1. The Agent Landscape

1.1 Introduction

Over the last decade or so, the notions underlying agent-based systems have become almost commonplace, yet were virtually unknown in earlier years. Not only have agent-based systems moved into the mainstream, they have spread beyond a niche area of interest in artificial intelligence, and have come to be a significant and generic computing technology. The dramatic and sustained growth of interest is demonstrated by the increasing number of major conferences and workshops in this very dynamic field, covering a depth and breadth of research that testifies to an impressive level of maturity for such a relatively young area.

As described in [122], an agent is a computer system capable of flexible autonomous action in a dynamic, unpredictable and open environment. Increasingly, agent technologies are being seen as playing a crucial role in application domains such as *Ambient Intelligence*, the seamless delivery of ubiquitous computing, continuous communications and intelligent user interfaces to consumer and industrial devices; *Grid Computing*, where multi-agent system approaches will enable efficient use of the resources of high-performance computing infrastructure in science, engineering, medical and commercial applications; *eBusiness*, where agent-based approaches are already supporting the automation and semi-automation of information-gathering activities and purchase transactions over the Internet; and the *Semantic Web*, where agents are needed both to provide services, and to make best use of the resources available, often in cooperation with others.

Agents offer an abstraction tool, or metaphor, for the design and construction of complex systems with multiple distinct and independent components. These abstractions can be used in the design and development of large systems, of individual agents, of ways in which agents may interact to support these concepts, and in the consideration of societal or macro-level issues such as organisations and their computational counterparts. They also enable the aggregation of different functionalities that have previously been distinct (such as planning, learning, coordination, and so on) in a conceptually embodied and situated whole.

Some of the reasons for the growth in popularity of the field (apart from the obvious intuitive appeal of the agent metaphor) can be seen in the progress made in complementary technologies [111], of which perhaps the most dramatic has been the emergence of the World Wide Web. The distribution of information and associated technologies lend themselves almost ideally to use by, in and for multi-agent

systems, while the problems that arise as a consequence suggest no solution quite as much as agents. The dual aspect of this interaction with the World Wide Web has thus been a major driving force. Other contributing factors include advances in distributed object technology that have provided an infrastructure without which the development of large-scale agent systems would become much more difficult and less effective. For example, the CORBA distributed computing platform [138] and the more recent Jini networking infrastructure [2], to handle low-level interoperation of heterogeneous distributed components, are both valuable technologies that can underpin the development of agent systems without the need for re-invention of fundamental techniques.

The contradiction of agent-based systems is that there is still an effort to provide a sound conceptual foundation despite the onward march of applications development. Indeed, while there are still disagreements over the nature of agents themselves, significant commercial and industrial research and development efforts have been underway for some time [22, 34, 140, 141], and are set to grow further.

A recurrent theme that is raised in one form or another at many agent conferences and workshops is the lack of agreement over what it is that actually constitutes an agent. It is difficult to know if this is a help or hindrance, but the truth is that it is probably both. On the one hand, the immediately engaging concepts and images that spring to mind when the term is mentioned are a prime reason for the popularisation of agent systems in the broader (and even public) community, and for the extremely rapid growth and development of the field. Indeed the elasticity in terminology and definition of agent concepts has led to the adoption of common terms for a broad range of research activity, providing an inclusive and encompassing set of interacting and cross-fertilising sub-fields. This is partly responsible for the richness of the area, and for the variety of approaches and applications. On the other hand, however, the lack of a common understanding leads to difficulties in communication, a lack of precision (and sometimes even confusion) in nomenclature, vast overuse and abuse of the terminology, and a proliferation of systems adopting the agent label without obvious justification for doing so. The discussion is valuable and important, for without a common language, there can be significant barriers to solid progress, but it is problematic to find a way to converge on such a language without constraining or excluding areas in the current spectrum of activity.

This book seeks to address the aforementioned problems by providing a sound conceptual framework with which to understand and organise the landscape of agent-based systems. The approach is not to constrain the use of terminology through rigid definition, but to provide an encompassing infrastructure that may be used to understand the nature of different systems. The benefit of this is that the richness of the agent metaphor is preserved throughout its diverse uses, while the distinct identities of different perspectives are highlighted and used to direct and focus research and development according to the particular objectives of a sub-area.

