BREAKING NEW GROUND IN SIMULATING KNOWLEDGE MANAGEMENT PROCESSES: *SIMISPACE2*

Martin Ihrig

Alan S. Abrahams

The Sol C. Snider Entrepreneurial Research Center
The Wharton School of the University of Pennsylvania
Vance Hall 4th Floor, 3733 Spruce Street, Philadelphia, PA 19104
E-mail: ihrig@wharton.upenn.edu, asa28@wharton.upenn.edu

KEYWORDS

Agent-based simulation, strategic knowledge management, I-Space, knowledge flows and structures.

ABSTRACT

The purpose of this paper is to explain exactly how *SimISpace2* works, i.e. to illustrate the distinctive features and underlying processes of this unique simulation environment. There is no other program available to researchers that can model strategic knowledge management processes in such a complex manner. By describing the inner workings of the simulation, we demonstrate the benefit of the new simulation software for future researchers. We show that the simulation model is a viable research tool for understanding the role of knowledge structures, knowledge value and knowledge accumulation and development strategies in various settings.

1 INTRODUCTION

SimISpace2 is an agent-based graphical simulation environment designed to simulate knowledge management processes, in particular knowledge flows and knowledge-based agent interactions. Its conceptual foundation is provided by Boisot's (1995; 1998) work on the Information-Space (*I-Space*). So far, *SimISpace2* and its predecessor *SimISpace* (Boisot, Canals, & MacMillan, 2003) have been used to study intellectual property rights policies (Boisot, MacMillan, & Han, 2007) and to configure simulations that model the knowledge-transfer dilemma facing the counter-terrorism community (MacMillan, Boisot, Abrahams, & Bharathy, 2005). *SimISpace2*'s latest application is in the field of entrepreneurship, to explore different opportunity recognition strategies.

"The I-Space approach is markedly different from existing approaches for modeling the physical world." (MacMillan et al., 2005: 5) In contrast to other simulations, *SimISpace2* makes it possible to model the multiple knowledge-specific activities required of the knowledge lifecycle. One can study the effects of knowledge management actions and do macro level explorations of knowledge processes. By enabling the researcher to model and analyze knowledge assets and their life-cycle, *SimISpace2* is a useful tool in the knowledge modeling context.

The structure of the paper is as follows. First, we briefly summarize the main characteristics of *I-Space* Theory and outline what the user can expect from the *SimISpace2* environment. We also explain the underlying concept of the simulation, including how the properties of the basic entities – agents and knowledge assets – are determined. Second, we describe in detail how knowledge assets are represented in the simulation model and what properties they can have. We also discuss the importance of knowledge networks, the difference between owning and possessing knowledge, and the value of knowledge. Third, we depict the different properties agents can have and give a detailed description of the various actions agents are able to perform.

2 FUNDAMENTALS

2.1 *I-Space*: a conceptual framework

The Information Space, or *I-Space*, is a conceptual framework that relates the degree of structure of knowledge (i.e. its level of codification and abstraction) to its value and diffusibility as that knowledge develops. Tacit knowledge (low codification and abstraction) flows very slowly between agents and often only in face-to-face situations. Codified and abstract knowledge, by contrast, can diffuse rapidly throughout a population, whether such diffusion is desired or not. The development of knowledge from tacit to codified and abstract can add value to knowledge to the extent that it does not diffuse to other agents, and to the extent that it succeeds in shedding noisy data without simultaneously shedding usable information.

Knowledge flows give rise to a six-step learning process within an agent population – the Social Learning Cycle or *SLC* – in which new knowledge is created, shared, internalised and applied. Different *SLC* configurations in the *I-Space* reveal the learning strengths and weaknesses of different agent groups. Some are given to hoarding their knowledge; others to sharing it.

Proceedings 21st European Conference on Modelling and Simulation
Ivan Zelinka, Zuzana Oplatková, Alessandra Orsoni ©ECMS 2007
ISBN 978-0-9553018-2-7 / ISBN 978-0-9553018-3-4 (CD)

The *I-Space* allows us to represent an agent's knowledge as a portfolio of knowledge assets that develop over time either through acts of individual or collective learning, or through transactions with other agents. These interactions can be studied under different, preset environmental contexts.

2.2 A brief description of the innovative simulation environment

SimISpace2 is an agent-based simulation model that is being developed to implement the main features of the I-Space (Boisot, 1995, 1998) as a conceptual framework. It is designed to serve as a general knowledge management engine that, through a user-friendly graphical interface, can be adapted to a wide range of knowledge-related applications. We start by creating domain-specific knowledge networks that agents are required to discover and exploit by investing either in individual learning processes – codification, abstraction, etc. – or in interactions with other agents. *SimISpace2* allows a user to specify different knowledge groups, agent groups, discovery conditions, knowledge spillover conditions, cost conditions and reward conditions. Complex scenarios can be modeled by assigning distributions for each of the properties of agent and knowledge groups.

During the simulation runs, agents compete and cooperate in performing three different kinds of actions:

- Managing knowledge (e. g. discovering, codifying & abstracting, protecting, exploiting, disposing, etc.)
- Exchanging knowledge (e.g. trading, licensing, alliancing, joint-venturing)
- Networking and being in motion (e.g. calling for meetings, encountering, relocating, relaxing, exiting & entering)

The above actions and interactions form the basis for the emergent properties of agents (e.g. stock of knowledge, financial and experience funds, location, etc.) and of knowledge assets (e.g. diffusion, location, structure, obsolescence, etc).

