Chapter 12
The Integration of Agent-Based Modelling and Geographical Information for Geospatial Simulation

Andrew T. Crooks and Christian J.E. Castle

Abstract Within this chapter we focus on the integration of Geographical Information System (GIS) and Agent-based modelling (ABM) and review a selection of toolkits which allow for such integration. Moreover, we identify current capabilities of modelling within a GIS and methods of coupling and integrating GIS with agent-based models. We then introduce suggested guidelines for developing geospatial simulations with ABM toolkits and offer practical guidelines for choosing a simulation/modelling system before providing a review of a number of simulation/modelling systems that allow for the creation of geospatial agent based models along with the identification of a number references for further information.

12.1 Introduction

The Agent-Based modelling (ABM) paradigm is developing into a powerful tool in many disciplines as seen in Crooks and Heppenstall (2012), Johansson and Kretz (2012) and Harland and Heppenstall (2012), but also in a other disciplines such as archaeology (Axtell et al. 2002), economics (Tesfatsion and Judd 2006), health (Epstein 2009), geography (Batty 2005) and computational social science more generally (see Cioffi-Revilla 2010 for a discussion). Such models allow researchers

A.T. Crooks (✉)
Department of Computational Social Science, George Mason University, Research 1, 4400 University Drive, Fairfax, VA 22030, USA
e-mail: acrooks2@gmu.edu

C.J.E. Castle
Mott MacDonald Group, Mott MacDonald House, 8-10 Sydenham Road, Croydon, Surrey CR0 2EE, UK
e-mail: Christian.Castle@mottmac.com

A.J. Heppenstall et al. (eds.), *Agent-Based Models of Geographical Systems*,
DOI 10.1007/978-90-481-8927-4_12,

to explore how through the interaction of many individuals more emergent phenomena arise. Moreover, it allows for practitioners to build models of complex social phenomenon by simulating the interactions of the many actors in such systems. Thus gaining insights that will lead to greater understanding and, in some cases, better management of the behaviour of complex social systems. The intention of this chapter is to the outline how one can develop geospatial agent-based models (i.e. that model spatially explicit geographic phenomena – where the nature of the features and movement that is represented varies over the Earth's surface). Essentially, geospatial models depend on the location of the features or phenomena being modelled, such that if one or more of those locations change, the results of the model change (Wegener 2000). Geographical Information Systems (GIS) are a particularly useful medium for representing model input and output of a geospatial nature. However, GIS are not well suited to dynamic modelling (Goodchild 2005; Maguire 2005) as will be discussed in Sect. 12.2. Consequently, Sect. 12.2.2 explores the opportunity of linking (through coupling or integration/embedding) a GIS with a simulation/modelling system purposely built for the task at hand (Sect. 12.3), and therefore better suited to supporting the requirements of ABM.

12.2 Modelling Within GIS: Current Capabilities

It can be difficult to comprehend how GIS technology, built essentially for handling maps and "map-related ideas", can be adapted to the needs of dynamic simulation modelling; especially when it is not even perceived as an optimal platform for modelling (Goodchild 2005). Particular criticisms of GIS with respect to modelling is their ability to handle time (Langran 1992; Peuquet 2005– see Sect. 12.2.1), the representation of continuous variation (Longley et al. 2005), and most have only rudimentary modelling capabilities (Maguire 2005). Nevertheless, there are several good reasons to justify why the use, or linkage of GIS with simulation/modelling systems (see Sect. 12.2.2), is an effective means of modelling when spatial and temporal analysis is necessary.

Current commercial and public domain GIS software systems all contain numerous tools for acquiring, pre-processing, and transforming data. Their use in modelling includes data management, format conversion, projection change, re-sampling, raster-vector conversion, etc. GIS also include excellent tools for visualisation/mapping, rendering, querying, and analysing model results, as well as assessing the accuracies and uncertainties associated with inputs and outputs.

Typically, all of the capabilities described above are accessible via end-user graphical and command line interfaces. However, these capabilities have recently become accessible through application programming interfaces (APIs), via software libraries. The exposure of APIs was a significant recent improvement in terms of GIS and spatial modelling, as external programmers now have access to the underlying software components upon which GIS software vendors base their end-user versions of systems. This is perhaps the most pertinent enhancement, as many

of the techniques used in GIS analysis are potentially far more robust if they can be linked with an extensive toolkit of methods for simulation; an issue which is addressed at greater length later in Sect. 12.2.2. GIS vendors have invited this situation as it allows GIS to be extended and customised for use in new application areas, thus expanding the market potential of their systems.

Alternatively, a model can be expressed as a sequence of GIS commands executed by a script (Maguire 2005). Recently in GIS there has been a move to use industry-standard low-level programming languages (e.g. Java, C++, and Visual Basic), and scripting languages (e.g. Python, VBScript, and Jscript), rather than proprietary, home grown scripting languages (e.g. ESRI's Arc Macro Language, AML, or Avenue). Interoperability standards such as the Microsoft. Net framework facilitate this process by allowing compliant packages to be called from the same script.

In addition to scripts, graphical flowcharts can be used to express sequences of operations that define a model. Longley et al. (2005) note that one of the first graphic platforms for conceptualising and implementing spatial models was probably the ERDAS IMAGINE software, which allows the user to build complex modelling sequences from primitive operations. ESRI is another GIS vendor that provides an environment that allows models to be authored and executed in a graphical environment: ModelBuilder within ArcGIS 9.x, which superseded Spatial Modeller within ArcView 3.

In principle, graphic-model building can be used for dynamic modelling via an iterative process, where the output of one time step becomes the input for the next. However, this method posses two dilemmas: (1) the GIS will not have been designed for an iterative process, requiring the user to re-enter the data at the beginning of each time step, and; (2) the time required to run a model could be considerable. The former of these problems can be overcome with scripting languages (e.g. Python in ArcGIS); both can potentially be overcome by integrating the GIS with a simulation/modelling system better equipped for the task at hand. Before exploring the possibilities of linking GIS and simulation/modelling systems (Sect. 12.2.2), the following section of this chapter evaluates the capability of GIS to handle space-time information, which computer simulations generate in volume, and has always been a limitation.

12.2.1 Representing Time and Change Within GIS

The subject of time within GIS has received a considerable amount of attention. Heywood et al. (2006) comments that ideally, GIS would be able to represent temporal change using methods that explicitly represent spatial change, as well as different states through time. Furthermore, methods allowing direct manipulation and comparison of simulated or observational data in a temporal and spatial dimensions should be catered for. In reality, two main challenges for the integration of time within GIS exist: (1) continuous data over a period of time are rarely available for an entity or system of interests; (2) data models and structures able to record, store,

and visualise information about an object in different temporal states are still in their infancy (Heywood et al. 2006). In the context of this chapter, the former challenge is less of a constraint since an agent-based computer simulation is capable of generating an abundance of data over a continuous period of time, while much progress has been made on the later issue. The following discussion outlines issues related to the representation of time and change, as well as approaches for incorporating space-time information within GIS.

The basic objective of any temporal database is to record change over time, where change can be thought of as an event or collection of events. An event might be a change in state of one or more locations, entities, or both. Changes that might affect an event can be distinguished in terms of their temporal pattern; Peuquet (2005) has suggested four types: (1) continuous – events occurring throughout some period of time; (2) majorative – events occurring most of the time; (3) sporadic – events occurring some of the time, and; (4) unique – events that only occur once. The distribution of events within these temporal patterns can also be very complex (e.g. chaotic, cyclic, or steady state), complicated further as change, to some extent, is always occurring at various rates as well (e.g. from sudden to gradual). Hence, duration and frequency are important descriptive characteristics within this taxonomy of temporal patterns.

There are three approaches for capturing space-time information within a GIS: (1) location-based; (2) time-based, and; (3) entity-based. The only method of viewing a data model within existing GIS, as a space-time representation, is as a temporal series of spatially-registered 'snapshots' (Peuquet 2005). Invariably this approach employs a raster data model, although vector has also been used, with only a single information type stored (e.g. elevation, density, precipitation, etc.) for each cell at any one point in time. Information for the entire layer is stored for each time step, regardless of whether change has occurred since the previous step. There are several criticisms of this approach. Firstly, the data volume increases enormously, because redundant data is stored in consecutive snapshots. The state of a spatial entity can only be retrieved by querying cells of adjacent snapshots, because information is stored implicitly between each time step. Finally, the exact point when change has occurred cannot be determined. Langran (1992) has proposed a modification of this approach. The temporal-raster (or grid) approach allows multiple values to be stored for each pixel. A new value, and the time at which change occurred for each pixel is stored, which can result in a variable number of records for each cell. Recording the time at which change has occurred allows for values to be sorted by time. The most recent value for each cell can therefore be retrieved, which represents the present state of the system. The obvious advantage to this approach is the reduction of redundant data stored for each cell.

Peuquet and Duan (1995) have proposed a time-based approach to storing space-time information within a GIS, where change is stored as a sequence of events through time. Time is stored in increasing order from an initial point, with the temporal interval correlating to successive events. An event is recorded at the time when the amount of accumulated change is considered significant, or by another domain-specific rule. This type of representation has the advantage of

facilitating time-based queries, and the addition of a new event is straight forward as it can simply be added to the end of the timeline. Furthermore, in terms of modelling an important capacity of any model is the ability to represent alternative versions of the same reality. The concept of representing multiple realities over time is called branching. Branching allows various model simulation runs to be compared, or simulation results to be compared to observed data. The time-based approach facilitates the branching of time in order to represent alternative or parallel sequences of events resulting from specific scenarios, because it is strictly an ordinal timeline.

Finally, several entity-based space-time models have been proposed. Conceptually these models extend the topological vector approach (e.g. coverage model); tracking changes in the geometry of entities incrementally through time. The amendment vector model was the first of this type, and extended frameworks have been proposed subsequently. Besides maintaining the integrity of entities and their changing topology, these approaches are able to represent asynchronous changes to entity geometries. However, the space-time topology of these vectors becomes increasingly complex as amendments accumulate through time. In addition, aspatial entity attributes can change over time. To record aspatial changes, a separate relational database is often used. However, if change occurs at a different rate between the spatial and aspatial aspects of an entity, maintaining the identity of individual entities becomes difficult, especially when entities split or merge.

Object-oriented data models have transformed the entity-based storage of space-time information within GIS (Zeiler 1999), and have become mainstream within commercial GIS (e.g. the geodatabase structure with ArcGIS). They have grown increasingly more sophisticated, catering for a powerful modelling environment. The object-oriented data model approach provides a cohesive representation that allows the identity of objects, as well as complex interrelationships to be maintained through time. Specifically, temporal and location behaviour can be assigned as an attribute of features rather than the space itself, which has the distinct advantage of allowing objects to be updated asynchronously. Despite the advantages of the object-oriented data model, Reitsma and Albrecht (2006) observe that, to date, no data model or data structure allows the representation of processes (i.e. recording a process that has changed the state of an object within a model).[1] Consequently, queries about where a process is occurring at an instant of time cannot be expressed with these current approaches. Notwithstanding, object-oriented data models are the canonical approach to the storage of space-time data generated by agent-based models, and their visualisation within GIS, given their complementarities. Nevertheless, the visualisation of agent-based models within GIS is still limited to a temporal series of snapshots.

