In Proceedings of the First International Conference on Multi-Agent Systems, 254-260, AAAI Press / MIT Press, 1995.

A Formal Framework for Agency and Autonomy

Michael Luck
Department of Computer Science
University of Warwick
Coventry, CV4 7AL
United Kingdom
mikeluck@dcs.warwick.ac.uk

Mark d'Inverno
School of Computer Science
University of Westminster
London, W1M 8JS
United Kingdom
dinverm@westminster.ac.uk

Abstract

With the recent rapid growth of interest in Multi-Agent Systems, both in artificial intelligence and software engineering, has come an associated difficulty concerning basic terms and concepts. In particular, the terms *agency* and *autonomy* are used with increasing frequency to denote different notions with different connotations. In this paper we lay the foundations for a principled theory of agency and autonomy, and specify the relationship between them. Using the Z specification language, we describe a three-tiered hierarchy comprising objects, agents and autonomous agents where agents are viewed as objects with *goals*, and autonomous agents are agents with *motivations*.

Introduction

Although there has been substantial progress in artificial intelligence for many years, research tended to focus on solving problems without regard to any real external environment or to the notion of a reasoning *agent*. In other words, the problems and their solutions, while significant, were limited in that they were divorced from real situations. More recently, however, the importance of these limitations has been recognised. One consequence is the rapid growth of interest in the design and construction of *agents* as systems exhibiting intelligent behaviour.

Concepts of *agents* and *agency* are increasingly being used in a range of areas in artificial intelligence (AI) and computer science (Wooldridge & Jennings 1995). However, these notions are often different, even within the same subfield, and the lack of a common understanding can stifle further research and development. In distributed AI, *agents* are often taken to be the same as *autonomous agents* and the two terms are used interchangeably, without regard to their relevance or significance. The difference between these related, but distinct, notions is both important and useful in considering aspects of intelligence.

In this paper, we define *autonomy* and *agency*, and explicate the relationship between them. We argue that autonomy is distinct and is achieved by *motivating* agency. Our concern is to develop a formal model of agency and autonomy that can be used both as the basis of an implementation, and also as a precise but general framework for further research.

Given the range of areas in which the terms *agent* and *agency* are applied, the lack of consensus over meaning is not surprising. As Shoham points out, the number of diverse uses of the term *agent* are so many that it is almost meaningless without reference to a particular notion of agenthood (Shoham 1993).

In a recent collection of papers, for example, several different views emerge. For Smith, an agent is a "persistent software entity dedicated to a specific purpose" (Smith, Cypher, & Spohrer 1994). Selker takes agents to be "computer programs that simulate a human relationship by doing something that another person could do for you" (Selker 1994). More loosely, Riecken refers to "integrated reasoning processes" as agents (Riecken 1994). Others avoid the issue completely and leave the interpretation of their agents to the reader. It is recognised, however, that there is no agreement on what it is that makes something an agent.

If we are to confront the problems that arise from this lack of agreement and definition, then the use of formalisms is appropriate since they allow unambiguous descriptions of complex systems, and also provide proof systems and sets of proof obligations which enable the construction of reliable and robust models. Formalization provides clarity in characterizing the nature of such important but ill-defined concepts as agency and autonomy.

In the current work, we have adopted the specification language Z (Spivey 1992) for two major reasons. First, it provides modularity and abstraction and is sufficiently expressive to allow a consistent, unified and structured account of a computer system and its associated operations. Such structured specifications enable the description of systems at different levels of abstraction, with system complexity being added at successively lower levels. Second, we view our enterprise as that of building programs. Z schemas are particularly suitable in squaring the demands of formal modelling with the need for implementation by providing clear and unambiguous definitions of state and operations

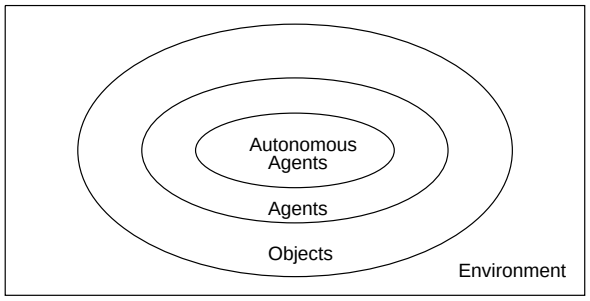

Figure 1: The Entity Hierarchy

on state which provide a basis for program development. Thus our approach to formal specification is pragmatic — we need to be formal to be precise about the concepts we discuss, yet we want to remain directly connected to issues of implementation. Z provides just those qualities that are needed, and is increasingly being used for specifying frameworks and systems in AI e.g. (Goodwin 1993).

