Jul 1st, 12:00 AM

An Architecture for Modelling Individual Behaviour and Landscape Scale Outcomes in an Intelligent Agent-Based Simulation of Environmental Management

L. J. Smith

R. Itami

I. D. Bishop

An Architecture for Modelling Individual Behaviour and Landscape Scale Outcomes in an Intelligent Agent-Based Simulation of Environmental Management

L.J. Smith[a], **R Itami**[b] **and I.D.Bishop**[a]

[a] *Department of Geomatics, The University of Melbourne, Parkville VIC 3010 Australia (lsmith@vicnet.net.au)*

[b] *GeoDimensions Pty Ltd, 22 Dunstan Avenue Brunswick VIC 3056 Australia*

Abstract: This work describes a software architecture that couples intelligent agent technology with Geographic Information System (GIS). The agents simulate individual behaviour, while GIS-based simulation models represent the landscape that is observed and acted upon by the agents. Our motivation in developing the software is to assist in understanding and resolving complex, semi-structured, environmental management problems, where multiple actors participate in decision-making and management. Like real stakeholders, agents are individuals and have different knowledge and interests. The agents' design is based upon the Belief, Desire, and Intention (BDI) theory of intelligent agents and implemented in Visual Basic. Agents are initialised with a set of goals they want to achieve and a library of plans that stores knowledge about landscape management actions. Each plan is like a 'if-then' rule where the 'if' is a set of conditions that need to be matched to agents beliefs for execution and the 'then' part, which describes an outcome when the plan is executed. Agents act by looking for plans that match their goals and their current beliefs. At each time step, for each outstanding goal they examine their plan library and choose the best plan, if any, which satisfies the goal. Plan choice is influenced by the agent's current belief state and values. Plans can be linked together in a hierarchy providing flexible and responsive reasoning to the agents. Landscape scale outcomes of the collective agent's actions are simulated using GIS-based simulation models. Agents can have goals and plans that relate to the system wide state simulated by the GIS models and can observe changes in system state. By comparing these changes with existing conditions agents can determine if catchment resources are degrading or improving. If there is a degradation in one or more landscape resources the agent will act to revise its management actions in an attempt to improve its performance and if the goal is a system wide resource the performance of the catchment as a whole. This process is iterated until no improvement in system performance is detected. We demonstrate the potential of this decision-making framework on an abstract landscape.

Keywords: Intelligent Agents, Simulation, Environmental Management, GIS

1. INTRODUCTION

Today more and better environmental information is available, communities are more informed and participatory and pressing environmental problems have become more obvious and urgent. As a result environmental planning decisions are increasingly complex and difficult to resolve.

Modelling and simulation are means to support complex environmental decision-making. They have long been used to provide evaluative information for proposed human actions that change the landscape. A simulation environment offers the ability to generate, evaluate and compare alternative hypothesis before implementation and hence can provide action or outcome-focused results.

With most environmental planning decisions multiple stakeholders are involved. These stakeholders usually have widely divergent interests, knowledge and values. Incorporating these values in alternative proposals and their evaluation is clearly needed for decision-making to be seen as accountable and participatory. There is a pressing need for environmental simulation tools that can illuminate the social and political nature of environmental planning and decision-making. Crucial to many environmental planning disciplines is an ability to incorporate and consider a wide range of stakeholder knowledge, values and interests explicitly as part of the planning process.

Intelligent agents offer a means to represent stakeholders explicitly in a simulated environment. Similar to real environmental planning situations it becomes possible to simulate the complex interactions between stakeholders and their environments. Just like real stakeholders these simulated agents can possess varying values, spatial domains, knowledge and interests. Together they can act to generate alternative environmental management scenarios.

2. AGENT REQUIREMENTS FOR ENVIRONMENTAL SIMULATION

In order to be useful for complex environmental management applications our agents need five attributes: 1) Deliberative reasoning combined with responsiveness to changes in system state 2) An ability to use complex spatial and domain specific knowledge that in many cases is held by a few stakeholders; 3) An ability to manipulate a GIS and observe the outputs of implemented environmental simulation models. 4) A capacity to be different to other agents in terms of the values, knowledge and behaviours implemented in the simulation. 5) An ability to be autonomous in terms of observing and acting in their environment.

2.1 Deliberative Reasoning and Responsiveness

Complex environmental management problems require deliberation and reasoning to determine the best course of action for a given situation. In a simulation setting, agent deliberation will take time. At the same time, the simulated environment in which the agents exist is dynamic and will change as other agents act and as the coupled simulation models execute. Agents need to remain responsive to these constant system changes for decisions and behaviour to remain relevant. They need to drop behaviours that conflict with new changes in the system and find new actions to deal with arising circumstances. They must also check to ensure their desired goals / system states are being met and modify behaviour appropriately.