[1] However this is an active research topic and holds much promise with respect to creating geospatial agent-based models (see Torrens 2009 for a more detailed discussion).

12.2.2 Linkage – Coupling Versus Integration/Embedding

Models implemented as direct extensions of an underlying GIS, through either graphic model-building or scripts, generally make two assumptions: (1) all operations required by the model are available in the GIS (or in another system called by the model); and, (2) the GIS provides sufficient performance to handle the execution of the model (Longley et al. 2005). In reality, a GIS will often fail to provide adequate performance, especially with very large datasets and a large number of iterations, because it has not been designed as a simulation/modelling engine. This one-size-fits-all approach inherent in GIS provides limited applicability, and attention has subsequently been devoted to linking, either through coupling or integration/embedding, GIS with simulation/modelling systems more directly suited to users needs. General classifications have been produced by numerous authors (e.g. Maguire 1995; Bernard and Krüger 2000; Westervelt 2002; Goodchild 2005; Longley et al. 2005; Maguire 2005). Several of their definitions now overlap as technological advance has blurred the boundaries of their classifications, whist some definitions are convoluted because terminology has been used interchangeably or sometimes inappropriately (e.g. coupling, linkage or integration). Nevertheless, categorisation of these techniques is possible, and a brief description of each is developed below, in an attempt to clarify the situation. This is followed by a critique of these different approaches, with a view to identifying an appropriate method for developing geospatial agent-based models.

In situations where GIS and simulation/modelling systems already exist (e.g. as commercial products), or the cost of rebuilding the functionality of one system into another is too great, the systems can be coupled together (Maguire 2005). Coupling can therefore be broadly defined as the linkage of two stand-alone systems by data transfer. Three types of coupling are distinguishable, although these are only a subset of the much larger fields of enterprise application integration (Linthicum 2000) and software interoperability (Sondheim et al. 2005). The attributes of each approach cascaded along the coupling continuum, from loose to tight/close (Table 12.1 summaries the competing objectives of the different coupling approaches; greyed boxes are considered more desirable characteristics – adapted from Westervelt 2002):

1. **Loose Coupling.** A loose connection usually involves the asynchronous operation of functions within each system, with data exchanged between systems in the form of files. For example, the GIS might be used to prepare inputs, which are then passed to the simulation/modelling system, where after execution the results of the model are returned to the GIS for display and analysis. This approach requires the GIS and simulation/modelling system to understand the same data format; if no common format is available an additional piece of software will be required to convert formats in both directions. Occasionally, specific new programmes must be developed to perform format modifications;
2. **Moderate Coupling.** Essentially this category encapsulates techniques between loose and tight/close coupling. For example, Westervelt (2002) advocates remote procedure calls and shared database access links between the GIS and

Table 12.1 Comparison of coupling approaches (Adapted from Westervelt 2002)

Objective and explanation	Loose	Moderate	Close/tight
Integration Speed: The programmer time involved in linking the programmes	Fast	Medium	Slow
Programmer Expertise: Required level of software development expertise	Low	High	Medium
Multiple Authorship Avoidance: In some instances it might be necessary for the programmer to modify the original software product. Any alteration reduces the ownership responsibility. Major alterations could totally sever this link, resulting in limited or no support by the original author(s)	High	Medium	Low
Execution Speed: How rapidly does the integrated software execute?	Slow	Medium	Fast
Simultaneous Execution: Can components of the system run simultaneously and communicate with one another? Can the components operate on separate platforms?	Low	Low	High
Debugging: How difficult is it to locate execution errors in the linked system?	Easy	Moderate	Hard

simulation/modelling system, allowing indirect communication between the systems. Inevitably, this reduces the execution speed of the integrated system, and decreases the ability to simultaneously execute components belonging to the different software; and,

3. **Tight or Close Coupling.** This type of linkage is characterised by the simultaneous operation of systems allowing direct inter-system communication during the programme execution. For example, standards such as Microsoft's COM and .NET allow a single script to invoke commands from both systems (Ungerer and Goodchild 2002). A variant of this approach allows inter-system communication by different processes that may be run on one of more networked computers (i.e. distributed processing).

Coupling has often been the preferred approach for linking GIS and simulation/modelling systems. However, this has tended to result in very specialised and isolated solutions, which have prevented the standardisation of general and generic linkage. An alternative to coupling is to embed or to integrate the required functionality of either the GIS or simulation/modelling system within the dominant system using its underlying programming language (Maguire 2005). The final system is either referred to as GIS-centric or modelling-centric depending on which system is dominant. In both instances, the GIS tools or modelling capabilities can be executed by calling functions from the dominant system, usually through a graphical user interface (GUI). Compared to coupling, an embedded or integrated system will appear seamless to a user (Maguire 1995). However, in the past integration has been based on existing closed and monolithic GIS and simulation systems, which poses a risk of designing systems that are also closed, monolithic, and therefore costly (Fedra 1996).

Interest in modelling-centric systems has increased considerably over recent years, predominately due to the development of simulation/modelling toolkits with scripting capabilities that do not require advanced computer programming skills (Gilbert and Bankes 2002). Often the simulation/modelling toolkit can access GIS functions, such as data management and visualisation capabilities, from a GIS software library. For example, the RepastJ (see Sect. 12.3.3.3) toolkit exploits functions from GeoTools (a Java GIS software library) for importing and exporting data, Java Topology Suite (JTS) for data manipulation, and OpenMap for visualisation. The toolkit itself maintains the agents and environment (i.e. their attributes), using identity relationships for communication between the different systems. Functions available from GIS software libraries reduce the development time of a model, and are likely to be more efficient because they have been developed over many years with attention to efficiency. Additionally, the use of standard GIS tools for spatial analysis improves functional transparency of a model, as it makes use of well known and understood algorithms. Alternatively, spatial data management and analysis functions can be developed within the modelling toolkit, although this strategy imposes huge costs, in terms of time to programme the model, and time required to frequently update spatial data or use spatial analysis functions within the model.

Conversely, the GIS-centric approach is an attractive alternative; not least because the large user-base of some GIS expands the potential user-base for the final model. Analogous to the modelling-centric approach, GIS-centric integration can be carried out using software libraries of simulation/modelling functions accessed through the GIS interface. There are many examples of simulation/modelling systems integrated within commercial GIS, including: the Consequences Assessment Tool Set (2011, CATS) system, designed for emergency response planning; the Hazard Prediction and Assessment Capability (2004, HPAC) system, for predicting the effect of hazardous material releases into the atmosphere; the NatureServe Vista (2011) system, for land use and conservation planners.

Brown et al. (2005) propose an alternative approach which straddles both the GIS-centric and modelling-centric frameworks. Rather than providing functionality within one system, the middleware-based approach manages connections between systems, allowing a model to make use of the functionality available within the GIS or the simulation/modelling toolkit most appropriate for a given task. Thus, the middleware approach allows the simulation/modelling toolkit to handle the identity and relationship of, and between agents and their environment. Conversely, the GIS would manage spatial features, as well as temporal and topological relationships of the model. Essentially, the simulation/modelling toolkit handles what it is designed for (i.e. implementing the model), while the GIS can be used to run the model, and visualise the output. An example of this approach is the ABM extension within ArcGIS (referred to as Agent Analyst), which allows users to create, edit, and run RepastPy models from within ArcGIS (Redlands Institute 2010). However, it is the opinion of the authors that only a dichotomy of integration classifications exists. A GIS is either integrated into a simulation/modelling toolkit, or vice versa. The definition of the middleware approach is essentially tight coupling (see above).

12.3 Developing Geospatial Simulations with Agent-Based Modelling Toolkits

The process of building an agent-based model begins with a conceptual model, where basic questions or goals, elements of the system (e.g. agent attributes, rules of agent interaction and behaviour, the model environment, etc.), and the measurable outcomes of interest are identified (Brown 2006). It is important to 'ground' a model during the conceptualisation process (i.e. establish whether simplifications made during the design process do not seriously detract from the credibility and likelihood that the model will provide important insights; Carley 1996). It is usual for a modeller to set forth a claim as to why the proposed model is reasonable. This claim will be enhanced if the applicability of the model is not over stated, and by defining the models limitations and scope. Grounding can be reinforced by demonstrating that other researchers have made similar or identical assumptions in their models, and by justifying how a proposed model will be of benefit in relation to pre-existing models.

Conceptualising the fundamental aspects of an agent-based model (i.e. one or more agents interacting within an environment), juxtaposed with the distinction between explanatory vs. predictive purposes of a model suggests a fourfold typology of agent and environment types (Table 12.2). Couclelis (2001) classifies agents and their environment as either being designed (i.e. explanatory) or analysed (i.e. predictive – empirically grounded). If designed, agents are endowed with attributes and behaviours that represent (often simplified) conditions for testing specific hypotheses about general cases. Analysed agents are intended to accurately mimic real-world entities, based on empirical data or ad hoc values that are realistic substitutes for observed processes. Similarly, the environment that agents are situated within can be designed (i.e. provided with characteristics that are simplified to focus on specific agent attributes), or analysed (i.e. represent a real-world location).

The boundary between designed and analyzed is not always distinct, especially when ad hoc data are employed. Subtle but profound differences, both practical and conceptual, exist between the design or analysis approach of developing agents and their environment. A major difference in practical terms is that designing something provides direct (partial or total) control over the outcome, whereas there can only be hope that something has been analyzed correctly (Couclelis 2001). Table 12.2 provides further details to consider when developing agents and their environment; including a brief description of the model, the purpose and intent of the model (see Parker et al. 2001), verification and validation strategies used to assess the model outputs (see Parker et al. 2001; Crooks et al. 2008), and appropriate software for the development of a model (see Sect. 12.3.2).

