

Chapter 2

Agent-Based Modeling and Simulation

Computer simulation, or just simulation, is a decision support technique that enables stakeholders to conduct experiments with models that represent real-world systems of interest [1]. Simulation modeling in healthcare commonly incorporates random variations to represent certain key characteristics or behaviours. The applications in healthcare in silico are of considerable value, since it is often not possible or too difficult, dangerous or unethical to do them in vivo. Several different classifications of simulations have been conducted. Jun et al. classified them into two categories: management of patient flow and resource allocation [2]. Brailsford identified three main groups of models in terms of detail levels on which the models are focusing: at the human body level, at the healthcare unit level, and at the system-wide level [3]. Most recently, after reviewing healthcare simulation literature that have been published between 1970 and 2007 in high-quality journals, Mustafee et al. identified four simulation techniques: Discrete Event Simulation (DES), Monte Carlo Simulation (MCS), System Dynamic (SD) and Agent-Based Simulation (ABS) [4]. In this review, we limit our scope to only discuss ABS and diabetes models.

This chapter discusses the software agent systems and the ABS in healthcare, and then focuses on diabetes models. Subsequently, we present several related applications and systems on a more detailed level. A summary is given in the last section of this chapter.

2.1 Software Agents

Software agent systems have been implemented in an ever-increasing application space, from workflow management to data mining, from business process reengineering to Personal Digital Assistants (PDAs), and from education to bioengineering. This section will discuss the basic concept of the software agent and its classification as well as multi-agent systems. TRILabs Execution Environment for

Mobile Agents (TEEMA) is also presented since it has been adopted as an Agent Execution Environment (AEE) in this work.

2.1.1 Basic Concepts

The concept of an agent can be traced back to the early days of research into the field of Distributed Artificial Intelligence (DAI) in the 1970s, which include Carl Hewitt's concurrent Actor model [5]. In this model, Hewitt proposed a term of 'actor', which proposes the concept of a self-contained, interactive and concurrently-executing software object. An actor has an encapsulated internal state, a mail address and behaviour, and can also communicate with other actors by messaging [6].

It is very difficult to precisely define an agent. Even within the software affiliation, the word 'agent' is really an umbrella term for a variety of research and development [5]. The response to this lack of definition is that some agent researchers have invented many synonyms such as softbots (software robot), personal agents and autonomous agents [5]. Nonetheless, we try to propose one definition here to make an agent clear to the audience. "An agent is referred to as a component of software and/or hardware which is capable of acting exactly in order to accomplish tasks on behalf of its user [5]." Additionally, commonly accepted concepts of software agents are listed as follows [5, 7, 8]:

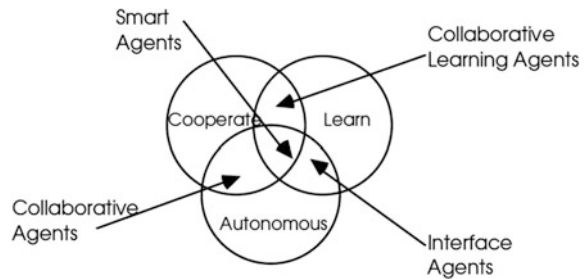
- **Autonomy:** Agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal state [9].
- **Social Ability/Cooperation:** Agents interact with other agents (and possibly humans) via some kind of agent-communication language [10].
- **Reactivity/Proactivity:** Agents perceive and respond to their environment in a timely fashion to adapt their behaviour accordingly. The environment may be the physical world, a user via a Graphical User Interface (GUI), a collection of other agents, the Internet, or perhaps even all of these combined.
- **Learn:** Agents would have to learn as they react and/or communicate with their peers and their environment.

2.1.2 Agent Classification

Software agents are difficult to define, as demonstrated, and similarly complicated to categorize. According to the overview work of Nwana [5], there are numerous ways to classify existing software agents.

Firstly, agents can be classified by their mobility: for instance, the ability of the agent to move around networks. Thus, agents can be categorized into the classes of either static or mobile agents.

Fig. 2.1 A part view of an agent typology [5]



Secondly, they may be classified in terms of either deliberative or reactive agents. Deliberative agents derive from the deliberative thinking paradigm in which the agents have an internal symbolic, reasoning model, and these agents communicate and negotiate with other agents to achieve coordination. By contrast, reactive agents do not have any internal, symbolic models of their environment, and they behave using a stimulus/response type of action by responding to the present state of the existing environment [11].