Initial Concepts

In this section, we introduce terms and concepts that will be used to explicate our understanding of *agents* and *agency*, and provide formal definitions that are used subsequently in developing the concept of *autonomy*.

Our view of agency best relates to that of Shoham who takes an *agent* to be any entity to which mental state can be ascribed (Shoham 1993). According to Shoham, such mental state consists of components such as beliefs, capabilities and commitments, but there is no unique correct selection of them. This is sensible, and we too do not demand that all agents necessarily have the same set of mental components. We do, however, specify what is minimally *required* of an entity for it to be considered an agent.

The definition of agency that follows is intended to subsume existing concepts as far as possible. It will be built up and extended over the course of the paper, and will involve several intermediate concepts. In short, we propose a three-tiered hierarchy of entities comprising *objects*, *agents* and *autonomous agents*. The basic idea underlying this hierarchy is that all known entities are objects. Of this set of objects, some are agents, and of these agents, some are autonomous agents. This is shown as a Venn diagram in Figure 1. The following sections define what is required for each of the entities in the hierarchy.

Before we can move to a definition of any of these entities, we must first define some primitives. The first of these is an *action* which is strongly related to the notion of *agency*.

Definition: An *action* is a discrete event which changes the state of the environment.

The second primitive that needs defining is an *attribute*. Attributes are simply features of the world, and are the only characteristics which are manifest. They need not be perceived by any particular entity, but must be potentially perceivable in an omniscient sense. (The notion of a feature here allows anything to be included.)

Definition: An *attribute* is a perceivable feature.

Using the notions of actions and attributes, we provide definitions of agency and autonomy. In order to do so, we must distinguish agents and autonomous agents from other entities. We call such other entities *objects*, and discuss them below.

At a basic level, an object can be defined in terms of its attributes and its actions. An object, in this sense, is just a describable 'thing' or entity that is capable of acting, and has no further defining characteristics. This provides us with the basic building block to develop our notion of agency.

Definition: An *object* is an entity that comprises a set of actions and a set of attributes.

In Z, before constructing a specification, we must first define types. Here we define the set of all actions and the set of all attributes:

$$[Action, Attribute]$$

Now a state schema can be constructed that defines an object. Z schemas have two parts: the upper, declarative, part which declares variables and their types, and the lower, predicate, part which relates and constrains those variables. The type of any schema can be considered as the cartesian product of the types of each of its variables, without any notion of order, but constrained by predicates.

The schema below has only a declarative part containing two variables. First, *capableof* is the set of actions of the object, and is sometimes referred to as the *competence* of the object. Second, *attributes* is the set of features of the object. Objects are therefore defined by their ability in terms of their actions, and their configuration (in a broad sense) in terms of their attributes. The actions include any event in which the object is involved. The configuration of an object includes references to the body of the object and its position, and is similar to Goodwin's notion of configuration (Goodwin 1993).

$$\begin{array}{|l} \hline Object \\ capableof : \mathbb{P}\, Action \\ attributes : \mathbb{P}\, Attribute \\ \hline \end{array}$$

Attributes are manifest and are potentially accessible by an observer. The capabilities of an object in terms of actions, by contrast, are not necessarily observable.

For example, consider a robot without a power supply. Since the robot has no power, its capabilities are severely limited and include just those which rely on its

physical presence, such as supporting things, weighing things down, and so on. The attributes of the robot specify that it is blue, that it is upright, that it is large, and other such features. As a second example, consider a coffee-cup. Its capabilities state that it can support things and that it can contain liquid, for example, while its attributes state that it is stable, white and made of china. (These examples will be developed further to distinguish between agents and autonomous agents.)