Once a model has been conceptualised, it must be formalised into a specification which can be developed into a computer programme (Grimm and Railsback 2012 and Abdou et al. 2012 offer constructive advice on this); if the model is required to be run as a computer simulation. The process of formalisation involves being precise about what an identified theory relating to a phenomena of interest means,

Table 12.2 Description, purpose/intent, verification and validation strategies, and appropriate development tools for agent-based models incorporating designed or analysed agents/environments (Adapted from Berger and Parker 2001)

		Agent	
		Designed	Analysed
Environment	Designed	Model description – Abstract Purpose/intent – Discovery of new relationships – Existence proof Verification and validation strategy – Theoretical comparison – Replication Appropriate development tools – Easy to implement simulation/ modelling system Example model – Filatova et al. (2009)	Model description – Experimental Purpose/intent – Role-playing games among stakeholders – Laboratory experiments Verification and validation strategy – Repetitions – Adequacy of design Appropriate development tools – Flexible simulation/modelling systems with well developed user interfaces Example model – Mooij et al. (2002)
	Analysed	Model description – Historical Purpose/intent – Explanation Verification and validation strategy – Qualitative: goodness of fit Appropriate development tools – Advanced simulation/modelling systems linked with GIS Example model – Mathevet et al. (2003)	Model description – Empirical Purpose/intent – Explanation – Projection – Scenario analysis Verification and validation strategy – Quantitative: goodness of fit Appropriate development tools – Low-level programming languages Example model – Jackson et al. (2008)

making sure that it is complete and coherent. There are several reasons why computer simulation is more appropriate for formalising social science theories than mathematics, which has often been used in the social sciences (Gilbert and Troitzsch 2005). First, programming languages are more expressive and less abstract than most mathematical techniques. Second, a computer simulation can deal more easily with parallel process and processes without well defined order or actions than systems of mathematical equations. Third, a computer model can include heterogeneous agents (e.g. pedestrians with varying degrees of knowledge about a building layout), while this is usually relatively difficult using mathematics. Finally, computer programmes are (or can easily be made to be) modular, so that major changes can be made to one part of the model without requiring large changes in other parts of the programme, an ability which mathematical systems often lack.

The object-oriented paradigm provides a very suitable medium for the development of agent-based models. In particular, it provides the aforementioned modularity useful for developing a computer simulation. It is not the intention of this chapter to outline the fundamental object-oriented concepts, this has been achieved by numerous others (refer to Booch (1994) for a seminal discussion and Armstrong (2006) for a useful evaluation and clarification of key object-oriented notions).

At the time of writing, there are many simulation/modelling systems available to assist the development stage of ABM. The majority of these simulation/modelling systems are programmed, and/or require the user to develop their model in an object-oriented language. The subsequent section of this chapter identifies some of the simulation/modelling systems available for ABM, highlighting key questions that should be considered for a user to determine an appropriate system for their needs.

12.3.1 Types of Simulation/Modelling Systems for Agent-Based Modelling

In general, two types of simulation/modelling systems are available to develop agent-based models: toolkits or software.[2] Based on this dichotomy, toolkits are simulation/modelling systems that provide a conceptual framework for organising and designing agent-based models. They provide appropriate libraries[3] of software functionality that include pre-defined routines/functions specifically designed for ABM. However, the object-oriented paradigm allows the integration of additional functionality from libraries not provided by the simulation/modelling toolkit, extending the capabilities of these toolkits. Of particular interest to this chapter is the integration of functionality from GIS software libraries (e.g. OpenMap, GeoTools, ESRI's ArcGIS, etc.), which provide ABM toolkits with greater data management and spatial analytical capabilities required for geospatial modelling (see Sect. 12.2).

The development of agent-based models can be greatly facilitated by the utilisation of simulation/modelling toolkits. They provide reliable templates for the design, implementation and visualisation of agent-based models, allowing modellers to focus on research (i.e. building models), rather than building fundamental tools necessary to run a computer simulation (see Tobias and Hofmann 2004; Railsback et al. 2006). In particular, the use of toolkits can reduce the burden modellers face programming parts

[2] An agent-based model could be programmed completely from scratch using a low-level programming language if a modeller has sufficient programming knowledge and experience; see below for disadvantages of this approach.

[3] A collection of programming classes grouped together, termed packages (i.e. classes with similar purpose).

of a simulation that are not content-specific (e.g. GUI, data import-export, visualisation/display of the model). It also increases the reliability and efficiency of the model, because complex parts have been created and optimised by professional developers, as standardised simulation/modelling functions. Unsurprisingly, there are limitations of using simulation/modelling systems to develop agent-based models, for example: a substantial amount of effort is required to understand how to design and implement a model in some toolkits; the programming code of demonstration models or models produced by other researchers can be difficult to understand or apply to another purpose; a modeller will have to learn or already have an understanding of the programming language required to use the toolkit; and finally the desired/required functionality may not be present, although additional tools might be available from the user community or from other software libraries. Benenson et al. (2005) also note that toolkit users are accompanied by the fear of discovering that a particular function cannot be used, will conflict, or is incompatible with another part of the model late in the development process.

Probably the earliest and most prominent toolkit was SWARM, although many other toolkits now exist. At the time of writing there are more than 100 toolkits available for ABM (see AgentLink 2007; SwarmWiki 2010; Nikolai and Madey 2009; Tesfatsion 2010; Wikipedia 2010 for comprehensive listings). However, variation between toolkits can be considerable. For example, their purpose (some toolkits have different design objectives e.g. Artificial Intelligence (AI) rather than social science focus, or network opposed to raster or vector model environments), level of development (e.g. some models are no longer supported or have ceased development), and modelling capabilities (e.g. the number of agents that can be modelled, degree of interaction between agents) can vary. A review of all toolkits currently available is beyond the scope of this chapter. However, we identify a selection of noteworthy simulation/modelling toolkits (e.g. Swarm, MASON, Repast, AnyLogic), highlighting their purpose and capabilities, as well as resources providing further information.

In addition to toolkits, software is available for developing agent-based models, which can simplify the implementation process. For example, simulation/modelling software often negates the need to develop an agent-based model via a low-level a programming language (e.g. Java, C++, Visual Basic, etc.). In particular, software for ABM is useful for the rapid development of basic or prototype models. However, modellers using software are restricted to the design framework advocated by the software. For instance, some ABM software will only have limited environments (e.g. raster only) in which to model, or agent neighbourhoods might be restricted in size (e.g. von Neumann or Moore). Furthermore, a modeller will be constrained to the functionality provided by the software (unlike ABM toolkits modellers will be unable to extend or integrate additional tools), especially if the toolkit is written in its own programming language (e.g. NetLogo). Section 12.3.3 identifies a selection of noteworthy software for the development of agent-based models; StarLogo, its derivative NetLogo, and AgentSheets.

12.3.2 Guidelines for Choosing a Simulation/Modelling System

Ideally, a modeller would have comprehensive practical experience in a range of modelling/simulation systems before choosing which system to use for a modelling endeavour. Unfortunately, this is not usually feasible. For this reason several authors (Najlis et al. 2001; Gilbert and Bankes 2002; Serenko and Detlor 2002; Tobias and Hofmann 2004; Dugdale 2004; Rixon et al. 2005; Robertson 2005; Andrade et al. 2008; Berryman 2008; Liebert et al. 2008; Nikolai and Madey 2009) have gained practical experience and/or have surveyed several systems, identifying key criteria that should be considered before making a decision. General criteria include, but are not limited to: ease of developing the model/using the system; size of the community using the system; availability of help or support (most probably from the user community); size of the community familiar with the programming language in which the system is implemented (if a programming language is necessary to implement the model); is the system still maintained and/or updated; availability of demonstration or template models; technical and how-to documentation, etc. Criteria relating specifically to a systems modelling functionality include: number of agents that can be modelled; degree of interaction between agents; ability to represent multiple organisational/hierarchical levels of agents; variety of model environments available (network, raster, and vector); possible topological relationship between agents; management of spatial relationships between agents, and agents with their environment; mechanisms for scheduling and sequencing events, etc. These criteria will be weighted differently depending on a modeller's personal preferences and abilities (e.g. the specification of the model to be developed, programming experience/knowledge, etc.).

Another important distinction separating simulation/modelling systems is there licensing policy; open source, shareware/freeware, or proprietary. Open source simulation/modelling systems constitute toolkits or software whose source code is published and made available to the public, enabling anyone to copy, modify and redistribute the system without paying royalties or fees. A key advantage of open source simulation/modelling systems relates to the transparency of their inner workings. The user can explore the source code, permitting the modification, extension and correction of the system if necessary. This is particularly useful for verifying a model (see Crooks et al. 2008). The predominant open source simulation/modelling systems are toolkits (e.g. MASON, Repast, Swarm, etc.). The distinction between an open source simulation/modelling system and a shareware/freeware system is subtle. There is no one accepted definition of the term shareware/freeware, but the expression is commonly used to describe a system that can be redistributed but not modified, primarily because the source code is unavailable. Consequently, shareware/freeware systems (e.g. StarLogo, NetLogo, etc.) do not have the same flexibility, extendibility or potential for verification (in relation to access to their source code), as open source systems. Similarly, shareware/freeware systems tend to be toolkits,

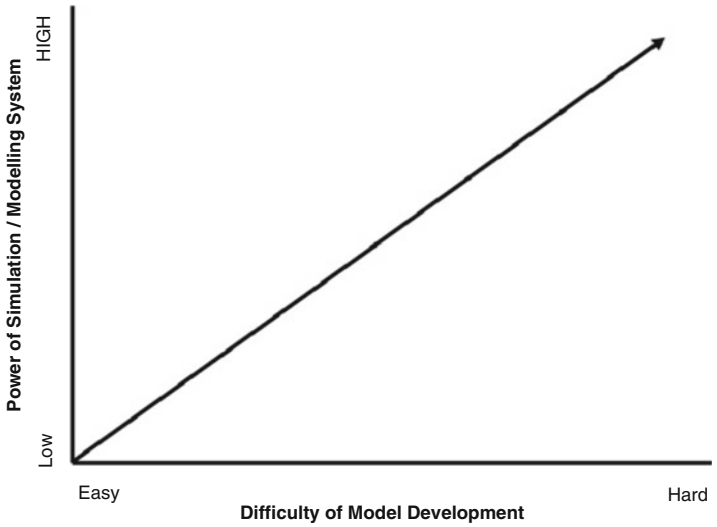

Fig. 12.1 Balance between power versus difficulty of developing a model with a simulation/modelling system

rather than software.[4] Finally, proprietary simulation/modelling systems are available for developing agent-based models. Proprietary systems are mainly software, developed by an organisation who exercises control over its distribution and use; most require a licence at a financial cost to the user. These systems have the advantage of being professionally designed and built for a specific use, and are often relatively simple to use. However, they often lack the community support found with open source or shareware/freeware systems. Moreover, since access to their source code is prohibited, a model developed with proprietary software is essentially black box. A modeller will therefore, to some extent, be left unsure about the inner validity of a model constructed with a proprietary system. This situation is compounded when the output of a model is emergent or unexpected.