1.2 Agents

1.2.1 Terminology

In artificial intelligence, the introduction of the notion of agents is partly due to the difficulties that have arisen when attempting to solve problems without regard to a real external environment or to the entity involved in that problem-solving process. Thus, though the solutions constructed to address these problems are in themselves important, they can be limited and inflexible in not coping well in real-world situations. In response, agents have been proposed as *situated* and *embodied* problem-solvers that are capable of functioning effectively and efficiently in complex environments. This means that the agent receives input from its environment through some sensory device, and acts so as to affect that environment in some way through effectors. Such a simple but powerful concept has been adopted with remarkable speed and vigour by many branches of computer science because of its usefulness and broad applicability.

Indeed, there is now a plethora of different labels for agents ranging from the generic *autonomous agents* [97], *software agents* [73], and *intelligent agents* [183] to the more specific *interface agents* [106], *virtual agents* [7], *mobile agents* [24, 178], *information agents* [104], and so on. The diverse range of applications for which agents are being touted include operating systems interfaces [61], processing satellite imaging data [169], electricity distribution management [96], air-traffic control [100] business process management [94], electronic commerce [82] and computer games [77], to name a few.

The richness of the agent metaphor that leads to such different uses of the term is both a strength and a weakness. Its strength lies in the fact that it can be applied in very many different ways in many situations for different purposes. The weakness, however, is that the term *agent* is now used so frequently that there is no commonly accepted notion of what it is that constitutes an agent. Given the range of areas in which the notions and terms are applied, this lack of consensus over meaning is not surprising. As Shoham [153] points out, the number of diverse uses of the term *agent* are so many that it is almost meaningless without reference to a particular concept of agent. Similarly, Connah and Wavish [27] state that the term agent has "almost as many meanings as there are instances of its use" and that this causes "considerable confusion".

That there is no agreement on what it is that makes something an agent is now generally recognised, and it is standard, therefore, for many researchers to provide their own definition. In a relatively early collection of papers, for example, several different views emerge. Smith [159] takes an agent to be a "persistent software entity dedicated to a specific purpose". Selker [152] views agents as "computer programs that simulate a human relationship by doing something that another person could do for you". More loosely, Riecken [145] refers to "integrated reasoning processes" as agents. Others take agents to be computer programs that behave in a manner analogous to human agents, such as travel agents or insurance agents [62] or software entities capable of autonomous goal-oriented behaviour in a heterogeneous comput-

ing environment [84], while some avoid the issue completely and leave the interpretation of their agents to the reader. Many such other agent definitions can be found in the excellent review by Franklin and Graesser [67], in advance of proposing their own definition.

Typically, however, agents are *characterised* along certain dimensions, rather than defined precisely. For example, in the now foundational survey of the field by Wooldridge and Jennings [183], a *weak notion* of agency is identified that involves *autonomy* or the ability to function without intervention, *social ability* by which agents interact with other agents, *reactivity* allowing agents to perceive and respond to a changing environment, and *pro-activeness* through which agents behave in a goal-directed fashion. To some extent, these characteristics are broadly accepted by many as representative of the key qualities that can be used to assess '*agentness*'.

Wooldridge and Jennings also describe a *strong notion* of agency, prevalent in AI which, in addition to the *weak* notion, also uses mental components such as belief, desire, intention, knowledge and so on. Similarly, Etzioni and Weld [62] summarise desirable agent characteristics as including *autonomy*, *temporal continuity* by which agents are not simply 'one-shot' computations, believable *personality* in order to facilitate effective interaction, *communication ability* with other agents or people, *adaptability* to user-preferences and *mobility* which allows agents to be transported across different machines and architectures. They further characterise the first of these, autonomy, as requiring that agents are *goal-oriented* and accept high-level requests, *collaborative* in that they can modify these requests and clarify them, *flexible* in not having hard, scripted actions, and *self-starting* in that they can sense changes and decide when to take action. Other characteristics are often considered, both implicitly and explicitly, with regard to notions of agency including, for example, *veracity*, *benevolence* and *rationality*.

Krogh [102], for example, notes that there have been many attempts to find one central common denominator and, with a certain amount of pessimism, predicts that any such attempts will fail. However, he does comment that these definitions are technically useful even though they are usually flawed. As an example, he cites the definition of software agents by Genesereth and Ketchpel [73].