SimISpace2 has the following innovative features that make it different from other simulation programs:

a) It models information flows by implementing a theoretical framework that is novel, powerful, and growing in popularity.
b) It is the only agent-based simulation of knowledge flows that is based on a comprehensive theory of agent-based knowledge evolution (the I-Space theory).
c) It can track and analyse multiple collectivities' social learning processes, their discovery strategies, and the agent interaction strategies.
d) By analysing the complex networks of agent interactions, it can track the emergence of institutional structures and gain an understanding of the reciprocal influence of process and structure in the knowledge domain.

2.3 Getting started: specifying different knowledge and agent groups

Two major forms of entities can be modeled with *SimISpace2*: agents and knowledge items/assets. When setting up the simulation, the user defines agent groups and knowledge groups with distinct properties. Individually definable distributions can be assigned to each property of each group (uniform, normal, triangular, exponential, or constant distribution). When the simulation runs, the individual group members (agents and knowledge items) are assigned characteristics in accordance with the distribution specified for the corresponding property for the group of which they are a member.

Agents and knowledge items cannot join or leave groups. They start off being a member of a group from the start of the simulation and remain a member until the end of the simulation. Even if they die, they remain associated with their group. The static nature of group membership here has been implemented to facilitate easier analysis. If groups had dynamic membership, a consistent analysis of the properties of their members would be harder since the actual members would change from period to period. To prevent overlapping distributions, a participant – an agent or a knowledge item – is only ever a member of a single group. Each group has its own distributions for each property, and if an agent (or knowledge item) were a member of more than one group then conflicts would arise as to which distribution to base their property values from.

As a result, group size is static. The number of members is entirely determined by the start number specified for the group by the user. During the start-of-simulation processing (bootstrapping) the simulation iterates through all the agents and knowledge items and uses random-number generation to assign each agent's and knowledge item's characteristics according to the distribution specified for each property. That way, the agent and knowledge groups contain agents and knowledge items with approximately the distribution specified for each property.

3 KNOWLEDGE ASSETS

3.1 Terminology

Knowledge in the simulation environment is defined as a 'global proposition'. The basic entities are *knowledge items*. Based on the *knowledge group* they belong to, those knowledge items have certain characteristics. All knowledge items together make up the *knowledge ocean* – a global pool of knowledge. Agents access the knowledge ocean, pick up knowledge items, and deposit them in *knowledge stores*. A knowledge store is an agent's personal storage place for a knowledge item. Each knowledge store is local to an agent, i.e. possessed by a single agent. Knowledge stores as

containers *hold* knowledge items as their contents. This happens after agents obtain a knowledge item. Examples of a knowledge store include books, files, tools, diskettes, and sections of an agent's brain. There is only one knowledge item per knowledge store, i.e. each knowledge item that an agent possesses has its own knowledge store. If an agent gets a new knowledge item (whether directly from the knowledge ocean or from other agents' knowledge stores), a new knowledge store for that item is generated to hold it.

3.2 Knowledge item vs. knowledge store

The concept of a knowledge item has been separated from the concept of a knowledge store so that there is some traceability of knowledge. If knowledge items are taken from the same common pool and stored in different agent's knowledge stores, then it is possible to see when two (or more) agents have the same knowledge. This is very useful for seeing how knowledge diffuses. Contrast this to the case where agents are allowed to arbitrarily create their own knowledge. This would make it very difficult to determine the provenance of knowledge, as it would be very hard to determine when two pieces of knowledge held by different agents are in fact the same knowledge.

The importance of the separation between a global pool of knowledge and local knowledge stores can also be illustrated by looking at abstraction and codification (which applies only to knowledge stores and not to knowledge items). Knowledge stores are held by an agent and held at a particular level of codification and abstraction. This means that if the agent codifies its knowledge and makes it more abstract, the properties of the knowledge item are not changed but the corresponding knowledge store gets a higher degree of codification and abstraction. So the knowledge item is held at a certain level of abstraction and codification in that knowledge store. Knowledge stores are about the *form*, knowledge items about the *content* of knowledge.

When it comes to codification and abstraction, the simulation uses a so-called *string-of-pearls* mechanism. Each time an agent codifies or abstracts knowledge (the knowledge store that holds a knowledge item), a new, i.e. an additional, knowledge store is created with the new level of codification or abstraction. The same applies to the opposite actions of codification and abstraction: absorption and impacting. The knowledge item in those knowledge stores is always the same, only the level of codification and abstraction of the knowledge stores changes as do their indices. The string-of-pearls is a means of implementing versioning of knowledge stores.

3.3 Properties / Characteristics

Each knowledge item belongs to a particular knowledge group. As already mentioned, the properties of those knowledge groups can be set by assigning distributions. A knowledge item's 'exogenous characteristics', i.e. the values of particular properties for a particular individual knowledge item at a particular point in time, depend on the distributions set on the group level. There are also 'endogenous characteristic', i.e. characteristics that evolve throughout the simulation, including those that have a starting level determined by a distribution set on the group level but are recalculated after each period based on occurrences in that period. Characteristics of knowledge stores are influenced by the characteristics of the knowledge items they hold and the actions agents who possess them perform. The following subsections describe the properties of knowledge items and knowledge stores.