Striking a balance between the aforementioned criteria is difficult. Unfortunately, while identifying a suitable system for the development of an agent-based model, too much time can often be expended trying to find this balance. This balance can be perceived as a trade off between the difficulty of developing a model (e.g. in terms of time required to programme the model, understand how to develop a model with a specific system, or acquiring experience and knowledge of a programming language if required, etc.), versus the modelling power provided by the simulation/modelling system (e.g. modelling capabilities and functionality, Fig. 12.1). The key

[4]Other shareware/freeware systems used for the creation of spatial agent-based models include OBEUS (Benenson et al. 2006) and CORMAS (Bousquet et al. 1998). These systems are not reviewed in this chapter for space requirements.

is striking a 'personal' balance between these criteria. For example, those more accustomed to programming may prefer the functionality and flexibility of a simulation/modelling toolkit. However, modellers that only wish to develop a basic or prototype model quickly and easily, possibly with little or no programming skills may prefer to use simulation/modelling software (see Railsback et al. 2006).

12.3.3 Simulation/Modelling Systems for Agent-Based Modelling

This section provides key criteria pertaining to a selection of simulation/modelling systems available for the development of agent-based models (the rationale for each criterion was described in Sect. 12.3.2). Although there are many systems available for developing agent-based models, this chapter reviews seven, separated into three categories of licensing policy (1) open source (Swarm, MASON and Repast); (2) shareware/freeware (StarLogo and NetLogo); and (3) proprietary systems (AgentSheets and AnyLogic). These systems were chosen because they fulfilled the (majority of the) following criteria, they are: maintained and still being developed; widely used and supported by a strong user community; accompanied by a variety of demonstration models and in some instances the model's programming script or source code is available; and finally they are capable of developing spatially explicit models, possibly via the integration of GIS functionality. Tables 12.3–12.5 tabularise information of each system for comparison purposes; categorised by their licensing policy (adapted from Najlis et al. 2001 and Parker 2001). The reminder of this section provides further information about each system, identifying examples of geospatial models that have been developed with the system. A caveat must be noted at this point, the information provided within this section is accurate at the time of publication. However, the systems reviewed are constantly being updated, thus modellers are advised to check each systems website to obtain up to date information.

12.3.3.1 Swarm

Swarm (Table 12.3) is an open source simulation/modelling system designed specifically for the development of multi-agent simulations of complex adaptive systems (Swarm 2010); although agent-based models can easily be develop using Swarm as well. Inspired by artificial life, Swarm was designed to study biological systems; attempting to infer mechanisms observable in biological phenomena (Minar et al. 1996). In addition to modelling biological systems (e.g. Railsback and Harvey 2002), Swarm has been used to develop models for anthropological, computer science, ecological, economic, geographical, and political science purposes. Useful examples of spatially explicit models include: the simulation of pedestrians in the urban centres (Schelhorn et al. 1999 and Haklay et al. 2001); and the examination of crowd congestion at London's Notting Hill carnival (Batty et al. 2003).

Table 12.3 Comparison of open source simulation/modelling systems (Adapted from Najlis et al. 2001 and Parker 2001)

Open source simulation/modelling systems			
	Swarm	MASON	Repast
Developers	Santa Fe Institute/SWARM Development Group, USA	Evolutionary Computation Laboratory and Center for Social Complexity, George Mason University, USA	University of Chicago, Department of Social Science Research Computing, USA
Date of inception	1996	2003	2000
Website	http://www.swarm.org	http://cs.gmu.edu/~eclab/projects/mason	http://repast.sourceforge.net
E-mail list	http://www.swarm.org/mailman/listinfo	https://listserv.gmu.edu/archives/mason-interest-l	https://lists.sourceforge.net/lists/listinfo/repast-interest
Implementation language	Objective-C/Java	Java	Java/Python/Microsoft.Net
Operating system	Windows, UNIX, Linux, Mac OSX	Windows, UNIX, Linux, Mac OSX	Windows, UNIX, Linux, Mac OSX
Required programming experience	Strong	Strong	Strong
Integrated GIS functionality	Yes (e.g. Kenge GIS library for Raster data: http://www.gis.usu.edu/swarm)	Yes	Yes (Repast 3 simulations can also be run within ArcGIS through an extension called Agent Analyst).
Integrated charting/graphing/statistics	Yes (e.g. R and S-plus statistical packages)	Yes (e.g. wrappers for JFreeChart)	Yes (e.g. Colt statistical package, and basic Repast functionality for simple network statistics) While in RepastS there are automated connections to R and VisAD
Availability of demonstration models	Yes	Yes	Yes
Source code of demonstration models	Yes	Yes	Yes

Tutorials/how-to documentation	Yes	Yes	Yes
Additional information	Minar et al. (1996) Contributed user example models: http://www.swarm.org/index.php/Swarm:_contributed_code	GeoMason website: http://www.cs.gmu.edu/~eclab/projects/mason/extensions/geomason/ Example publications: Luke et al. (2004), Kennedy et al. (2010) Useful weblog: http://www.gisagents.blogspot.com	Agent Analyst Extension (http://www.institute.redlands.edu/agentanalyst) Useful weblogs: http://www.gisagents.blogspot.com http://crimesim.blogspot.com/

Table 12.4 Comparison of shareware/freeware simulation/modelling systems (Adapted from Najlis et al. 2001 and Parker 2001)

Shareware/freeware simulation/modelling systems		
System name	StarLogo	NetLogo
Developers	Media Laboratory, Massachusetts Institute of Technology, USA	Centre for Connected Learning and Computer-Based Modelling, Northwestern University, USA
Date of inception	Early 1990s, Java based version 2000	1999
Website	http://education.mit.edu/starlogo/	http://ccl.northwestern.edu/netlogo
E-mail list	http://education.mit.edu/pipermail/starlogo-users	None
Implementation language	Proprietary scripting	Proprietary scripting
Operating system	Windows, UNIX, Linux, Mac OSX	Windows, UNIX, Linux, Mac OSX
Required programming experience	Basic	Basic
Integrated GIS functionality	None	Yes
Integrated charting/graphing/statistics	Yes	Yes
Availability of demonstration models	Yes	Yes
Source code of demonstration models	Yes	Yes
Tutorials/how-to Documentation	Yes	Yes
Additional information	OpenStarLogo website: http://education.mit.edu/openstarlogo/	http://groups.yahoo.com/group/netlogo-users http://ccl.northwestern.edu/netlogo/docs/gis.html http://backspaces.net/wiki/NetLogo_Bag_of_Tricks

Najlis et al. (2001) identify the steep learning curve of Swarm as a significant factor to consider before choosing this system to develop an agent-based model; although this should be less of a problem for a modeller with strong programming skills.

12.3.3.2 MASON

MASON (Multi Agent Simulation Of Neighbourhood – Table 12.3) is developed by the Evolutionary Computation Laboratory (ECLab) and the Centre for Social Complexity at George Mason University (see Luke et al. 2005). Currently MASON provides much of the same functionality as Repast, for example, dynamically charting

Table 12.5 Comparison of proprietary simulation/modelling systems (Adapted from Najlis et al. 2001 and Parker 2001)

Proprietary simulation/modelling systems		
	AgentSheets	AnyLogic
Developers	AgentSheets Inc., USA	XJ Technologies, Russia
Date of inception	1991	Unknown
Website	http://www.agentsheets.com	http://www.xjtek.com
E-mail list	None	None
Implementation language	Proprietary scripting	Proprietary scripting
Operating system	Windows, UNIX, Linux, Mac OSX	Windows, UNIX, Linux, Mac OSX
Required programming experience	None – Basic	Moderate
Integrated GIS functionality	None	None
Integrated charting/graphing/statistics	Yes	Yes
Availability of demonstration models	Yes http://repast.sourceforge.net/examples/index.html	Yes http://repast.sourceforge.net/examples/index.html
Source code of demonstration models	N/A	N/A
Tutorials/how-to documentation	Yes	Yes
Additional information	Carvalho 2000 and Repenning et al. 2000	http://www.xjtek.com/support/forums/general

(e.g. histograms, line graphs, etc.) and model output during a simulation. A recent addition to MASON is GeoMASON (2010) which allows GIS vector data to be imported/exported. In addition MASON also supports the use of raster data in the creation of geospatial agent-based models (e.g. Kennedy et al. 2010) as shown in Fig. 12.2.

MASON has a growing set of technical documents and well commented Javadocs and a user group which is actively supports its e-mail list. MASONs how-to documentation, demonstration models (e.g. the seminal heat bugs example, network models, etc.), and several publications detailing the implementation and/or application of MASON are available for a prospective modeller to evaluate the system further (MASON 2010). Examples of spatially explicit models utilizing MASONs GIS functionally include exploring conflict between herdsmen and farmers in East Africa (Kennedy et al. 2010), pastoralists in Inner Asia (Cioffi-Revilla et al. 2010), residential dynamics in Arlington County, Virginia (Hailegiorgis 2010) and understanding the Afghan drug industry (Łatek et al. 2010).

12.3.3.3 Repast

Repast (Recursive Porous Agent Simulation Toolkit – Table 12.3) was originally developed at the University of Chicago, and is currently maintained by Argonne

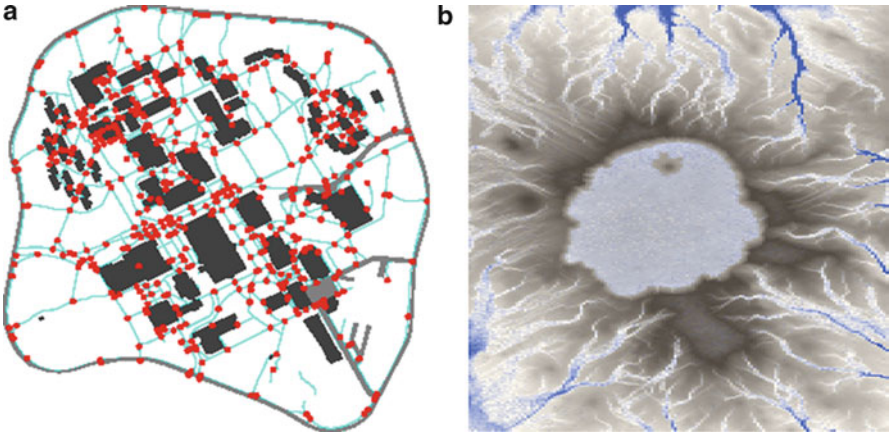

Fig. 12.2 Examples of raster and vector agent-based models in MASON. (**a**) Agents are *red* points which move around the footpaths (*Blue Lines*). (**b**) A rainfall model where agents are *blue* and flow down the Terrain (Built from a Digital Elevation Model)

National Laboratory and managed by the Repast Organisation for Architecture and Development (ROAD). Earlier incarnations of Repast catered for the implementation of models in three programming languages: Python (RepastPy); Java (RepastJ and Repast Simphony); and Microsoft.Net (Repast.Net). RepastPy allows basic models to be developed by modellers with limited programming experience via a 'point-and-click' GUI (Collier and North 2005). RepastPy models can subsequently be exported/converted into Java for further development in RepastJ. Repast.Net and RepastJ allow for more advanced models to be developed (Vos 2005), because more complex functionality can be programmed into a model. Agent Analyst is an ABM extension that allows users to create, edit, and run Repast models from within ArcGIS (Redlands Institute 2010). For further information of earlier versions of Repast, readers are referred to Crooks (2007). Repast has a relatively large user group and an actively supported e-mail list, as well as extensive how-to documentation and demonstration models available from the system website.

