



City Research Online

City, University of London Institutional Repository

Citation: Rodrigues, M. A. S. (1999). The development of Spatial Intelligent Agents With Geographic Information Systems. (Unpublished Doctoral thesis, City, University of London)

This is the accepted version of the paper.

This version of the publication may differ from the final published version.

Permanent repository link: <https://openaccess.city.ac.uk/id/eprint/30665/>

Link to published version:

Copyright: City Research Online aims to make research outputs of City, University of London available to a wider audience. Copyright and Moral Rights remain with the author(s) and/or copyright holders. URLs from City Research Online may be freely distributed and linked to.

Reuse: Copies of full items can be used for personal research or study, educational, or not-for-profit purposes without prior permission or charge. Provided that the authors, title and full bibliographic details are credited, a hyperlink and/or URL is given for the original metadata page and the content is not changed in any way.



CITY
University

*The development of Spatial Intelligent Agents
With Geographic Information Systems*

by
Maria Armanda Simenta Rodrigues

a Dissertation submitted in partial fulfilment
of the requirements for the
degree of Doctor of Philosophy

in
City University
Department of Information Science
London, UK

August, 1999

To my parents and Cédric

Table of Contents

TABLE OF CONTENTS	5
TABLE OF FIGURES	13
ACKNOWLEDGMENTS	17
ABSTRACT	20
CHAPTER 1-INTRODUCTION	21
The GISc research agenda and this thesis.....	22
Problem Definition	28
Objectives	31
Limitations	34
Thesis organisation	34
CHAPTER 2-LITERATURE REVIEW ON INTELLIGENT AGENTS AND SPATIAL ISSUES.....	37
2.1 Defining Agents	38
2.2 Agents in the Context of Computer Science	41
2.2.1 Artificial Intelligence	41
2.2.2 Expert Systems.....	42
2.2.3 Object-orientation.....	42
2.3 Origins of Agents	44
2.3.1 Distributed Artificial Intelligence	45
2.3.2 Complexity Handling.....	47
2.4 Implementation of Agents	52

2.4.1	Modelling agents	52
2.4.1.1	Philosophical issues	52
2.4.1.2	Models of Agency	53
2.4.1.2.1	Rational Agency : Logical perspective.....	53
2.4.1.2.2	Rational Agency: Economic perspective.....	54
2.4.1.2.3	Social Agency.....	55
2.4.1.2.4	Interactive Agency.....	56
2.4.1.2.5	Adaptive Agency.....	56
2.4.2	Agent Architectures.....	57
2.4.2.1	Classical Approaches: Deliberative Architectures	57
2.4.2.2	Alternative Approaches: Reactive Architectures.....	59
2.4.2.3	Hybrid Architectures.....	61
2.5	Agents and spatial issues.....	63
2.5.1	Fundamental spatial concepts.....	64
2.5.1.1	Euclidean Space	66
2.5.1.2	Set-based geometry of space.....	68
2.5.1.3	Topological spaces	69
2.5.1.4	Network spaces.....	71
2.5.2	The Geographic Individual.....	72
2.5.3	Computational methods for representing geographic concepts.....	74
2.5.3.1	Spatial Data Modelling.....	75
2.5.3.2	Spatial Data Types.....	77
2.5.3.3	Spatial Access Methods.....	78
2.5.4	Agents in spatial problems	79
2.5.4.1	Spatial Simulation	79
2.5.4.1.1	Individual-based Models.....	81
2.5.4.2	Spatial Decision Making.....	82
2.5.4.3	Interface agents in GIS.....	84
2.6	Discussion.....	84

CHAPTER 3-INTERFACE AGENTS FOR SPATIAL KNOWLEDGE

HANDLING 91

3.1 Approaches to building interface agents 93

3.1.1 End-user programming approach 94

3.1.2 The knowledge-based approach 95

3.1.3 The machine learning approach 95

3.1.3.1 Acquiring competence 97

3.1.3.2 Examples of learning interface agents 99

3.1.4 Questioning interface agents 101

3.2 Interface agents that assist users performing geographic tasks 103

3.2.1 Previous work 104

3.2.2 Memory-based framework for developing spatial interface agents 106

3.2.2.1 Analysis of the spatial task 107

3.2.2.2 User Model 109

3.2.2.3 Design of a spatial interface agent 110

3.2.2.3.1 Memory-based reasoning 112

The overlap metric 113

Weighted feature metric 114

Value differences 115

Restricting the Database 116

Deciding (or Acting) 116

3.2.2.3.2 Spatial issues 117

3.3 Discussion 118

CHAPTER 4-ADAPTIVE AGENTS FOR SPATIAL SIMULATION .. 121

4.1 Simulation 122

4.1.1 Theory of simulation 124

Objects	125
Set of models of the subsystems	125
Update functional	125
Simulation	126
Implementation	126
4.1.1.1 Emergence.....	126
Examples.....	127
4.1.1.2 Simulation and Emergence	128
Examples.....	130
Studying Emergence of S^2 at L^2	130
4.2 Traditional simulation techniques and their limitations	131
4.3 Individual structures in simulation	133
4.3.1 Reactive agents for simulation – characteristics of reactive agents	134
Cognitive cost and cognitive economy	134
Situation.....	134
Self-sufficiency.....	135
Robustness and fault tolerance	135
Flexibility and adaptability	135
4.3.1.1 Feedback.....	135
4.3.2 Multi-agent simulation.....	136
4.4 Spatial Simulation.....	138
4.5 Framework for developing adaptive agent-based spatial simulations ..	141
4.5.1 Learning	143
4.5.1.1 Reinforcement learning	144
An active learner	146
Using Q-learning.....	146
Updating the Q matrix.....	147
Exploration function	148

4.5.2	Analysis of the spatial simulation	149
4.5.3	Spatial agent design.....	150
4.6	Discussion	151

CHAPTER 5-FIRST CASE-STUDY: AN ASSISTANT FOR PRINTING AND PLOTTING FOR SMALLWORLD GIS 155

5.1	Smallworld GIS	156
5.2	The agent architecture	156
5.3	The drafting and plotting assistant.....	157
5.3.1	The drafting and plotting task	161
5.3.2	State Transitions.....	162
5.3.3	Suggestions and actions.....	164
5.4	Discussion	165

CHAPTER 6-CASE-STUDY: INTELLIGENT ASSISTANT FOR SPATIAL INFORMATION ACCESS..... 167

6.1	A Spatial Information Facilitator on the World Wide Web	171
6.1.1	Online Mapping with ESRI's Mapobjects and MapObjects Internet Map server	172
6.1.2	Using metadata to search for fitting information	173
6.1.3	Metadata map server.....	174
6.1.4	Client requests.....	175
6.2	Using Memory-based reasoning to personalise the use of the Spatial Information Facilitator	177
6.2.1	Memory-based information structure.....	179

6.2.1.1	Mapping between the methodology definitions and the Memory database structure	179
6.2.1.2	Memory implementation.....	182
6.2.1.3	Memory Manager's overnight update	187
6.2.1.4	Action Prediction.....	187
6.2.2	SIFIA – The Spatial Information Facilitator Interface Assistant	188
6.3	Recommendation example	190
6.4	User testing.....	193
6.4.1	User Profiles	193
	Public type users.....	195
	Urban Planner type users.....	195
	Central Administration type users	195
	Business Professional type users	195
6.4.2	Tests Results	196
6.4.3	Interpretation of results	200
6.5	Discussion.....	202
 CHAPTER 7-CAR PARK AGENT SIMULATION		205
7.1	Simulation description.....	206
7.1.1	Environment – car park.....	207
7.1.2	Agents – Cars.....	209
7.1.3	Passive Elements	210
	7.1.3.1 Parking spaces.....	211
	7.1.3.2 Entrances and Exits.....	214
7.1.4	Time and space	215
7.1.5	Changing the environment and interacting	216
7.1.6	Percepts.....	217

7.2	Learning	218
7.2.1	Q matrix	219
7.2.2	Spatial reward	220
7.3	Implementation issues	226
7.3.1	Running the simulation	228
7.3.1.1	Testing results	231
7.4	Emergent properties	235
7.5	Discussion	237
7.5.1	Increase of complexity	239
7.5.2	Conclusions from the implementation	240
 CHAPTER 8-CONCLUSIONS AND FURTHER DEVELOPMENTS		. 243
8.1	Evaluation of research area from literature review	245
8.1.1	Spatial Simulation	245
8.1.2	Spatial Decision Making.....	246
8.1.3	Interface agents for Geographic Information Systems (GIS)	246
8.2	Specific findings of the research	246
8.3	Key Conclusions	252
8.4	Further developments of the research	253
 REFERENCES AND BIBLIOGRAPHY		255
 APPENDIX A		257
 APPENDIX B		307
 APPENDIX C		343

APPENDIX D385

APPENDIX E421

TABLE OF FIGURES

Figure 1 - The agent's learning process (taken from Maes, 1994). The interface agent learns in four different ways: it observes and imitates the user's behaviour, it adapts based on user feedback, it can be trained by the user on the basis of examples and it can ask for advice from other agents assisting other users.	98
Figure 2 - The decomposition of the agent: the agent interface, the reasoning component, the memory and the memory manager.	111
Figure 3 - An agent structure in Russell and Norvig's framework	141
Figure 4 - Q-learning algorithm for an active learner agent with exploration function.....	149
Figure 5 - The agent controller user interface.....	159
Figure 6 - The drafting agent calls a specific HTML page that provides information on the current state of the drafting task.....	160
Figure 7 - The agent finishes the execution of the task on behalf of the user (once it has sufficient information to do it)	161
Figure 8 - Conceptual State Transition Diagram	164
Figure 9-MapObjects IMS Architecture	173
Figure 10 - The client interface-a java applet which communicates with the map server	176
Figure 11 - The Visual Basic functions that implement the metric for the spatial region. The region in ext1 is placed in the intersect structure before executing the intersection with ext2. The result of the intersection is placed in the same intersect structure	182

Figure 12 – Design of memory table MBR, which contains all of requests ever sent by a user to the metadata map server 183

Figure 13 – SQL query that generates the values in view `frequence_cmd` 184

Figure 14 – SQL query that generates the first version of the table `frequence_cmd_gcr`. This table has to go through further processes so that the existence of the frequencies of similar regions not identified at runtime may be evaluated 184

Figure 15 - Resulting `frequence_cmd_gcr` after process of identification of similar regions..... 185

Figure 16 – Example of one of the weights table. The weight of each predictor region is measured according to the goal field `cmd` 185

Figure 17 – Structure and values in `d_gcmd_cmd` table. The `d` value represents the comparison of two records in terms of the values of the predictor field `gcmd` and of the records’ values for the predictor field `cmd`..... 186

Figure 18 – Measure of distance between each record in the memory (after predictor restriction) and the last request made by the user, in terms of the predictor field `gcmd`. As can be seen, the assistant recommends the retrieval of information. 186

Figure 19 – The Implemented architecture: basis for SIFIA (The Spatial Information Facilitator Interface Assistant) 188

Figure 20 – The Interface Assistant Window 190

Figure 21 – In blue, examples of requests selected by the agent after the user has selected the region in red..... 191

Figure 22 – The highest rated regions after one spatial selection. Their spatial mapping is presented in Figure 21. 192

Figure 23 – Generation of information for recommendation..... 192

Figure 24- Metadata results for the example. This figure shows the display of the number of entities selected by the sum of relevant regions and by the region itself	193
Figure 25-Testing results. The tests evaluate the behaviour of 12 users identified with 3 group profiles.	199
Figure 26 – An example of the car park environment: here there is one marked entrance and one marked exit and 18 parking spaces. As a car reaches a parking space (like car 19) just before parking, it asks permission to the parking space. A red car is looking for a place to park, while a blue car is moving towards the exit, after parking.	208
Figure 27 – Parking function of the car agent. The car will only change its state if parking has been successful near the parking space.	213
Figure 28 – Parking function of the parking space. This is a synchronised method, to ensure that only one car tries to park at each parking space at a time.	213
Figure 29 – Distance function used in the car park simulation. This needs to be changed if the configuration and the method of movement changes in the environment.....	213
Figure 30 – Parking space function betterCar. This function analyses if there are any car agents better position to park in it. The result of this function will influence the reward function.	215
Figure 31 – Example of a Q matrix of a car agent that has learned very little and which has entered an initially full car park.	221
Figure 32 – Part of the reward function referring to agents that were previously moving towards an empty parking space.....	221
Figure 33 - Part of the reward function referring to agents that were previously moving towards an awaiting position (action=centre).	222

Figure 34 – Reward function algorithm..... 224

Figure 35 – part of the `max_action` function showing the implementation of exploration in this simulation. 226

Figure 36 – Example of the simulation with agents that do not hold any previous knowledge. As the placing is done iteratively, they search for parking spaces in a series. The red cars are looking for a space to park, while blue cars are leaving the car park..... 228

Figure 37 – Example of the simulation with experienced agents whose primary concern is to park as fast as possible. These cars have had twenty previous experiences of parking, with random configurations each time. 229

Figure 38 - Example of the simulation with experienced agents whose primary concern is to park as fast as possible. These cars have had twenty previous experiences of parking, always with the current configuration of cars. 230

Figure 39 - Comparison of three different experience profiles over 8 different simulation configurations. For each run the number of targets considered by each agent is taken into account, as well as the number of steps taken by the agent before parking..... 232

Figure 40 - Individual agent values for the 25 agents in the first tested configuration and whose results are presented in Figure 39. The values presented reflect the number of parking spaces considered as targets by the agents and the number of steps taken by each agent until they reach a parked state. 233

Acknowledgements

First of all, I would like to thank my first supervisor, Prof. Jonathan Raper, not only for his support through out the period while I was working on the dissertation, but also for his intelligence, his capacity for creative thinking, his ability for keeping calm when panic is in order and for showing me how interesting it is to work in research.