3.3.1 Properties of knowledge items

Obsolescence and obsolescence rate

As an 'endogenous characteristic', the obsolescence property for each knowledge item is recalculated at the end of each period. Obsolescence at the end of the period is the Obsolescence at the start plus the obsolescence rate ($O_n = O_{n-1} + o_r$). Obsolescence varies from 0 to 1, where 1 means the knowledge is worthless (i.e. completely obsolete). The obsolescence rate property for each knowledge item specifies at what rate (units per period, rather than percent per period) that knowledge item becomes obsolete.

Base value

It is the basic value of a knowledge item, which may be different from the price that agents set for that knowledge item when they trade it (see further down for more information on the value of knowledge).

Per period carrying cost

It is the cost of knowing or holding a particular knowledge item, per period. Knowledge with a higher level of abstraction and codification has a lower carrying cost.

Abstraction increment & codification increment

It is the increment by which abstraction and codification can increase. Note that a knowledge item itself does not have a level of abstraction or codification – only a knowledge store has a level of abstraction and codification. However, knowledge items have an abstraction increment and a codification increment, that helps determine how fast the knowledge stores containing them can be abstracted and codified. Whether a smaller or a larger proportion of the abstraction or codification increment is used is a function of the experience an agent has accumulated.

Diffusion

Diffusion is calculated as the percentage of agents who possess a particular piece of knowledge. When calculating the number of agents, those agents who are cropped (i.e. those who died after exhausting their resources), exited (and not re-entered), or have merged, are not counted. Those agents formed out of merger are counted, but not those parent agents forming the merger. The diffusion property for each knowledge item is calculated dynamically at the end of each period

(endogenous characteristic). As it is an emergent property of the simulation, it is not a property that can be set in advance. Consequently, diffusion rate is not allowed to be stored or set, as rate of diffusion is also an emergent property of the system. Diffusion has two components. First, it is a consequence of voluntary exchange through agent interactions such as trading, joint ventures (JVs), alliances, licensing. Second, it results from spillovers through scanning. Those spillovers reflect the degrees of codification and abstraction of knowledge stores (and, by implication, of knowledge elements in the stores), as more abstract and codified knowledge is easier to scan. Since scanning is more likely to happen on more codified and abstract knowledge, diffusion will be faster for more codified and abstract knowledge. Patented and copyrighted knowledge is also easy to scan, but it is harder to exploit (see further down for details).

3.3.2 Properties of knowledge stores

Location

Knowledge stores have a specific two-dimensional location (X/Y) in the SimWorld. Both X and Y can have values of between 0 and 100.

Codification & abstraction

Abstraction and codification have values of between 0 and 1. Codification and abstraction are properties of knowledge stores, and not properties of knowledge items. This is because knowledge items are held by multiple agents and one agent's investment in codification or abstraction does not influence the codification and abstraction of the same knowledge item in other agents' knowledge stores. So, knowledge items do not have a level of abstraction and codification. Only in knowledge stores is there a level of abstraction and codification. This is because abstraction and codification are local (i.e. particular to a given knowledge store, which resides in some location), as opposed to global. However, the user can set a 'starting degree' of codification and abstraction for each knowledge group. Once a particular knowledge item is scanned by an agent, this property determines the level of codification and abstraction of the knowledge store that holds the newly obtained item. Therefore, there are knowledge groups with knowledge items that will start more codified and abstract and knowledge groups with knowledge items that will start less codified and abstract.

3.4 Knowledge networks

3.4.1 Linked knowledge

When configuring the simulation, the user can specify links between individual knowledge items in the knowledge ocean (also across knowledge groups). This results in knowledge networks. Agents cannot link knowledge themselves during the simulation runs, since knowledge links are defined as being natural, and discovered, rather than man-made. Links are set by the user a priori and created during the start-of-simulation processing. This also allows the user to specify exactly the density of knowledge links. Depending on the knowledge group properties, each of the linked knowledge items has a different starting degree of codification and abstraction.

Linked knowledge is important for the individual discovery process. The more codified and abstract a knowledge item gets, the higher the probability that the agent who possesses that knowledge item (who holds it in his knowledge stores) sees the knowledge items that are linked to that particular knowledge item. The same applies for knowledge items that get less codified and abstract (through absorbing and impacting of the knowledge stores): as absorbing and impacting advances, it is more probable that the agent sees the associated network. However, there is a big difference between going up in the I-Space (codifying and abstracting) and going down in the I-Space (absorbing and impacting). Going up in the I-Space makes it more probable to see the more codified and abstract knowledge items in the network. Going down in the I-Space makes it more probable to see the less codified and abstract knowledge items in the network. When talking about more or less codified and abstract knowledge items in the network, the 'starting degree' of codification and abstraction is referred to.

Ontological connections are created by the user at the beginning of the simulation. The agent tries to discover the connections (linked knowledge items) epistemologically by investing in increased codification and abstraction (or absorption and impacting). The agent's experience and investment increases the number of knowledge items that are visible to the agent and hence amenable to appropriation.

3.4.2 Discovered through investment (DTI) knowledge

DTI knowledge is a special kind of knowledge, which is discovered through investing in related (child) knowledge. DTI knowledge items cannot be discovered through scanning (an agent's action to obtain knowledge items). The user chooses up to five knowledge items to be children of a DTI knowledge item (DTI network; separate from linked knowledge). The only way for an agent to discover DTI items is to successfully scan the children and then to abstract and codify them beyond a user-specified value. Once this level is reached the agent automatically obtains the DTI knowledge item. Investing in the child items, i.e. scanning, codifying and abstracting them, is the primary means of discovering DTI knowledge. By specifying the value characteristics of the DTI knowledge item, the user can indirectly determine the value of the respective DTI network.

