting conceptual drift, Middleware tries to cast a **Model** on heterogeneous applications.



Middleware and Models (3)

- Before: you have a total mess.
 - A lot of systems, using different technologies.
 - *Ad-hoc* interactions, irregular structure.
 - Each piece must be described in its own reference frame.
- Then: the Integration Middleware (IM) comes.
 - A new, shiny *Model* is supported by the IM.
 - Existing systems are re-cast under the *Model*.
 - New *Model*-compliant software is developed.
- After: you have the same total mess.
 - But you can reason about it effectively and coherently way.



Agent Technology Transfer

- **MASs as Integration Middleware**
 - The powerful abstractions used in MASs can be used to build the Model.
 - Semantic models deepen integration.
 - But, agent infrastructures are also needed.
 - Moreover, methodologies have their role.
- **Mainstreaming Agent Technology**
 - Addressing the whole transfer chain.



Agent Technology Transfer

- Technology transfer affected by three dimensions.
 - Conceptual Paradigm.
 - Development Process and Tools.
 - Economical Environment.
- Middleware can help in all three.
 - Essential to really mainstream agents.





Conceptual Paradigm

- Kuhn's concept of *Paradigm Shift*.
 - Became more and more abused.
 - *Normal science* periods must dominate.
 - Evolutionary rather than revolutionary.
- Agent Middleware helps change.
 - Turns Agent Technology model into a tool.
 - Creeping revolution from Object Orientation.
 - Do we agree about Object Technology at least?



Development Process

- Conceptual transfer must be backed up by processes and tools.
 - Skill transfer towards users of previous approaches.
 - Tools that extend the familiar development environment to the new technology.
- Agent Middleware helps process.
 - Tool and methodology integration.





Economical Environment

- New technologies impact business.
 - Agents as a *Disruptive Technology*?
 - Acknowledging general business rules.
 - Shifting from **Services** to **Products**.
- Agent Middleware helps business.
 - A common infrastructure reduces costs.
 - A solid infrastructure reduces risks.





Seamless Agent Middleware

- Agents have a powerful feature.
 - They are **situated**.
 - Situatedness directly supports integration within heterogeneous open systems.
- An agent situatedness is twofold.
 - The agent is ***physically situated***.
 - The agent is ***socially situated***.





Seamless Agent Middleware

- Social situatedness is very well supported by the ACL linguistic connector.
 - Speech Act Theory, Intentional Stance, ...
- Not the same for physical situatedness.
 - Speech acts require two language-capable peers
 - The environment has a lot of passive, service-level components (artifacts?).
 - Issue: Representing the Agent-Artifact interaction.



Seamless Agent Middleware

- Option 1: **Wrapper Agents**.
 - Reducing physical situatedness to social.
 - Using ACL to talk with artifacts via wrapper.
- Option 2: **Artifact-Specific** means.
 - Each non-intentional portion of the environment has its own way to affect the agent and of being affected by the agent.





Seamless Agent Middleware

- Def.: **Seamless Agent Middleware**
- *“An agent middleware technology that allows to set the social/physical situatedness balance wherever it better suits the application”.*
- With respect to using agent wrappers.
 - This means that you **never have to**.
 - But this means also that you **always can**.



