Integrating BDI Agents with Agent-Based Simulation Platforms*

(JAAMAS Extended Abstract)

Dhirendra Singh
Computer Science, RMIT,
Melbourne, Australia
dhirendra.singh@rmit.edu.au

Lin Padgham
Computer Science, RMIT,
Melbourne, Australia
lin.padgham@rmit.edu.au

Brian Logan
School of Computer Science,
University of Nottingham, UK
brian.logan@nottingham.ac.uk

ABSTRACT

This paper describes an integration framework that allows development of simulations where the cognitive reasoning and decision making is programmed and executed within an existing BDI (Belief, Desire, Intention) system, and the simulation is played out in an existing ABM (Agent Based Modelling) system. The framework has a generic layer which manages communication and synchronisation, a system layer which integrates specific BDI and ABM systems, and the application layer which contains the program code for a particular application. The code is available on GitHub: `https://github.com/agentsoz/bdi-abm-integration`

1. INTRODUCTION

Agent-based simulations are often built using toolkits such as Repast [4] or NetLogo [7], which provide a graphical development interface and a suite of tools to assist in the analysis of simulation results. In such toolkits, the agents are relatively simple entities that respond reactively to their environment. However, for social science simulations involving humans, it can be challenging to represent human-like behaviour using simple reactive agents. In our simulation work with Emergency Management personnel (e.g., [5, 6]), a key concern has been that people behave in complex ways, and unless this variation can be convincingly captured, models of community response have only limited value. Informal validation by emergency services personnel often involves determining how *believable* the modelled behaviours of residents are, based on what they typically see during actual incidents. We have found the Belief, Desire, Intention (BDI) model to be a good candidate for this, as it is able to capture complex and highly variable behaviours in a compact way using a goal-plan hierarchy, while at the same time being intuitive for non-programmers.

Having identified BDI as a suitable representation of agent behaviours, we needed a way to integrate BDI agent modelling with agent-based modelling and simulation, such as MATSim [3]. This need has driven the development of the framework reported here. Although we have focussed on BDI systems, the framework can support many of the cognitive approaches to modelling humans in simulation surveyed in [1], provided they are based on agents receiving percepts, and affecting the environment via actions. Similarly, a variety of different ABMs can be accommodated.

2. ARCHITECTURAL FRAMEWORK

Conceptually, the infrastructure provides a mechanism whereby some agents in the simulation have a "brain" (decision making component) in the BDI system, while the "body" of the agent carries out actions in the ABM system. The ABM contains the environment, and it is here that percepts originate and are communicated to the brain, and actions determined by the BDI system are carried out with their consequent effects. These two systems operate synchronously, with the ABM allowing input from the BDI system at each time step. Some simple agents may not require a BDI "brain" while others may influence the simulation by communication with other agents in the BDI system, but not require embodiment in the ABM.

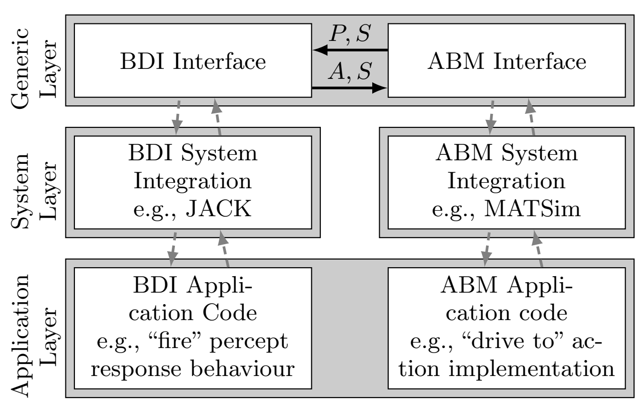

Figure 1: BDI-ABM integration architecture: ABM sends percepts list P to BDI & receives actions list A; S is list of actions status (pass, fail, suspend, abort)

The integration architecture has three distinct layers as shown in Figure 1: a *generic layer*, which realises the conceptual model of brain and body; a *system layer*, which

*We thank AOS for JACK use, Telematics Trust for partial funding of bushfire work, Kai Nagel for MATSim assistance, the RMIT Foundation for an International Visiting Fellowship that allowed the third author to visit RMIT University, as well as many contributing students and colleagues.

Appears in: *Proc. of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017), S. Das, E. Durfee, K. Larson, M. Winikoff (eds.), May 8–12, 2017, São Paulo, Brazil.*

provides the code necessary for linking a particular BDI or ABM system into the generic layer; and an *application layer*, which provides the application-specific code including agent behaviour and reasoning. We have used 3 different BDI systems and 4 different ABM simulations; however, we have most experience with the JACK/MATSim combination.