> "An entity is a software agent if and only if it communicates correctly in an agent communication language such as ACL".

Krogh argues that this definition is inappropriate for the following reasons.

- If it does not communicate 'correctly' then it is not an agent.
- If it does not communicate at all then it is not appropriate to ascribe agenthood to an entity.

Instead, Krogh argues that there are many situations in which we would wish to ascribe agenthood to entities that cannot communicate correctly or cannot communicate at all. Without fully elaborating, Krogh further suggests that in some cases it is appropriate to consider entities that are not computer programs as agents.

Recognising that there is no commonly accepted definition of what constitutes an agent, Krogh chooses to *delineate* a class of agents that have certain *dimensions*

as described above. In particular, these are independent, selfish, interacting, heterogeneous and persistent. However, the terms are not defining in themselves and can introduce even more ambiguity since, as stated earlier, the meanings attributed to these dimensions are not themselves uniform.

1.2.2 Problems with Definition

Wooldridge and Jennings recognise that many such qualities have been proposed by others as being necessary for agenthood but, in a joint paper with Sycara [95], suggest that the four characteristics enumerated in their *weak notion* above are the "essence" of agenthood. Despite some broad acceptance of this view, there are still many problems. For example, in a more recent paper, Müller [133] seeks to survey *autonomous* agent architectures by considering the three strands of *reactive* agents, *deliberative* (or pro-active) agents and *interacting* (or social) agents. The properties here correspond perfectly to three of these four key characteristics, but instead of being used to represent all agents, they are used to break down the classes of agents into three distinct streams of research.

The difficulty with this approach of *characterising* agents through identifying their properties is exemplified by considering *mobile agents* [24, 178], which are quite distinct and identifiable in the focus on movement of code between host machines. Here, the key characteristic is precisely this mobility, and indeed mobility has been regarded by some as an intrinsic agent property. A critical analysis of the area of mobile agents would, however, unearth a recognition that this mobility augments other, more central agent characteristics in mobile agents, so that mobility is valuable in identifying the kind of agent, rather than understanding all agents. Similarly, some of the more specific labels for agents describe other characteristics that do not impact on agents as a whole, but relate to a particular domain or capability.

This area is fraught with difficulty, yet there have been several efforts to address these issues. For example, in attempting to distinguish agents from programs, Franklin and Graesser constructed an agent taxonomy [67] aimed at identifying the key features of agent systems in relation to different branches of the field. Their aim, amply described by the title of the paper, "Is it an agent or just a program?", highlights what might be regarded as the problem of the *Emperor's clothes*, as to whether there is any value to the notion of agents. The definition provided, that an "autonomous agent is a system situated within and a part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to affect what it sense in the future," serves to distinguish some non-agent programs from agents through the introduction of *temporal continuity*, for example, but still suffers from simply providing a *characterisation*. Using this, Franklin and Graesser then move to classify existing notions of agents within a taxonomic hierarchy. While interesting and valuable, it still does not provide a solution to the problem of identifying agentness. As Petrie points out, for example, *autonomy* remains *unelaborated*, yet it is a key part of the definition [143].

In somewhat similar fashion, Müller [133] also provides a taxonomy of intelligent agents that reflects different application areas, and which can be used to iden-

tify classes of agent architectures that are suited to particular problems. While this is also a valuable aid to understanding the range of work done in this area, it does not help in clarifying the issues discussed above. Nwana [137], too, offers an interesting typology of agent systems in his review of the field, but importantly warns against the dangers associated with the "rampant" use of the agent buzzword, its overselling and the possibility of it becoming a "passing fad" as a result.

Thus, while agent properties illustrate the range and diversity both of the design and potential application of agents, such a discussion is inadequate for a more detailed and precise analysis of the basic underlying concepts. If we are to be able to make sense of this rapidly growing area, then we need to progress beyond a vague appreciation of the nature of agents. Indeed, as Wooldridge argues [182], to avoid the term 'agent' becoming meaningless and empty and attached to everything, only those systems that merit the agent label should have it.