Thirdly, agents may be categorized based on several ideal and primary attributes which agents should exhibit. Nwana et al. identified three attributes: autonomy, learning and cooperation, rather than the list in Sect. 2.1.1, this is because Nwana et al. classify reactivity a learning method. Nonetheless, they use these three attributes and classify agents as collaborative agents, collaborative learning agents, interface agents and truly smart agents (Fig. 2.1). Truly smart agents, however, have not yet been developed. As Maes notes, “current commercially available agents barely justify the name” [12]. Foner is even more emphatic [13].

Fourthly, agents can also be classified by other characteristics, i.e. roles, or any combination of two or more attributes.

Nwana finally classifies software agents into seven types to cover most currently existing agents: collaborative agents, interface agents, mobile agents, information/Internet agents, reactive agents, hybrid agents, and smart agents [5].

We discuss mobile agents in the next section, as the technique that we use in this work follows in this typology.

2.1.3 Mobile Agents

Mobile agents are computational software processes accomplished by roaming Wide Area Networks (WANs) such as the World Wide Web (WWW), communicating with other types of hosts, accumulating information on behalf of its owner and coming ‘back home’ having achieved the duties set by its user [5]. However, not all agents are mobile and mobility does not guarantee an agent [14]. Mobile agents are agents, as they possess at least one of the attributes listed in Sect. 2.1.1.

A key point about mobile agents is that agents need not be stationary. An agent can just stay in one place and interact with its environment through conventional means, such as remote procedure calling and messaging [14]. We call these agents stationary agents. A mobile agent, by contrast, is not bound to the system on which it executes [15]. The idea is that mobility in the agent can provide significant benefits in certain applications in comparing to their stationary counterparts. The benefits of mobility are identified below as a number of practical advantages [5, 14].

- **Reduced communication costs:** Transferring only useful information instead of raw data can be very time-efficient and will save cost bandwidth.
- **Overcoming limited local resources:** Mobile agents are capable of using other machines that have more processing power and storage than local counterparts.
- **Easier coordination:** Collating all the results locally can be simpler to coordinate a number of remote and independent requests.
- **Asynchronous computing:** Mobile agents may be ‘set off’ when you can do something else and come back later. They may also perform in other machines offline.
- **A flexible distributed computing architecture:** Mobile agents offer an innovative way of doing distributed computation which functions differently from static set-ups.
- **Dynamic adaption:** Mobile agents can react autonomously to changes by sensing their execution environment. Multiple mobile agents possess unique capabilities of distributing themselves among a complex architecture to maintain the optimal configuration for solving a particular problem.

As discussed, mobile agents can travel around networks and interact with other agents; therefore, in most cases, they embed in a MAS.

2.1.4 *Multi-agent Systems*

MAS are a body of multiple autonomous agents that interact, cooperate, and negotiate with each other in order to satisfy their design objectives [16–18]. Two main characteristics can be derived from the definition. Firstly, each agent is autonomous and is able to solve the problem in its domain. However, it only has incomplete information or limited capabilities for solving the whole problem and, thus, has a limited viewpoint. Secondly, through agent interaction, the system can address a complex problem that is beyond the capability of individual agents. Furthermore, the interaction can be either cooperative or competition/negotiation in order to move close to an optimal solution.

If a problem domain is especially complex, huge, or/and unpredictable (i.e. modeling of the healthcare system), then the only way it can reasonably be addressed is to develop a number of functionally specific agents that are specialized for solving a specific problem aspect. For instance, it is impossible to simulate the

healthcare system in a single model because of its large scale and enormous complexity. An ideal alternative is to model each individual component of the system, such as a patient, a physician, and a hospital ward, instead of the whole healthcare system. Through observing the interaction of these components, the system response can be evaluated. The advantages of this method are incremental development, simpler individual component development, a capability to incorporate complex/competing goals and/or constraints, etc.; as discussed previously.

2.1.5 TEEMA Agent Platform

To execute, agents need an Agent Execution Environment (AEE) or Agent Platform (AP). This is a software system that provides a runtime environment for agents to execute and a standard interface for interactions, services for creation, migration and termination of mobile agents, and supports agent mobility and communication while providing security for both hosts and agents [19].

TEEMA is such an AEE and was adopted as the platform in this work because of its availability and its familiarity to the authors. It has been developed in Java jointly by TRILabs Regina and the University of Regina. Just like any other AEE, TEEMA provides standard libraries to support various types of operations for agents such as addressing, naming, messaging, mobility, security and logging [19, 20]. TEEMA also supports a multi-agent system and provides multitasking services.