3.5 Owning vs. possessing knowledge

An agent can own or possess knowledge. Agents possess the knowledge if they possess a knowledge store that contains the knowledge item. This happens after they have scanned a knowledge item or acquired

it through licensing, an alliance or a merger (see further down for details on individual action types). It can also be the result of an original endowment of knowledge the user specifies when determining the simulation settings. Possession ends when the agent finishes possessing the knowledge store that contains the knowledge item. This happens after the agent disposes of all knowledge stores containing the knowledge item, or when the agent dies (i.e. is cropped, or merges with another agent).

An agent may possess a piece of knowledge, but not own it. Ownership indicates when an agent has legal rights to benefit from a knowledge item (e.g. via trade, JV, licensing, etc.). Ownership happens through patenting or copyrighting (of a knowledge item). Knowledge, i.e. the knowledge store containing the knowledge item, must be above a certain codification and abstraction level (0.5 for both) in order to be patentable. Knowledge must be above a certain codification level (0.5) in order to be copyrightable. This means that knowledge can only be owned once the knowledge store has reached a certain level of codification and abstraction (however, all knowledge can be possessed). An agent cannot patent or copyright a knowledge item that also another agent holds at a codification level of ≥ 0.5 (for details see description of the codification and abstraction process further down). All of the agent's knowledge stores that hold a patented or copyrighted knowledge item are considered to be owned by that agent.

Ownership also happens when the knowledge is bought in a trade or acquired through a joint-venture, an alliance, or a merger. This means, that multiple agents may jointly own a piece of knowledge, e.g. when the knowledge is joint-ventured. It also means that some actions, e.g. trade, bring about transfer of ownership, in which case ownership for one agent (the seller) ends and ownership for another agent (the buyer) begins. Ownership ends when the agent dies (i.e. is cropped or merges with another agent) or when the agent disposes of all knowledge stores containing the knowledge item. It also ends when knowledge, i.e. a knowledge store containing a knowledge item, is sold in a trade or when the duration of the patent or copyright ends. Note that in these cases though the agent loses ownership it still possesses the knowledge. Not all knowledge is owned, e.g. with tacit knowledge, we are more concerned with possessing than with owning. An agent possessing knowledge will be able to exploit it and learn from it whether or not that agent owns it (see further down for details). Strengthening an ownership claim on particular knowledge – patenting or copyrighting – however decreases the ability of others who only possess it to exploit it.

3.6 What is knowledge worth?

Agents can capitalize on their knowledge by exploiting, trading or (exclusively and non-exclusively) licensing it. The base value of a knowledge item will then be transformed into a *trade amount* or *license amount* or *exploit amount*. This is done according to the level of abstraction and codification of the knowledge store which holds the knowledge item, and according to the diffusion of the knowledge item. The user can define an industry specific table of revenue multipliers based on abstraction and codification level. When determining amounts, the abstraction and codification level of the knowledge store is read, and then the corresponding relevant abstraction-codification-multiplier from the industry-specific revenue-multiplier-for-abstraction-and-codification-level table is looked up. In the I-Space, the value of knowledge is a function of both utility (how codified and abstract) and scarcity (how far diffused). Usually, the higher the codification and abstraction level, the higher the revenue multiplier, i.e. more codified and abstract knowledge is worth more. More codified and abstract knowledge, however, is also more likely to be diffused. This erodes the value of knowledge. In addition to that, the calculations include obsolescence, which also has a negative effect on value: completely obsolete knowledge is worthless. Whereas revenue multipliers depend on the characteristics of the knowledge store (level of codification and abstraction), obsolescence solely depends on the properties of the knowledge item.

Note the distinction between capital value and rental value of knowledge. When knowledge is traded, it is traded for its capital value, and this is a once off transfer of funds. In contrast, when knowledge is licensed, the revenue of the license is recurring revenue, charged by the licensor, to the licensee, at the end of every period, for the duration of the license.

The *trade amount* is calculated as follows:
Trade amount = (base value) x [(relevant abstraction-codification-multiplier for knowledge store) / (diffusion for knowledge item)] x (price multiplier attribute for that agent) x (1 - obsolescence for knowledge item)
The price multiplier is used to account for the fact that certain agents routinely price knowledge higher than others.

The e*xclusive license* amount is:
Exclusive license amount = (base value) x 6% x [(relevant abstraction-codification-multiplier for knowledge store) / (diffusion for knowledge item)] x (1 - obsolescence for knowledge item)

The *non-exclusive license amount* is:
Non-exclusive license amount = (base value) x 3% x [(relevant abstraction-codification-multiplier for knowledge store) / (diffusion for knowledge item)] x (1 - obsolescence for knowledge item)

The *exploit amount* is:
Exploit amount = (base value) x (efficiency as exploiter) x [(relevant abstraction-codification-multiplier for knowledge store) / (diffusion for knowledge item)] x (1 – obsolescence for knowledge item)

The more efficient agents are, the more revenue they are able to generate. Trading and licensing are – from a conceptual perspective – mutually agreed prices, so efficiency does not influence revenues there. Since the price is agreed in a two-party exchange, efficiency only influences the cost of a trade action or a license action (see description of action specific properties further down for how efficiency influences the cost of performing an action). Exploitation is a one-sided action, so if agents are efficient at it, they will make more money in revenues.