To summarise, there is a distinct lack of precision and consensus in work dealing with agents and their dimensions. Consequently, there is no common currency for the notion of agenthood, or indeed for dimensions of agency. For example, the notions of reflexive and reactive agents are often confused. This is not an isolated case: terms are often used interchangeably without real regard for their significance and relevance. Another example, of this kind, is that agency is often taken to imply some degree of autonomy and the two terms are often used interchangeably [67].

This book adopts the stance that agency and autonomy relate to very specific and distinct, though related, qualities. It offers a precise understanding of these terms and the relationship between them, both of which are of fundamental importance in defining the nature of agent-based systems.

1.3 Multi-Agent Systems

Now, multi-agent systems are typically distributed systems in which several distinct components, each of which is an independent problem-solving agent come together to form some coherent whole. There is generally no pre-established architecture or configuration incorporating the agents, and the interactions between them are not pre-defined, as is usually the case with traditional processes in concurrent programs. More importantly, there is no global system goal, the agents being heterogeneous with their own goals and capabilities. In consequence, agents in a multi-agent system need to *coordinate* their activities and cooperate with each other, in order to avoid duplication of effort, to avoid unwittingly hindering other agents in achieving goals, and to exploit other agents' capabilities. These basic points motivate the consideration of several additional issues regarding agents that don't arise when considering individual agents in isolation. Each is outlined next.

Agent Modelling
As discussed, an agent may need a *model* of its world. If this world contains agents then it may be beneficial to model these other agents, too.

Multi-Agent Planning
In some cases agents will share plans in order to coordinate their behaviour or to achieve a goal using others.

Social Relationships
Agents may have *social relationships* with other agents. For example, if one agent has performed a service for another agent, the second agent may be under an *obligation* to reciprocate in some way. If two agents are working together to achieve a task, the agents are typically said to be *cooperating*.

Interaction
Agents may *interact*. In a multi-agent world in which interaction is not pre-defined, agents may need models of each other to decide how to interact, and to decide on their success or failure. This may impact in different ways on the social relationships between the agents.

Communication
Agents may *communicate* to exploit interaction and ensure coordination. An agent may persuade others to adopt its goals and alter their plans [15].

The same problems regarding the undefined nature of agents discussed earlier arise in multi-agent systems. It is difficult to consider these issues in any structured or principled way without agreement on the basic components that are involved. In order to understand fully the issues introduced above, it is first necessary to understand the nature of individual agents themselves.

1.4 Desiderata for a Conceptual View of Agents

Agent systems that can act independently in complex environments are very appealing and, judging by the recent effort invested in their development and application, are here to stay. However, single-agent systems are fundamentally limited by their own particular dimensions. Individual agents in multi-agent systems, by contrast, can exploit the capabilities of other agents, allowing for a much greater range of collective functionality than an isolated agent. Multi-agent systems, however, are more complex in design and construction since this increased functionality arises through the interaction of the individual agents involved. Furthermore, in many cases, these agents are autonomous and the interaction emerges in a fashion that is essentially unplanned. An understanding of the way in which such systems operate can only be achieved through an analysis of the relationships that arise between agents, and any pre-existing architectural structure.

This book provides a detailed examination and analysis of the general social organisation of multi-agent systems and develops models of dimensions (or capa-

bilities) of agents that need to function effectively and efficiently within them. A key aspect of this analysis is that it must be directed at the individual interacting agents themselves. More specifically, this book has the following salient concerns.

- To provide principled *definitions* for agency and autonomy, and to explicate the nature of the relationship between them. The distinction between agency and autonomy is critical to understanding the nature of the relationships in multi-agent systems.
- To provide a *unifying framework* that incorporates these definitions within which existing work and definitions can be situated. This is achieved by constructing high-level models of agents and autonomous agents and their operation in a way that is not architecture-specific. The framework should serve as a foundation both for the development of agent systems and for analysing agent relationships.
- To *analyse* and define key relationships that arise between agents. These relationships are universal and fundamental, arising as a natural consequence of the definitions of agency and autonomy.
- To build models of the *dimensions* of *deliberative* agents that are required for them to recognise and exploit social relationships in order that they may interact effectively in multi-agent systems.
- To demonstrate the practical applicability of the models constructed to existing theories and systems so that they may be readily analysed and evaluated.