I would also like to thank Prof. António Câmara for the support that he made available to me, for his ideas and suggestions and for always being motivated to improve the World and Portugal.

To Eng. Rui Goncalves Henriques, for all the support that he provided as a scientist and as the president of Centro Nacional de Informação Geográfica (CNIG) and for being the first to suggest that I should embark on this PhD adventure.

To Junta Nacional de Investigação Científica e Tecnológica (JNICT) and later to Fundação para a Ciência e Tecnologia (FCT), for the funding they provided me with, which made all this work possible.

Thanks are also required to CNIG for the support provided to the research work presented in this thesis.

To all my colleagues at CNIG, I thank their friendship, support and availability to help.

I would also like to thank the people at the Geography Department at Birkbeck College, especially Tim McCarthy and Nathan Williams, for their time and coffee.

I thank Elsa João for helping me with everything you need to know about doing a PhD in England.

I thank Paulo Cabrita for his tips on Java and Cédric Grueau for his help on defining the user profiles for the testing of SIFIA.

To all my friends, especially Luísa, Eduarda, Angela and João Pedro, I thank their friendship and care.

To Carmen Morgado and Miguel Boavida I thank that last minute help.

To my parents and my sister I thank their unquestionable support, comprehension and love. Thank you for always being there.

Finally, to Cédric I thank his firm belief that I could do this, for that was what made me do it.

“I grant powers of discretion to the University Librarian to allow this thesis to be copied in whole or in part without further reference to me. This permission covers only single copies made for study purposes, subject to normal conditions of acknowledgement”.

Abstract

The manipulation of geographic information (GI) has been considered, by most of its users, as a very complex process and has involved the development of specific applications called geographic information systems (GIS). A new subdiscipline of Information Science called Geographic Information Science (GISc) has been proposed by the GIS research community on the grounds that there are specific characteristics not only to GI but also to the processes involved in its manipulation.

The thesis draws on several research issues which are part of the agenda in GISc: The complexity of handling spatial/geographic information, spatial reasoning in dynamical systems, integration of several types of application, human-computer interaction and spatio-temporal issues.

In this context, this dissertation proposes the application of a new computational paradigm, intelligent agents in GISc. Intelligent agents are "computational systems that inhabit some complex dynamic environment, sense and act autonomously, and by doing so realise a set of goals or tasks for which they are designed " (Maes, 1995).

The aim of this dissertation is to analyse the potential of research in intelligent agents in GISc and to explore the use of simple learning techniques to improve the adaptability of spatial intelligent agents. The thesis involves the following objectives: to analyse the needs of research in GISc in the areas of reasoning about geographic space; to study the potential of intelligent agents in that area of research; to explore the use of simple learning techniques to improve the adaptability of intelligent agents for geographic information; and to explore the implementation environments of GIS software for the integration of intelligent agent systems.

The primary contributions of this research are three case studies which use intelligent agents in a spatial or geographic context: a simple non-adaptive interface assistant for the printing and plotting tool of Smallworld GIS; an intelligent assistant that uses memory-based reasoning to identify and locate specific-purpose geographic information; a simulation of a car park where agents are cars that use reinforcement learning techniques to improve their parking performance.

Ch. 1

Introduction

The manipulation of geographic information has been considered, by most of its users, as a very complex process and has involved the development of specific applications called geographic information systems (GIS). The research community has proposed a research agenda for a new subdiscipline called Geographic Information Science (GISc) (Goodchild, 1992; NCGIA, 1995). This agenda asserts that there are specific characteristics not only for geographic information itself but also in the way the information should be handled.

According to Goodchild (1992), the specificity and complexity of geographic information lies on the multi-dimensional property of the spatial key associated with data. In fact, this key may even be defined for a continuous domain in each

dimension. If a dataset is defined as two-dimensional, then any coordinate pair location (x,y) in the dataset can be queried and must present a value if addressed. This involves large volumes of data, and complex processes for manipulation. The complexity rises if other dimensions are added (height or temporal).

The implication of this is that there exist many different types of conceptual data models for geographic information, and that the choice between them for a given phenomenon is a fundamental issue in geographic data handling (Goodchild, 1992).

Another specific characteristic of geographic data is spatial dependence, which is the propensity for nearby locations to influence each other and to possess similar attributes (Goodchild, 1992). Locations that are near each other will normally have similar attribute values. However, the structure of spatial dependence is unusual, relying on both dimensions (x,y) of the spatial key, with the level of similarity determined by a metric.

The final specific feature of geographic data is that they are distributed over the curved surface of the earth, a fact often forgotten in the limited study areas of many GIS projects (Goodchild, 1992). For centuries, efforts were put into portraying the earth's surface on a sheet of paper, and extensive technology was developed for map projections. As a result, very few methods exist for analysing data on the sphere. The potential of current electronic display can now be used to create views of the globe itself.

The GISc research agenda and this thesis

Given these particular characteristics of geographical information, NCGIA (1995) outlined their view of the content of GISc. Through out their discussion, several problems in GIS and GISc research, which still need a solution, are described. Some specific problems of reasoning about geographic space form the basis for the work described in this dissertation.

Goodchild (1992), divides the content of GISc into the following parts: Data collection and measurement; data capture; spatial statistics; data modelling and theories of spatial data; data structures, algorithms and processes; display; analytical tools; and institutional, managerial and ethical issues.

Most of these areas involve computer science questions that were also considered by Worboys (1995). With the focus on the technological issues, Worboys analysed Goodchild's concerns and delivered a list of computer science focussed GIS/GISc research agenda. This list included the following research areas: models and user requirements, representations, spatial access methods, computational paradigms, architectures and data management, interfaces and languages, finite resolution, data quality and multiple resolution systems and, extensions to the two-dimensional spatial model of geo-information.

Several problems concerning geographic information and its manipulation have been identified in these research agendas which define important questions for this dissertation. These problems can be classified in the following manner:

- The complexity of handling spatial/geographic information;
- Spatial reasoning in dynamical systems;
- Integration of several types of application;
- Human-computer interaction issues;
- Spatio-temporal issues.

The following sections examine these issues in more detail to illustrate the key theoretical questions requiring resolution for progress in reasoning about geographic space.

Goodchild (1992) relates the problems of handling the complexity of geographic information to the continuous dimension of the data as well as to the fact that large volumes of data become available every day, particularly now with the growth of the Internet. The amount of information becoming available online is explained not only

by developments in online mapping technology (see chapter 4 of Plewe, 1997, for choices on commercial distributed geographic information software) but, more importantly, by National and International data availability policies (Rodrigues, 1998; SNIG, 1998).

In Europe, several countries are creating their own national geographic information infrastructures, the first one being the Portuguese National Geographic Information Infrastructure (SNIG¹). One of the aims of this infrastructure is to direct users to information about the data they are looking for and, if possible, to provide access to that data. Other efforts of this kind are the National Clearinghouse for Geo-Information (in the Netherlands) and the Spatial Information Directory in GIS Flanders (Belgium), Rodrigues, 1998. There are also several Supra-national efforts in the making, like the European Spatial Metadata Infrastructure (ESMI) project and MEGRIN (Rodrigues, 1998).

All of these efforts involve the publishing of metadata on available geographic information. Some of them also involve using this metadata to locate and identify relevant data. With so many providers, information overload is an issue that must be dealt with.

Another area where large volumes of geographic data are becoming available is that of distributed geolibraries. The issue of distributed digital libraries has become very popular with technological developments that have enabled the publishing and organisation of multimedia data and its search and retrieval in a timely fashion (Raper, 1996).

The NSF/DARPA/NASA Digital Library Initiative (CISE, 1999) is an organised effort funded by the US National Science Foundation (NSF), The Defense Advanced Research Projects Agency (DARPA) and The National Aeronautics & Space

¹The SNIG is the responsibility of the Portuguese Centre for Geographic Information (CNIG: <<http://www.cnig.pt>>. This system is based on the World-Wide Web and joins all the producers of Geographic information in Portugal. Access to the several producer entities in the system is provided either through direct access to their web pages or through the use of the SNIG's metadata infrastructure, which includes detailed information on the data provided by the entities.

Administration (NASA), where research into different areas (applied into digital libraries) is involved. One of those projects is the Alexandria Digital Library (ADL, 1998), the first Digital Geolibrary, where material is organised according to their geographic reference.

As stated by Goodchild (1998), a geolibrary is a library filled with georeferenced information. This means that “information is found and retrieved by matching the area for which information is needed with the footprints² of items in the library, and by matching other requirements, although the footprints always provide the primary basis of search”. It also means that this search can be conducted in several remote sites at the same time.

The geographic location of each item in the library is the primary index for searching for information. Therefore, the manipulation of this type of index should be very sophisticated.

Digital geolibraries are appearing and growing every day, through the growing availability of datasets and through the improvement of computational techniques for handling them. Examples of the possibilities of implementation of digital geolibraries using existing technology can be found at (MSC, 1999a).

Information overload, a problem in itself, is related to another problem in spatial information called *surfacing*. This is the time-lag between data and information gathering, storage and retrieval, as well as what proportion of such data and information can be analysed, interpreted and used. It is difficult to process and absorb new datasets. In fact, there is a general inability to recognise the need to cope with the overload of data and to adapt, recognise and redefine what is seen, needed and absorbed.

There exists a need for efficient methods for storage and access that are capable of dealing with the large volumes of information that are becoming available. These

methods, associated with the definition of metadata standards that can capture the specific features of this data can improve the situation of information overload.

Also important are the problems related to handling uncertainty in geographic issues. These involve not only geographic objects with indeterminate boundaries but also issues related to scale, generalisation, quality of information and error control and propagation. These problems are being handled through the use of set theory (particularly fuzzy set theory) or statistical and probability techniques (Fisher and Wood, 1999; Fisher, 1999; Fisher and Langford, 1996).

Another important area is that of spatial reasoning in dynamical systems. Spatial modelling and simulation techniques are mostly based on the property of spatial dependence (Goodchild, 1992). However, early individual-based models concentrate on sequential processes applied to spatial locations in the study area (e.g. cellular automata). The limitations of these methods are clear. A model of reality will be more real if there is the possibility of events happening in a parallel fashion. Also, reality includes active and passive entities populating a spatial environment. The model of the cell representing simply a location in the environment, which may or may not change state, is also limiting.

A related issue is that of spatio-temporal information. Current concerns are usually connected with representation of temporal series of information and with the methods for integration and interpretation of these series (Worboys, 1999; 1996; 1996a; 1994). However, little work has been put into the connection between space and time in spatial modelling and simulating.

Another relevant problem in this dissertation is that of the integration of several types of spatial applications in order to fulfil one spatial task. Included here are the issues of transferring information between tools which are based on different

²Footprint is the word used in the Alexandria Digital Library to name the several techniques used to refer to geographic locations (coordinates, geographic units, naming,...).

technologies and the integration of GIS in several types of spatial processes (e.g. spatial simulation, spatial analysis).

The techniques in, for example, loosely-coupled analysis (as described by Goodchild, 1992), where an independent analysis module relies on a GIS for its input data, and for such functions as display, can also be used in simulation (Guerrin *et al*, 1998). However, this type of process often involves the loss of higher level structures of information like, topology, object identity, metadata or various kinds of relationships. There is a need for research into tight coupling, in which a whole spatial process can be organised, passing through several types of tools, without loss of high level information (Goodchild, 1992).

The last of the key problems in geographic information science that bear upon this dissertation are related to human-computer interaction. Specifically, issues related to user interfaces in GIS and spatial information systems are one of the areas where further research is very much needed.

All of the different application areas of GIS have different object sets, variable or fixed scale and dynamic or static inputs. This means that it is very difficult to develop a common interface for all types of GIS. As described by Raper and Rhind (1990), the knowledge required to use a system is complex and it becomes easy to make mistakes. Moreover, the use of the command structure and sequence of standard tasks in GIS often require a thorough knowledge of spatial analysis and theory.

These difficulties are also related to the necessity of handling multi-source, non-standard, heterogeneous data, associated with inadequate standards and in some cases, variable expectations of the users. After results are provided, users often will come back with new requests, which may imply problems in integrating required data (non-availability of required scales and time periods).

The required interfaces will, not only facilitate the use of the tools to the non-expert user, mostly through task-dependent definitions, but also limit the users choices,

by restricting requests that do not make sense or cannot be executed for lack of required data (Goodchild, 1992). These interfaces will be adaptable to the user's needs and to the available data. Another required facility associated with user interfaces is recovering from errors by backtracking to save intermediate results (Goodchild, 1992). The matter of the technology to use to solve these problems must be addressed. Worboys (1995) proposes research into the application of new computational paradigms to the handling of spatial information.

The same author also defends research into new metaphors in user interfaces (in which he is supported by Goodchild, 1992). Worboys states that an interface to a spatial information system should help users to take advantage of the data within. This can be accomplished not only through the use of new metaphors in user interfaces but also through new visualisation methods and approaches to metadata handling. Data mining and exploratory uses of information systems also hold promise.

Problem Definition

Given the various research problems identified above, this dissertation proposes the application of intelligent agents, a new computational paradigm, in GISc. The problems described above are of major importance in the further development of GISc. The lack of current solutions for them or the awkwardness of existing ones is, in most cases, preventing the development of useful tools for handling and managing spatial information or modelling and simulating spatial processes. The use of agents in these areas can provide solutions for some of these problems or improve current implementations. This thesis explores the extent to which intelligent agents can help solve some of these problems.