Whilst still being maintained RepastJ, Repast.Net and RepastPy have now reached maturity and are no longer being developed. They have been superseded by Repast Simphony (RepastS), which provides all the core functionality of RepastJ or Repast.Net, although limited to implementation in Java. For a comparison of RepastS and previous versions readers are referred to North and Macal (2009). RepastS was initially released in late 2006 and now provides the same GIS functionality of previous versions. The main improvements with RepastS over Repast 3.0 is a new optional GUI point-and-click environment for model development that generates Java classes, however models can still be coded manually. Secondly a improved runtime GUI, the GUI can now be used to build displays (both in 2 and 3D) or charts, output data, interrogate agents, and interface with other programs (like R for statistics) via a point-and-click interface at run time. This means that these tasks are

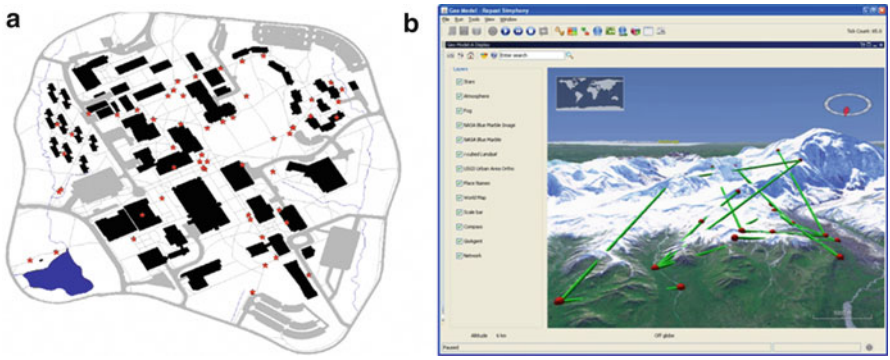

Fig. 12.3 Examples of vector agent-based models in RepastS. (**a**) Agents (*Red Dots*) moving about on footpaths (*Grey Lines*). (**b**) An agent-based model overlaid on NASA world wind (Source: Repast 2011)

done more quickly after the model has been built and compiled, and do not feature in the underlying code at all, unlike previous Repast implementations.

The Repast development team have provided a series of articles regarding RepastS. The architecture and core functionality are introduced by North et al. (2005a), and the development environment is discussed by Howe et al. (2006). The storage, display and behaviour/interaction of agents, as well as features for data analysis (i.e. via the integration of the R statistics package) and presentation of models within Repast S are outlined by North et al. (2005b). Tatara et al. (2006) provide a detailed discussion outlining how-to develop a "simple wolf-sheep predation" model; illustrating RepastS modelling capabilities. In relation to the integration of GIS functionality the reader is referred to the tutorials by Malleson, (2008) which demonstrates how to create a virtual city via the importation of shapefiles, create agents and then move the agents around a road network (this tutorial was used for the creation of Fig. 12.3a). Furthermore, within RepastS it is possible to embed spatially explicit agent-based models directly into a 3D GIS display. For this RepastS provides methods to directly visualise agent-based models to NASA's (2010) virtual globe – World Wind. This new interactive 3D GIS display allows one to visualise agents with satellite imagery, elevated terrain and other scientific datasets as shown in Fig. 12.3b. RepastS also supports the importation of NetLogo (see Sect. 12.3.3.5) models into the Repast framework via ReLogo (Ozik 2010). Such functionality aims to allow for rapid prototyping of agent-based models by first building simple agent-based models in NetLogo and once satisfied allowing one to migrate and extend them in RepastS. Not only does RepastS provide tools for the conversion of simple models from NetLogo, it also supports high performance distributed computing, via Repast for High Performance Computing (Repast HPC, see Collier 2010).

Useful examples of spatially explicit models created using Repast include the studying of segregation, and residential and firm location (Crooks 2006, 2010),

residential dynamics (Jackson et al. 2008) crime (Malleson et al. 2010) and the evacuation of pedestrians from within an underground station (Castle 2007).

12.3.3.4 StarLogo

StarLogo (Table 12.4) is an shareware/freeware modelling system developed at the Media Laboratory, Massachusetts Institute of Technology (MIT) It has undergone some change, the original StarLogo modelling system has been released as an open source project (see OpenStarLogo 2010) however, it is still included in this section as the new version, StarLogo TNG (The New Generation) is still shareware/freeware. StarLogo TNG moves StarLogo from the 2D to the 3D realm through the use OpenGL graphics API and aims to lower the barrier for programming agent-based models through the use of a drag and drop programming graphical interface. Modellers can drag commands from a set of model building blocks (a block based graphical language) rather than creating models using the StarLogo syntax thus allowing for rapid model development. StarLogo TNG uses OpenGL for displaying the models at run time therefore providing a 3D display termed 'SpaceLand'. The terrain within such models is editable and can be manually shaped. Agents can also be programmed to move in x, y and z directions.

StarLogo lacks the same flexibility offered by open source systems, since modellers are constrained to functionality provided by the system. Despite this limitation, StarLogo is very easy to use, notably for people with very little programming experience. Dynamic charting functionality of model output during a simulation is provided. In addition, a number of demonstration models and detailed how-to documentation relating to these models is supplied with StarLogo, and many more are available to download from the World Wide Web (WWW). While StarLogo does not support GIS per se, it does allow one to import GIFs, therefore allow pixels to be converted into patches. Batty et al. (1998) used this approach to examine visitor movement within London's British Tate Gallery, specifically how changes in room configuration can affect movement between exhibits.

12.3.3.5 NetLogo

NetLogo (originally named StarLogoT – Table 12.4) is a variant of StarLogo, originally developed at the Centre for Connected Learning and Computer-Based Modelling at Northwestern University, to allow StarLogo models to be developed on computers using the Macintosh operating system. It is now possible to create StarLogo models on a computer using a Macintosh operating system, thus the critically distinction between the two simulation/modelling systems is that NetLogo is specifically designed for the deployment of models over the internet (NetLogo 2010). Initially both NetLogo and StarLogo only provided functionality to import image files, which can be used to define the environments within which agents are

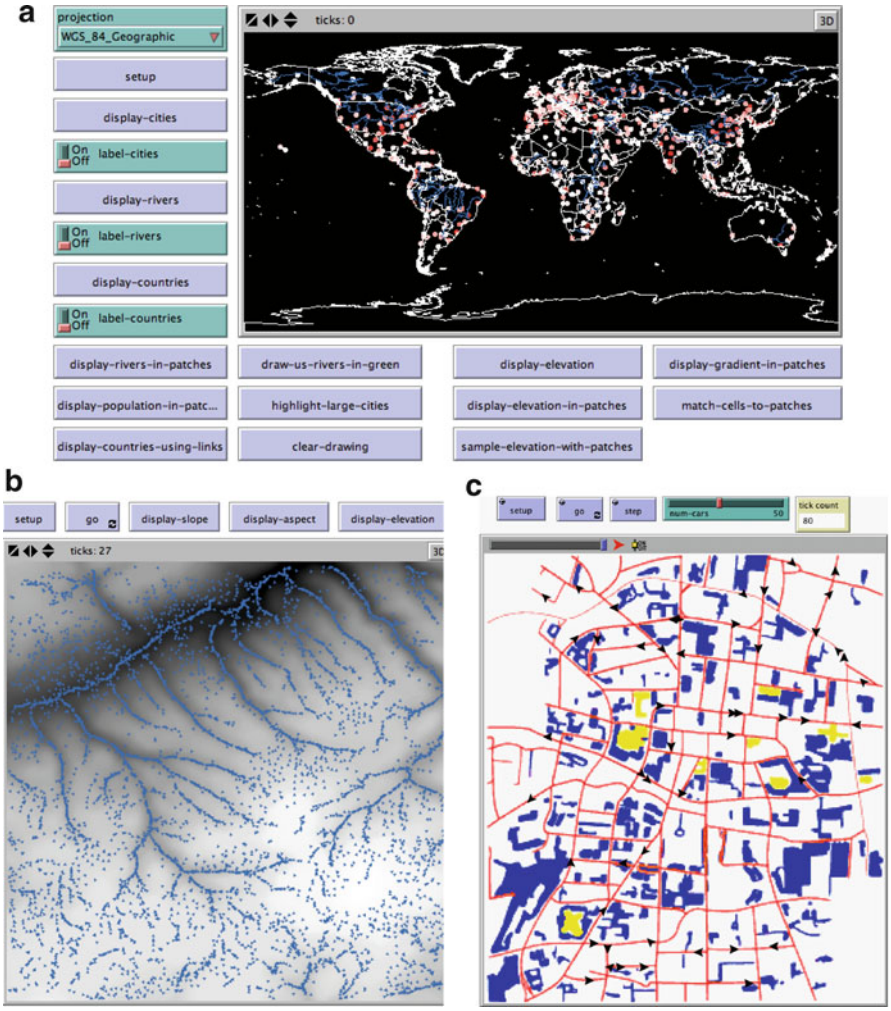

Fig. 12.4 Example of GIS integration in NetLogo. (**a**) Demonstration model of using *point*, *line* and *polygon* shapefiles for creating a landscape. (**b**) NetLogo's gradient example and (**c**) the cruising model where cars move along the roads (Red lines) (Source: NetLogo 2010)

located, thus facilitating the development of spatial models (Fig. 12.4). However, within NetLogo it is now possible to import both raster (in the form of .asc files) and vector data (shapefiles). This new ability opens up a range of possibilities for the easy creation of spatial agent based models. For example, for the studying of surface erosion (Wilensky 2006) as shown in Fig. 12.4b.

The NetLogo installation comes with two demonstration models highlighting this functionality. For vector data, four different GIS datasets: a point file of world cities, a polyline file of world rivers, a polygon file of countries (however there is

no way to distinguish if the polygon has holes in it) are imported into a NetLogo model and converted into patches as shown in Fig. 12.4a. For the raster example, a raster file of surface elevation is loaded into a NetLogo model to demonstrate the possibilities of working with spatial data as shown in Fig. 12.4b. In this example, Agents follow the surface to lower elevations. Such functionality potentially lowers the barrier between coupling agent-based models and GIS to none expert programmers. For example, the gradient example presented above could be used to model process that relies on cost surfaces such as emergency evacuation of buildings (see Crooks et al. 2008, for an example). As with StarLogo TNG (Sect. 12.3.3.4), models within NetLogo can be viewed in a 3D environment however unlike StarLogo TNG it is only the agents that appear in 3D while the surface remains a 2D plane.