4 AGENTS

The 'SimWorld' is populated by agents. Depending on the agent group, those agents have certain *characteristics* and perform numerous *actions*.

4.1 Properties / Characteristics

Each agent belongs to a particular agent group. As already mentioned, the properties of those agent groups can be set by assigning distributions. The value of a particular property for a particular individual agent at the start of the simulation depends on the distributions set on the group level. The agent's property values (agent's characteristic) will change over time as the simulation proceeds. There are various properties that agents can have. They can be either general properties or action specific properties.

The user can directly endow agent groups with a prior stock of knowledge, i.e. they can specify which particular knowledge items agents possess from period one on. In addition to that, 'agent-knowledge-densities' come out as emergent properties of a simulation. There are various action-types which allow agents to acquire knowledge (e.g. trade, license, etc.) and each agent has some propensity of indulging in that action-type.

4.1.1 General properties for agents

Location

Agents have a specific two-dimensional location (X/Y) in the SimWorld. Both X and Y can have values of between 0 and 100. For simplicity, it is assumed that more than one agent and knowledge store can occupy the same co-ordinate in space. This rule pertains to actions such as agent relocating. If an agent re-enters the SimWorld, having exited earlier, its co-ordinate is randomly assigned.

Experience funds

Agents have *financial funds* and *experience funds*. The financial funds are reduced / increased when a purchase / sale (i.e. trade) is made. The experience fund is not usually reduced, and may increase during both purchase and sale. When an action is undertaken, the base experience gained or lost is added to the base experience fund and the base financial cost is subtracted from the financial funds (see further down for more details). Agent experience has the following effects: agents with higher experience have lower costs, higher efficiency, and higher effectiveness. The experience an agent has accumulated also influences whether a smaller or a larger proportion of the abstraction or codification increment is used. Experience funds is not updated at the end of each period, it is only changed during / after each successful action.

Financial funds

As an 'endogenous characteristic', the financial funds of each agent need to be recomputed at the end of each period. The 'per period carrying cost' for all knowledge items held by the agent during a period is subtracted from the agent's financial funds. The exact amount subtracted depends on the abstraction and codification of that knowledge. Knowledge with a higher level of abstraction and codification has a lower carrying cost. That is, the actual amount subtracted from the financial funds is the 'per period carrying cost' times (2 minus (abstraction times codification)). This is because codification and abstraction should lower the cost of carrying knowledge. Note that financial funds are also updated *during* each action – costs of action are subtracted if the action can be successfully taken, and the revenues from the action are added to financial funds (and payments resulting from the action are subtracted). The licensing costs and revenues are also subtracted from or added to an agent's financial funds each period for the length of the license. Remember that license amounts are per period, whereas trade amounts are once off. At the end of each period, an agent has subtracted from its financial funds the license amount for each knowledge item for which it is licensee and has added to its financial funds the license amount for each item for which it is licensor. At the end of each period, the agent's income per period (i.e. income from external sources, not relating to specific transactions) is added to their financial funds and their expense per period (i.e. expenses from external sources, not relating to specific transactions) is subtracted from their financial funds. Agents currently cannot go into credit, i.e. agents cannot have negative financial funds.

Income and expense per period

Income per period is the base income per period from external sources, not relating to specific transactions, i.e. it does not include money made from trading, licensing, and other transactions. Thereby, income per period is merely a fixed, recurring income. Similarly, agents can have base expenses per period from external sources. Those do not relate to specific transactions, i.e. they do not include costs of taking actions, or costs relating to trading, licensing, and other transactions. Thus, expense per period is merely a fixed, recurring cost.

Vision

Vision determines how far the agent can see spatially. An agent's vision is a certain radius from its current location within which it can *scan* and call for *meetings*. Thereby, it establishes the size of the market within which the agent operates. Some will be village markets and some will be global markets.

Price multiplier
Some agents habitually sell at a higher price than other agents. Hoarders have a high price multiplier – greater than 1 – whereas Sharers have a low price multiplier – less than 1. The price multiplier characteristic is normally assigned a value between 0.5 and 1.5.

Activity rate
Activity rate is the number of actions that an agent takes per period. By setting the activity rate, users can set up groups that work frantically (many actions per period) and groups that are laid back (with few actions per period). To create agents that work-hard-and-play-hard, one can set the activity rate high, but set the propensity for the relax action high as well, meaning that the agent will relax often. Whereas *vision* affects the range of an agent's interaction, *activity rate* affects the rate of interaction.

4.1.2 Action specific properties for agents

Propensity
Propensity refers to the likelihood of performing a particular action. Propensity is specified for each action-type. Each time an agent takes an action it must decide which particular action to take. The agent will attempt to embark on a given action-type with the probability (in this case, the propensity) drawn from the distribution associated with the propensity property for that action type for the respective agent group. That is, propensity is the probability of the agent attempting a particular action, each time the agent takes an action. Again, agents may take zero or more actions per period, according to their activity rate. The propensities across all action-types for a given agent must sum to 1. If the sum comes out to more than one or less than one, then the propensities are scaled to 1. This is the reason for why, when it comes to creating distributions, each propensity attribute must have a constant distribution and the sum of the constant distribution values for all the propensity properties for a particular agent group must sum to 1.

Effectiveness
Effectiveness is used to determine the likelihood of succeeding in a particular action-type. Whether an action succeeds depends on the effectiveness of all agents participating in the action. The average of the effectiveness of all agents participating in the event gives the probability of the action succeeding, given the action is undertaken. It is assumed that the environment does not introduce any additional random variation. Effectiveness is partly a function of the agent's experience fund. This gives agents a good reason to invest in such funds.