As in any new area, several definitions of agents have been put forward (see chapter 2 for definitions of agents). Pattie Maes (1995), defines agents as "computational systems that inhabit some complex dynamic environment, sense and act autonomously in this environment, and by doing so realise a set of goals or tasks for which they are designed."

Agents have been criticised generally as a computational paradigm in several occasions (e.g. Shneiderman, 1995). The criticisms fall into claims that any simple program can be considered an agent if evaluated according to agent-oriented concepts. In this dissertation the claim that agents cannot be used to implement spatial reasoning will be falsified. The prototypes implemented in the context of this work are agent-based, are concerned with solving spatial problems and have been implemented using an agent-oriented methodology.

Agents and Interfaces to GIS

This work proposes the use of a new type of metaphor in user interfaces: the intelligent assistant (Maes, 1995). Unlike many other current metaphors, this is not graphically-based, but concept-based. Its origin is motivated by the fact that non-technical people are becoming common computer users and tasks like acquiring news and information and receiving and sending mail are becoming computer-based. From this, Maes (1995) argues that it is necessary to make interaction metaphors evolve. By using a complementary style of interaction, called *indirect management* (Kay, 1990), the user is engaged in a cooperative process in which human and computer agents both initiate communication, monitor events and perform tasks. Maes (1994) argues that this type of interaction can be implemented using autonomous agents that represent *personal assistants* collaborating with the user in their work environment. The author also states that the set of tasks or applications an agent can assist as virtually unlimited (Maes, 1994): information filtering, information retrieval, mail management, meeting scheduling, selection of books, movies, music and so forth.

Agents and information access

Handling spatial-information involves being able to control information overload and lower surfacing. Intelligent agents (personal assistants) have been in use in information management for some time (Maes, 1994; Kozierek and Maes, 1993; Dent *et al*, 1992; Mitchell *et al*, 1994). Specifically, personal assistants enable the

personalisation of the characteristics of a user's environment in a working session. Learning techniques provide the building of knowledge by the agent on user preferences and the evolution of these preferences as user's habits and tasks evolve in time.

When executing spatial tasks in a spatial environment, a user may benefit from the help of a personal assistant that can remember his/her favourite type of information, spatial region, scale, and sequence of operations applied. The assistant can also prevent the user from trying to integrate information that cannot be integrated or perform sequences of operations that do not make sense. By remembering previous working sessions of the user, the agent can also change the working interface, including not only operations that are often used but also others that may prove to be useful.

This type of work contributes not only to improve the control of information overload (by providing the user with relevant information) but also to the creation of better user interfaces. This is done by using the metaphor of the personal assistant to aid the non-familiar user with the interface and by facilitating its use through personalisation.

Agents and individual-based modelling

The other area of spatial information science to which this work contributes is to spatial reasoning in dynamical systems. The use of agents in this area enables the creation of truly individual-based environments. Here, agents represent active actors in the environment, individuals that are located in space, that hold spatial characteristics and whose actions take spatial conditions into account. These models can act in a truly parallel fashion, if specific conditions are given to them, and evaluate space and time as a continuum, taking into account that, in a real dynamic system, conditions for actions may change at any moment.

These models enable the association of the object-oriented, reactive, intelligent capabilities of agent systems (see chapter 2) to the spatial characteristics of

geographic entities (individuals) as described by earlier theorists (Harvey, 1969; Nunes, 1991; Chapman; 1997).

In this case, the application of simple learning techniques with some simple extensions to spatial operations enables agents in a spatial environment to develop spatial reasoning.

Objectives

The aim of this dissertation is to analyse the potential of research in intelligent agents in geographic information science and to explore the use of simple learning techniques to improve the adaptability of spatial intelligent agents.

Therefore, the objectives of the thesis are:

1. To analyse the needs of research in GISc in the areas of reasoning about geographic space. Firstly, the identification and access of special interest geographic information through the use of metadata structures, by a non-expert user with very specific needs. Secondly, the integration of these processes in adaptable user interfaces for spatial information systems. Finally, and at a different level of research, the development of systems that can provide the simulation of spatial processes resulting from the individual execution of spatial tasks;
2. To study the potential of intelligent agents for the above research questions: specific-purpose geographic information location (identification of the searched information) and access; the improvement of spatial information systems interfaces and the simulation of evolving spatial environments from the modelling of spatially-aware individuals;
3. To explore the use of simple learning techniques to improve the adaptability of intelligent agents for geographic information;
4. To explore the implementation environments of GIS software for the integration of intelligent agent systems.

To achieve these objectives, a review of the generic use of intelligent agents as well as in the specific areas of spatial simulation, spatial decision making and in the development of user interfaces is performed to support the proposed developments. This aims to provide an overview of the areas of application of intelligent agents and of the state of the art of current implementations in intelligent agents with geographic information.

The prototype assistant for printing and plotting in the Smallworld GIS and the Spatial Information Facilitator for the World-Wide Web aim to give two different views of how a user interface may be adapted to the preferences of the user.

The first one is a simple implementation, integrated in the Smallworld GIS framework, which performs a simplification of the printing and plotting task once the user has chosen sufficient attributes for the agent to execute the task.

The Spatial Information Facilitator provides identification and location of specific-purpose spatial information for a user through the search of a metadatabase of available geographic information. It uses memory-based reasoning to evaluate previous working sessions of each user and when it finds patterns of behaviour, it automatically provides the user with the requests most often issued. Personalisation here is made not only on the operations executed by the user but also on the specificity of the required information.

The Car Park Agent simulation is an agent-based simulation where several car agents search for a parking space in a car park. The aim of the simulation is to park the cars as quickly as possible in car spaces that obey the preferences of the cars. The cars use learning to review their parking decisions as they move in the car park. Their parking experiences are stored and used in new visits to the car park. Spatial considerations have been added to the utility information used by each car agent so that properties like position, distance and speed are of relevance to the decisions made.

The first prototype is only a simple experience to illustrate the potential of using agents within GIS as well as their implementation issues when integrated in a GIS. The last two go further: agents with spatial tasks to fulfil use simple learning techniques to improve their performance in time. These learning techniques are adapted to reason over spatial features, so that learning can also take place in terms of the spatial characteristics of decision and action.

These two prototypes illustrate the possibilities of using agents in spatial tasks, in domains as different as information access and spatial simulation. They also explore the potential of using simple learning techniques to improve the agents' performance.

The first claim in this thesis, as discussed above, is the falsifying of the negative hypothesis about agents systems, which asserts that any simple program can be considered an agent if evaluated according to agent-oriented concepts. This thesis aims to falsify this general claim specifically through the use of intelligent agents for spatial reasoning. It is argued here that the development of the prototypes in the context of this work, which are concerned with solving spatial problems and have been implemented using an agent-oriented methodology, are sufficient to defend that claim.

The second claim in this thesis, which needs further justification, is that simple learning added to the spatially-aware agents will add to the knowledge they hold of the spatial domain they are embedded in and will enable them to improve their performance. This claim is supported by the tests results provided in chapters 6 and 7.

In the case of the car park agent simulation, improvement in performance is demonstrated through quantitative performance measures defined there. In the case of the Spatial Information Facilitator, the improvement is recognised in terms of the rapidity of information retrieval forming part of the preference profile of the

user, and in terms of the placing of the user (by the agent) in a specific group, through the experience given by previous behaviour.

Limitations

The aim of this dissertation is to analyse the potential of using intelligent agents in the areas of spatial information handling and spatial reasoning. It is not the objective of this thesis to prove that agent technology is the best choice for system development in this context. As a result, the developed prototypes aim to illustrate the global approach, but they do not correspond to complete products, and only the most relevant functionalities for each case study were implemented. In this context, intensive software testing was not performed, but a more informal and continuous approach into development was adopted. This thesis demonstrates that these concepts have now developed to the point where the GIS industry can now begin to exploit them.

Thesis organisation

Chapter 2 of this dissertation lays the ground for the subject of the thesis by presenting intelligent agents in the context of computer science. It describes the origins of agents and the existing models and architectures for their implementation. It then evaluates the potential for using intelligent agents in spatial issues. It concludes by addressing this thesis' three relevant areas of research in spatial information in an intelligent agent context: Spatial simulation, spatial-decision making and interface agents for GIS. Finally, it describes the needs for further research in the area and sets the ground for the original work, which has been developed in the context of this dissertation.

Chapter 3 describes the different approaches currently used for the development of interface agents and explains the choice made in this work for the machine learning approach. It then describes the specific learning technique used and addresses the specific spatial issues, which were considered in its use.

Chapter 4 addresses the theory of simulation and the traditional techniques used in the implementation of simulation. The limitations of these techniques are discussed and the use of individual structures in simulation (such as reactive agents) are put forward as the technique chosen in this dissertation. The specificity of spatial simulation is described and a framework for developing adaptive agent-based spatial simulations using reinforcement learning is presented.

Chapters 5, 6 and 7 present the case-studies developed for this dissertation and discuss the advantages and limitations of the developed methodologies in the context of these applications.

Finally, chapter 8 draws the conclusions for this work and presents some possibilities for future work in this area.

Ch. 2

Literature review on intelligent agents and spatial issues

*“The most stable elements , Clarice, appear in the middle of the periodic table,
roughly between iron and silver.”*

Hannibal Lecter writing to Clarice Starling,
in Thomas Harris, Hannibal, 1999

Although the most widely available agent applications are quite recent, research into agents has a long history, the first major work being *“The Society of Mind”* by Marvin Minsky (1985). In this book, Minsky describes the structure of the mind through a series of small, one page chapters. He argues that the mind is composed of small non-intelligent pieces that when considered as a whole can develop intelligent thought. He calls this concept, the Society of Mind. The book itself

consists of a set of small, common-sense ideas that together can explain “the strangest mysteries of mind”. He uses, for his writing of the book, the same structure that he puts forward for the organisation of the mind. This seminal insight has taken some time to be fully exploited but is now responsible for an explosion of agent software applications.

2.1 Defining Agents

Several authors have defined agents. There are several very different definitions, depending on the area of application of the author’s work. The decision taken here was not to choose one definition to go by, but to quote several which identify the agents’ properties considered to be the most relevant in this work:

- (Russell and Norvig, 1995): “An Agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors.”
- Hayes-Roth, 1995): “Intelligent agents continuously perform three functions: perception of dynamic conditions in the environment; action to affect conditions in the environment; and reasoning to interpret perceptions, solve problems, draw inferences and determine actions”.
- (Maes, 1995): “Autonomous agents are computational systems that inhabit some complex dynamic environment, sense and act autonomously in this environment, and by doing so realise a set of goals or tasks for which they are designed.”
- (Wooldridge, 1997): “An intelligent agent is generally regarded as an autonomous decision-making system, which senses and acts in some environment.”
- (Huhns and Singh, 1998): “Agents are active, persistent (software) components that perceive, reason, act and communicate.”
- (Wooldridge, M., Jennings, N.R., 1995):

“A weak Notion of Agency: A software-based computer system that enjoys the following properties:

- *autonomy*: agents operate without the direct intervention of humans or others, and have some kind of control over their actions and internal state;
- *social ability*: agents interact with other agents (and possibly humans) via some kind of *agent-communication language*;
- *reactivity*: agents perceive their environment, (which may be the physical world, a user via a graphical user interface, a collection of other agents, the INTERNET, or perhaps all of these combined), and respond in a timely fashion to changes that occur in it;
- *pro-activeness*: agents do not simply act in response to their environment, they are able to exhibit goal-directed behaviour by *taking the initiative*.

A Stronger Notion of Agency: For some researchers – particularly those working in AI - the term ‘agent’ has a stronger and more specific meaning than that sketched out above. These researchers generally mean an agent to be a computer system that, in addition to having the properties identified above, is either conceptualised or implemented using concepts that are more usually applied to humans. For example, it is quite common in AI to characterise an agent using *mentalistic* notions, such as knowledge, belief, intention and obligation. Some AI researchers have gone further, and considered *emotional* agents.”

Considering the first definition (Russell and Norvig, 1995), the authors emphasise that there are three components that seem essential for agents to be agents:

- The environment – every object that surrounds the agent and with which it has to interact. The state of the environment is dynamic and can be changed by the agent or by external forces;

- Sensors – structures that inform the agent about the state and changes in the environment;
- Actuators – structures that enable the agent to act on the environment.

However, according to Franklin and Graesser (1996), when depending heavily on what we take as the environment and on what sensing and acting are, it is necessary to restrict the meaning of these concepts. “If we define the environment as whatever provides input and receives output, and take receiving input to be sensing and producing output to be acting, every program is an agent”.

Considering the next definition (Hayes-Roth, 1995), we find some help on restricting these concepts: sensors allow agents to perceive the dynamic conditions of the environment, while actuators enable it to affect conditions in it. There is a new element added, that is, reasoning. Reasoning will provide agents with the potential to interpret perceptions, solve problems, draw inferences and determine actions, and therefore, to decide.

(Maes, 1995) adds another relevant property to which Hayes-Roth had already alluded: she states that agents are autonomous, that is, they can act without direct intervention from humans or any other processing modules. Hayes-Roth partly deals with this by saying that agents perform continuously, they do not depend on being started up, shut down, or called by someone or something. Another addition given by Maes is the property of having goals and working towards realising them. Agents are initially assigned a task or goal, which will be their primary aim, and all their processing is explained by it.

Wooldridge (1997) refers to an agent as being a decision-making system. This is not a general property but it may be relevant in some domains. If the agent is autonomous and acting it will definitely need to decide which action to take given a specific state of the environment.

Huhns and Singh (1998) introduce a property that Wooldridge and Jennings (1995) confirm: the social ability of agents. They think that some agents may become more

efficient if, instead of working selfishly towards their specific goals, they communicate and cooperate with other agents in order to reach common goals.