In satisfying these aims, this book adheres to four main principles. First, it is concerned with the development of a theory that subsumes, as far as possible, existing work, so that it is useful and widely applicable. It is not our intention to produce yet another set of definitions that do not relate to any previous attempts. Second, a major criticism of many existing definitions and concepts is that they are vague and ambiguous. In order to avoid such problems, we must ensure that we are precise and unambiguous at all times. The book therefore uses formal methods to provide a rigorous and precise underpinning to the work in this book. Third, abstraction in analysis and design is an important tool because it enables an appropriate level of description to be chosen. Consequently, the book provides a means of moving between different levels of agent specification from the most abstract to the least abstract, from primitive definitional specifications through deliberative agent dimensions to instances of systems and theories. Finally, agent theories should serve as specifications [184]. This book aims to provide an explicit design and specification environment for the development of real systems; the formal models will relate directly to computational systems.

1.5 A Formal Framework for Agent Definition and Development

1.5.1 Formal Frameworks

To address the concerns identified above in relation to the agent field, it is sensible for a well-defined and precise vocabulary for the fundamental elements of agents

and multi-agent systems to be developed. If such a vocabulary is also situated in a structured framework, it can provide the right kind of platform on which to base further research. Formal specification techniques are appropriate for this task.

It has been claimed elsewhere that formal specification can be used to construct *formal frameworks* within which common properties of a family of systems can be identified [57, 69–71]. As a result of such specifications, it becomes possible to consider different systems as instances of one design, and to show how new designs can be constructed from an existing design framework.

More precisely, a formal framework must satisfy three distinct requirements, as follows.

- It must provide meanings for common concepts and terms precisely and unambiguously, and do so in a readable and understandable manner. The availability of readable explicit notations allows a movement from a vague and conflicting understanding of a class of models towards a common conceptual framework. A common conceptual framework exists if there is a generally held understanding of the salient features and issues involved in the relevant class of models.
- It must be sufficiently well-structured to provide a foundation for subsequent development of new and increasingly more refined concepts. In particular, it is important that a practitioner is in a position to choose the level of abstraction suitable for their current purpose.
- It must enable alternative designs of particular models and systems to be presented explicitly, compared and evaluated. It must provide a description of the common abstractions found within that class of models as well as a means of further refining these descriptions to detail particular models and systems.

Over the course of this book, a principled theory of agency is developed by describing just such a framework, called the SMART *agent framework*. Using the Z specification language, a sophisticated model of agents and their relationships is built up and illustrated with application to three distinct case-studies.

1.5.2 Notation

There is a large and growing number of formal techniques and languages available to specify properties of software systems [42]. These include state-based languages such as VDM [98], Z [164] and B [105], process-based languages such as CCS [128] and CSP [88], temporal logics [60], modal logics [23] and statecharts [174], with each technique having its advocates for use in modelling various aspects of computing.

This book adopts the language Z, deliberately selecting a technique which not only enables designs of systems to be developed formally, but allows for the systematic refinement of these specifications to implementations. The choice of Z is a direct response to (arguably) the most problematic aspect of many formal techniques for agent-based computing — that they do not directly relate to the construction of agent software.

Furthermore, Z is a specification language that is increasingly being used both in industry and academia, as a strong and elegant means of formal specification, and is supported by a large array of books (e.g. [10, 83]), articles (e.g. [11, 12]) industrial case studies (e.g. [26, 36, 176]), well-documented refinement methods (e.g. [179]), and available tools for animation (e.g. [85, 149, 163]).

Additionally, Z has other benefits.

- It is more *accessible* than many other formalisms since it is based on existing *elementary* components such as set theory and first order predicate calculus. (A summary of the notation is provided in Table 1.1, but a more detailed tutorial introduction is provided in the Appendix.)
- It is an extremely expressive language, allowing a consistent, unified and structured account of a computer system and its associated operations.
- It is gaining increasing acceptance as a tool within the artificial intelligence community (e.g. [35, 76, 129, 181]) and is therefore appropriate for the current work in terms of standards and dissemination capabilities.