2.2 Agent-Based Simulation in Healthcare

ABS is a relatively new approach for modeling systems consisting of autonomous and interacting agents. A growing number of agent-based applications, most of which are MAS, have been developed to deal with many different types of problems in the health care domain. Such applications can be found in areas such as patient scheduling, community care, organ and tissue transplant, information access, decision support systems, training, internal hospital tasks, senior citizen care etc. [21].

This area draws the attention of researchers due to its many advantages. Firstly, ABS allows people to explicitly model the complexity inherent in the healthcare system arising from individual actions and interactions in the system, which is either not possible or not readily accommodated utilizing traditional modeling techniques, such as DES or SD [22]. A second advantage of ABS is that it enables the interconnection and interoperation of multiple existing legacy systems. It also provides solutions that efficiently utilize distributed information and expertise. Another advantage comes from distributed computing such as computational effectiveness, reliability and maintainability

References

1. M. Pidd, *Computer simulation in management science*: Wiley, (2004)
2. J. Jun, S. Jacobson, J. Swisher, Application of discrete-event simulation in health care clinics: a survey. *J. Oper. Res. Soc.* **50**, 109–123 (1999)
3. S.C. Brailsford, Advances and challenges in healthcare simulation modeling: tutorial, 2007, pp. 1436–1448
4. N. Mustafee, K. Katsaliaki, S.J.E. Taylor, Profiling literature in healthcare simulation. *Simulation* **86**, 543 (2010)
5. H.S. Nwana, Software agents: An overview. *Knowl. Eng. Rev* **11**, 205–244 (1996)
6. C. Hewitt, Viewing control structures as patterns of passing messages. *Artif. Intell.* **8**, 323–364 (1977)
7. M. Wooldridge, N.R. Jennings, Intelligent agents: theory and practice. *Knowl. Eng. Rev.* **10**, 115–152 (1995)
8. S. Gill, R. Paranjape, A Review of recent contribution in agent-based healthcare modeling, in *Multi-agent Systems for Healthcare Simulation and Modeling: Applications for System Improvement*, ed. by R. Paranjape, A. Sadanand. Information Science Reference-Imprint of: IGI Publishing, 2009, pp. 26–44
9. C. Castelfranchi, Guarantees for autonomy in cognitive agent architecture, *Intelligent Agents*, pp. 56–70, 1995
10. M.R. Genesereth, S.P. Ketchpel, Software agents, *Commun. ACM*, vol. 37, pp. 48–53, 147, 1994
11. J. Ferber, Simulating with reactive agents. *Many Agent Simul. Artif. Life* **36**, 8–28 (1994)
12. P. Maes, Intelligent software, 1997, pp. 41–43
13. L.N. Foner, What’s an agent, anyway? a sociological case study, *FTP Report MIT Media Lab*, vol. 1, 1993
14. D.B. Lange, M. Oshima, Seven good reasons for mobile agents. *Commun. ACM* **42**, 88–89 (1999)
15. D. Lange, M. Oshima, O. Mishuru, Programming and deploying Java mobile agents with Aglets, 1998
16. M.J. Wooldridge, *An introduction to multiagent systems*: Wiley, 2002
17. E. Oliveira, K. Fischer, O. Stepankova, Multi-agent systems: which research for which applications. *Robot Auton. Syst.* **27**, 91–106 (1999)
18. K.P. Sycara, Multiagent systems. *AI Mag* **19**, 79 (1998)
19. C. Gibbs, TEEMA Reference Guide, Version 1.0, 2000
20. R. Martens, L. Benedicenti, TEEMA TRILabs Execution Environment for Mobile Agents, 2001
21. J. Nealon, A. Moreno, Agent-based applications in health care, *Applications of software agent technology in the health care domain*, pp. 3–18, 2003
22. P.O. Siebers, C.M. Macal, J. Garnett, D. Buxton, M. Pidd, Discrete-event simulation is dead, long live agent-based simulation! *J. Simul.* **4**, 204–210 (2010)

The Diabetic Patient Agent

Modeling Disease in Humans and the Healthcare
System Response

Paranjape, R.; Wang, Z.G.; Gill, S.

2018, XIV, 123 p. 81 illus., 74 illus. in color., Hardcover

ISBN: 978-3-662-56289-5