Wooldridge and Jennings (1995) classify agents' properties and create a much more comprehensive definition that includes most of the elements referenced by the previous authors. They provide a "weak" and a "stronger" notion of agent, where the former describes the type of agent this work will be concerned with, and the latter refers to the choice of modelling an agent using human-like features, mentalistic or emotional properties. These properties will be analysed in the section dedicated to the models of agency.

2.2 Agents in the Context of Computer Science

In his paper "Agent-based software engineering" (Wooldridge, 1997), Wooldridge considers the problem of building agent-based systems as a software engineering enterprise. He compares agents with other software development techniques that have previously been considered, as agents are now, as the solutions for the current existing problems in software development.

2.2.1 Artificial Intelligence

Intelligent Agents are strongly identified with Artificial Intelligence (AI). In fact, Russell and Norvig have written an AI text book based on agents (Russell and Norvig, 1995). However, as Wooldridge (1997) states, we must distinguish between the intelligence that is the ultimate goal of the AI community, and the intelligence that we aim to provide with agents. According to Wooldridge, "the only intelligence requirement we generally make of our agents is that they can make an acceptable decision about what action to perform next in their environment, in time for this decision to be useful". Other requirements will be defined by the problem at hand. That is why, although agents are usually built using AI techniques, their requirement to act in time to be useful, must make them, fundamentally, a computer science problem.

2.2.2 Expert Systems

An expert system (ES) is a system that is capable of solving problems or giving advice in some knowledge-rich domain (Wooldridge, 1997). During the 1980s, expert systems were very popular, and they were built for each area of business or science where the advice of an expert was of major importance. As Wooldridge (1997) notes, there are several differences between ES and agents:

- ES are not in direct contact with the environment in which they have to make decisions about. They do not perceive the environment through sensors, they receive relevant information from users, in direct contact with the problem at hand;
- They do not, usually, have to produce a solution (or an action) in real time;
- They do not, usually, have to cooperate with other expert systems.

There are exceptions, one being ARCHON, the set of expert systems that provided answers in real time (Jennings *et al*, 1996).

2.2.3 Object-orientation

An *object* is an entity that encapsulates some state and a collection of methods, corresponding to operations that may be performed on that state (Wooldridge, 1997). The concept of object includes one static part, information, and one dynamic part, the behaviour of the system (Worboys, 1995). The static aspect of the object is expressed by a collection of named *attributes*. The set of values given to each attribute for one specific object comprises its *state*. The dynamic part of the system (its *behaviour*) is expressed as a set of operations that the object will perform under specific conditions and which are called *methods*. Methods are typically invoked as the result of messages sent to the object. The behaviour of the object is, thus, the complete set of responses that it may give to messages.

Objects that present similar behaviour are considered of the same *type*. This is a semantic notion that must be represented by specific computational objects at the

design and implementation stages. *Object classes* are structures that enable the implementation of this semantic type. A class represents a group of similar objects with corresponding data structures and methods. An object is created belonging to a specific class, and is automatically given the structure and methods associated with that class. The object will be an *instance* of the chosen class.

The object-oriented (OO) approach has been promoted especially in areas of software development for which existing technology was found to be problematic. Object-oriented development was seen as a new way of thinking about software, based on abstractions that exist in the real world. The essence of object-oriented development is the identification and organisation of application-domain concepts, rather than their final representation in a programming language (Rumbaugh, 1991). This approach has been applied in different areas of software development leading to the following meanings (Worboys, 1995):

- Object-oriented programs – allow the system construction process to be carried through using the object-oriented approach from the phases of analysis through implementation. Examples of OO programming languages are Simula, Smalltalk, C++ and more recently Java. It is important to state that OO programming languages enable the creation and manipulation of objects but there is usually no guarantee for the support of persistence of the objects created;
- Object-oriented user interfaces – (e.g.: the Apple Macintosh graphic user interface), referred to as object-oriented mainly because they can deal with high-level representations and because, in some cases, they encapsulate methods with objects;
- Object-oriented databases – which enable the support for persistent objects.

Some OO tools have been widely used for the development of agents systems because of the characteristics underlying them and which are helpful in agent development: modularity, inheritance, encapsulation, dynamic binding and multi-

threading (Rodrigues and Raper, 1999). On the other hand, Wooldridge notes the following differences between object and agent systems:

- In traditional object-oriented programs there is a single thread of control. In contrast, agents are process-like, concurrently executing entities;
- An agent is a rational decision-making system: agents are capable of reactive and proactive behaviour, and of interleaving these types of behaviour as the situation demands. The object-oriented research community is not concerned with this problem, whilst it is of major importance for the intelligent agents research community;
- The object-oriented community has not addressed issues like cooperation, competition, negotiation, computational economies, etc, which form the foundation of multi-agent systems development.

Some variants to the object model have been developed that approximate objects to agents: e.g. Object-based concurrent programming models (like ACTORS: Agha, 1986), have long been recognised as an elegant model for concurrent computation and have contributed enormously both to the study of objects and agents.

2.3 Origins of Agents

It is argued in this thesis that we can establish two defined schools in the study of agents: the area of artificial intelligence called *Distributed artificial intelligence* and the body of research emerging from the study of complex systems which, in this thesis, will be called *Complexity Handling*. It is interesting to compare these two fields, as they seem to be approximating each other although they start from two very opposing positions. Distributed Artificial Intelligence became popular in the mid eighties as a way of solving the very difficult problems of Symbolic Artificial Intelligence (Bond and Gasser, 1988; Huhns, 1987; Adler and Cottman, 1989). It was created by Artificial Intelligence (AI) researchers as an approach to AI where intelligence was distributed through several entities in a system.

The study of complexity emerged from the work of researchers in the natural sciences who started to develop systems to simulate the experiences they were interested in (Langton, 1989). What is interesting in this is that, as is presented below, AI researchers, trapped by the complex nature of their systems saw the solution of their problems in creating simpler symbolic, distributed systems. Natural sciences researchers initially used very simple structures and have evolved into creating simulations whose emerging behaviour is based on agents (Reynolds, 1987; Ferrand, 1995). Each of these approaches is discussed in more detail below.

2.3.1 Distributed Artificial Intelligence

Appearing in the scientific literature related to Artificial Intelligence around the mid eighties, Distributed Artificial Intelligence (DAI) was introduced as a new research area dealing with the solution of complex problems by networks of autonomous, cooperating computational processes (Adler and Cottman, 1989 quoting Huhns, 1987). At the time, related research focused almost exclusively in areas where agents would cooperate to solve a single complex task (e.g.: data fusion, situation assessment or speech understanding, Adler and Cottman, 1989). DAI scientific literature at the time concentrated on research issues contemplating: distribution and task allocation; coherence and coordination; languages, structures and protocols for interaction and case-studies (Huhns, 1987; Bond and Gasser, 1988).

In Bond and Gasser (1988), Lesser and Corkill (1988) present what they called at the time *Functionally Accurate, Cooperative Distributed problems*, which would be the basis for defining the best problems for using distributed artificial intelligence. In fact, they argued that, up to that date, distributed processing had only been successfully implemented in process control and distributed databases problems. These were applications characterised by task decompositions in which the data can be partitioned in such a way that each subtask can usually be performed completely by a single node – without the need for the node to see the intermediate states of processing at other nodes. In these conventional distributed systems a node rarely needs the assistance of another node in carrying out its problem-

solving function. This type of distributed processing decomposition was called *completely accurate, nearly autonomous* (CA/NA), because each node's algorithm operates on complete and correct information and because each node usually has in its local database the information it requires to complete its processing correctly.

This approach is/was not suitable for applications in which algorithms and control structures cannot be replicated or partitioned effectively so as to match the natural distribution of data in the network. In this case, the implementation of CA/NA is very expensive because of the high communication and synchronisation costs required to guarantee completeness and consistency of local databases. Lesser and Corkill (1988) defended that the almost exclusive use of the CA/NA approach restricted the types of application which were implemented in a distributed manner. The approach presented by Lesser and Corkill (1988) suggested that in a distributed system, each node could perform useful processing using incomplete input data while simultaneously exchanging the intermediate results of its processing with other nodes to, cooperatively, construct a complete solution. They argued that the amount of communication necessary to exchange these results would be much less than the communication of raw data and processing results which would be required using the CA/NA approach. The *functionally accurate* (FA) problem-solving structure exhibited acceptable system input/output behaviour as opposed to *completely accurate* problem-solving structures where all intermediate results shared among subtasks were required to be correct and consistent.

As in FA problem solving, a node may have to perform useful processing using incomplete, incorrect and/or inconsistent tentative results, nodes need to cooperate in order to eliminate incorrect results and to converge to a complete and consistent solution. Therefore, this type of system was called *functionally accurate and cooperative* (FA/C). These systems were the basis for what is now called Distributed Artificial Intelligence (DAI). Three arenas for research were defined and are still active today (Bond and Gasser, 1988, pp. 3-36):

- Distributed Problem Solving (DPS) – concerned with how the work of solving a particular problem can be divided among a number of modules, or “nodes” that cooperate at the level of dividing and sharing knowledge about the problem and about the developing solution (Lesser and Corkill, 1987; Smith and Davis, 1981);
- Multiagent Systems (MA) – coordinating intelligent behaviour among a collection of (possibly pre-existing) autonomous intelligent “agents”, how they can coordinate their knowledge, goals, skills, and plans jointly to take action or to solve problems (Hewitt, 1985; Hewitt, 1986);
- Parallel AI (PAI) – dealing with developing parallel computer architectures, languages and algorithms for AI. The focus here was in solving performance problems of AI systems and not toward conceptual advances in understanding the nature of reasoning and intelligent behaviour among multiple agents. This area was not initially seen as a sub-part of research in DAI (Davis, 1980; Davis, 1982; Fehling and Erman, 1983).

As we can see, DAI was a field of research mainly concerned with the development of intelligent distributed systems that could do useful work even if each node could not have access to complete and consistent information. The emphasis was on solving distributed problems, on making nodes (“agents”) communicate and cooperate and on improving performance of highly distributed systems.

2.3.2 Complexity Handling

As Franklin (1995) states, early AI researchers worked from the homocentric view that minds are a specifically human characteristic, and thus, concentrated on simulating human tasks like chess playing, theorem proving and language translation. The classic approach to that, Symbolic AI, was seen as a top-down approach to the study of mind by virtue of the high-level abstract tasks upon which it was focused.

Artificial Life (AL) is, according to Langton (1989), the study of man-made systems that exhibit behaviours characteristic of natural living systems. It complements the

traditional biological sciences concern with the analysis of living organisms by attempting to synthesise life-like behaviours within computers and other artificial media. Like symbolic approaches to AI, biology has traditionally started at the top, viewing a living organism as a complex biochemical machine, and worked *analytically* downwards from there - through organs, tissues, cells, organelles, membranes, and finally molecules - in its pursuit of the mechanisms of life. Artificial Life starts at the bottom, viewing an organism as a large population of simple machines, and works upwards *synthetically* from there - constructing large aggregates of simple, rule-governed objects which interact with one another nonlinearly in the support of life-like, global dynamics. This leads to what Langton calls *emergent behaviour*, life emerging from the organised interactions of a great number of entities, with no global controller responsible for the behaviour of every part. In fact, every part is a behaviour in itself, and life is what emerges from all the local interactions among individuals.

AL was initially conceived as a tool for the natural sciences and several researchers have been active in developing specific tools to prove their theories. Disenchanted Symbolic AI researchers also started focusing their efforts in a bottom-up approach for developing intelligent systems, as they expected to learn more from building complex models of simple systems (such as animals) than from building simple models of complex systems (e.g. Brooks, 1986; Brooks, 1990). This is where an area that was initially fed by computer technology, actually provides input and feedback to fundamental computer science and AI research. The idea of creating software using techniques that have made life evolve is now a very popular one and has had results in techniques like neural networks, genetic algorithms and some approaches to agent systems.

As Langton (1989) states, the roots of complex behaviour can be traced back to the attempts that man has made of building imitations of living things. Initially, these attempts involved one central program responsible for the model's dynamic behaviour. Langton believes that this was the source of the failure of these models

and the source of failure of the whole program of modelling complex systems that followed, including much of the work in Artificial Intelligence. He goes on to say that the most promising approaches to modelling complex systems like life or intelligence are those which have dispensed with the notion of a centralised global controller, and have focused instead on mechanisms for the distributed control of behaviour. This work has begun by looking at the way in which behaviour is generated in a bottom-up fashion in living systems. These mechanisms were then generalised so that they could be applied in artificial systems. The great contrast, once again, lies in the exceedingly parallel and distributed nature of the operation of living systems, when opposed with the singularly serial and centralised control structures associated with 'developed' systems.

The two most relevant biological concepts taken into account were:

- *The Genotype*: The complete set of genetic instructions encoded in the linear sequence of nucleotide bases that makes up an organism's DNA;
- *The Phenotype*: The organism itself - the structures that emerge in space and time as the result of the interpretation of the genotype in the context of a particular environment.

The process by which the phenotype develops through time under the directions of the genotype is called *Morphogenesis*.

In the context of AL, and to be able to apply these notions into the development of artificial systems, it was necessary to generalise the notions of genotype and phenotype, so that they could be applied in non-biological situations. Therefore, the following definitions appeared:

- *Generalised genotype* - GTYPE – any largely unordered set of low-level rules;
- *Generalised phenotype* – PTYPE – behaviours and/or structures that emerge out of the interactions among these low-level rules when they are activated within some specific environment.

PTYPES draw on the full combinatorial potential implicit in the set of possible interactions between the low-level rules that compose the GTYPE. The consequences of this are twofold: The extremely rich variety of possible PTYPES and the extreme difficulty in predicting the PTYPES that will emerge from specific GTYPES, given initial structures.