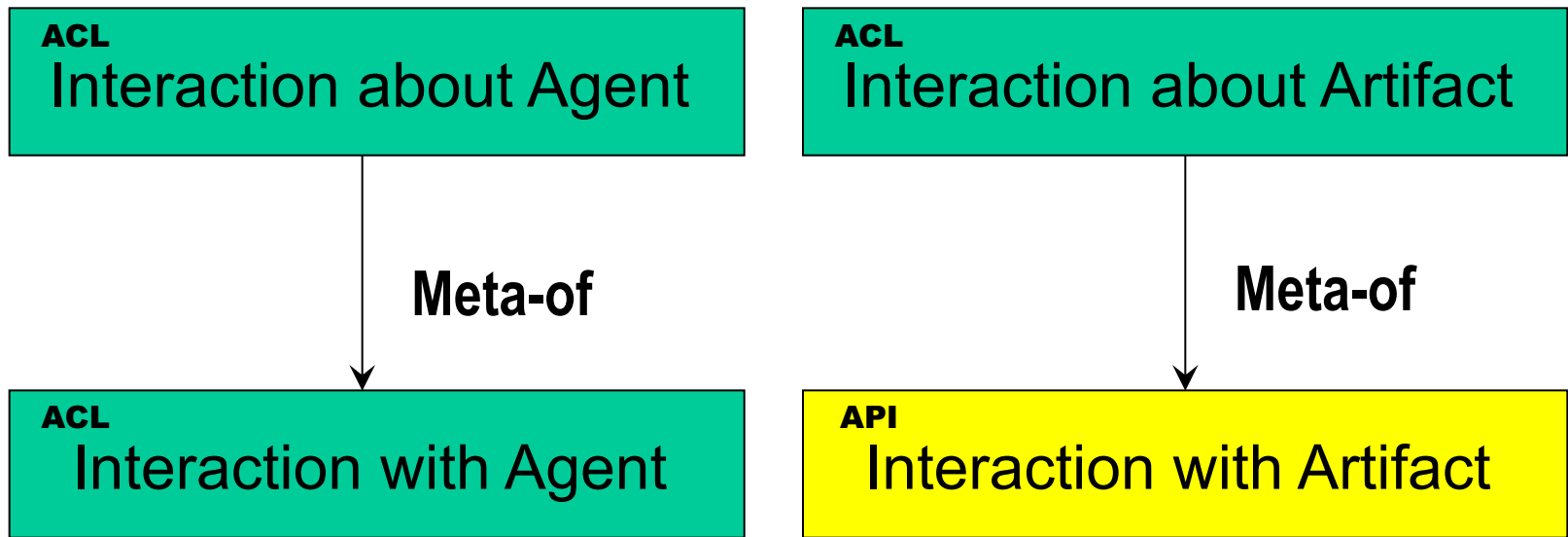
Metaphorical Guiding Light

- Metaphor Terms
 - Intentional software components $\Rightarrow$ People.
 - Non-intentional software components $\Rightarrow$ Things.
 - ACL $\Rightarrow$ Language.
- Language entails **conscious interaction**.
 - People talk with people.
 - People do not talk with things.
- Language entails **description**.
 - People talk about people.
 - People talk about things.



Metaphorical Guiding Light

- The metaphor, then, suggests the following relation between ACL and a service-level API.





A Paradigmatic Domain

- The interaction / coordination dimension
 - ACL only: coordination-is-sociality.
 - Pheromons only: coordination-is-situatedness.
- Relating to Activity Theory
 - Co-ordination / Co-operation / Co-construction
 - Subjective & Objective Coordination
 - subjective co-ordination overseeing and/or reconfiguring objective co-operation
- The previous schema applies perfectly!



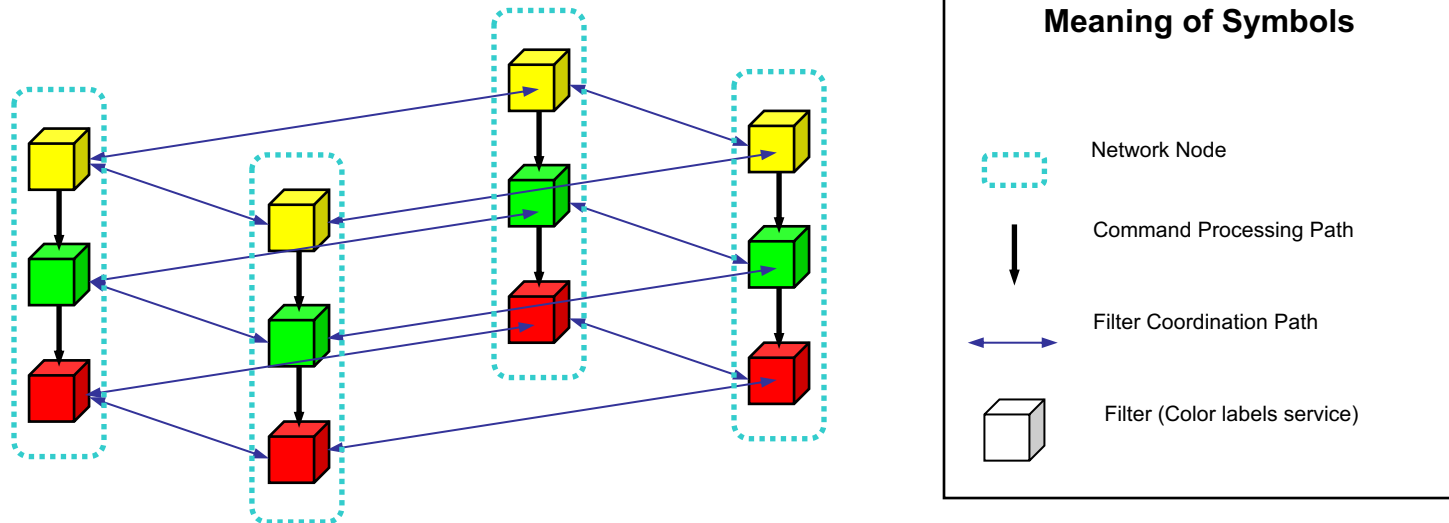
The JADE Example

- JADE Middleware Evolution.
 - First Versions of JADE: whole JVM.
 - Since JADE 2.4: In-Process Interface.
 - JADE within Web apps or app servers.
 - New kernel architecture: kernel services.
- Achieving Seamlessness.
Agents + Services = Applications





Distributed Coordinated Filters



- The major concepts, in principle, are:
 - Service, Command, Filter (**Processing** aspect).
 - ServiceSlice, Node (**Distribution** aspect).
 - ServiceInteraction, ServiceProtocol (**Introspection** aspect).



Conceptual Class Diagram





Conclusion and Future Work

- Seamless situatedness and open systems.
 - Varied environment and society.
 - Shifting balance of Agents vs. Artifacts.
 - Linguistic Interaction as common meta-level.
- Applying the new JADE architecture.
 - New baseline services (e.g. Security).
 - ASAM project at Whitestein.
 - Coordination Service with TuCSoN.





Troublesome Questions

- Methodological.
 - How do we set the threshold between agents and artifacts?
- Technological.
 - How shall we model the non-linguistic part within the linguistic part?