Efficiency
Efficiency is a weighting factor, used to adjust cost of participation in an action-type. In other words, it is a cost modifier for an action. As applies to effectiveness, efficiency is partly a function of the agent's experience fund. This gives agents a good reason to invest in such funds.

Base financial cost
It is the base cost of participating in a particular action. Every action has a financial cost, which must be paid for by the agent when embarking on that action. The agent must pay this cost, from its financial funds, immaterial of whether the action succeeds or not. Financial funds are updated during each action – costs of action are subtracted if the action can be successfully taken. If agents have insufficient funds to undertake an action, then the action cannot be successfully taken, and their financial funds are not depleted. The final cost particular to an agent when undertaking an action equals the base financial cost for that action divided by efficiency for that action.

Base experience gained or lost
It is the amount of experience gained, or lost, by participating in an action-type. Each action undertaken by the agent normally increases the agent's base experience by a certain amount. There is only one experience fund per agent, no accumulated experience per action-type.

Experience threshold required
It is the minimum amount of experience required to undertake a specific action.

4.2 Actions

The artificial agents in the simulation are able to perform various *actions* and thereby to adopt different role types. All actions – aside from entering – require the agent to be inside the SimWorld. Actions are assumed to have zero duration, start and end in the same period and are purposefully taken by agents. This stands in contrast to *states* that have some duration and that are inferred by the system based on the actions taken in the previous periods. For example, the system will infer that the agent starts *owning* a knowledge item if the agent bought the knowledge item during the period; similarly the system will infer that the previous owner finishes *owning* the knowledge upon selling ("owning" is a state). An agent will be permitted to undertake an action, if and only if, the environment and the agent state permit the action. Most importantly, an agent can only perform an action, if it has sufficient financial funds and experience. An agent will not be able to undertake an action if it cannot pay the cost of action. Entry into the SimWorld is only possible, if the agent is outside and alive and the carrying capacity of the SimWorld has not been exceeded. Also, an agent can only license or trade a piece of knowledge if they own the knowledge item. The state of the world as well as that of the agent (and the knowledge) changes after an action is successfully undertaken. What follows next is a description of each action-type. When deciding what to do in a period, agents pick from this list of actions.

Scanning (storing)
An agent can scan knowledge. Scanning means picking up a random knowledge item, whether from the knowledge ocean or from other agents' knowledge

stores. An agent can scan any knowledge item in the knowledge ocean, but can only scan knowledge items in knowledge stores within its vision. An agent can scan knowledge possessed or owned (patented or copyrighted) by other agents. Agents only try to pick up knowledge items that they do not already possess at that level of abstraction and codification. If a knowledge item is successfully scanned, it starts off in a new knowledge store possessed (but not owned) by the agent. Depending on the origin of the knowledge item, the new knowledge store picks up the level of codification and abstraction from the knowledge group the knowledge item belongs to (knowledge item from knowledge ocean) or from the knowledge store it found the item in (knowledge item from another agent). The ease with which knowledge is scanned is a function of the degree of codification and abstraction of the knowledge involved. Knowledge items in knowledge stores with higher abstraction and codification have a higher propensity of being scanned, so the program weights the probability of being scanned by the abstraction and codification levels of the knowledge chosen. Remember the special role of networks. The more an agent codifies and abstracts the knowledge stores of a particular knowledge item which is linked in the knowledge ocean, the more likely it is that the agent will discover, through scanning, those linked knowledge items that have a high starting level of codification and abstraction. The more an agent absorbs and impacts the knowledge stores of a particular knowledge item which is linked in the knowledge ocean, the more likely it is that the agent will discover those linked knowledge items that have a low starting level of codification and abstraction. Also remember that once an agent has scanned all the child items of a DTI knowledge item and has codified and abstracted them up to a certain level, then the agent automatically gets the DTI knowledge item associated to that DTI network (see earlier).

Learning

An agent can learn existing knowledge. Learning enables agents to exploit the knowledge items they learned of. Before knowledge items can be exploited, learning has to take place. Agents can only learn from a knowledge store they possess. Their chances of successfully learning are higher for more codified knowledge.

Abstracting and Codifying

An agent can abstract and codify a knowledge store (two separate actions). Abstraction increases knowledge abstraction; codification increases knowledge codification. Both actions only occur on knowledge stores (not on knowledge items). The agent must possess the knowledge store to carry out abstraction or codification. The abstraction level and codification level of the knowledge store is updated after each abstracting or codifying action. Abstraction and codification of the knowledge store increases by the abstraction increments or respectively the codification increments of the (single) knowledge item in the store. The exact increase is weighted by the agent's experience: agents with more experience get a larger proportion of the abstraction or codification increment. The knowledge can be abstracted and codified without any limits. However, the level of abstraction and the level of abstraction cannot exceed 1.

Absorbing and impacting

We speak of absorption if an agent's knowledge gets more uncodified, and of impacting if an agent's knowledge gets more concrete. They can be considered as negative codifying and abstracting. However, the user can set separate characteristics for each of them.