One general approach to building GTYPE/PTYPE systems is based on the methodology of *recursively generated objects*. In this methodology, the conceptual “object” is a structure that has sub-parts. The rules of the system specify how to modify the most elementary, “atomic” sub-parts, and are usually sensitive to the *context* in which these atomic sub-parts are embedded. The “neighbourhood” of an atomic sub-part is taken into account in determining which rule to apply in order to modify that sub-part. Usually, there are no rules that apply to the entire structure, that is, no use is made of global information. Each piece is modified solely on the basis of its own state and the state of the pieces nearby. Following are short descriptions of examples of recursively generated objects (Langton, 1989).

Lindenmayer systems (L-systems): sets of rules for rewriting strings of symbols bearing strong relationships to the formal grammars treated by Chomsky (Lindenmayer and Prusinkiewicz, 1989).

Cellular Automata (CA): another example of the recursive application of a simple set of rules to a structure. In a CA, the structure that is being updated is the entire universe: a lattice of finite automata. The local rule set - GTYPE in this case - is the transition function obeyed homogeneously by every automaton in the lattice. The local context, taken into account in updating the state of each automaton, is the state of the automata in its immediate neighbourhood. The transition function for the automata constitutes a *local-physics* for a simple, discrete space/time universe. The universe is updated by applying the local physics to each “cell” of its structure

over and over again. Thus, although the physical structure itself doesn't develop over time, its *state* does. For more information on CA see Wolfram (1994).

“Boids”: The previous examples were largely concerned with the growth and development of a *structural* PTYPE. Now, we give an example of the development of a *behavioural* PTYPE. Reynolds (1987), implemented a general platform for studying the qualitatively similar phenomena of flocking, herding and schooling. This model includes a large collection of autonomous but interacting objects (“Boids”), inhabiting a common simulated environment. The modeller can specify the manner in which the individual boids will respond to local events or conditions. The global behaviour of the aggregate boids is strictly an emergent phenomenon, none of the rules for the individual boids depend on global information and the updating of the global state is done on the basis of individual boids responding to local conditions.

Langton (1989) noted that *context-sensitive* rules in GTYPE/PTYPE systems provided the possibility for nonlinear interactions among parts. The lack of context sensitivity would make it impossible to add meaningful flow of information to the system, because it would become linearly decomposable. Thus, there would be no place for complex long-range dependencies between remote parts of the structure.

From Reynolds work, the thought of structures that will employ agents in the place of Boids, is almost immediate. This transition would employ mainly reactive agents (see agent architectures below) and there are several examples already in the literature. Several researchers that work in natural or social sciences, trying to build spatial simulations of the phenomena they were concerned with, started with a CA approach and later evolved to a reactive multi-agent approach (Touret, 1995; Vigneron, 1995; Ferrand, 1995). Using agents as local entities in simulation environments has become quite popular presently (some examples are presented later in section 2.5.4.1).

Working in complexity and Artificial Life led to the creation of systems built from the bottom-up, composed of very simple parts that would enable global intelligent behaviour. Also in this area, the emphasis was on distributed behaviour towards the realisation of one global objective.

Before moving on, it is important to identify the concepts that DAI and AL share, and what separates them. Both areas stress the importance of distributed behaviour in making agents systems successful. What DAI refers to as communication and cooperation can be seen as working towards a global solution in AL (although these concepts are not identical). However, DAI agents are much more complex as entities than AL's, requiring a concept of "agency" with explicit components being defined.

2.4 Implementation of Agents

In this section, the implementation of agents is addressed by focusing on the existing models of agency and on how these models are implemented in specific agent architectures. Philosophical issues in addressing reality can be taken into account when analysing these models and specifically the following items, based in psychology and in the philosophy of Science, are of major importance.

2.4.1 Modelling agents

2.4.1.1 Philosophical issues

The first step towards building agents systems is to define the model under which they will be conceived. The study of agents has included the definition of models of agency. These models have been created and used according to three philosophical approaches, which are described below (Huhns and Singh, 1998):

- *Behaviourism* - This is a doctrine that considers only the direct behaviours of agents. The universe is seen as sequences of events where causality does not exist as a concept. Agents do not have mental states such as intentions or beliefs and multiagents systems do not have social states such as

commitments. Any existing patterns are caught externally. This is a traditional view used in distributed computing;

- *Subjectivism* – In contrast, this approach considers that agents do have intentions and commitments, but only as represented in agents, and the universe has causation but only as represented in the agents' definition; this is the type of model traditionally used in Artificial Intelligence;
- *Realism* – This is the most attractive position but also more difficult to implement. Abstractions are included but their definitions must be related to real world concepts. In this approach, the agents' perspective guide their actions (which cannot be done in Behaviourism), the evaluators' perspective is used for testing compliance (which is impossible in subjectivism). The linkage between the two perspectives is provided in the designers' perspective.

2.4.1.2 Models of Agency

Given these approaches, the most successful models of agency are presented below using the scheme proposed by Huhns and Singh(1998). The application of the above doctrines on the following models may depend on the choice taken by the developers. However, there are certain combinations which are not possible (e.g. applying the logical perspective of rational agency under the behaviourist doctrine):

2.4.1.2.1 Rational Agency : Logical perspective

The BDI model (Beliefs, Desires and Intentions) relies on the definition of the following logical qualitative concepts (Rao and Georgeff, 1995; Wooldridge, 1997):

- *Beliefs* – Information on the current state of the environment which is updated appropriately after each sensing action. This is the informative component of system state and will typically be represented symbolically;
- *Desires* – Information about the objectives to be accomplished (priorities and payoffs associated with the various current objectives). Desires represent the motivational state of the system, the tasks that are allocated to a specific agent;

- *Intentions* – An agent's intentions represent desires that it is committed to achieving.

As it is thought that agents will not, generally, be able to achieve all their desires, even if these desires are consistent, they must therefore fix upon some subset of available desires and commit resources to achieving them. Intentions should be used as feedback in future decision-making: an agent should not adopt intentions that conflict with those it currently holds.

The use of the belief abstraction involves the maintenance of consistency in a system enabled through the capture of relationships of the beliefs of agents (Huhns and Singh, 1998). Consistency can be maintained according to two views:

- *well-foundedness*, where all beliefs, except premises, should be justified by other beliefs and these justifications should contain no cycles (a belief cannot be used as justification and be justified in the same series of thought);
- *coherence*, which states that beliefs should hold together as a coherent body (locally) even if they cannot be externally justified.

Human behaviour is closer to coherence than well-foundedness although traditional logic favours the latter. Both these approaches are used successfully in tools called *Truth Maintenance Systems* (Huhns and Bridgeland, 1991).

2.4.1.2.2 Rational Agency: Economic perspective

When the perspective has an economic concern, the agent's preferences are decided from the knowledge of the effects of its actions. The chosen action will be the one that maximises preferences. This approach aims to reduce the preferences of the agent into a single scalar that can be compared with other scalars. This will require a careful selection of the target problem. The most successful applications of Economic Rationality are connected with the following activities (Huhns and Singh, 1998):

- *Decision-theoretic planning*: Costs and effects of actions are modelled quantitatively and probabilistically. If probabilities can be reliably estimated, effective plans can be created from this approach (Haddawy, 1996; Horvitz and Rutledge, 1991 quoted by Huhns and Singh, 1998);
- *Negotiation*: A small set of agents with a common language, problem abstraction and solution negotiate deals with a concern on the utility of the deal for their specific goals (Rosenchein and Zlotkin, 1994);
- *Computational markets*: an approach to distributed computation based on market mechanisms (Wellman, 1995). The challenge here is to build computational economies to solve problems of distributed resource allocation. The system aims to reach an equilibrium corresponding to an allocation of resources, dictating the activities and consumptions of the agents.

2.4.1.2.3 Social Agency

According to Huhns and Singh (1998), modern applications call for a true peer-to-peer distributed and flexible paradigm that can only be accomplished through cooperation and sociability. They believe that this is where agent technology can be most useful. The mental model cannot capture all aspects of social interactions, and economic rationality essentially reduces an agent to a selfish agent. The notion of commitment is extended to include *Social Commitments*. These are commitments that an agent makes in relation to another agent. Social agents manipulate new concepts like *Witnesses* (Castelfranchi, 1995) and *Contexts* (Singh, 1997).

Coordination in a system that involves agents performing some activity in a shared environment, becomes essential, especially when considered as *Cooperation*, if the agents in the system are not antagonistic (Huhns and Singh, 1998; Jennings, 1992).

Social Agency is best applied in problems where it is necessary to achieve *Coherence* (how well a system operates as a whole) and *Optimality* (which are intimately related), Huhns and Singh (1998). Nwana (1996) describes work at

Carnegie Mellon University where an architecture of collaborative agents was created to organise decision making on the Infosphere (a distributed set of on-line information services). The same architecture was used to create agents systems in the areas of financial portfolio management, emergency medical care and electronic commerce (Sycara, 1995). Nwana (1996) also mentions two prototype applications at British Telecom Labs for business process reengineering (O'Brien and Wiegand, 1996) decentralised management and control of consumer electronics (Titmuss et al, 1996).

2.4.1.2.4 Interactive Agency

Interactions occur when agents exist and act in close proximity. There are unintended interactions, which mainly result in resource contention, and intended interactions, called *communications*, which can occur through shared resources (shared memory). Communication implies shared conventions based on language. In this type of model, the equilibrium between communication and computation must be carefully measured. Communication is generally more expensive and less reliable than computation. However, it may be impossible to reconstruct information locally and communication can only be avoided when agents are conceived to share all knowledge *a priori* (Huhns and Singh, 1998; Hern, 1988). Davis and Smith (1983) evaluate the implications of communication among agents: communication must be efficient in order not to saturate the available channels, and the grain size of messages passed must be carefully chosen to balance between the benefits of information sharing (among agents) with the difficulty of handling complex messages. Haddadi (1995) developed a formal theory for describing the reasoning processes that lead agents to communication - towards potential cooperation.

2.4.1.2.5 Adaptive Agency

The final model of agency is the one that supposes that agents are adaptive and therefore are persistent and can learn. A large amount of learning in agents has to do with learning values for parameters, which is mostly applied to personalisation

of a user interface through the modelling of the user. One of the challenges in this area is to make autonomous agents learn from the environment (Shen, 1993). The Autonomous Agents group, in MIT Media Laboratory, led by Pattie Maes has developed a large amount of work on Interface agents and information agents that learn (Maes, 1994; Maes, 1994a; Maes, 1995). Weiß, G. (1997) edited a volume on DAI and Machine Learning, and Stone and Veloso (1997) have organised existing work on Machine Learning in Multi-agent systems in a thorough review paper.

2.4.2 Agent Architectures

Agent Architectures are the result of trying to apply agent theories (described above) in implementation. Wooldridge and Jennings (1995) have written a thorough review of the types of architectures available for agent development. They present the following classification (presented in sections 2.4.2.1, 2.4.2.2 and 2.4.2.3), considering the issues surrounding the construction of agent systems that satisfy the properties of existing agent theories. These architectures are divided into classical (viewing agents as a particular type of knowledge-based system), alternative and hybrid approaches.

2.4.2.1 Classical Approaches: Deliberative Architectures

Deliberative (cognitive) Agents are agents that explicitly include a symbolic model of the world and where decisions are made via logical (or at least pseudo-logical) reasoning, based on pattern matching and symbolic manipulation (Wooldridge and Jennings, 1995). The symbolic AI paradigm has its roots in the *physical-symbol system hypothesis*, which was formulated by Newell and Simon (1976). "A physical symbol system is a physically realisable set of physical entities (symbols) that can be combined to form structures, and which is capable of running processes that operate on those symbols according to symbolically coded sets of instructions" (Wooldridge and Jennings, 1995). The physical symbol system hypothesis states that this type of system is capable of general intelligent action.

Theoretically, the idea of deliberative agents, based on logical reasoning, was considered very attractive but very hard to achieve in practice, because of the computational complexity of symbol manipulation, even in very simple theorem proving. Quoting Wooldridge and Jennings (1995): "(...) first-order logic is not even *decidable* and modal extensions to it (...) tend to be highly *undecidable*. Even if one rejects a purely logical approach, one still finds that some key problems (such as planning (...)) appear to be *intractable*."

These problems have led researchers to work on alternative approaches for building agents. However, let us first consider some of the most outstanding examples of Deliberative Architectures described by Wooldridge and Jennings (1995), and which are discussed below:

Planning Agents: This type of architecture originates from the AI planning community, and its concern with the design of artificial agents. They claimed that the central component of an artificial agent must be some kind of AI planning system. Planning results from the design of a sequence of actions that will result in the achievement of the desired goal. Several attempts have been made for constructing agent systems whose primary component is a planner (Ambros-Ingerson and Steel, 1988; Wood, 1993; Etzioni *et al*, 1994; Cohen *et al*, 1989). However, Chapman (1987) concluded that planning systems, even those using the most refined techniques, would ultimately turn out to be unusable in a system where results must be given in a time-constrained period. These conclusions have led researchers to question the whole symbolic AI paradigm and have led to new approaches in agent architectures, like the ones described in sections 2.4.2.2 and 2.4.2.3.

BDI based architectures: One of the examples of a Beliefs, Desires, Intentions architecture is IRMA (*Intelligent Resource-bounded Machine Architecture*), (Bratman

et al, 1988; Pollack and Ringuette, 1990). Besides the symbolic representation of beliefs, desires and intentions, the architecture includes a *Plan Library*. Reasoning about the world is accomplished through the *Reasoner*. The *Opportunity Analyser*, monitors the environment looking for options for the agent to take. The choice of action is determined by the *Filtering Process* and the *Deliberation Process*, where the former determines the potential courses of action, depending on the agent's intentions, and the latter is responsible for the final choice of action.