An object must be situated in an environment. We take an environment simply to be a set of attributes that describes all of the features of the world and all of the entities (objects, agents or autonomous agents) in the world.

$$Environment == \mathbb{P}\ Attribute$$

An *interaction* is simply that which happens when actions are performed in an environment. The effects of an interaction on the environment are determined by applying the *effectinteraction* function in the axiom definition below to the current environment and the actions taken. This axiom definition is a global variable and is consequently always in scope. We require only one function to describe all interactions, since an action will result in the same change to an environment whether taken by an object, agent or autonomous agent.

$$effectinteraction : Environment \longrightarrow \mathbb{P}\ Action \nrightarrow Environment$$

Agency

There are many dictionary definitions for an agent. A recent paper (Wooldridge & Jennings 1994) quotes the definition of an agent as "one who, or that which, exerts power or produces an effect."[1] However, they omitted the second sense of agent which is given as "one who acts for another ...". This is important, for it is not the acting alone that defines agency, but the acting for *someone or something* that is defining. Indeed, Wooldridge and Jennings acknowledge the difficulties in a purely action-based analysis of agency.

To understand better what actually constitutes an agent, consider the example of a coffee-cup. A cup is an object. We can regard it as an agent and ascribe to it mental state, but it serves no useful purpose to do so without considering the circumstances. A cup is an agent *if* it is containing a liquid and it is doing so to some end. In other words, if I fill a cup with coffee, then the cup is my agent — it serves my purpose. Alternatively, the cup would also be an agent if it was placed upside down on a stack of papers and used as a paperweight. It would *not* be an agent if it was just sitting on a table without serving any purpose to any agent. In this case it would be an object. Note that we do not require an entity to be intelligent for it to be an agent.

Thus agents are just objects with certain dispositions. They may always be agents, or they may revert to being objects in certain circumstances. This will be explored further later. For the moment, we will concentrate on the nature of the disposition that characterizes an agent. An object is an agent if it serves a useful purpose either to a different agent, or to itself, in which case the agent is *autonomous*. This latter case is discussed further later. Specifically, an agent is something that satisfies a goal or set of goals (often of another). Thus if I want to store coffee in a cup, then the cup is my agent for storing coffee. It has been *ascribed* or, if we anthropomorphize, has 'adopted', my goal to have the coffee stored. An agent is thus defined in relation to its goals. We take a traditional view of goals as describable environmental states.

Definition: A *goal* is a state of affairs to be achieved in the environment.

Definition: An *agent* is an instantiation of an object together with an associated goal or set of goals.

Before we can define a schema for agency, we must define goals to be just a set of attributes that describe a state of affairs in the world:

$$Goal == \mathbb{P}\ Attribute$$

The schema for an agent is simply that of an object but with the addition of goals.

$$\begin{array}{|l} \textit{Agent} \\ \hline Object \\ goals : \mathbb{P}\ Goal \\ \hline goals \neq \{\} \\ \hline \end{array}$$

Thus an agent has, or is *ascribed*, a set of goals which it retains over any instantiation (or lifetime). One object may give rise to different instantiations of agents. An agent is instantiated from an object in response to another agent. Thus agency is *transient*, and an object which becomes an agent at some time, may subsequently revert to being an object.

Returning to the cup example, we have an agent with the same attributes and actions as the cup object, but now it can be ascribed the goal — my goal — of *storing my coffee*. Not everyone will know that it is an agent in this way, however. If, for example, I am in a cafe and there is a half-full cup of lemon-tea on my table, there are several views that can be taken. It can be regarded by the waiter as an agent for me, storing my tea, or it can be regarded as an object serving no purpose if the waiter thinks it is not mine. The view of the cup as an object or agent is relevant to whether the waiter will remove the cup or leave it at the table. Note that we are not suggesting that the cup actually possesses a goal, just that there is a goal that it is satisfying.

Consider also the robot example, and suppose now that the robot has a power supply. If the robot has no

[1] *The Concise Oxford Dictionary of Current English (7th edition)*, Oxford University Press, 1988.

goal, then it cannot use its actuators in any sensible way but only, perhaps, in a random way, and must be considered an object. Alternatively, if the robot has some goal or set of goals, which allow it to employ its actuators in some directed way, such as picking up a cup or riveting a panel onto a hull, then it is an agent. The goal need not be explicitly represented, but can instead be implicit in the hardware or software design of the robot. It is merely necessary for there to be a goal of some kind.

These examples highlight the range of behaviour that is available from agents. The coffee-cup is passive and has goals *imposed* upon and *ascribed* to it, while the robot is capable of actively manipulating the environment by performing actions designed to satisfy its goals.