Patenting and Copyrighting

An agent can patent or copyright knowledge (two separate actions) for a certain duration and with a specific strength. An agent can only patent or copyright a knowledge item it possesses. For patenting the agent has to hold the knowledge item in a knowledge store that has an abstraction level of at least 0.5 and a codification level of at least 0.5. For copyrighting, the knowledge store must have a codification of at least 0.5. That is, an agent can only invest in patenting after it has invested in codifying and abstracting. For copyrighting the agent has to have invested in codifying. Each patent and each copyright has a particular strength and duration. The user can assign a distribution for both characteristics of both actions (general setting for all knowledge). Once an agent patents or copyrights an item, it owns that item. As a consequence, all of the agent's knowledge stores that hold the newly patented or copyrighted knowledge item are then eligible for the actions that require ownership (trading, licensing, etc.). An agent may not patent or copyright a knowledge item that is already possessed by another agent in a knowledge store that has a codification level of ≥ 0.5. The codification level of ≥ 0.5 ensures that there is a possible 'race to codify/abstract that brings patent reward' and that possessed knowledge in lower regions of the I-Space (ideas / thoughts in an agent's head) do not inhibit other agents to patent / copyright their already developed knowledge. Patents or copyrights are not diffusion blocking. They just restrain other agents from doing certain things with their knowledge items, e.g. selling, licensing, successfully exploiting.

Exploiting

An agent can exploit knowledge to gain value. Exploitation means capitalizing on internalized knowledge. This means that an agent must learn the knowledge prior to exploiting it, i.e. perform the learning action on the knowledge item. The financial funds of the exploiter agent are increased by the value of the exploiting (see exploit amount above). An agent can exploit a piece of knowledge that it possesses, even if it does not own it. The higher the strength of the patents and copyrights held on the knowledge by other agents, the lower the probability of successfully exploiting, if the agent does not have a license for that

knowledge. If the agent holds the patent or copyright or has a license for the knowledge, then their chance of successfully exploiting the knowledge is high.

Meeting

An agent can meet with another agent. Only agents who have initiated the meeting (initiator) and those who have responded positively are allowed to attend. An agent can only meet with agents within its vision. Meeting is a prerequisite for a trade, license, merger, joint venture and alliance.

Buying / selling knowledge (trading)

An agent can buy (sell) knowledge stores from (to) another agent for a certain price (sale amount). In contrast to scanning, buying only targets knowledge that is owned by other agents. Meeting is a prerequisite for trading, and mutual consensus is necessary. Agents can only sell knowledge stores that they own, i.e. knowledge that is copyrighted or patented. This also implies that if you want to trade knowledge, it must be up to a certain level of codification and abstraction. The buyer acquires ownership and the seller loses ownership. This means that the patent or copyright is terminated for the seller, and the rest of the patent or copyright (remaining time) is transferred to the buyer. Note that the seller still possesses the knowledge and is still in a position to learn and to exploit it. An agent can only attempt to acquire a knowledge item that it does not already own at that level of abstraction and codification (since agents would be wasting money buying something they already own). The trading price is determined using the formula for trade amount mentioned earlier. The financial funds of the seller agent are increased by the sale value for the trade, and the financial funds of the buyer agent are decreased by the sale value for the trade.

Licensing – exclusive and non-exclusive

An agent can license knowledge, exclusively or non-exclusively (two separate actions), to another agent for a certain per period license amount / fee. Meeting is a prerequisite for licensing, and mutual consensus is necessary. Agents can only be licensor for knowledge that they own. Furthermore, agents only attempt to be licensee for a knowledge item that they do not already own at that level of abstraction and codification. The exclusive licensor gives the knowledge to one licensee. That is, only one exclusive license can be given. The non-exclusive licensor gives the knowledge to various licensees for joint possession. That is, multiple non-exclusive licenses may be given. The licensee does not acquire ownership of the knowledge. Though they own the knowledge, licensors may not exploit the knowledge – only the licensees may exploit the knowledge. For simplicity the special type of license where licensors can continue to exploit their knowledge is omitted from the simulation. At the end of each period, and for the duration of the license, the financial funds of the licensor agent are increased by the license amount. Similarly, for the duration of the license, and at the end of each period, the financial funds of the licensee agent are decreased by the license amount. The price of the license, i.e. the license amount, is determined using the formula for license amount shown earlier.

Forming joint ventures

An agent can form joint ventures with another agent. The goal is to exchange knowledge (a knowledge store). Meeting is a prerequisite for a joint venture, and mutual consensus is necessary. Agents can only joint venture knowledge that they own. An agent does not lose ownership of the knowledge when it is joint-ventured. The partner in the joint venture gains (joint) ownership of the joint-ventured knowledge, i.e. gets a copy of the patent or copyright. If one of the partners trades or licenses the joint-ventured knowledge, the revenues are split.

Making alliances

An agent can form an alliance with another agent. Meeting is a prerequisite for making an alliance, and mutual consensus is necessary. When an alliance is made, both parties gain (joint) ownership of all the knowledge that is held by the other party. If one party trades or licenses that knowledge later, the revenues are split. An alliance is equivalent to the joint-venturing of all knowledge held by both agents joining the alliance.

Merging

An agent can merge with another agent to form a new entity (agent). Meeting is a prerequisite for merging, and mutual consensus is necessary. When an agent is merged, the old agents cease to exist, and a new agent comes into being. On merger, assets are pooled, but 20 percent of financial funds and experience funds are liquidated on merger to account for the costs of merger. Also, on merger, the new agent gains ownership of all the knowledge owned by the merged agents, and the merged agents, which are now dead, lose ownership of that knowledge. For the new agent's properties the program randomly picks of the properties of the agents that are merging.