3. KEY ISSUES

There were a number of important issues to be considered in developing this framework. Critically, the BDI agent brain must be able to be both autonomous and pro-active. It cannot simply react to percepts from the environment, and must be able to modify previous decisions or take new decisions at any time. This necessitates infrastructure which we included in the generic layer, for both starting, aborting and suspending actions based on BDI reasoning.

Decisions must also be made about where agent communication should happen — within the BDI or the ABM system. There are arguments that, conceptually, communication should happen within the environment of the ABM. However there are many reasons why this may not be appropriate in all situations. If the communication has only to do with decision making regarding actions, then there may be no need to model it in the ABM. Also the size of ABM timesteps may be such that it makes no sense to model it there. An urban simulation may step each year, but there may be communications affecting cognitive decisions that happen within that year and therefore are better modelled in the BDI system. Consequently we allow either option.

Mapping of actions from the BDI system to the ABM is another important issue. Typically, the granularity of actions about which an agent reasons is much coarser than that which is required for a physically based simulation. For example, an agent does not reason about every step it takes on the way to its destination: it makes a decision to walk to somewhere. However, the ABM must progress the agent over (possibly) multiple timesteps, observing any obstacles arising. The BDI system must therefore allow actions that take multiple timesteps to complete (*durative* actions), and the ABM system must be able to recognise and communicate action completion, as well as situations that should be communicated back to the BDI system. The mechanism for managing durative actions is part of the system specific layer. We have developed a generic approach, but the details depend on the particular BDI system. Mapping of coarser granularity actions to multi step sequences in the ABM, including recognition of completion, is application specific and thus belongs at the application layer. However there are common actions such as `drive-to(loc)` which naturally become part of a reusable library for a given BDI/ABM integration.

Which system is master and which slave is also a key issue. We have found it simplest to have the ABM be the master, and the BDI system the slave, called by the ABM at each timestep. However, a model in which both are slaves to some external controller that can also integrate other subsystems may be preferable. This is a focus of future work.

An additional issue is how to determine when the BDI system should return control to the ABM system. As the BDI system is (typically) event based rather than time stepped, this requires determining when all agents have handled all their current events and finished progressing all current intentions. The possibility of communication within the BDI system results in some technical nuances, the solutions to which are described within the full paper.

4. EXAMPLE APPLICATION(S)

We have built five applications for very different domains using our infrastructure. These include a bushfire evacuation, a conservation ethics model, a simple taxi service application, a vaccination simulation and the zombies example used by Repast.

Our most complex integration is the bushfire evacuation, where we have developed a number of end-user applications with BDI modelling of agent behaviour and MATSim as the ABM. These applications aid in the planning and preparation for evacuations of regional towns or outlying city areas in the event of imminent bushfires. Road networks are extracted from OpenStreetMap,[1] and the agent population is constructed using demographics data from census tables. Our largest evacuation simulation [2] contains over 35,000 agents, each representing a household in a specific bushfire prone region. Agents representing evacuating residents exhibit some of the known behaviours of residents in bushfires, such as driving to pick up children from school and/or loved ones from nearby locations before driving to designated evacuation centres. Important percepts include `fire-alert` and `road-congestion`, with key BDI actions being `drive-to(loc)` (used in all our simulations with MATSim) and `take-route-to(loc, route)` which allows for greater reasoning about viable routes based on knowledge of the fire.

REFERENCES

[1] T. Balke and N. Gilbert. How do agents make decisions? A survey. *Journal of Artificial Societies and Social Simulation (JASSS)*, 17(4):13, 2014.

[2] C. Bulmulla, L. Padgham, D. Singh, and J. Chan. The importance of modelling realistic human behaviour when planning evacuation schedules. In *Autonomous Agents and Multi-Agent Systems (AAMAS)*, Sao Paolo, Brazil, May 2017.

[3] A. Horni, K. Nagel, and K. Axhausen, editors. *The Multi-Agent Transport Simulation MATSim*. Ubiquity Press, London, 2016.

[4] M. J. North, T. R. Howe, N. T. Collier, and J. R. Vos. *Advancing Social Simulation: The First World Congress*, chapter A Declarative Model Assembly Infrastructure for Verification and Validation, pages 129–140. Springer Japan, Tokyo, Japan, 2007.

[5] L. Padgham, R. Horne, D. Singh, and T. Moore. Planning for sandbagging as a response to flooding: A tool and case study. *Australian Journal of Emergency Management (AJEM)*, 29:26–31, Oct. 2014.

[6] D. Scerri, S. Hickmott, K. Bosomworth, and L. Padgham. Using modular simulation and agent based modelling to explore emergency management scenarios. *Australian Journal of Emergency Management (AJEM)*, 27:44–48, July 2012.

[7] S. Tisue and U. Wilensky. Netlogo: A simple environment for modeling complexity. In *International Conference on Complex Systems (ICCS)*, pages 16–21, Boston, MA, USA, 2004.

[1] `www.openstreetmap.org`