We now introduce perception. An agent in an environment may have a set of percepts available. These are the possible attributes that an agent could perceive, subject to its capabilities and current state. However, due to limited resources, an agent will not normally be able to perceive all those attributes possible, and bases action on a subset, which we call the *actual* percepts of an agent. Some agents will not be able to perceive at all. In the case of a cup, for example, the set of possible percepts will be empty and consequently the set of actual percepts will also be empty. The robot, however, may have several sensors which allow it to perceive. Thus it is not a requirement of an agent that it is able to perceive.

For clarity of exposition, we define a *View* to be the perception of an *Environment* by an agent, and which is just a set of attributes.

$$View == \mathbb{P}\ Attribute$$

It is also important to note that it is only meaningful in our model to consider perceptual abilities in the context of goals. Thus when considering objects which have no goals, perceptual abilities are not relevant. Objects respond directly to their environments and make no use of percepts even if they are available. We say that perceptual capabilities are *inert* in the context of objects.

In the schema for agent perception, *AgentPercepts*, we add further detail to the definition of agency, and so include the schema *Agent*. An agent has a set of perceiving actions, *perceivingactions*, which are a subset of the capabilities of an agent. The function, *canperceive*, determines the attributes that are potentially available to an agent through its perception capabilities. When applied, its arguments are the current environment and the agent's capabilities. The second predicate line states that those capabilities will be precisely the set of perceptual capabilities. Finally, the function, *willperceive*, describes those attributes which are actually perceived by an agent, and will always be applied to its goals.

AgentPercepts

$$\begin{array}{l} Agent \\ perceivingactions : \mathbb{P}\ Action \\ canperceive : Environment \longrightarrow \\ \qquad \mathbb{P}\ Action \nrightarrow Environment \\ willperceive : \mathbb{P}\ Goal \longrightarrow Environment \longrightarrow View \\ \hline perceivingactions \subseteq capableof \\ \forall\, env : Environment; as : \mathbb{P}\ Action \bullet \\ \qquad as \in \mathsf{dom}\,(canperceive\ env) \Rightarrow \\ \qquad\qquad as = perceivingactions \\ \mathsf{dom}\ willperceive = \{goals\} \end{array}$$

Directly corresponding to the goal or goals of an agent is an action-selection function, dependent on the goals, current environment and the actual perceptions. This is specified in *AgentAct* below, with the first predicate ensuring that the function returns a set of actions within the agent's competence. Note also that if there are no perceptions, then the action-selection function is dependent only on the environment.

AgentAct

$$\begin{array}{l} Agent \\ agentactions : \mathbb{P}\ Goal \longrightarrow View \longrightarrow \\ \qquad Environment \longrightarrow \mathbb{P}\ Action \\ \hline \forall\, gs : \mathbb{P}\ Goal; v : View; env : Environment \bullet \\ \qquad (agentactions\ gs\ v\ env) \subseteq capableof \\ \mathsf{dom}\ agentactions = \{goals\} \end{array}$$

The state of an agent includes four new variables: *posspercepts*, describing those percepts possible in the current environment; *actualpercepts*, a subset of these which are the current (actual) percepts of the agent; *willdo*, the next set of actions the agent will perform; and *history*, the sequence of actions previously performed by the agent.

AgentState

$$\begin{array}{l} environment : Environment \\ AgentPercepts \\ AgentAct \\ posspercepts : View \\ actualpercepts : View \\ history : \mathsf{seq}\,(\mathbb{P}\ Action) \\ willdo : \mathbb{P}\ Action \\ \hline \forall\, as : \mathsf{ran}\ history \bullet as \subseteq capableof \\ actualpercepts \subseteq posspercepts \\ posspercepts = \\ \qquad canperceive\ environment\ perceivingactions \\ actualpercepts = willperceive\ goals\ posspercepts \\ perceivingactions = \{\,\} \Rightarrow posspercepts = \{\,\} \\ willdo = \\ \quad agentactions\ goals\ actualpercepts\ environment \end{array}$$

Next, we define which of the agent state variables remain unchanged after a set of actions has been performed by that agent. If any of these did change, a different agent schema would have to be instantiated.