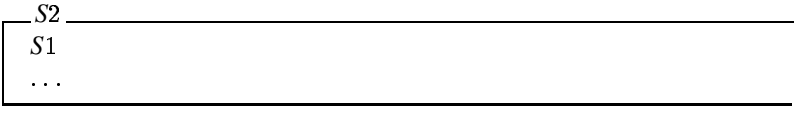
1.5.3 Specification Structure Diagrams

Over the course of this book, the formal description of the framework is presented in Z, but the structure of the specification is also presented graphically. In particular, diagrams are used to detail the way in which schemas are used to produce a *specification structure*, which provides a graphical overview of the way in which the formal models in this book are constructed. The key to these diagrams is presented in Figure 1.1 and is explained below.

State Schema
State schemas are represented by a box enclosing the schema name.

Operation Schema
Operation schemas are represented by a hexagon enclosing the schema name.

Schema Inclusion
A solid arrow between boxes represents state schema inclusion. In the case shown in Figure 1.1, $S1$ is included in $S2$.



Variable Inclusion
A dashed arrow between boxes represents a schema being included in the declarative part of the second schema as a type. In the case shown in Figure 1.1 a variable included in the state schema $S2$ is declared in terms of the type defined by state schema $S1$. For example, the schema below includes a variable that is defined as a set of elements of the schema type $S1$.

Table 1.1. Summary of Z Notation

Definitions and declarations

a, b	Identifiers
p, q	Predicates
s, t	Sequences
x, y	Expressions
A, B	Sets
$a == x$	Abbreviated definition
$[a]$	Introduction of given set
$a ::= b\langle\langle B \rangle\rangle \mid c\langle\langle C \rangle\rangle$	Free type declaration
$\mu\, d \mid P$	Definite description

Logic

$\neg p$	Logical negation
$p \wedge q$	Logical conjunction
$p \vee q$	Logical disjunction
$p \Rightarrow q$	Logical implication
$p \Leftrightarrow q$	Logical equivalence
$\forall X \bullet q$	Universal quantification
$\exists X \bullet q$	Existential quantification

Sets

$x \in y$	Set membership
$x \notin y$	Non-membership
$\{\ \}$	Empty set
$A \subseteq B$	Set inclusion
$A \subset B$	Strict set inclusion
$\{x, y, \ldots\}$	Set of elements
$(x, y, \ldots)$	Ordered tuple
$A \times B \times \ldots$	Cartesian product
$\mathbb{P} A$	Power set
$\mathbb{P}_1 A$	Non-empty power set
$A \cap B$	Set intersection
$A \cup B$	Set union
$A \setminus B$	Set difference
$\bigcup A$	Generalised union
$\bigcap A$	Generalised intersection
$\#A$	Size of finite set

Relations

$A \leftrightarrow B$	Relation
$\mathrm{dom}\, R$	Domain of relation
$\mathrm{ran}\, R$	Range of relation
$R^{\sim}$	Inverse of relation
$R \rhd A$	Range restriction
$A \lhd R$	Anti-domain restriction
$R^{\sim}$	Relational Inverse
R^{+}	Transitive Closure
$R(\!\mid A \mid\!)$	Relational Image
$R_1 \oplus R_2$	Relational Overriding

Functions

$A \pfun B$	Partial function
$A \fun B$	Total function
$A \surj B$	Total Surjection
$A \pinj B$	Partial Injection
$A \bij B$	Bijection

Sequences

$\langle x, y, \ldots \rangle$	Sequence
$\mathrm{seq}\, A$	Finite sequences
$\mathrm{seq}_1\, A$	Non-empty seqs
$\mathrm{iseq}\, A$	Injective seqs
$\mathrm{iseq}_1\, A$	Non-empty inj seqs
$s \frown t$	Concatenation
$head\ s$	First element of seq
$last\ s$	Last element of seq
$s \mathrm{in}\, t$	Subsequence

Schema notation

S d p	Vertical schema
d p	Axiomatic definition
S T d p	Schema inclusion
ΔS S S'	Operation schema
$z.a$	Component inclusion

Conventions

$a?$	Input to an operation
$a!$	Output from an op
a	Variable before op
a'	Variable after op
S	Schema before op
S'	Schema after op
ΔS	Change of state
ΞS	No change of state