Deliberation and responsiveness then must balanced for agents to act appropriately in dynamic, unpredictable systems. Without this balance agents will either deliberate for so long that their actions are no longer relevant to the current system state or simulation performance is unacceptably slow. Otherwise, if the agent is too reactive, an agent may implement actions without necessarily finding a good or the best action to execute.

2.2 Domain and Location Specific Knowledge

Environmental management problems tend to be domain and location specific. Often the general information that is available is not suitably detailed to be useful for specific real-world environmental management problems. As such many environmental management problems rely on the expertise of domain specific experts or local stakeholders. Our agents need to be able integrate and leverage the general, expert and local knowledge that is available to provide useful results.

2.3 GIS and Environmental Model Integration

Our agents will be situated within a simulated environment. They need to be able manipulate and observe this environment. The environment will be simulated using standard Geographic Information Software (ESRI ArcGIS 8.1) and some off-the-shelf environmental simulation models. These simulation models will need to use the GIS and a record of the agents actions as inputs for updating the GIS and generating simulation outputs. Our agents then need to be able to observe these results and incorporate them into their reasoning. ArcGIS 8.1 now provides a COM programmable interface called ArcObjects as the standard customisation interface. Agents programmed in a COM compliant programming language will be able to manipulate and observe the GIS and other simulation outputs stored as database tables.

2.4 Variation in Agent Values, Knowledge and Behaviour

Environmental management problems involve many stakeholder views, interests and behaviours . Accurate simulation requires that our agents be variable in terms of the knowledge, values and actions they implement. The reasoning behind this behaviour needs to defensible and communicable as outputs to the end users of the simulation.

2.5 Encapsulation and Autonomy

The reasoning and behaviours of the agent need to be encapsulated within the agent as much as possible to reduce complexity of integration. Agents also need to be autonomous in terms of pursuing their own goals and implementing actions. They need to autonomously observe relevant system changes and change their behaviours based upon these observations.

3. RELATED WORK

Agent-based simulations have been applied to several applications where humans interact with a natural system, for example recreation behaviour (Itami et.al. 1997), ancient societies (Dean et.al. 1999). Itami's et.al. (1997) RBSim II agents have many of the same requirements as our agents. RBSim II agents plan and move along complex routes. In order to handle the complexity of moving along these networks the agents use deliberate reasoning to find the best path and reactive reasoning to respond to local events and conditions. Similar to our requirements the RBSIM II agents

alternate between reactive behaviour and deliberative reasoning. Limiting the amount of forward reasoning required finding the next best step or move reduces complexity.

Our requirements differ somewhat however from RBSim II. The complexity of simulation, where the environment is unpredictable and highly dynamic and the agents can implement many different behaviours requires a different approach to agent reasoning and action. It would be unfeasible to plan all possible alternatives at the beginning of the simulation and have these alternatives remain relevant to the changing system. Our agents need a more fine-grained approach to alternating deliberation and action.

Itami's (1997) and Dean's et.al. (1999) agents use rules to capture and model complex human behaviour. By following executing a set of rules when appropriate, the agents can display complex behaviours like seeking out food, planting crops, finding scenic viewpoints, using toilets or finding cover in weather events. These rule-based approaches demonstrate a feasible way to capture and model complex, domain specific knowledge of recreation, ancient and other human communities.

4. BDI AGENTS

The Belief Desire Intention (BDI) theory of agency provides a well-documented theoretical framework for implementing intelligent agents. The BDI framework was developed in the mid 1980's by Georgeff and Lansky (1986), and refined later by Ingrand et.al. (1992) It has since been implemented as a several functioning software platforms, current examples being Jack Intelligent Agents (Busetta et.al. 1999) and Jam Agents (Huber 1999). It provides a convenient terminology and structure for describing intelligent agents. Unlike many other agent systems the BDI framework has been applied to many practical applications, making the theory and terminology clearer and more generic than other systems.

BDI agents fulfil the requirements needed for our framework. Firstly the BDI framework provides a way to interleave deliberation with responsiveness and limit the amount of forward deliberation required to act rationally. BDI agents can partially search and expand planned actions allowing them to select good alternatives, while avoiding constant deliberation and its associated time penalty.

BDI agents have also been applied and are suited to highly dynamic and unpredictable situations. By only partially expanding alternative plans of actions they can remain responsive to changes in system state. The frameworks of Huber (1999) and Busetta et.al. (1999) also describe ways to recover from failed actions, to customise reasoning for specific situations, to handle conflicting actions and goals, and to modify already executing actions, all of which are needed in dynamic simulations like environmental management applications.