$\Delta AgentState$

$AgentState$
$AgentState'$

$capableof' = capableof$
$goals' = goals$
$perceivingactions' = perceivingactions$
$canperceive' = canperceive$
$willperceive' = willperceive$
$agentactions' = agentactions$

We now specify how an agent interacts with its environment. As a result of an interaction, the environment and the $AgentState$ change with certain variables unaffected as defined in $\Delta AgentState$. The performed set of actions, $willdo$, is added to the $history$, and the environment is altered accordingly.

$AgentEnvInteract$

$\Delta AgentState$

$history' = history \frown \langle willdo \rangle$
$environment' =$
$\quad effectinteraction\ environment\ willdo$

Autonomy as Motivated Agency

So far we have developed a definition of agency. However, the definition relies upon the existence of other agents which provide goals that are adopted in order to instantiate an agent. In order to ground this chain of goal adoption, to escape what could be an infinite regress, and also to bring out the notion of *autonomy*, we introduce *motivation*.

Grounding the hierarchies of goal adoption demands that we have some agents which can generate their own goals. These agents are *autonomous* agents since they are not dependent on the goals of others. Autonomous agents possess goals which are *generated* from within rather than *adopted* from other agents. These goals are generated from *motivations* which are higher-level non-derivative components characterizing the nature of the agent, but which are related to goals. Motivations are, however, qualitatively different from goals in that they are not describable states of affairs in the environment. For example, consider the motivation *greed*. This does not specify a state of affairs to be achieved, nor is it describable in terms of the environment, but it may (if other motivations permit) give rise to the generation of a goal to rob a bank. The distinction between the motivation of greed and the goal of robbing a bank is clear, with the former providing a reason to do the latter, and the latter specifying what must be done.

The following definition draws on Kunda's work on motivation in psychology (Kunda 1990).

Definition: A *motivation* is any desire or preference that can lead to the generation and adoption of goals and which affects the outcome of the reasoning or behavioural task intended to satisfy those goals.

A *motivated agent* is thus an agent that pursues its own agenda for reasoning and behaviour in accordance with its internal motivation. Since motivations ground the goal-generation regress, we claim that motivation is critical in achieving autonomy. An *autonomous agent* must necessarily be a *motivated* agent.

Definition: An *autonomous agent* is an instantiation of an agent together with an associated set of motivations.

We can now specify an autonomous agent. First we define the set of all motivations as a base type.

$[Motivation]$

An autonomous agent is defined as an agent with motivations and some potential means of evaluating behaviour in terms of the environment and these motivations. In other words, the behaviour of the agent is determined by both external and internal factors. This is qualitatively different from an agent with goals because motivations are non-derivative and governed by internal inaccessible rules, while goals are derivative and relate directly to motivations.

$AutonomousAgent$

$Agent$
$motivations : \mathbb{P}\ Motivation$

$motivations \neq \{\}$

In illustration of these ideas, note that the cup cannot be considered autonomous because, while it can have goals ascribed to it, it cannot *generate* its own goals. The robot, however, is potentially autonomous in the sense that it may have a mechanism for internal goal generation depending on its environment. Suppose the robot has motivations of achievement, hunger and self-preservation, where achievement is defined in terms of fixing tyres onto a car on a production line, hunger is defined in terms of maintaining power levels, and self-preservation is defined in terms of avoiding system breakdowns. In normal operation, the robot will generate goals to attach tyres to cars through a series of subgoals. If its power levels are low, however, it may replace the goal of attaching tyres with a newly-generated goal of recharging its batteries. A third possibility is that in satisfying its achievement motivation, it works for too long and is in danger of overheating. In this case, the robot can generate a goal of pausing for an appropriate period in order to avoid any damage to its components. Such a robot is autonomous because its goals are not imposed, but are generated in response to its environment.

Autonomous agents also perceive, but motivations, as well as goals, filter relevant aspects of the environment. In the schema below, the function $autowillperceive$ is a more complex, but related, version of an agent's $willperceive$. However, that which an autonomous agent is *capable* of perceiving at any time is independent of its motivations. Indeed, it will

always be independent of goals and motivations, and there is consequently no equivalent increase in functionality to *canperceive*.

AutonomousAgentPercepts

$$
\begin{array}{l}
AutonomousAgent \\
AgentPercepts \\
autowillperceive : \mathbb{P}\, Motivation \longrightarrow \mathbb{P}\, Goal \longrightarrow \\
\qquad\qquad Environment \longrightarrow View \\
\hline
willperceive = autowillperceive\ motivations \\
\mathsf{dom}\ autowillperceive = \{motivations\}
\end{array}
$$

The next schema defines the action-selection function and includes the previous schema definitions for *AgentAct* and *AutonomousAgent*. The action-selection function for an autonomous agent is produced at every instance by the motivations of the agent, and is always and only ever applied to the motivations of the autonomous agent.

