The LuCe Coordination Technology for MAS Design and Development on the Internet

Andrea Omicini, Enrico Denti, and Vladimiro Toschi

LIA, Laboratorio di Informatica Avanzata
DEIS - Università di Bologna
Viale Risorgimento, 2 40136 – Bologna, Italy
{edenti,aomicini,vtoschi}@deis.unibo.it

Abstract. Internet-based multi-agent systems call for new metaphors, abstractions, methodologies and *enabling technologies* specifically tailored to agent-oriented engineering. While *coordination models* define the framework to manage the space of agent interaction, ruling social behaviours and accomplishing social tasks, their impact on system design and development calls for an effective *coordination technology*.
This paper presents LuCe, a coordination technology that integrates Java, Prolog and the notion of *logic tuple centre*, a programmable coordination medium, into a coherent framework. The power of the LuCe coordination technology is first discussed in general, then shown in the context of a simple yet significant system: a TicTacToe game among intelligent software agents and human players on the Internet.

1 Introduction

Multi-agent systems (MAS) are becoming an ubiquitous paradigm for building complex software applications, introducing in the AI field new issues coming from the Software Engineering area, such as the need for models and methodologies for MAS engineering, and the availability of *enabling technologies* [5]. This convergence results in a new research area, called *multi-agent system engineering* [9]. There, the emphasis is put on *task-oriented design*, where the global system goals are delegated to the responsibility of either individual agents (*individual tasks*) or agent societies (*social tasks*) [3,20,13].

While *coordination* deals in general with managing the interaction among components [16], in this context it addresses the issue of how agents interact: as agent architectures and languages [21] support the design and development of individual agents [14], *coordination models* [12,2,10,4,17,18,19] provide the metaphors and abstractions required to build agent societies [1]. In particular *coordination media*, which embed social rules as coordination laws, can work as the core for agent societies pursuing social tasks. However, coordination models alone cannot face the intrinsic complexity of building agent societies: the full exploitation of a coordination model for the design and development of multi-agent social behaviour requires a suitable *coordination system*, providing engineers with the enabling technology they need.

A. Porto and G.-C. Roman (Eds.): COORDINATION 2000, LNCS 1906, pp. 305–310, 2000.

This paper discusses how a *coordination technology* [1] like the LuCe (Logic Tuple Centres) coordination system can be exploited for the engineering of Internet-based MAS, exploiting the model's metaphors to provide ad-hoc development tools. As a case-study, we discuss a TicTacToe game among intelligent software agents and human players on the Internet, showing how the LuCe technology can effectively support the system design and development.

2 LuCe: Model and Technology

LuCe agents interact by exchanging *logic tuples* through a multiplicity of communication abstractions called *tuple centres*. The communication language is based on first-order logic: a tuple is a Prolog ground fact, any unitary Prolog clause is an admissible tuple template, and unification is the tuple matching mechanism. Agents perceive a LuCe tuple centre as a logic tuple space [11], which can be accessed through the typical Linda-like operations (`out`, `in`, `rd`, `inp`, `rdp`). Each tuple centre is uniquely identified by a ground Prolog term, and any ground term can be used to denote a tuple centre.

What makes a tuple centre different from a tuple space is the notion of *behaviour specification*, which defines how a tuple centre reacts to an incoming/outgoing communication event. From the agent's viewpoint, a tuple centre looks like a tuple space, and is accessed via the same interface: however, the tuple centre's behaviour in response to communication events is not fixed once and for all, but can be extended so as to embed the coordination laws into the coordination medium [6]. In particular, LuCe adopts the tuple centre model defined in [7], where the behaviour is defined in terms of *reactions* to communication events, expressed as first-order logic tuples, the *specification tuples*.

So, a logic tuple centre is conceptually structured in two parts: the *tuple space*, with ordinary communication tuples, and the *specification space*, with specification tuples. Agents can then be designed around their individual tasks, defining interaction protocols according to their own perception of the interaction space: tuple centres' coordination laws will take care of "gluing things together".

The clear distinction between the coordinating entities (tuple centres) and the coordinated entities (agents) naturally decouples individual and social tasks: the system social behaviour can then be changed just by changing the specification tuples. Moreover, the same coordination rules can be reused for other problems of the same class, applying the same behaviour specification to a different domain.

As far as technology is concerned, the LuCe system is built around three major ingredients: the Java technology (for the system development and the Web interface), Prolog, as the *lingua franca* for communication among heterogeneous Web agents, and tuple centre technology, exploited as the core of the LuCe system. In terms of development tools, since a tuple centre is characterised at any time by the set T of its tuples, the set W of its pending queries, and the set S of its reaction specifications, LuCe supplies a set of *Inspectors* to view, edit and control tuple centres from the data, the pending query and the specification viewpoints.