NetLogo has been used to develop applications in disciplines varying from biology and physics to the social sciences. Extensive how-to documentation/tutorials and demonstration models are available from the system website, and functionality can be extended through APIs, although the source code for the system is currently unavailable. Useful examples of spatially explicit models created using NetLogo include the study of gentrification (Torrens and Nara 2007), residential housing demand (Fontaine and Rounsevell 2009) and the emergence of settlement patterns (Graham and Steiner 2006) and the reimplementation of Axtell et al. (2002) artificial Anasazi model by Janssen (2009).

12.3.3.6 AgentSheets

AgentSheets (Table 12.5) is a proprietary simulation/modelling system that allows modellers with limited programming experience to develop an agent-based model, because models are developed through a GUI (Repenning et al. 2000). A number of demonstration models are available from the system website. For example, Sustainopolis is a simulation analogous to the computer game SimCity; exploring pollution dispersal within a city (Fig. 12.5). Furthermore, AgentSheets can be linked to real time information over the internet (Repenning and Ioannidou 2004). For example, AgentSheets has been used in conjunction with real time weather feeds and used to make mountain biking recommendations in Boulder County. Within the model, agents represent locations that are possible candidates for biking featuring real time, web accessible weather sensors. This information is then used by the biker to reach a decision on where to go biking. Carvalho (2000) has used AgentSheets extensively to teach undergraduate students. He comments that it is easy to use the system to develop models quickly and provides students with hands-on experience of ABM without the need to learn a programming language. However, he also found that models created with AgentSheets were limited in their sophistication (notably in terms of the complexity of representation of agent behaviour and interaction). Furthermore, agents are limited to movement within a two-dimensional cell-based environment.

Fig. 12.5 The Sustainopolis model developed in AgentSheets (2010)

12.3.3.7 AnyLogic

AnyLogic (Table 12.5) incorporates a range of functionality for the development of agent-based models. For example, models can dynamically read and write data to spreadsheets or databases during a simulation run, as well as dynamically chart model output. Furthermore, external programmes can be initiated from within an AnyLogic model for dynamic communication of information, and vice versa. However, AnyLogic models can only be created on Microsoft operating systems, although a simulation can be run on any Java-enabled operating system once compiled (e.g. a Macintosh operating system). The AnyLogic website notes that models have been developed for a diverse range of applications including: the study of social, urban (Fig. 12.6) and ecosystem dynamics (e.g. a predator-prey system); planning of healthcare schemes (e.g. the impact of safe syringe usage on HIV diffusion); computer and telecommunication networks (e.g. the placement of cellular phone base stations); and the location of emergency services and call centres. Further information pertaining AnyLogic modelling applications can be found in Parinov (2007), these include imitating the functioning of a emergency department in a large hospital. However, the source code of these examples and/or documentation of these

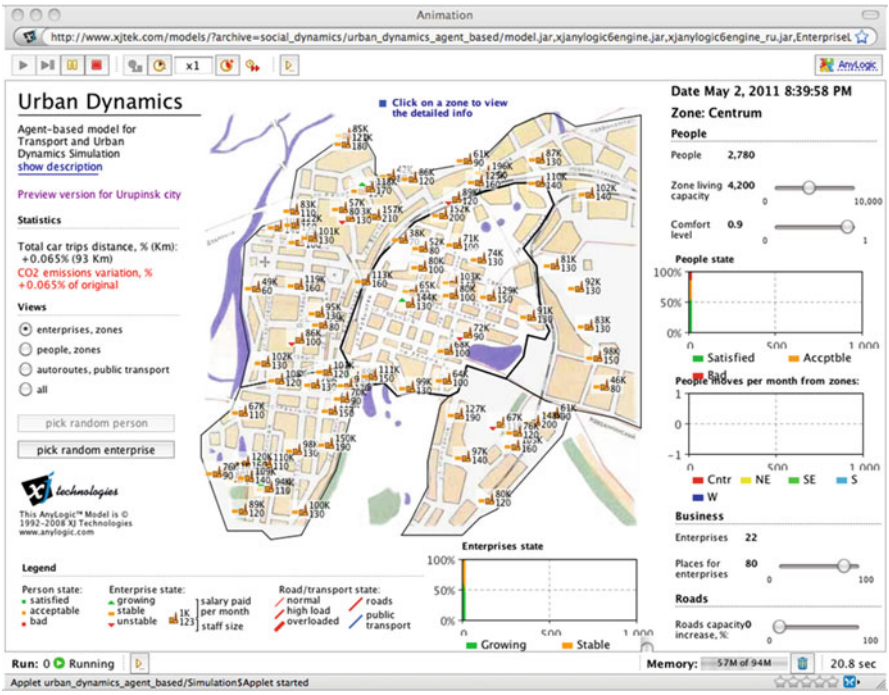

Fig. 12.6 An urban and transport dynamics model developed in AnyLogic (2010)

models is unavailable. Example applications utilizing AnyLogic for spatial agent-based modelling include: Makarov et al. (2008) who studied traffic jams in Moscow and explored different scenarios for reducing such events either by road pricing or new road building. Johnson and Sieber (2009) used AnyLogic to explore tourism in Nova Scotia, while Pint et al. (2010) used AnyLogic to explore organised crime in Rio's favelas.

12.4 Summary

This chapter has reviewed the current capabilities of modelling within a GIS and suggests that agent-based modellers interested in developing geospatial models involving many (possibly tens of thousands) interacting agents with complex behaviours and interactions between themselves, and their environment should consider either GIS-centric or modelling-centric integration. Moreover, we have discussed considerations one should take when thinking about utilizing an agent-based simulation/modelling system. Furthermore, we have outlined a selection of simulation/modelling systems which can be used for the creation of geospatial agent-based models along with providing examples of applications.

Each of simulation/modelling systems discussed within this chapter can be positioned within the continuum illustrated in Fig. 12.1 (power versus difficulty of developing a model with a simulation/modelling system). However, the exact location of each system is very subjective (i.e. dependant upon a modeller's knowledge and experience of ABM in general, and each simulation/modelling system in particular). The information presented within this chapter is aimed at providing the reader with a selection of useful criteria to assess the seven simulation/modelling systems presented, allowing each system to be (approximately) located within this continuum based on the readers own knowledge and experience. That is not to say that the selection criteria cannot be utilized for other simulation/modelling systems and once a candidate system(s) has been identified the reader will need to investigate the potential suitable of the system(s) further.

However, it needs to be noted that while such tools exist, integrating GIS data for ABM is still a difficult process (Gilbert 2007) and many considerations are needed such as what data is needed, how should the data be utilised, how should agents interact with the data, etc. Nevertheless, such systems lower the entry level needed to create geospatial agent-based models and thus allowing a greater number of social scientists to create geospatial agent-based models. One note of caution however is needed, that is there is still a computational challenge when it comes to the creation of geospatial agent-based models with thousands of agents operating and interacting with raster or vector features (see Kennedy et al. 2009 for a discussion) but over time this should be reduced with increased computational power.

References

Abdou, M., Hamill, L., & Gilbert, N. (2012). Designing and building an agent-based model. In A.J. Heppenstall, A.T. Crooks, L.M. See, & M. Batty (Eds.), *Agent-based models of geographical systems* (pp. 141–166). Dordrecht: Springer.

AgentLink. (2007). Software. Available at http://eprints.agentlink.org/view/type/software.html. Accessed on 12 July 2010.

AgentSheets. (2010). AgentSheets: The Sustainopolis model. Available at http://www.agentsheets.com/Applets/sustainopolis/. Accessed on 12 July 2010.

Andrade, P., Monteiro, A., & Câmara, G. (2008). Entities and relations for agent-based modelling of complex spatial systems. *BWSS 2008: 1st Brazilian Workshop on Social Simulation*, Bahia.

AnyLogic. (2010). AnyLogic: Urban dynamics agent based model. Available at http://www.xjtek.com/anylogic/demo_models/48/. Accessed on 12 July 2010.

Armstrong, D. J. (2006). The quarks of object-oriented development. *Communication of the ACM, 49*(2), 123–128.

Axtell, R., Epstein, J. M., Dean, J. S., Gumerman, G. J., Swedlund, A. C., Harburger, J., Chakravarty, S., Hammond, R., Parker, J., & Parker, M. (2002). Population growth and collapse in a multiagent model of the Kayenta Anasazi in Long House Valley. *Proceedings of the National Academy of Sciences of the United States of America (PNAS), 99*(3), 7275–7279.

Batty, M. (2005). *Cities and complexity: Understanding cities with cellular automata, agent-based models, and fractals*. Cambridge, MA: The MIT Press.

Batty, M., Conroy, R., Hillier, B., Jiang, B., Desyllas, J., Mottram, C., Penn, A., Smith, A., & Turner, A. (1998). *The virtual tate*, Working paper 5. London: Centre for Advanced Spatial Analysis (University College London).

Batty, M., Desyllas, J., & Duxbury, E. (2003). Safety in numbers? Modelling crowds and designing control for the Notting Hill carnival. *Urban Studies, 40*(8), 1573–1590.

Benenson, I., Aronovich, S., & Noam, S. (2005). Let's talk objects: Generic methodology for urban high-resolution simulation. *Computers, Environment and Urban Systems, 29*(4), 425–453.

Benenson, I., Birfur, S., & Kharbash, V. (2006). Geographic automata systems and the OBEUS software for their implementation. In J. Portugali (Ed.), *Complex artificial environments: Simulation, cognition and VR in the study and planning of cities* (pp. 137–153). Berlin: Springer.

Berger, T., & Parker, D. C. (2001). Examples of specific research: Introduction to standard descriptions of projects'. In D. C. Parker, T. Berger, & S. M. Manson (Eds.), *Meeting the Challenge of Complexity* (pp. 48–54). *Proceedings of a Special Workshop on Land-Use/Land-Cover Change*, Irvine.

Bernard, L., & Krüger, T. (2000). Integration of GIS and spatio-temporal simulation models: Interoperable components for different simulation strategies. *Transactions in GIS, 4*(3), 197–215.

Berryman, M. (2008). *Review of software platforms for agent based models*. Edinburgh: Defence Science and Technology Organisation, DSTO-GD-0532.

Booch, G. (1994). *Object-oriented analysis and design with applications*. Redwood City: Benjamin/Cummings.

Bousquet, F., Bakam, I., Proton, H., & Le Page, C. (1998). CORMAS: Common-pool resources and multi-agent systems. In A. P. Pobil, J. Mira, & A. Moonis (Eds.), *Lecture notes in artificial intelligence 1416* (pp. 826–838). Berlin: Springer.

Brown, D. G. (2006). Agent-based models. In H. Geist (Ed.), *The Earth's changing land: An encyclopaedia of land-use and land-cover change* (pp. 7–13). Westport: Greenwood Publishing Group.

