Validation of BDI Agents

Jan Sudeikat[1,2], Lars Braubach[1], Alexander Pokahr[1], Winfried Lamersdorf[1], and Wolfgang Renz[2]

[1] Distributed Systems and Information Systems,
Computer Science Department, University of Hamburg,
Vogt–Kölln–Str. 30, 22527 Hamburg, Germany
Tel.: +49-40-42883-2091
{4sudeika,braubach,pokahr,lamersd}@informatik.uni-hamburg.de
[2] Multimedia Systems Laboratory,
Department of Information and Electrical Engineering
Hamburg University of Applied Sciences,
Berliner Tor 7, 20099 Hamburg, Germany
Tel.: +49-40-42875-8304
{sudeikat,wr}@informatik.haw-hamburg.de

Abstract. Testing and Debugging multi-agent systems (MAS) - which are inherently concurrent and distributed – is a challenging task. While complex application scenarios demand intelligent entities with autonomous reasoning capabilities, the applied reasoning mechanisms impair current approaches to validate MAS implementations. Reactive planning systems, namely the well-known *Belief Desire Intention* (BDI) architecture, have been successfully applied to implement these intelligent entities by means of goal directed agents. Despite testing and debugging, used to validate whether implementations behave as intended, are crucial to serious development efforts, only minor attention has been payed to corresponding tool support and testing procedures for BDI–based MAS. In this paper, we examine how the reasoning mechanism inside agent implementations can be checked and how static analysis of agent declarations can be used to visualize and check the overall communication structure in closed MAS. We present corresponding tool support, which relies on the definition of crosscutting concerns in BDI agents and enables both approaches to the Jadex Agent Platform.

1 Introduction

Agent-orientation proposes autonomous, proactive entities, so-called agents [1], as an atomic design and development metaphor for software systems. These entities enable a lifelike decomposition of software systems as independent actors, interacting with each other. Besides simple reactive agents [2] have been successfully applied in various application domains, the BDI architecture has been established to develop *deliberative* agents [3,4]. Methodologies and development tools are in active development to support the construction of software systems, utilizing this specific architecture. Implementations of this model use the concrete concepts of *beliefs*, *goals* and *plans*, to design and implement individual

R.H. Bordini et al. (Eds.): ProMAS 2006, LNAI 4411, pp. 185–200, 2007.

agents [5,6]. Beliefs denote the local knowledge of individual agents, goals describe the agents objectives and plans are the executable means by which agents achieve their goals. These concepts allow agents to reason pro–actively about which actions to take, i. e. plans to execute.

The autonomous nature of these entities, their complex interactions and their individual memory and reasoning capabilities introduce unprecedented levels of uncertainty [7] to these software systems. While traditional development approaches design the single flow of control in a software system, the individual agent knowledge and reasoning capabilities may lead to unexpected individual behaviors, inhibiting predictions of agent actions and interactions.

Testing and debugging are of equal importance to the efficient development of functionally correct agent systems. While testing (or validation in general) is a more or less systematic approach of discovering unknown bugs, debugging is the process of tracking down and finally removing an already known bug. Much work in the MAS area has been devoted solely to debugging (see e.g. [8,9,10]). This paper discusses approaches, which address validation issues, focusing on approaches for BDI-based agents. Concretely, we present how assertions can be used in BDI concepts to support these activities for BDI agents. In addition, we examine how agent declarations can be used to analyse structural properties of MAS communication and internal agent processing.

This paper is structured as follows. In the coming section we review the BDI architecture. In section 3 testing and validation approaches to MAS are examined and current approaches, particularly concerned with BDI architectures, are discussed. The following section 4 presents our approaches to validate agent implementations. We introduce a mechanism to execute assertions on BDI concepts and two static analysis approaches. After exemplifying their usage and implementation for the *Jadex* system (section 5), we conclude and give prospects for future work.

2 The BDI Agent Architecture

A successful architecture to develop deliberative agents is the BDI model. Bratman [3] developed a theory of human practical reasoning, which describes rational behavior by the notions *Belief*, *Desire* and *Intention*. Implementations of this model replaced the latter two by the concrete concepts *goals* and *plans*, leading to a formal theory and an executable model [4,11].

Beliefs represent the local information of agents about both the environment and its internal state. The structure of the beliefs defines a domain–dependent abstraction of the actual environment. It can be regarded as the *view point* of an agent towards its surrounding. The goals represent agents' desires, which are commonly expressed by certain target states inside the beliefs. This general concept enables pro–active agent behaviors. Agents carry out these goals on their own (see [12] for a discussion of goals in BDI systems). Finally, plans are the executable means by which agents achieve their goals. Agents can access a library of plans and deliberate which plans to execute, in order to reach a desired