The BDI framework also provides a means to use complex domain-specific behaviour. There are numerous published examples (see for example Busetta 1999) that demonstrate complex domain specific behaviour being captured and modelled in intelligent agent applications.

The BDI agent has a view of the world – a library of *beliefs,* a set of goals or *desires* and a set of plans*,* or actions that the agent can *intend* to carry out. In a simulation run the agent is able to act autonomously. It pursues its goals by intending and executing plans that are appropriate according to its current belief set. By giving an agent a set of precompiled plans it becomes possible to imbue an agent with a number of roles. The plan library contained within the agent defines a set of behaviours appropriate to many different situations. These predefined plans allow the agent to react quickly and rationally to situations that arise during a simulation.

Each agent consists of five main components: a world model, a goal library, a plan library, an intention stack and an interpreter (Huber 1998). The world model is a database containing a collection of beliefs that represent the agent's current view of its world. The plan library contains a set of defined actions or procedures that achieve the agent's goals. The intention stack is a model of the agent's current goals and currently executing plans, tracking the progress and status of goal and plan execution. The interpreter performs agent reasoning by searching the plan library for plans relevant to the currently held goals.

4.1 Interpreter

The agent's internal interpreter carries out reasoning. The interpreter operates in a loop.

The reasoning loop switches between execution and deliberation. At each time-step the agent both considers the actions to undertake next and also executes any plans that have been previously added to the intention stack.

Reasoning happens in four main steps: Starting with the goal library, at any time there will be a series of pending, unachieved goals and a set of beliefs that describe the agent's state or world view. These facts and goals trigger plan searching and selection. The best plan found to be relevant to the current goal and current beliefs is intended for execution by adding it to the intention stack.

The execution thread then switches to the intention stack. Any plans existing in the intention stack are executed. Plan execution may result in new facts being added to the agent's belief database, primitive actions being carried out on the external

environment or new goals being added to the agent's goal stack. Execution within a plan continues until it reaches a defined end point.

Hierarchical Reasoning

Plans are hierarchical when they post sub goals during execution and wait while they are achieved. At the next time step, the interpreter will attempt to achieve the sub goal by finding and executing a suitable plan. At two time steps after starting, if the sub goal has been achieved, the first plan will continue execution. In this way large, branched hierarchies of goals and plans can be built. At higher levels the goals and plans are more abstract consisting of more sub goals than direct primitive actions. As execution continues through the hierarchy more specific actions and goals are executed.

The alternate deliberation / execution of the plan / goal hierarchy allows the interpreter to only partially expand the hierarchy. This limits the amount of reasoning time required before acting. The agent can remain responsive to changes in their beliefs and states, while still remaining committed to unachieved goals. The partial expansion of the hierarchy also allows the agent to recover from failures. If one plan fails to achieve a sub goal then the interpreter can go back and try other plans that were appropriate. If at any point changes in beliefs result in a plan failing or a goal no longer being relevant the goal can be cancelled or restarted.

4.2 Beliefs

The agent beliefs are held in database. Each belief consists of a statement and a Yes / No / Unknown value that represent whether the statement is true false or unknown. During execution beliefs can be added, removed or modified through plan execution.

4.3 Goals

The agent begins the simulation with a set of goals that it will pursue during the simulation. Goals have a number of states allowing the agent to track them. These are: *pending, successful, failed* or *in progress*. These states allow the interpreter to manage and determine the sequence of execution of intended plans.

4.4 Plans

Plans are the central component of the agent's reasoning system. A plan represents procedural knowledge - it defines a set of actions the agent will do, the circumstances under which to use the plan and the results of executing the plan. Plans are based on a standard Visual Basic class with the following properties and methods:

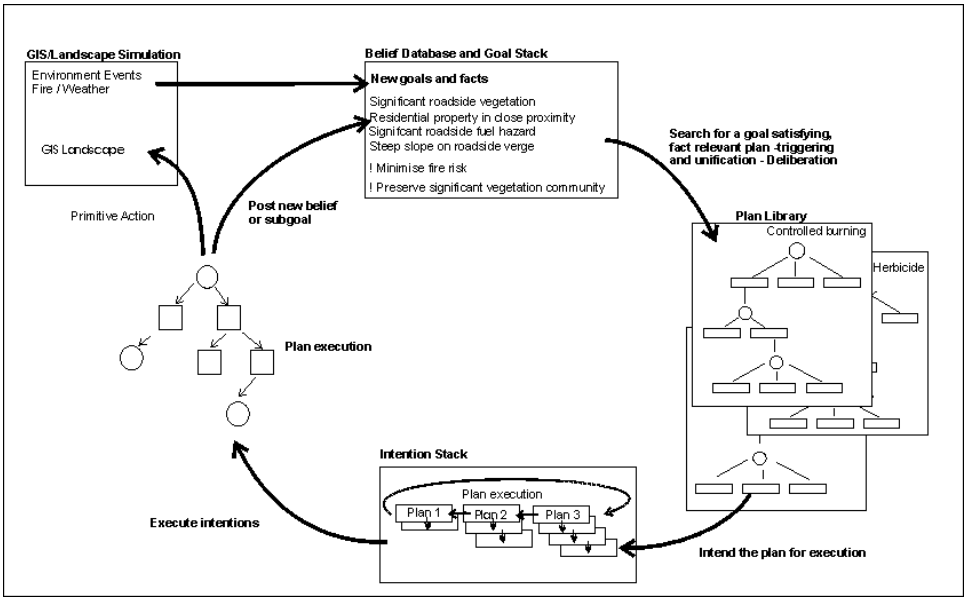

Figure 2 Interpreter Reasoning Loop

State Property

Plans have a number of states that allow the intention stack of the agent to track them. A plan can be idle, executing, failed or successful. At the end of each cycle of the reasoning loop, by querying the state of all plans on the intention structure the agent can add, remove, suspend and update the goals and plans currently executing.

Goal Handled Property:

Each plan handles a single goal. When an agent is trying to achieve a goal it searches the plan library by querying and comparing the goal handled property of each plan to the search goal.

Relevant Method:

After a plan is found that handles the search goal the agent queries the *relevant* method. A plan is determined to be relevant if the current beliefs of the agent are not in conflict with any of the conditions set out in the relevant method. Any plans that pass the relevant method are added to an Applicable Plan List (APL).

The APL is a collection of plans that have been found to be relevant to a goal. From this collection a single plan is then chosen. Plans will be either chosen at random or by calculating the utility of each plan using the context method.

Context Method:

A plan's context method is used when the utility of the plan is being used for selection. By comparing the utility of each of the plans in an applicable plan list the agent can select the plan and intend the best plan. Calculation of the utility, like the *relevant* method, is based upon the agent's current beliefs and values.

4.5 Plan Execution

The plan's execute method defines what is to occur once a plan reaches a state of implementation.

From this method any logic can be implemented. There are four main agent primitives that can be called from an execute method: *Post Goal, Add Belief, Remove Belief, Update Belief,* and *Implement Treatment.*

Post Goal

New goals can be posted to the goal stack from plans. This means plans can trigger further reasoning and behaviour. The plan can wait for the goal to be achieved before continuing supporting hierarchical linking and branching of multiple plans.

Add Belief, Remove Belief, Update Belief

As a plan executes it becomes necessary to add or remove beliefs to the agent's belief set. For example the agent may remember which plans it has tried, goals it has satisfied, the time a plan was completed or other changes as a result of carrying out actions.

Implement Treatment

Execution of the plan results in the agent implementing a change on its spatial domain.

5. AGENT BASED SIMULATION

Our agents exist within a simulated landscape. The simulated landscape contains data on vegetation, roads, property boundaries, buildings and slope and is displayed to the user through a GIS interface. Agents are situated at discrete locations in the landscape.

Figure 2 shows a simplified class diagram of the agent-modelling framework. At the top of the framework the Simulation Engine object initialises and runs the simulations. The simulation contains a collection of agents and a series of objects that represent the landscape described in the GIS. The simulation engine initialises the simulation using a simulation scenario.

Simulation scenarios define the parameters of the simulation. They consist of the GIS data layers, a collection of agents and their associated spatial domains, global environmental events like weather conditions or fire events and schedule of when they occur, the length and start date for the simulation as well as the outputs to record in the simulation.

5.1 The Simulation Engine

The simulation engine executes a simulation scenario. The engine reads the parameters of a simulation scenario from the database and initialises the necessary objects in the Visual Basic application. At the beginning of a simulation the GIS data is loaded into memory. The agent collection is initialised. The agents are allocated their plan library, their initial goals and beliefs and their spatial domains (GIS polygons) upon which they can carry out actions. Global events are scheduled. If output statistics are requested the output database is initialised. The graphics windows are initialised according to the parameters