AutonomousAgentAct

$$
\begin{array}{l}
AutonomousAgent \\
AgentAct \\
autoactions : \mathbb{P}\, Motivation \longrightarrow \mathbb{P}\, Goal \longrightarrow \\
\qquad View \longrightarrow Environment \longrightarrow \mathbb{P}\, Action \\
\hline
\mathsf{dom}\ autoactions = \{motivations\} \\
agentactions = autoactions\ motivations
\end{array}
$$

The state of an autonomous agent in an environment is defined as follows.

AutonomousAgentState

$$
\begin{array}{l}
AutonomousAgentPercepts \\
AutonomousAgentAct \\
AgentState
\end{array}
$$

As with objects and agents, we define which variables remain unchanged after a set of actions has been performed by that agent.

ΔAutonomousAgentState

$$
\begin{array}{l}
AutonomousAgentState \\
AutonomousAgentState' \\
\hline
capableof' = capableof \\
perceivingactions' = perceivingactions \\
canperceive' = canperceive \\
autowillperceive' = autowillperceive \\
autoactions' = autoactions
\end{array}
$$

Now we specify the operation of an autonomous agent performing its next set of actions. Notice that while no explicit mention is made of any change in motivations, they may change in response to changes in the environment. If they do change, then the agent functions *willperceive* and *agentactions* will also change. Further, motivations may generate new and different goals for the agent to pursue. In any of these cases, the characterizing features of an agent are in flux so that an autonomous agent can be regarded as a continually re-instantiated non-autonomous agent. In this sense, autonomous agents are permanent as opposed to transient non-autonomous agents (which may revert to being objects).

AutonomousAgentEnvInteract

$$
\begin{array}{l}
\Delta AutonomousAgentState \\
\hline
history' = history \frown \langle willdo \rangle \\
environment' = \\
\qquad effectinteraction\ environment\ willdo \\
willperceive' = autowillperceive\ motivations' \\
agentactions' = autoactions\ motivations'
\end{array}
$$

Discussion

There exists a small body of work that provides a similar view to that presented here. For example, Covrigaru and Lindsay describe a set of properties that *characterize* autonomous systems to some "degree", relating to such factors as type and number of goals, complexity, interaction, robustness, and so on (Covrigaru & Lindsay 1991). In contrast, we *define* what is necessary for a system to be autonomous in very precise terms, and we distinguish clearly between objectness, agency and autonomy. One particular consequence of the difference in views is that we allow a rock, for example, to be considered an agent *if* it is being used for some purpose, such as a hammer for tent-pegs. Covrigaru and Lindsay deny the rock the quality of autonomy because it is not goal-directed, but ignore the possibility of agency, skipping over an important part of our framework. Other work includes that by Tokoro who offers a related view in which he distinguishes objects, concurrent objects, autonomous agents and volitional agents, similar in spirit to our own view (Tokoro 1993). In addition, Castelfranchi also characterizes autonomy through the use of motivation (Castelfranchi 1994). Our work differs in that we take autonomy to be an absolute concept which is constant regardless of the context in which it occurs. It either exists or it does not.

The notion of motivation is not new, and has been used elsewhere. Simon, for example, takes motivation to be "that which controls attention at any given time," and explores the relation of motivation to information-processing behaviour, but from a cognitive perspective (Simon 1979). More recently, Sloman has elaborated on Simon's work, showing how motivations are relevant to emotions and the development of a computational theory of mind (Sloman & Croucher 1981; Sloman 1987). Others have used motivation and related notions in developing computational architectures for autonomous agents such as *motives* (Norman & Long 1994), and *concerns* (Moffat & Frijda 1994). What is new about our work is the role of motivation in defining autonomy.