In the GRATE* system, social behaviour is added to this type of architecture (Jennings, 1993). In this work, agents not only have beliefs, desires and intentions but they also have *joint intentions*. This is a layered architecture divided into a *Domain level system* (which solves the domain related problems) and the *Cooperation control layer* which enables cooperation between agents. The emphasis is the social behaviour of the agents and ensuring that their domain level activities are coordinated with those of others. GRATE* has been evaluated against agents which only have individual intentions and against selfish agents, in electricity transportation management. A significant improvement was noted when the situation became complex and dynamic (Jennings, 1995).

2.4.2.2 Alternative Approaches: Reactive Architectures

The problems associated with symbolic AI have led to work on alternative approaches and specifically to the development of Reactive Architectures. These are a reaction to deliberative architectures and the symbolic AI paradigm. They are composed of Reactive Agents, with no central symbolic world model and using no complex symbolic reasoning. Agents are simple structures, which together form an intelligent being, capable of solving the most complex problems. This type of architecture is very near the ideas presented by Minsky (1985) when describing the simple pieces that compose a mind. It is also closely related to the work accomplished by Artificial Life researchers (see section 2.3.2 above).

The Subsumption Architecture (Brooks, 1986) resulted from the work of Brooks, an MIT researcher who became frustrated by classical AI approaches to building

control mechanisms for autonomous robots. According to Brooks (1990), Symbolic AI systems decompose intelligence into functional information-processing modules, where each module has a specific function to carry out but cannot do much by itself. Brooks presents a system where intelligence is decomposed into individual behaviour-generating modules, and each module can, by itself, actually generate some behaviour. Brooks also defends that creating correct symbol descriptions of the world must be task dependent and that perception provides the relevant description for the task in order. He further defends that the designed systems should also be task dependent. A *subsumption architecture* is a hierarchy of task-accomplishing behaviours, which compete to control the actions of a robot. This is also a layered architecture, working as a natural system would when concerned with its necessities. Lower layers provide more primitive behaviours and have precedence over higher layers. The system is, computationally, a very simple one, with no explicit reasoning or pattern matching. However, their results were impressive in certain types of tasks (Steels, 1990).

Running Arguments: For Agre and Chapman (1987), most everyday activities are routine, requiring little new abstract reasoning. This architecture is based on the concept of running arguments, where most decisions can be encoded into a low-level structure, which only needs updating for handling new kinds of problems.

Situated Automata: This is a very interesting architecture, which seems to combine the best part of reactive and symbolic, declarative systems. A declarative system is a knowledge-based system built in terms of what it knows. Agents are built by adding sentences to it, one at a time, that comprise their knowledge of the environment. The construction of the agent can be very simplified by this approach (Russell and Norvig, 1995, pg. 153). In a situated automata architecture, agents are specified through declarative sentences that can be compiled down to a digital machine, which satisfies the declarative specification (Kaelbling and Rosenchein, 1990). The reader might wonder why this architecture has not been classified as a hybrid. In fact, although agents are specified in a declarative form, there exist no

symbolic representations or symbol manipulation in the system (Wooldridge and Jennings, 1995).

Agent network architecture: In this architecture, presented by Maes (1990), agents are composed of a set of competence modules, which are similar to the definition of Brooks' behaviours. There are also similarities with neural networks architectures with the relevant difference, however, that neural net nodes are only meaningful within the context of the network, whilst, competence modules are defined in declarative terms, and, thus, have a well-defined meaning.

From these definitions we can conclude that most reactive architectures include very simple structures, with almost hard-wired knowledge of their basic capabilities. Learning and adaptive behaviour is possible, from the conjunctions of these very simple "behaviours", evolving from the local actions of agents to the global performance of the system as a whole.

2.4.2.3 Hybrid Architectures

The existence of the above approaches has led some researchers into combining them, and trying architectures that have both classic and reactive properties. In fact, the last reactive architectures presented in the previous section already present hybrid characteristics. Below we present a few examples where you can identify two main types of hybrids:

The Procedural Reasoning Systems (PRS) architecture, proposed by Georgeff and Lansky (1987), and evaluated by Georgeff and Ingrand (1989), is a belief-desire-intention architecture with deliberative and reactive structures. Beliefs are represented as facts, and desires as system behaviours. Intentions are active *Knowledge Areas*, which are partially elaborated plans that can be activated in goal-driven or data-driven fashion. Knowledge Areas can also act reactively, which allows the system to respond rapidly to changes in the environment.

A similar architecture, with elements both from PRS and IRMA, **COSY** (Haddadi, 1994), is a hybrid BDI-architecture with five components: Sensors, Actuators, Communications, Cognition and Intention. The cognition component mediates between the intentions of the agents and its beliefs. An agenda is maintained, which contains active scripts that can be invoked in a goal-driven fashion (to follow agent's intentions) or in a data-driven fashion (depending on the state of the environment), just like knowledge areas in PRS.

The **TouringMachines** (Ferguson, 1992) hybrid agent architecture uses three control layers embedded in a control framework. The *reactive layer* generates actions when events happen very rapidly and it is necessary to react immediately to them. The *planning layer* constructs plans and selects actions depending on the agent's goals. The *modelling layer* maintains symbolic representations of other entities in the agent's environment. These models are manipulated in order to identify and resolve goal conflicts.

Another layered architecture is **InterRRap** (Müller *et al*, 1995), where each layer represents a higher level of abstraction than the one below it. The lowest level manages the communication between the agent and its environment. Just above, the behaviour-based component implements and controls the basic reactive properties of the agent. The next layer is concerned with planning and the highest one ensures that cooperation between agents is possible. Each layer contains several knowledge bases.

As we have seen, hybrid architectures generally have a reactive component that enables the system to react very rapidly to changes in the environment. The first two examples have several similarities, the most important being that intentions can be activated in a goal-driven fashion (in order to accomplish the agent's intentions) or in a data-driven fashion (depending on the state of the system). The two layered architectures are structured according to the type of behaviour necessary at a given time. The lowest level will generally be associated with reactive

behaviour and, as we go up in the structure, the layers demonstrate more structured and deliberative behaviour.

From the described agent models and architectures it is now possible to state the approaches that are most appropriate for the implementation of spatially-aware intelligent agents. The rational model involves the creation of agents that will include strategies for a rational decision. The economic perspective will be most appropriate for this type of work, because it will quantitatively measure the choices of action given to the agent. It is also clear that these agents will be adaptive to changes in the environment. In terms of architectures, the choice for reactive agents with embedded reinforcement techniques will enable the integration of spatial parameters into the agents' adaptive behaviour.

2.5 Agents and spatial issues

Agents with a spatial awareness are a relatively old concept. In Minsky (1986), space is described as "just a society of nearness relations between places" and the global geography of a space as "nothing more than hints about which pairs of points lie near one another". Minsky also suggests that several layers of agents build the maps inside our brains, each layer being composed of agents that are responsible for regions of the space and whose function is to detect which other agents are the nearest to them.

Before addressing the potential of the use of agents in spatial issues, it is necessary to know the computational issues of handling spatial concepts (a review of these is presented in Egenhofer *et al*, 1999). The specific characteristics of spatial information systems as well as the underlying properties of space must be taken into account when inserting agents in spatially-aware environments. Taking into account that the knowledge representation must be task-dependent in order to be successful (as argued by Brooks, 1990), it is of major importance to understand spatial concepts in order to create agents that manipulate them. In the following

sections, the most relevant spatial concepts and the computational structures that manipulate them are discussed.

2.5.1 Fundamental spatial concepts

Space has been defined through the years using different views and for different disciplines. As Nunes (1991) notes, “perhaps the most widely accepted conception of space is that of a container or framework in which things exist. ” This is so because it is intuitively what people think of space, something empty to be filled by things (Nunes, 1991).

Worboys (1995, pg. 97), states that space may be conceptualised in two distinct ways: absolute space, a set of locations with properties; and relative space, a set of objects with spatial properties. This dichotomy was realised at early stages in the development of *Geographical Information Systems* (GIS) by Chrisman (1975, 1978), and became so important that it is a major part of the discussion on the concept of geographic individual, for the definition of the domain of geography (section 2.5.2).

A *GIS* is a computer-based information system that enables capture, modelling, manipulation, retrieval, analysis and presentation of geographically referenced data (Worboys, 1995, pg. 1).

According to Nunes (1991), the first goal of GIS was to reproduce, inside the computer, the graphical and spatial nature and contents of maps, and to provide for some of the operations of analysis or presentation commonly carried out by maps. Since the beginning, two ways to represent the graphical rendering of maps appeared, each referring to the conceptualisations of space mentioned above (Nunes, 1991):

- *Raster-based* : square grid cell partitioning at a fixed resolution of the surface to be represented, each cell with a coded value corresponding to the feature at the location;
- *Vector-based* : vector encoding of series of pairs of coordinates corresponding to points that, taken successively as starting and ending points of straight line

segments, approximate the shape of the boundaries of a given cartographic feature, identified by a code or label.

This dual conceptual view of space mapped into GIS, and the lack of conceptual reflection in early stages of GIS development led raster and vector models to progressively acquire a status of conceptual models of geographic space that they do not have (Nunes, 1991).

Nunes argues that *geographic space* must be a relative space, where objects are the space. He states that, to define geographic space or a conceptual model for it will involve defining and studying the geographical objects, their attributes and relationships.

The formalisms that represent the abstract properties of structures within space are provided by *Geometries*. In 1872, in the *Erlangen program*, Klein defined *Geometry* as the study of invariances of a set of objects, subject to a group of transformations. From this definition, it becomes clear that the objects that form the spaces studied by geometry have no real, physical, sensible existence. Geometry is not a description of physical space, but it provides formal structures that can be used as representations or tools for representing physical space, if a mapping between the both domains can be defined (Nunes, 1991).

In the context of this dissertation's first objective, to analyse the needs of research in Geographic Information Science (GISc) in the areas of reasoning about geographic space, a review on the geometries proven to be most useful in GIS is presented. The aim of this review is to analyse the abstract formalisms that have been used to represent space. The current discussion (in subsections 2.5.1.1, 2.5.1.2, 2.5.1.3 and 2.5.1.4) presents the study of space through differently conceptualised geometries and sets the ground for the discussion of the subject of geography and the geographic individual (section 2.5.2). The approach taken in this review follows Worboys (1995, chapter 3), and aims to evaluate the specific properties and complexity of spatial concepts.

2.5.1.1 Euclidean Space

Euclidean Space is a coordinate-referenced space that enables measurements of distances and bearings between points according to the usual formulas. Euclidean space transforms spatial properties into properties of tuples of real numbers.

To present the several components of Euclidean geometry, let us assume a two dimensional model (called an *Euclidean plane*). Let us also set up a coordinate frame composed of a fixed point called *origin* and of two orthogonal lines (*axes*) which intersect at the origin. Euclidean space includes the following types of objects:

- *Point objects* – a point in the plane of axes is associated with a unique pair of real numbers (x, y) . These numbers represent the distance from the origin in the direction of each axis. The collection of all the points in the plane is called *Cartesian plane* or R^2 . Cartesian points (x, y) can also be viewed as *vectors*, drawn as one directed line segment from the origin to the point (x, y) . The distance between two points $a = (x_i, y_i)$ and $b = (x_j, y_j)$, $|ab|$ in a Euclidean plane is given by the equation below

$$|ab| = \sqrt{((x_i - x_j)^2 + (y_i - y_j)^2)};$$

- *Line objects* – In a GIS, lines represent the spatial attributes of objects and their boundaries. It is important to differentiate *line* (an infinite line passing by two defined points) from *line segment* (a finite line which begins and ends in two defined points) and *half line* (a half line begins in a defined point passes through a second one but does not have an end);
- *Polygonal objects* – to define *polygon*, it necessary to begin by the notion of *polyline*. A polyline (in R^2) is a finite set of line segments (*edges*) where each edge end-point is shared by exactly two edges, except possibly for two points, called the *extremes* of the polyline. Further, if no two edges intersect at any

place, other than possibly at their end-points, the polyline is defined to be a *simple polyline*. When a polyline has no extreme points it is a *closed polyline*. The area enclosed by a simple closed polyline is called a (*simple*) *polygon* (in \mathcal{R}^2). The polyline forms the *boundary* of the polygon and each end-point of an edge of the polyline is called a *vertex* of the polygon. The definition of polygon can be extended to include polygons that contain holes, islands within holes and the like.

The Euclidean plane includes some common transformations in \mathcal{R}^2 . These are functions from \mathcal{R}^2 to itself. This means that every point of the plane is transformed into another point (maybe the same). Some of the transformations preserve specific properties of the objects in the plane. Following is a description of the most relevant transformations.

- *Euclidean transformations (congruences)* – these transformations preserve the shape and size of the objects embedded in the plane (e.g. translation in a Euclidean plane);
- *Similarity transformations* – these transformations preserve the shape but may not preserve the size of the objects. All Euclidean transformations are similarities;
- *Affine transformations* – these preserve the affine properties of the objects in the plane, such as parallelism (e.g. rotations, reflections and shears are all similarities);
- *Projective transformations* – projective transformations preserve the projective properties of embedded objects (e.g. the projections of a circle may result in an ellipse). All affine transformations are projective;
- *Topological transformations (homeomorphisms, bicontinuous map)* – these preserve topological properties of embedded objects and are discussed below.

Euclidean geometry is the basis for the representation of geographic objects in GIS. The versatility of the above transformations associated with the operations included in the following space types has been responsible for the growth of the application of GIS in different disciplines (Egenhofer *et al*, 1999).