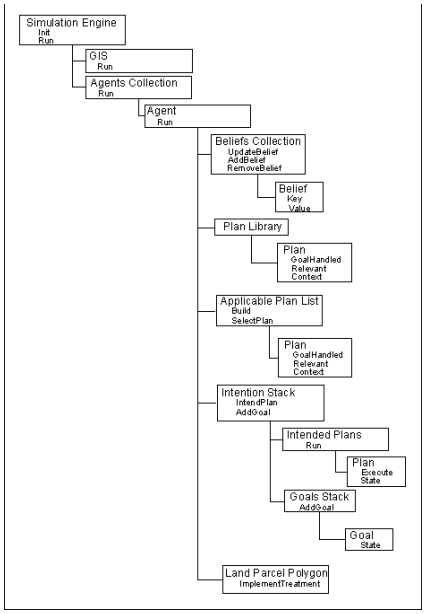

Figure 3 Simplified Class Hierarchy Showing Main Methods

requested by the user. For example the user may monitor changes in land cover or other management actions graphically with changes in display colours in the GIS.

The simulation is then ready to start. The simulation engine starts the simulation clock. At

each time step each agent in the collection is given a turn to deliberate and act. When the execution thread is given to an agent, the agent's interpreter searches for any relevant plans and executes any already intended plans. When the agent interpreter has finished a complete loop, execution returns to the simulation engine who then calls the next agent and so on. After calling all of the agents the simulation engine then runs any schedule events for the time step and any process models that are part of the simulation. Next any outputs defined in the simulations scenario are written to the output database. To avoid problems arising out of the synchronous execution of agents, the collection of agents is then reordered at random before the next loop of the simulation engine begins.

5.2 Agent Communication and Cooperation

At present the agents in our model do not communicate, cooperate or even rationally compete for resources in the simulation. This is an area for further work and there are many examples of BDI agents that do communicate and cooperate in a multi-agent environment.

6. PROOF OF CONCEPT

We intend to implement a pilot demonstration of the framework described here as the first step in implementing a fully functional simulation model based upon a real environmental management situation.

The demonstration will consist of three agents each having a single land polygon to manage. The agents will be initialised with four goals – two relating to their land parcel and two relating the system wide resources of all of the land parcels. Each agent will also be initialised with the same library of eight plans. Each plan will result in different outcomes upon the resources of the individual land and the system as a whole.

A simple simulation model will model three, abstract catchment resources. The plans implemented by the agents will affect the current state of each resource at any one time. Two of goals the agents will be pursuing will relate to the levels of these resources. Some plans will result in improvements in these system wide resources and some will degrade them.

We aim demonstrate that agents will modify their behaviours over time to improve system performance. We want the agents to respond to system changes brought about by other agents as evidenced in the outputs of a simple surrogate catchment model, drop any incompatible plans and find new plans to improve system performance. The results and a more thorough description of this prototype will be subject of future work.

7. CONCLUSION

Agent based modelling allows numerous simulations to be run to investigate alternative outcomes of combined environmental / social processes. We can systematically alter the quantitative parameters of a model or introduce completely new procedures, behaviours and events to the simulation. The approach allows for continual improvement, any knowledge, values, models and other information can be plugged into the simulation environment. While the agent system may never completely mimic real systems it can provide a means to rationally, cumulatively make progress on complex environmental management issues that are difficult to understand or manage otherwise.

There still remains considerable work in developing the framework described here. Agent communication, agent roles, agent cooperation and competition for resources, spatial dependencies all need further work. Model validation, transparency and the how the results will be used by decision makers are other issues requiring further exploration. These issues are the subjects of current research.

8. REFERENCES

Busetta P, Ronnquist R, Hodgson A, Lucus A. Jack Intelligent Agents- AOS Technical Report 1. December 1999 (see www.agent-software.com.au)

Dean, J., Gunnerman, G., Epstein, J., Axtell, R., Swedlund, A., Parker, M., McCarrol, S. Understanding Anasazi Culture Change Through Agent-Based Modelling. In Dynamics of Human and Primate Societies, Oxford University Press 1999.

Georgeff M,. Lansky, A. Procedural Knowledge Proceedings of the IEEE Vol 74 (10) pp 1383 – 1398 October 1986.

Gimblett R; Itami R. Modelling the Spatial Dynamics and Social Interaction of Human Recreationists' MODSIM 97 Tasmania. 1997.

Huber M. Jam: A BDI-theoretic Mobile Agent Architecture. Third International Conference on Autonomous Agents (Agents '99) pp236-243. Seattle, WA May 1999.

Ingrand F, Georgeff M, Rao S. An Architecture for Real-Time Reasoning and System Control. IEEE Expert 7(6) 33-44 December 1992.

Lucas A, Goss S. The Potential for Agents in Defence Simulation Information, Decision and Control (IDC '99) pp579-583. October 1999.