Brown, D. G., Riolo, R., Robinson, D. T., North, M. J., & Rand, W. (2005). Spatial process and data models: Toward integration of agent-based models and GIS. *Journal of Geographical Systems, 7*(1), 25–47.

Carley, K. M. (1996). *Validating computational models*, Working Paper. Pittsburgh: Carnegie Mellon University.

Carvalho, J. (2000). Using agentSheets to teach simulation to undergraduate students. *Journal of Artificial Societies and Social Simulation, 3*(3). Available at http://jasss.soc.surrey.ac.uk/3/3/forum/2.html

Castle, C. J. E. (2007). *Agent-based modelling of pedestrian evacuation: A study of London's king's cross underground station.* Ph.D thesis, University College London, London.

Cioffi-Revilla, C. (2010). Computational social science. *Wiley Interdisciplinary Reviews: Computational Statistics, 2*(3), 259–271.

Cioffi-Revilla, C., Rogers, J. D., & Latek, M. (2010). The MASON householdWorlds of pastoral nomad societies. In K. Takadama, C. Cioffi-Revilla, & G. Deffaunt (Eds.), *Simulating interacting agents and social phenomena: The second world congress in social simulation*. Berlin: Springer.

Collier, N. T. (2010). *Repast HPC manual.* Available at http://repast.sourceforge.net/docs.html

Collier, N. T., & North, M. J. (2005). Repast for Python Scripting. *Annual Conference of the North American Association for Computational Social and Organizational Science (NAACSOS)*, Notre Dame. Available at http://www.casos.cs.cmu.edu/events/conferences/2005/2005_proceedings/Collier.pdf

Consequences Assessment Tool Set. (2011). Incident manager and first responder tools. Available at http://www.saic.com/products/security/cats/. Accessed on 12 Apr 2011.

Couclelis, H. (2001). Why I no longer work with agents: A challenge for ABMs of human-environment interactions. In D. C. Parker, T. Berger, & S. M. Manson (Eds.), *Meeting the challenge of complexity* (pp. 14–16). *Proceedings of a Special Workshop on Land-Use/Land-Cover Change*, Irvine.

Crooks, A. T. (2006). Exploring cities using agent-based models and GIS. In D. Sallach, C. M. Macal, & M. J. North (Eds.), *Proceedings of the Agent 2006 Conference on Social Agents:*

Results and Prospects, University of Chicago and Argonne National Laboratory, Chicago. Available at http://agent2007.anl.gov/2006procpdf/Agent_2006.pdf

Crooks, A. T. (2007). *The repast simulation/modelling system for geospatial simulation*, Working Paper 123. London: Centre for Advanced Spatial Analysis (University College London).

Crooks, A. T. (2010). Constructing and implementing an agent-based model of residential segregation through vector GIS. *International Journal of GIS, 24*(5), 661–675.

Crooks, A. T., Castle, C. J. E., & Batty, M. (2008). Key challenges in agent-based modelling for geo-spatial simulation. *Computer, Environment and Urban Systems, 32*(6), 417–430.

Crooks, A.T., & Heppenstall, A.J. (2012), Introduction to agent-based modeling. In A.J. Heppenstall, A.T. Crooks, L.M. See, & M. Batty (Eds.), *Agent-based models of geographical systems* (pp. 85–105). Dordrecht: Springer.

Dugdale, J. (2004). An evaluation of seven software simulation tools for use in the social sciences. Available at http://www.irit.fr/COSI/training/evaluationoftools/Evaluation-Of-Simulation-Tools.htm. Accessed on 12 July 2010.

Epstein, J. M. (2009). Modelling to contain pandemics. *Nature, 460*, 687.

Fedra, K. (1996). Distributed models and embedded GIS: Integration strategies and case studies. In M. F. Goodchild, L. T. Steyaert, & B. O. Parks (Eds.), *GIS and environmental modelling: Progress and research issues* (pp. 413–417). Fort Collins: GIS World Books.

Filatova, T., Parker, D., & van der Veen, A. (2009). Agent-based urban land markets: Agent's pricing behavior, land prices and urban land use change. *Journal of Artificial Societies and Social Simulation, 12*(1). Available at http://jasss.soc.surrey.ac.uk/12/1/3.html

Fontaine, C. M., & Rounsevell, M. (2009). An agent-based approach to model future residential pressure on a regional landscape. *Landscape Ecology, 24*(9), 1237–1254.

GeoMASON. (2010). GIS integration for MASON. Available at https://mason.dev.java.net/source/browse/mason/contrib/geomason/. Accessed on 14 July 2010.

Gilbert, N. (2007). *Agent-based models*. London: Sage.

Gilbert, N., & Bankes, S. (2002). Platforms and methods for agent-based modelling. *Proceeding of the National Academy of Sciences of the USA, 99*(3), 7197–7198.

Gilbert, N., & Troitzsch, K. G. (2005). *Simulation for the social scientist*. Milton Keynes: Open University Press.

Goodchild, M. F. (2005). GIS, spatial analysis, and modelling overview. In D. J. Maguire, M. Batty, & M. F. Goodchild (Eds.), *GIS, spatial analysis and modelling* (pp. 1–18). Redlands: ESRI Press.

Graham, S., & Steiner, J. (2006). Travellersim: Growing settlement structures and territories with agent-based modelling. In J. T. Clark & E. M. Hagemeister (Eds.), Digital discovery: Exploring new frontiers in human heritage. *Proceedings of the 34th Conference on Computer Applications and Quantitative Methods in Archaeology*, Fargo.

Grimm, V., & Railsback, S. F. (2012). Designing, formulating and communicating agent-based models. In A. J. Heppenstall, A. T. Crooks, L. M. See, & M. Batty (Eds.) *Agent-based models of geographical systems* (pp. 361–377). Dordrecht: Springer.

Hailegiorgis, A. B. (2010). Changing residence in the city: An agent based model of residential mobility in Arlington County. *The Association of American Geographers (AAG) Annual Meeting*, Washington, DC, 14–18 Apr 2010.

Haklay, M., O'Sullivan, D., Thurstain-Goodwin, M., & Schelhorn, T. (2001). "So go downtown": Simulating pedestrian movement in town centres. *Environment and Planning B, 28*(3), 343–359.

Harland, K., & Heppenstall, A. J. (2012). Using agent-based models for education planning: Is the UK education system agent-based. In A. J. Heppenstall, A. T. Crooks, L. M. See, & M. Batty (Eds.), *Agent-based models of geographical systems* (pp. 481–497). Dordrecht: Springer.

Hazard Prediction and Assessment Capability. (2004). Available at http://computing.ornl.gov/cse_home/hpac.shtml. Accessed on 12 Apr 2011.

Heywood, I., Cornelius, S., & Carver, S. (2006). *An introduction to geographical information systems* (3rd ed.). Harlow: Pearson Education.

Howe, T. R., Collier, N. T., North, M. J., Parker, M. T., & Vos, J. R. (2006). Containing agents: Contexts, projections, and agents. In D. Sallach, C. M. Macal, & M. J. North (Eds.), *Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects*, University of Chicago and Argonne National Laboratory, Chicago. Available at http://agent2007.anl.gov/2006procpdf/Agent_2006.pdf

Jackson, J., Forest, B., & Sengupta, R. (2008). Agent-based simulation of urban residential dynamics and land rent change in a gentrifying area of Boston. *Transactions in GIS, 12*(4), 475–491.

Janssen, M. A. (2009). Understanding artificial Anasazi. *Journal of Artificial Societies and Social Simulation, 12*(4). Available at http://jasss.soc.surrey.ac.uk/12/4/13.html

Johnasson, A., & Kretz, T. (2012). Applied pedestrian modeling. In A. J. Heppenstall, A. T. Crooks, L. M. See, & M. Batty (Eds.), *Agent-based models of geographical systems* (pp. 451–462). Dordrecht: Springer.

Johnson, P. A., & Sieber, R. E. (2009). Agent-based modeling: A dynamic scenario planning approach to tourism PSS. In S. Geertman & J. Stillwell (Eds.), *Planning support systems: Best practices and new methods* (pp. 211–226). Berlin: Springer.

Kennedy, R. C., Lane, K. E., Fuentes, A., Hollocher, H., & Madey, G. (2009). Spatially aware agents: An effective and efficient use of GIS data within an agent-based model. *Agent-Directed Simulation Symposium (ADS'09)*, San Diego, 22–27 Mar 2009.

Kennedy, W. B., Hailegiorgis, A. B., Rouleau, M., Bassett, J. K., Coletti, M., Balan, G. C., & Gulden, T. (2010). An agent-based model of conflict in East Africa and the effect of watering holes. *Behavior Representation in Modeling and Simulation (BRiMS) Conference*, Charleston.

Langran, G. (1992). *Time in geographic information systems*. London: Taylor & Francis.

Łatek, M. M., Rizi, S. M. M., & Geller, A. (2010). Persistence in the political economy of conflict: The case of the Afghan Drug Industry. *Complex Adaptive Systems: Resilience, Robustness, and Evolvability Workshop, Fall Symposium of Association for the Advancement of Artificial Intelligence*, 11–13 Nov 2010, Arlington. Available at http://www.css.gmu.edu/projects/irregularwarfare/

Liebert, K. D., Earnest, D. C., & Tolk, A. (2008). Using GIS vector data to build virtual environments for agent based models. *Proceedings of the 2008 Spring simulation multiconference* (pp. 45–52), Ottawa.

Linthicum, D. S. (2000). *Enterprise application integration*. Boston: Addison-Wesley.

Longley, P. A., Goodchild, M. F., Maguire, D. J., & Rhind, D. W. (2005). *Geographical information systems and science* (2nd ed.). New York: Wiley.

Luke, S., Cioffi-Revilla, C., Panait, L., & Sullivan, K. (2004). MASON: A new multi-agent simulation toolkit. *SwarmFest 2004, Eighth Annual Swarm Users/Researchers Conference*, University of Michigan, Ann Arbor. Available at http://www.cscs.umich.edu/swarmfest04/Program/PapersSlides/seanLuke-SwarmFest04-040507-2100.pdf

Luke, S., Cioffi-Revilla, C., Panait, L., Sullivan, K., & Balan, G. (2005). MASON: A multi-agent simulation environment. *Simulation, 81*(7), 517–527.

Maguire, D. J. (1995). Implementing spatial analysis and GIS applications for business and service planning. In P. A. Longley & G. Clarke (Eds.), *GIS for business and service planning* (pp. 171–191). Cambridge: GeoInformation International.

Maguire, D. J. (2005). Towards a GIS platform for spatial analysis and modelling. In D. J. Maguire, M. Batty, & M. F. Goodchild (Eds.), *GIS, spatial analysis and modelling*. Redlands: ESRI Press.