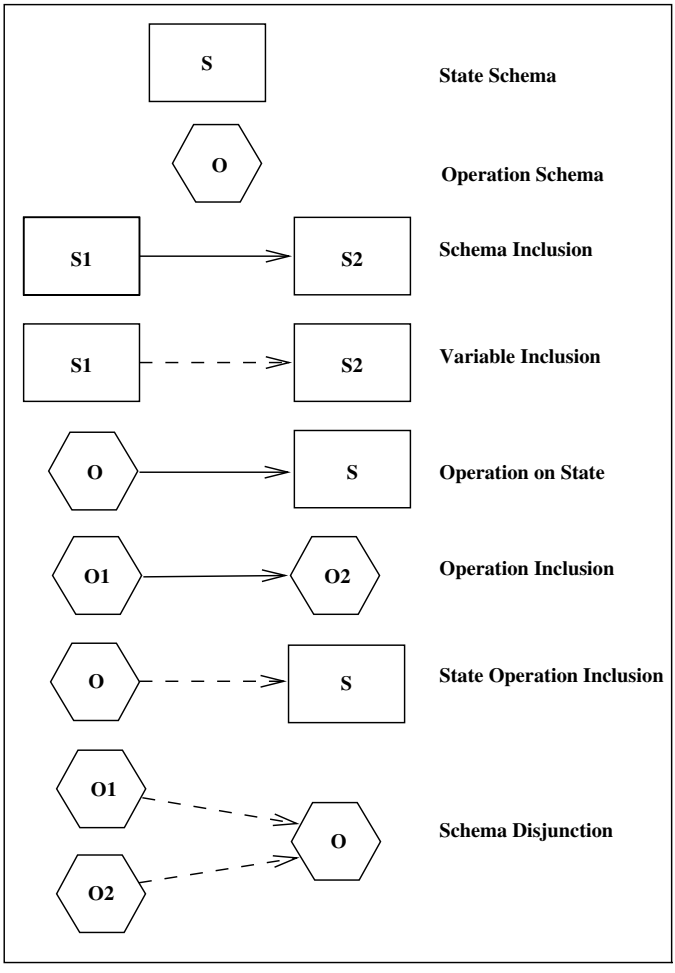
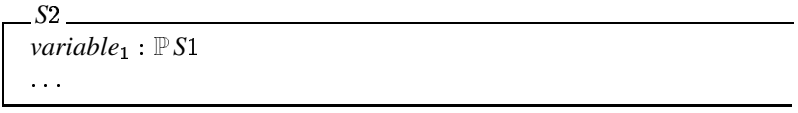
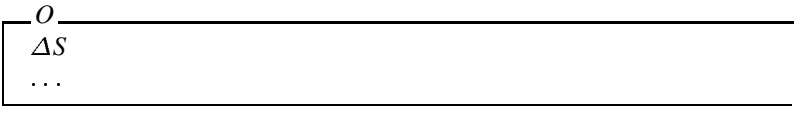


Fig. 1.1. Structuring Schemas



Operation on State
A solid arrow between a hexagon, O, and a box, S, indicates that the operation schema, O, is defined in terms of a state change to the state schema, S.



Operation Inclusion
A solid arrow between two hexagons represents operation inclusion. In the case shown in Figure 1.1 the operation schema $O2$ includes the operation schema $O1$.

State Operation Inclusion
A dashed arrow between a hexagon and a box indicates that the state schema has been included in the definition of an operation schema. In the case shown in Figure 1.1 the state schema S is included but its state is unaffected.

Schema Disjunction
A set of converging dashed arrows between a set of hexagons $O1$, $O2$ and another hexagon O indicates that the operation schema O is defined as the disjunction of the operation schemas $O1$ and $O2$. The pre-condition of O is the logical disjunction of the preconditions of $O1$ and $O2$.

In this book, the SMART agent framework is developed mathematically using Z, but it is also completely described in the accompanying text. The specification structure diagrams described above are also used throughout the book to illustrate how the formal description is organised. They serve as a reference point for all of the schemas that comprise the SMART framework. All of the concepts are introduced intuitively and informally before proceeding to a formal specification, so that the mathematically naive reader will also benefit from the book. Nevertheless, a tutorial introduction to Z is provided in the Appendix, and readers unfamiliar with the notation (which is summarised here in Table 1.1) may choose to consult it before proceeding. Alternatively, the texts mentioned above may also be valuable.

http://www.springer.com/978-3-540-40700-3

Understanding Agent Systems
d'Inverno, M.; Luck, M.
2004, XIX, 244 p., Hardcover
ISBN: 978-3-540-40700-3