Disposing

An agent can dispose a knowledge store. Agents cannot dispose of knowledge items since these are eternal propositions.

Relocating

An agent can relocate a certain distance. Relocating implies moving closer to / further from other agents or knowledge stores. The distance an agent moves per relocation depends on the respective agent group's distance settings for the relocate action (i.e. on the distribution of the distance property of the relocate action for the particular agent group). Relocation is important in the context of scanning and for calling a meeting as it affects what you see and who you see. As agents can only scan and call for meeting within the radius of their vision, they are only able to pick up knowledge or meet people in a different area by moving / relocating. When agents relocate, they leave their knowledge stores behind (N.B.: A new knowledge store is always given the same location as the agent).

Exiting and entering

An agent can exit the SimWorld. It can do so in any particular period with its current rent earnings. An example of exiting is retiring to the Bahamas to live a life of luxury. Once the agent has exited, the only action possible for the agent is entrance. An agent can enter the SimWorld if it is outside the simulation world. *Being in SimWorld* is a particular state an agent has or has not. It indicates when an agent is alive and is a citizen of the SimWorld (i.e. has not retired).

Relaxing

An agent can relax for a specified duration. Relaxing means that the agent will undertake no action. The duration specifies the number of periods the agent is relaxing for. Actions are not queued while the agent is relaxing; they simply do not try to undertake any actions whatsoever. Agents will only start trying to undertake actions after their relaxation duration is over. Relaxation is currently the only action-type that has a duration. The simulation determines the duration randomly by assigning a value between 1 and 10.

The death of an agent – cropping

Agents die if there are insufficient resources to sustain them. The agent's financial funds must be zero or negative for the crop action type to occur. Cropping happens each period after the updating of financial funds which happens during end-of-period processing. Once the agent is cropped, no action is possible for the agent, since the agent is dead. After cropping, states like 'owning' and 'is in SimWorld' are updated, because a dead agent can no longer own anything or be in the SimWorld. All agents with financial funds of zero must be cropped. This goes almost without saying since an agent without financial funds cannot afford to perform any action as they cannot afford the costs of performing any action. However, the simulation still needs to crop / kill them, so the user can see which agents died and when. It is assumed that dead agents cannot have an estate, and that they merely forfeit ownership when they die, rather than have ownership vest in the estate of the deceased agent. Note that an agent also dies when it merges. A merger results in a new agent living. Being *dead* or *alive* is a particular state an agent has.

5 CONCLUSION

Having described the special features of this unique agent-based graphical simulation environment in detail, the informed reader can see that *SimISpace2* is an exceptional tool to simulate knowledge management processes. The user can model and analyze very complex scenarios to study agents' (inter-)actions together with knowledge assets and their life-cycles. There are numerous practical problems that can be simulated using *SimISpace2*. Examples of practical applications would be intellectual property rights, cognitive arms races (e.g. two pharmaceutical firms racing to discover a new molecule), countries competing to attract intellectual capital, and intra-firm technology transfer strategies.

6 REFERENCES

Boisot, M., Canals, A., & MacMillan, I. C. (2003). *Neoclassical versus Schumpeterian approaches to learning: A knowledge-based simulation approach.* Paper presented at the ABS 2003: 4th Workshop on Agent-Based Simulation (Society for Modeling and Simulation International), Montpellier, France.

Boisot, M., MacMillan, I. C., & Han, K. S. (2007). Property rights and information flows: a simulation approach. *Journal of Evolutionary Economics, 17*(1), 63.

Boisot, M. H. (1995). *Information Space: A Framework for Learning in Organizations, Institutions and Culture* London: Routledge.

Boisot, M. H. (1998). *Knowledge assets - securing competitive advantage in the information economy*. New York: Oxford University Press.

MacMillan, I. C., Boisot, M., Abrahams, A. S., & Bharathy, G. (2005). *Simulating the knowledge transfer dilemma: Lessons for security and counter-terrorism.* Paper presented at the 2005 Summer Computer Simulation Conference (SCSC'05), Philadelphia, PA.

AUTHOR BIOGRAPHIES

MARTIN IHRIG is a visiting scholar at the Sol C. Snider Entrepreneurial Research Center at the Wharton School of the University of Pennsylvania. Previously, he was a research assistant at Otto-Friedrich-University of Bamberg, Germany and member of the EXIST - High Technology Entrepreneurship Postgraduate Program. Ihrig received his Master of Business Studies in Strategic Management and Planning from UCD Michael Smurfit School of Business, Ireland in 2003. Previously, he had studied in Spain (Carlos III de Madrid, International Economics), in France (École Supérieure de Commerce de Grenoble, International Business) and in Germany (University of Bamberg, European Business Administration and Economics). His research in the fields of entrepreneurship and strategic knowledge management focuses on modeling opportunity recognition processes with the help of agent-based simulations.

His e-mail address is: ihrig@wharton.upenn.edu

ALAN S. ABRAHAMS is a lecturer in the Department of Operations and Information Management, The Wharton School. He holds a PhD from the University of Cambridge, and a Bachelor of Business Science degree from the University of Cape Town. His research focuses on Entrepreneurial Decision Support, with particular interests in data mining, new opportunity identification and valuation, medical informatics for entrepreneurial philanthropy (HIV/AIDS clinical decision support), automated interpretation and enforcement of regulations, and development of simulation environments for modeling knowledge transfer.

His e-mail address is: asa28@wharton.upenn.edu