Makarov, V. L., Bakhtizin, A. R., & Zhitkov, V. A. (2008). Agent-based model for traffic jams in Moscow. *ABS2 Agent Based Spatial Simulation International Workshop*, 24–25 Nov 2008, Paris. Available at http://s4.csregistry.org/ABS2

Malleson, N. (2008). RepastCity – A demo virtual city. Available at http://portal.ncess.ac.uk/access/wiki/site/mass/repastcity.html. Accessed on 21 Jan 2009.

Malleson, N., Heppenstall, A., & See, L. (2010). Crime reduction through simulation: An agent-based model of burglary. *Computers, Environment and Urban Systems, 34*(3), 236–250.

MASON. (2010). Multi agent simulation of neighbourhood. Available at http://cs.gmu.edu/~eclab/projects/mason/. Accessed on 12 July 2010.

Mathevet, R., Bousquet, F., Le Page, C., & Antona, M. (2003). Agent-based simulations of interactions between duck population, farming decisions and leasing of hunting rights in the Camargue (Southern France). *Ecological Modelling, 165*(2–3), 107–126.

Minar, N., Burkhart, R., Langton, C. and Askenazi, M. (1996). The swarm simulation system: A toolkit for building multi-agent simulations. Available at http://www.santafe.edu/media/workingpapers/96-06-042.pdf. Accessed on 12 July 2010.

Mooij, W. M., Bennetts, R. E., Kitchens, W. M., & DeAngelis, D. L. (2002). Exploring the effect of drought extent and interval on the Florida snail kite: Interplay between spatial and temporal scales. *Ecological Modelling, 149*(1–2), 25–39.

Najlis, R., Janssen, M. A., & Parker, D. C. (2001). Software tools and communication issues. In D. C. Parker, T. Berger, & S. M. Manson (Eds.), *Meeting the Challenge of Complexity: Proceedings of a Special Workshop on Land-Use/Land-Cover Change* (pp. 17–30), Irvine. Available at http://www.csiss.org/resources/maslucc/ABM-LUCC.pdf

NASA. (2010). World wind. Available at http://worldwind.arc.nasa.gov/java/. Accessed on 15 July 2010.

NatureServe Vista. (2011). Decision support for better planning, Available at http://www.natureserve.org/prodServices/vista/overview.jsp. Accessed on 12 Apr 2011.

NetLogo. (2010). NetLogo: Multi-agent programmable modelling environment. Available at http://ccl.northwestern.edu/netlogo/. Accessed on 12 July 2010.

Nikolai, C., & Madey, G. (2009). Tools of the trade: A survey of various agent based modeling platforms. *Journal of Artificial Societies and Social Simulation, 12*(2). Available at http://jasss.soc.surrey.ac.uk/12/2/2.html

North, M. J., & Macal, C. M. (2009). Foundations of and recent advances in artificial life modeling with Repast 3 and Repast Simphony. In A. Adamatzky & M. Komosinski (Eds.), *Artificial life models in software* (2nd ed., pp. 37–60). New York: Springer.

North, M. J., Howe, T. R., Collier, N. T., & Vos, J. R. (2005a). The Repast Simphony development environment. In C. M. Macal & M. J. North (Eds.), *Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms*, Chicago. Available at http://agent2007.anl.gov/Agent_2005.pdf

North, M. J., Howe, T. R., Collier, N. T., & Vos, J. R. (2005b). The Repast Simphony runtime system. In C. M. Macal, M. J. North, & D. Sallach (Eds.), *Proceedings of the Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms*, Chicago. Available at http://agent2007.anl.gov/Agent_2005.pdf

OpenStarLogo. (2010). OpenStarLogo homepage. Available at http://education.mit.edu/openstarlogo/. Accessed on 12 July 2010.

Ozik, J. (2010). *RELOGO getting started guide.* Available at http://repast.sourceforge.net/docs/ReLogoGettingStarted.pdf

Parinov, S. I. (2007). New possibilities for simulations of socio-economic systems. *Artificial Societies, 2*(3–4): 23–53. Available at http://www.artsoc.ru/html_e/journal.htm

Parker, D. C. (2001). Object-orientated packages for agent-based modelling. Available at http://mason.gmu.edu/~dparker3/spat_abm/lectures/lecture2_tables.pdf Accessed on 12 June 2010.

Parker, D. C., Berger, T., & Manson, S. M. (2001). *Proceedings of an International Workshop on Agent-Based Models of Land-Use and Land-Cover Change,* Irvine. Available at http://www.csiss.org/maslucc/ABM-LUCC.htm

Peuquet, D. J. (2005). Time in GIS and geographical databases. In P. A. Longley, M. F. Goodchild, D. J. Maguire, & D. W. Rhind (Eds.), *Geographical information systems: Principles, techniques, management and applications (Abridged Edition)* (pp. 91–103). Hoboken: Wiley.

Peuquet, D. J., & Duan, N. (1995). An event-based spatio-temporal data model (ESTDM) for temporal analysis of geographic data. *International Journal of Geographical Information Systems, 9*(1), 2–24.

Pint, B., Crooks, A. T., & Geller, A. (2010). An agent-based model of organized crime: Favelas and the drug trade. *2nd Brazilian Workshop on Social Simulation*, Sao Bernardo do Campo.

Railsback, S. F., & Harvey, B. C. (2002). Analysis of habitat selection rules using an individual-based model. *Ecology, 83*(7), 1817–1830.

Railsback, S. F., Lytinen, S. L., & Jackson, S. K. (2006). Agent-based simulation platforms: Review and development recommendations. *Simulation, 82*(9), 609–623.

Redlands Institute. (2010). What is agent analyst? Available at http://www.institute.redlands.edu/agentanalyst. Accessed on 12 July 2010.

Reitsma, F., & Albrecht, J. (2006). A process-oriented data model. In J. Drummond, R. Billen, D. Forrest, & E. João (Eds.), *Dynamic and mobile GIS: Investigating change in space and time* (pp. 77–87). London: Taylor & Francis.

Repast. (2011). Recursive porous agent simulation toolkit. Available at http://repast.sourceforge.net/. Accessed on 7 Mar 2011.

Repenning, A., & Ioannidou, A. (2004). Agent-based end-user development. *Communications of the Association of Computing Machinery, 47*(9), 43–46.

Repenning, A., Ioannidou, A., & Zola, J. (2000). AgentSheets: End-user programmable simulations. *Journal of Artificial Societies and Social Simulation, 3*(3). Available at http://jasss.soc.surrey.ac.uk/3/3/forum/1.html.

Rixon, A., Moglia, M., & Burn, S. (2005). Bottom-up approaches to building agent-based models: Discussing the need for a platform. *Proceedings of the Joint Conference on Multi-Agent Modelling for Environmental Management*, Bourg-St-Maurice.

Robertson, D. A. (2005). Agent-based modeling toolkits. *Academy of Management Learning and Education, 4*(4), 525–527.

Schelhorn, T., O'Sullivan, D., Haklay, M., & Thurstain-Goodwin, M. (1999). *STREETS: An agent-based pedestrian model*, Working paper. London: Centre for Advanced Spatial Analysis (University College London).

Serenko, A., & Detlor, B. (2002). *Agent toolkits: A general overview of the market and an assessment of instructor satisfaction with utilizing toolkits in the classroom,* Working Paper 455. Hamilton: McMaster University (School of Business). Available at http://merc.mcmaster.ca/wpapers/wpaper455.html

Sondheim, M., Gardels, K., & Buehler, K. (2005). GIS interoperability. In P. A. Longley, M. F. Goodchild, D. J. Maguire, & D. W. Rhind (Eds.), *Geographical information systems: Principles, techniques, management and applications (Abridged Edition)* (pp. 347–358). Hoboken: Wiley.

Swarm. (2010). Swarm: A platform for agent-based models. Available at http://www.swarm.org/. Accessed on 12 July 2010.

SwarmWiki. (2010). Tools for Agent-Based Modelling. Available at http://www.swarm.org/wiki/Tools_for_Agent-Based_Modelling. Accessed on 12 July 2010.

Tatara, E., North, M. J., Howe, T. R., Collier, N. T., & Vos, J. R. (2006). An introduction to repast simphony modelling using a simple predator-prey example. In D. Sallach, C. M. Macal, & M. J. North (Eds.), *Proceedings of the Agent 2006 Conference on Social Agents: Results and Prospects*, University of Chicago and Argonne National Laboratory, Chicago. Available at http://agent2007.anl.gov/2006procpdf/Agent_2006.pdf

Tesfatsion, L. (2010). General software and toolkits. Available at http://www.econ.iastate.edu/tesfatsi/acecode.htm. Accessed on 12 July 2010.

Tesfatsion, L., & Judd, K. L. (Eds.). (2006). *Handbook of computational economics: Agent-based computational economics* (Vol. 2). Amsterdam: North-Holland.

Tobias, R., & Hofmann, C. (2004). Evaluation of free java-libraries for social-scientific agent based simulation. *Journal of Artificial Societies and Social Simulation, 7*(1). Available at http://jasss.soc.surrey.ac.uk/7/1/6.html

Torrens, P. M. (2009). *Process models and next-generation geographic information technology* (GIS best practices: Essays on geography and GIS, Vol. 2, pp. 63–75). Redlands: ESRI Press.

Torrens, P. M., & Nara, A. (2007). Modelling gentrification dynamics: A hybrid approach. *Computers, Environment and Urban Systems, 31*(3), 337–361.

Ungerer, M. J., & Goodchild, M. F. (2002). Integrating spatial data analysis and GIS: A new implementation using component object model (COM). *International Journal of Geographic Information Science, 16*(1), 41–53.

Vos, J. R. (2005). Repast .NET: The Repast framework implemented in the .NET. *Annual Conference of the North American Association for Computational Social and Organizational Science (NAACSOS)*, Notre Dame. Available at http://www.casos.cs.cmu.edu/events/conferences/2005/2005_proceedings/Vos.pdf

Wegener, M. (2000). Spatial models and GIS. In A. S. Fotheringham & M. Wegener (Eds.), *Spatial models and GIS: New potential and new models* (pp. 3–20). London: Taylor & Francis.

Westervelt, J. D. (2002). Geographic information systems and agent-based modelling. In H. R. Gimblett (Ed.), *Integrating geographic information systems and agent-based modelling techniques for simulating social and ecological processes* (pp. 83–104). Oxford: Oxford University Press.

Wikipedia. (2010). Comparison of agent-based modeling software. Available at http://en.wikipedia.org/wiki/Comparison_of_agent-based_modeling_software. Accessed on 12 July 2010.

Wilensky, U. (2006). NetLogo grand canyon model. Available at http://ccl.northwestern.edu/netlogo/models/GrandCanyon. Accessed on 15 July 2010.

Zeiler, M. (1999). *Modelling our world: The ESRI guide to geodatabase design*. Redlands: ESRI Press.