2.5.1.2 Set-based geometry of space

The set-based model of space is not as rich as the Euclidean plane, in terms of constructs. It is composed of *elements*, *collections of elements* (which are called *sets*) and the *membership* relation between an element and a collection in which it occurs. However, set theory is fundamental to the modelling of hierarchical relationships in spatial information systems (or GIS). For example, relationships between different base units of spatial reference may be modelled using set theory.

In classical set theory, the membership relation is a binary one: an object is either an element of a specific set, or it is not. There can be cases where the membership relation has a probability value (fuzzy set theory) associated with each pair element and set, and that can be very useful in geographic applications (see the discussion section of chapter 6 (section 0) for an example of this).

From the basic constructs considered above, a number of modelling tools might be defined. Considering two sets *S* and *T*, below is a description of a few of these tools:

- *Equality* – a relationship between two sets that holds when the sets contain exactly the same members;
- *Subset* – a relationship between two sets where every member of the first set is also a member of the second (denoted $S \subseteq T$);
- *Power set* – the set of all subsets of a specific set (denoted $P(S)$);
- *Empty set* – the set containing no members (denoted \emptyset);
- *Cardinality* – the number of members in a set (denoted $\#S$);

- *Intersection* – this is a binary operation that takes two sets and returns the set of the elements that are common to both sets (denoted $S \cap T$);
- *Union* – this is a binary operation that takes two sets and returns the set of the elements that are members of at least one of the original sets (denoted $S \cup T$);
- *Difference* - the binary operation that takes two sets and returns the set of the elements that are members of the first set but not of the second one (denoted $S \setminus T$)
- *Complement* – a unary operation that when applied to a set returns the set of elements that are not in the set (the complement of S is denoted S^c). This operation is executed with reference to a universal set;

The possibility of creating relations and functions between elements of sets also provides very important tools in set-based geometry (Worboys, 1995, pp. 104-110). Specifically, the operations referred to above are mostly used for relative evaluation of geographic objects. The use of this type of geometry associated with probability techniques has led to work on fuzzy concepts which is currently widely used in GISc to study problems like scale, generalisation, quality of information, overlay and error control and propagation (Fisher and Wood, 1999; Fisher and Langford, 1996).

2.5.1.3 Topological spaces

Topology is a branch of geometry, concerned with the set of geometrical properties that remain invariant under *topological transformations*, that is, under continuous transformation of a surface. Laurini and Thompson (1992, pg. 188), state the following elements as invariant:

- Incidences (points on lines, etc);
- Intersections;
- Adjacencies (neighbours for polygons);

- Inclusions (points in polygons).

Topological transformation can be exemplified by the changes reflected on a figure that has been drawn on a rubber band that can stretch and contract. We can define a *topological property* as one that is preserved by topological transformations in space.

A topological space can be defined in many different ways. One of the possibilities is based on the single primary notion of neighbourhood. The formal definition is the following:

Let S be a given set of points. A *topological space* is a collection of subsets of S , called *neighbourhoods*, which satisfy the following conditions:

1. Every point in S is in some neighbourhood;
2. The intersection of any two neighbourhoods of any point x in S contains a neighbourhood of x .

From this definition of neighbourhood, all familiar topological properties can be defined (e.g. adjacency, boundary and nearness). The most important example of topological space is the *usual topology* for the Euclidean plane, so called because it is the topology that comes to mind when working with the Euclidean plane and which corresponds to the rubber-band stretching and contracting mentioned above.

Topology can implement the relative nature of space through the relative relations of objects positioned in space. Topological concepts can be applied not only to tiled polygons but also to network spaces. The description of topological relations is of major importance in the definition of operators that enable the spatial evaluation of objects and phenomena (finding locations near a specific feature or the use of the adjacency property).

These operators are part of the computational treatment of geographic issues and enable the definition of the specific properties of geographic objects in a computational representation of geographic space (see the spatial sections of

chapters 3: section 3.2.2.3.2, 4: section 4.5.3, 6:section 6.2.2 and 7: section 7.2.2). They are also the basis for the relations mentioned by Minsky (1986) to describe space inside the human mind (section 2.5).

2.5.1.4 Network spaces

Network spaces are based on the concept of graph. A *graph* G is a finite non-empty set of *nodes* with a set of unordered pairs of distinct nodes (called *edges*). If x and y are nodes of G and the edge $e = (x, y)$ belongs to G , then e *joins* x to y , or is *incident* with x and y . On the other hand, x and y are *incident* with edge e . A graph represents connectedness between elements in the space. There are some extensions to the definition of graph that can be quite useful for modelling spatial relationships:

- *Directed graph* – a graph where each edge is assigned a direction. Direction is often indicated by representing edges as arrows (e.g. modelling a road network in city centre where there are usually many one-way streets);
- *Labelled graph* – In this type of graph, each edge is assigned a label (e.g. a number which can be associated with the cost of going from one node to the other).

There are several concepts related to graphs, which are often useful for modelling spatial relationships. A list of a few of these is presented below:

- *Degree* – the degree of a node is the number of edges with which it is incident;
- *Path* - a path between two nodes is a connected sequence of edges between the nodes, and is usually denoted by the nodes that make up the path. A *connected graph* is one in which there always exists a path between any two of its nodes;
- *Isomorphism* – when two graphs present exactly the same connectivity relationships they are said to be *isomorphic*;

- *Cycle* – a cycle is a path from a node to itself traversing at least one edge. A graph with no cycles is an *acyclic graph*. Acyclic graphs lead the way for a very useful set of graphs, connected acyclic graphs, also known as *trees*.

These were simple descriptions of the most widely used geometries for handling spatial phenomena in computational systems and specifically in GIS. Next, the issue of high-level modelling is addressed in a spatial context using the spatial concepts presented above.

2.5.2 The Geographic Individual

Harvey (1969) argues that the crucial issue in Geography is the definition of its domain: what are the geographic individuals, what are the geographic populations or sets, what are the appropriate scales of measurement. Even now, according to Nunes (1991), the explicit treatment of this question is still relatively unresolved. Without actually providing a definition, Harvey (1969) classifies individuals in the following way:

- Continuous vs. discrete individuals;
- Individuals that can be identified substantively (through a set of attributes) vs. individuals that can be identified spatially (through spatial or space-time coordinates);
- Natural individuals (e.g. farms, countries, lakes or rivers) vs. artificial individuals (spatial units of measurement, especially suited for continuously varying phenomena);
- Singular versus collective individuals;

Nunes (1991) points out that Harvey does not discuss these alternatives any further nor does he try to integrate them in a comprehensive framework. However, Chapman (1997), takes the concept of object, “something which can be observed as a discrete entity by some perceptual mechanism, operated by man”, and differentiates three kinds of objects (Nunes, 1991):

- *Proper objects* – the first order object of study or structurally integrated object. These are objects that have the properties of wholes. For Chapman, these are the only true objects;
- *Areal aggregate object* – something perceivable (e.g. wood);
- *Non-areal aggregate object* – a class intellectually constructed.

From this, it is clear that space or simple spatial objects like locations, places and regions cannot be the objects of geography. Chapman sees space as a property of objects, making full sense of the notion of relative space. If two objects are of the same kind, they must have different space values in order to be different.

Finally, Nunes (1991), defending a view of geographic space as a set of geographic entities, defines the latter as “a part of the Earth surface defined by virtue of one or several spatial and/or non-spatial properties, which make it a space in itself, physically and/or functionally different of its surrounding environment, and therefore, capable of carrying out an individual existence persistent in time, and capable of being perceived by human beings as such individual space and entity.”

Taking a different approach, while addressing the construction of Geographic Information (GI), Goodchild *et al* (1999), use the term *geographic concepts* to describe all of the generic components of GI, which the authors list in the following way:

- The concepts that provide the basis for GI itself (e.g., the geodetic system, projections, metrics of distance);
- Elements of geographic thesauri, the standard authorities that define a common vocabulary of feature types (e.g., *lake, reservoir, river, city, building*);
- The contents of gazeteers that establish the positions of named places;
- The terms and phrases that define relationships between geographic point sets (e.g., *near to, north of, crosses, intersects with*); and

- The generic classes of *things* that define the phenomena present at geographic locations (e.g., variables such as *elevation*, *temperature*; land cover classes; land ownership, zoning regulations).

The above definitions and considerations present a cognitive concern with the treatment of geographic individuals. Research work into the cognitive models of geographic space is currently underway in the Varenus Project (Mark *et al*, 1999) with major research themes addressing the acquisition of geographic knowledge, the mental representations of geographic knowledge, the use of knowledge and the communication of geographic information. These themes have been organised in the following research topics (where the first three are considered of higher priority by the project's team): *formal concepts of geographic detail; cognition of dynamic phenomena and their representations; multiple models and multiple frames of reference for spatial knowledge; ontology of geographic entities; mental maps and formalising spatial relations*. Some other less-well defined topics were also identified.

The representation of the concepts described in this section in a computational framework have been addressed by Egenhofer *et al* (1999), also in the context of project Varenus. In the next section the computational methods for representing geographic concepts are addressed according to the views given by Egenhofer *et al* (1999) and Worboys (1995, pp. 145-179).

2.5.3 Computational methods for representing geographic concepts

In Egenhofer *et al* (1999), the state of progress in computational methods for representing geographic concepts is reviewed by considering the specific characteristics of geographic information where computational manipulation is concerned; by evaluating the existing foundation in modelling and implementing geographic concepts; and by observing trends that are starting or already underway. These considerations are taken into account in the following subsections (sections 2.5.3.1, 2.5.3.2 and 2.5.3.3).

2.5.3.1 Spatial Data Modelling

When describing the modelling process and spatial data models, Worboys (1995), starts by defining model, an artificial construction in which parts of one domain (*source domain*) are represented in another domain (*target domain*). The purpose of the model is to simplify and abstract from the source domain. The elements that constitute the source domain, and which are relevant for the work in question (entities, relationships, processes or any other phenomena of interest) are translated by the model into the target domain and viewed and analysed in this context. The results given by the model can then be interpreted in the source domain. To measure how useful a model is, it is necessary to analyse how closely it can simulate the source domain and how easy it is to move between the two domains. There is a mathematical concept behind this, the notion of *morphism*, which is a function from one domain to the other that is capable of preserving some of the structure in the translation.

In GIS, models may exist at different levels, ranging from models of particular application domains (e.g. transportation models) to specific computer-based models of the physical information in the system (Worboys, 1995). Nunes (1991) drew attention to the distinction between two different and opposing classes of models for Geographic Information:

- *Field-based* models treat information as collections of spatial distributions, where each distribution may be formalised as a mathematical function from a spatial framework to an attribute domain;
- *Object-based* models treat the information space as a set of discrete, identifiable entities, each with a geo-reference.

As seen above, these two models result in two opposing GIS implementation approaches: raster and vector.

In the field-based approach, each field defines the spatial variation of an attribute in the relation as a function from the set of locations (*spatial framework*) to an

attribute domain. The relation is conceptualised into variations of single or multiple attributes. The spatial framework is a partition of the region studied into a finite tessellation of spatial objects. The individual areal components of the partition are called *locations*.

Object-based models decompose an information space into *objects* or *entities*. According to Mattos *et al* (1993), an entity must be identifiable, relevant and describable. This description is provided by static properties (describing the state of the entity), behavioural characteristics (methods associated with the entity) and structural characteristics (placing the object in the overall structure of the information space).

Worboys (1995) states that it is not necessary to implement object-based models using the object-oriented approach (see section 2.2.3 above) and that the object-oriented approach can be used as a framework for describing both field-based and object-based spatial models.

Whatever the chosen model, implementation issues of the model must be considered. Egenhofer *et al* (1999) describe the specific characteristics that make spatial data and processes unique in the way that they must be addressed.

Spatial databases contain multi-dimensional data with explicit knowledge about objects, their extent, and their location in space. The relative position of objects may be stored explicitly or implicitly. Moreover, the structure of a spatial object may be composed of either a single point or several thousands of polygons, with no regular shapes or patterns of distribution across space. Therefore, it is very difficult to store collections of these objects in a single relational table with a fixed tuple size (Egenhofer *et al*, 1999).

The highly dynamic nature of spatial data is also important: new objects are added, old ones are deleted and some are simply updated in their position, shape or in their alphanumeric attributes. Spatial data structures have to support this dynamic behaviour without deteriorating over time. Because of the large volumes of

information it is necessary to be able to seamlessly manage the data not only in memory but also in secondary as well as tertiary storage facilities.

These problems have been addressed by adopting new developments in database research, specifically, extensions to the relational data model and flexible object-oriented approaches for spatial information. Egenhofer *et al* (1999) survey some of these efforts.

2.5.3.2 Spatial Data Types

Traditional commercial databases do not provide spatial data types. However, as stated by Egenhofer *et al* (1999), they are a crucial requirement in the processing of geographic data. The authors describe several representations, some of which, being quite primitive, have been replaced by more powerful methods.

Vertex lists: a vertex list is a list of polygon vertices. It is sufficient for basic graphic output and it can support certain similarity operators. However, it presents several problems in the comparison of polygons and in the application of set operations, making it quite difficult to determine if two vertex lists represent congruent or similar polygons. Redundancy also becomes quite difficult to identify, in the case of polygons that coincide in one or more points. This type of structure has been replaced by representations with an improved capacity for capturing topological properties.

4D point structures: the representation of a line as a 4-dimensional point (and of a polygon as a list of such lines) is another structure that presents similar deficiencies to vertex lists. Concerns here lie with the significant loss in semantics and with the major problem of relying on coordinates to determine identity between objects.

Long fields: most commercial (relational) databases provide long fields (also known as binary large objects – BLOBs) which can store geometric data structures. These fields simply serve as containers for the data, and only the specific application

programs can decode the information inside. The problem with these structures is that they cannot be queried using SQL.