Conclusions

In the previous sections, we have constructed a formal specification which identifies and characterizes those entities that are called agents and autonomous agents. The work is not based on any existing classifications or notions because there is no consensus. Recent papers define agents in wildly different ways if at all, and this makes it extremely difficult to be explicit about their nature and functionality. The taxonomy given here provides clear and precise definitions for objects, agents and autonomous agents that explicates those factors that are necessary for agency and autonomy. In addition, our formalization of these concepts is written in the well-known language Z, which, through being based on elementary set theory, is accessible to a wide audience. Most usefully, perhaps, the specification is constructed in such a way as to allow further levels of specification to be added to describe particular agent designs and architectures.

The framework provides an important basis for reference. We can classify both human and artificial agents equally well. Consider the relationship of a programmer to a program. Programs are always designed to satisfy goals, but these goals are rarely explicit or able to be modified independently of the programmer. The programs lack goal-generating motivations, but can be ascribed goals. In this respect, they are agents. Programmers typically develop programs according to several motivations which determine how the program is constructed. Time and effort must be balanced against cost, ease of use, simplicity, functionality and other factors. Programmers consider these factors in determining the design of programs and in the goal or goals of programs. Programmers can change the goals of programs by modifying code if desired, and can modify their own goals to suit circumstances. In this respect, programmers are autonomous agents.

The difference between these kinds of programs as agents and much recent use of the term is that the relationship between the user (or programmer) and the program has become explicit. Software agents assist users. They adopt the goals of the users in the tasks that they perform. Whether or not they are autonomous depends on the ability of the agents to function independently of those users, and to modify their goals in relation to circumstances. This paper has made explicit the relationships that have previously been implicit across the board in the vast range of work on agency and autonomy, and provides a strong formal base on which to build.

One aspect not addressed here, however, is exactly *how* goals are generated from motivations. This is a difficult issue, but work is progressing on developing goal-generation mechanisms including efforts described above. The next stage of this work is to formalize a model of goal generation in the same way as the framework has been constructed here, developing a complete architecture for autonomous agents.

Acknowledgements

Thanks to John Campbell, Jennifer Goodwin, Paul Howells, Colin Myers, Mark Priestley, John Wolstencroft and the anonymous referees for comments and discussions on earlier drafts of this paper.

References

Castelfranchi, C. 1994. Guarantees for autonomy in cognitive agent architecture. In *Proceedings of the 1994 Workshop on Agent Theories, Architectures, and Languages.*

Covrigaru, A., and Lindsay, R. 1991. Deterministic autonomous systems. *AI Magazine* 12(3):110–117.

Goodwin, R. 1993. Formalizing properties of agents. Technical Report CMU-CS-93-159, Carnegie-Mellon University.

Kunda, Z. 1990. The case for motivated reasoning. *Psychological Bulletin* 108(3):480–498.

Moffat, D., and Frijda, N. H. 1994. An agent architecture: Will. In *Proceedings of the 1994 Workshop on Agent Theories, Architectures, and Languages.*

Norman, T. J., and Long, D. 1994. A proposal for goal creation in motivated agents. In *Proceedings of the 1994 Workshop on Agent Theories, Architectures, and Languages.*

Riecken, D. 1994. An architecture of integrated agents. *Communications of the ACM* 37(7):107–116.

Selker, T. 1994. A teaching agent that learns. *Communications of the ACM* 37(7):92–99.

Shoham, Y. 1993. Agent-oriented programming. *Artificial Intelligence* 60:51–92.

Simon, H. A. 1979. Motivational and emotional controls of cognition. In *Models of Thought.* Yale University Press. 29–38.

Sloman, A., and Croucher, M. 1981. Why robots will have emotions. In *Proceedings of the Seventh International Joint Conference on Artificial Intelligence*, 197–202.

Sloman, A. 1987. Motives, mechanisms, and emotions. *Cognition and Emotion* 1(3):217–233.

Smith, D. C.; Cypher, A.; and Spohrer, J. 1994. Programming agents without a programming language. *Communications of the ACM* 37(7):55–67.

Spivey, J. M. 1992. *The Z Notation.* Hemel Hempstead: Prentice Hall, 2nd edition.

Tokoro, M. 1993. The society of objects. Technical Report SCSL-TR-93-018, Sony CSL.

Wooldridge, M. J., and Jennings, N. R. 1994. Agent theories, architectures, and languages: A survey. In *Proceedings of the 1994 Workshop on Agent Theories, Architectures, and Languages.*

Wooldridge, M. J., and Jennings, N. R. 1995. Intelligent agents: Theory and practice. *Knowledge Engineering Review* 10(2).