Abstract Data Types (ADT): the authors quote this as the more robust way of integrating complex types into database systems. Object-oriented and object-relational database systems use the concept of abstract data type to define the structure of object classes. These can then be used to encapsulate the implementation of a data type in such a way that it can only be manipulated through a set of well-defined operators. Egenhofer *et al* (1999) believe that object-oriented concepts and the properties associated with their implementation can easily be adapted to the implementation of spatial data types and operators.

2.5.3.3 Spatial Access Methods

According to Egenhofer *et al* (1999), retrieval queries on a spatial database often require the fast execution of geometric search operations (e.g. point or range queries or spatial joins). The most important characteristic is, normally, the massive size of the spatial datasets that need to be searched. Therefore, GIS applications are for the most part, disk resident. Sequential search being too slow for this type of application, spatial search operators need support at the physical level that guarantees good performance for spatial query processing. This support becomes more important as the database grows.

Access methods for secondary storage management coordinate operations with the operating system, to optimise overall performance. The goal in the design of spatial access methods is to minimise the number of operations to secondary storage bearing in mind the physical organisation of storage devices. Although most spatial searches are I/O-based³, as opposed to CPU-based⁴, applications with objects of complex shapes may need a higher percentage of CPU and thus may imply a change in the balance with I/O operations (Egenhofer *et al* (1999).

³Meaning that an important percentage of the time of the search is spent in accessing peripheral devices, in this case secondary storage devices.

⁴Where most of the time spent executing the search is devoted to calculus.

As current secondary storage devices are linearly structured (based on relational tables), and because most spatial queries and interesting geographic configurations are related to the neighbourhood of a specific phenomenon, the existence of a total ordering among multi-dimensional spatial objects that would preserve spatial proximity would be of major importance (Egenhofer *et al*, 1999).

One-dimensional access methods (e.g., *linear hashing, extendible hashing, the B-tree*) are an important foundation for the creation of multidimensional access methods. From these, Egenhofer *et al* (1999) describe some approaches that have been developed to handle multidimensional search queries. The first one consists in the consecutive application of one single key structure per dimension. Another interesting approach was to extend hashing by using a hash function that takes a d-dimensional vector as argument.

The development of spatial access methods has been one of the most extensively investigated in the computational implementations of GIS. Gaede and Güther (1998) describe a large variety of these.

2.5.4 Agents in spatial problems

There exists very little work on Intelligent Agents that are in any way related with spatial issues. In this thesis, three major areas of space and agents research are identified: Spatial Simulation, Spatial Decision Making and Interface Agents in GIS.

2.5.4.1 Spatial Simulation

The implementation of spatial simulation has had its roots in what are called recursively-generated objects in section 2.3.2 (Wolfram 1994). Cellular automata, one of the models that can be included in this classification, were and are currently used to model and simulate forest fires (Gonçalves, 1997), and scientific processes which involve molecular diffusion. Nobre and Câmara (1995) use cellular automata associated with sketching to generate visual evolution rules of spatial objects. This technique was later used to extract spatial and temporal information from objects

moving on a video (Cámara and Nobre, 1997). The authors applied this technique to an oil spill model.

As stated in section 2.3.2, existing recursively-generated objects have evolved in several directions and some of the results significantly resemble the concept of the agent. To Ferrand (1996), the idea of simple components that, as a whole, produce complicated patterns of behaviour can be compared to sets of reactive agents, interacting simple entities, acting on reaction to stimuli.

This thread has been pursued at the Santa Fe Institute, where the Swarm Simulation System was developed with the objective of providing researchers with standardised simulation tools (Minar *et al.* 1996). A Swarm is a collection of independent agents interacting through discrete events. Swarms can be used to build truly structured dynamic hierarchical systems, as each agent in a Swarm may be a complex system (another Swarm) and vice-versa.

Swarms have already been used to create a combat model in a project that uses fuzzy-genetic decision optimisation for positioning of Military combat units. This project is an ongoing effort to provide decision support to tactical operations at the brigade and battalion level (Kewley *et al.*, 1998).

Another project using Swarms, uses several simulations to determine under which conditions economic agents would agree to pay extra cost of devices to prevent polluting the environment (Weisbuch *et al.*, 1994). Economic agents' opinions depend on space and time, because they exchange information among them and because they receive information on pollution diffusion. The model takes into account several coupled dynamics: the dynamics of pollution, of agent internal representations and of their choices.

Ferrand (1995) quotes work at LAMA-IGA (France) where the limitations of Cellular Automata for dealing with complex diffusion processes led to the conversion of the work towards Multi-Reactive-Agents Systems (MRAS). In project AGRIPA, the assessment of the risk of urban area scattering depending on different road

building scenarios is done with the help of a simulation of accessibility of space through the road network from any point. Agents are entities that move in the space according to local constraints and that, at each stop, mark the cost they have computed. Diffusion between points of space is described by a transition schema linked to the nature of the point (be it a road, a forest, etc). Agents reproduce themselves to explore all possible directions at specific points. The new clone lives as long as the cost it holds is lower than the marked one (Touret, 1995; Vigneron, 1995).

The interest of this approach, when compared with normal diffusion cost calculus, is that it can deal with any type of complex diffusion process, like paths crossing above bridges (Ferrand, 1995). This system is built on top of raster GIS, GRASS. However, Ferrand (1995), also argues that these systems can be built on raster or vector data. When working with raster data, the raster will represent the environment while agents are external processors that act upon it. In the case of vector data, the notion of environment disappears and it is necessary to define which objects belong to the environment and which will be active structures represented as agents.

2.5.4.1.1 Individual-based Models

Individual-based Models are simulations based on the global consequences of local interactions of members of a population (Reynolds, 1997)⁵. These *individuals* might represent plants and animals in ecosystems, vehicles in traffic, people in crowds, or autonomous characters in animation and games. The structure of these models generally includes an *environment* where the interactions occur and a set of individuals that are defined by their *behaviours* and characteristic *parameters*.

Individual-based models are also known as *entity* or *agent* based models, and as individual/entity/agent-based simulations. Some individual-based models are

⁵Reynolds organised an Internet site, <<http://hmt.com/cwr/ibm.html>>, as an annotated list of links on Individual-Based Models. In this site, the reader can reach several examples of spatial simulation using this type of structure.

spatially explicit, which means that the individuals are associated with a location in geometrical space. These are the models relevant to this work, and Reynolds quotes several projects which are or have been using this technology, some of these referenced above (e.g.: the work which uses Swarms as the key structure for building agent systems). For more examples of this, refer to (Reynolds, 1997).

Reynolds also mentions the overlap of these models with CA. He states that CA are similar to spatially-explicit, grid-based, immobile individual-based models. However, CA are always homogeneous and dense (all cells are identical) where as a grid-based individual-based model might occupy only a few grid cells, and more than one distinct type of individual might live on the same grid. This type of argument could be fought by CA researchers but, as Reynolds concludes, the philosophical issue is whether the simulation is based on a dense and uniform dissection of space (CA) or based on specific individuals distributed within space.

The agents created for spatial simulation are usually very simple as a structure and as local entities. The architectures built are reactive ones, and the results provided are based on the emergent properties of this type of system. Most of this work has either been created for the purpose at hand or has been based on the infrastructure given by raster GIS using several layers of Cellular Automata to model individuals (Slothower *et al*, 1996). It is more difficult to find work that involves entities moving in a spatial environment that is continuous in space and time.

2.5.4.2 Spatial Decision Making

In the area of Multi-criterion Spatial Decision Support, Ferrand (1996), proposes the use of agents to: solve the complex spatial optimisation problems encountered in the search for least environmental impact area for infrastructures and to support and simulate the exchange and dynamics of spatial representations and policies. SMAALA, a Support System for Spatial Analysis, developed at IMAG in Grenoble, integrates sensitivity maps given by experts, structural constraints of the project and political positions through the use of reactive agents that represent objects in space. Each political position is provided by a different representation of the area of

study and the process of negotiation is then simulated through the use of attraction/repulsion forces that represent the constraints presented by each actor (political position) in the process. If the system manages to reach an equilibrium, the final planning decision will have been reached.

Ferrand (1995) also quotes a project related to cartographic generalisation where MRAS have been in use. It aims to “support the creation of maps starting from a set of data, taking into account a thematical focus and graphical constraints linked to the used symbols and the final resolution” (see Baeijs *et al*, 1996, for more information).

In Papadias and Egenhofer (1995), agents are being used in Qualitative Collaborative Planning. Agents represent topological, direction and distance constraints that are applied to a spatial planning problem. The process of planning is executed in three steps: modelling the various constraints in a unified framework (enabling interoperability among agents); checking the satisfiability of the imposed constraints and searching the spatial database using these constraints. The focus is on using spatial access methods that can efficiently process qualitative constraints (represented by agents).

Also, Toomey *et al* (1994) describe a project that aims to implement software agents to help the dissemination of remote sensing data and presents an architecture in which agents communicate with each other in order to locate or, if necessary, produce the information requested by the user. An agent-based Remote Terrestrial Sensing (RTS) data dissemination environment has been built using funding from NASA's technology commercialisation program. This application allows the user to specify the desired imagery's geographic region location (by drawing it directly on a world map) and to specify constraints on other image attributes. These constraints are introduced using generic RTS domain terms instead of database specific ones. The Agents in the system are in charge of data sources. A Data Broker Agent leads the communication between agents.

These agents present properties of communication as well as collaboration, typical of interactive and social models of agency. The area of spatial decision making, be that on what spatial movement to take in order to solve a problem or what spatial information to retrieve can gain from becoming more adaptive and from integrating learning of previous experiences into the developed systems. The reformulating of objectives based on those experiences can also be successful.

2.5.4.3 Interface agents in GIS

Interface Agents are semi-intelligent, semi-autonomous systems that assist users when dealing with one or more computer applications (Kozierok and Maes 1993; Maes 1994). The metaphor used is that of a personal assistant collaborating with the user in their work environment. Current GIS applications, manipulating large volumes of data in complex and intense processes have been considered difficult to use and lacking in flexibility. The use of interface agents in GIS has been shown to improve the usability of applications.

Campos *et al* (1996), describes a knowledge-based interface agent for ARC/INFO that receives and processes user's requests in plain English. The agent takes this information and generates sequences of commands that ARC/INFO can understand. If the concepts known to the user are not confirmed by the ARC/INFO database, it interacts with him/her to clarify the misconception. After execution, the agent delivers and presents the results to the user.

The work in this area usually involves adaptive systems and learning. However, the specificity of learning about space does not seem to have been thoroughly researched, especially while developing GIS interfaces.

2.6 Discussion

As stated by Rasmussen *et al* (1994), pg. 1, "Computer-based interfaces are being inserted between humans and their work and advanced communication networks serve to integrate the operation of large scale distributed systems". Because of this,

the development of computer applications has diversified in such a way that (Rasmussen *et al*, 1994):

- Designers need to select and integrate data into information representing the work domain;
- The development of display formats takes into account hypotheses about the nature of mental models and staff working strategies.

The authors introduce two different perspectives to conceptualise work systems: *functional abstraction* versus *structural decomposition*. Functional abstraction is based on the space of the possible actions an actor may take while structural decomposition addresses the study of the elements that compose the system.

Rasmussen *et al* (1994) also draw a distinction between *relational* and *causal* representations. Relational representations involve mathematical relations between identified variables; causal representations describe the temporal sequences of events in the system. The former approach can be used to represent deterministic relationships between sharply defined and discrete variables (quantitative representation of relationships) while the latter establishes the causal relations between prototypical events and objects (given a well-documented text).

These two views are put together to form the functional, relational perspective and the structural cause-and-effect perspective applied to different work control strategies.

The Open-Loop control strategy involves planning of the work task up until the moment where work is in progress. It is not possible to correct errors afterwards. To implement this strategy, the structural causal perspective is used. The functional, relational perspective is mostly used in closed-loop situations that can be controlled through feedback.

Rasmussen *et al* (1994, pp. 15-16) argue that the open-loop strategy has been extensively used where the pace of evolution in the execution of the task was slow.

Software development, with the fast pace of technology and the quick evolution of users abilities with software packages, needs to be addressed according to the closed-loop strategy.

As the authors state(pg. 24), “the analysis and design of modern dynamic work systems cannot be expressed in terms of stable task procedures”. Rather, the analysis of work systems must involve the shaping of behaviours through goals and constraints that define the boundary of a space within which the actors can improvise, guided by their subjective performance criteria.

It is argued in this thesis that intelligent agents are capable of closed-loop operation unlike most modelling procedures, which use open-loop approaches. The learning facilities associated with agents enable them to review their behaviour during execution, in simulation environments or in the manipulation of application interfaces. Moreover, by studying agent behaviour, it is possible to identify the “prototypical work situations” (Rasmussen *et al*, 1994, pg. 29) that are important in a system and customise them accordingly.

Taking into account the aim of this dissertation, to analyse the potential of research in intelligent agents in geographic information science and to explore the use of simple learning techniques to improve the adaptability of spatial intelligent agents, a list of objectives were defined.

The review presented in this chapter aimed to analyse the needs of research in GISc in the areas of reasoning about geographic space, specifically for the following research topics: the identification of and access to special interest geographic information through the use of metadata structures, by a non-expert user with very specific needs; the integration of these processes into adaptable user interfaces for spatial information systems; and the development of systems that can provide the simulation of spatial processes resulting from the individual execution of spatial tasks.

It was also of importance to review the potential application of intelligent agents in the study of the above research questions. Specifically, it was important to determine how agents could be used for specific-purpose geographic information location (identification of searched for information) and access; the improvement of spatial information systems interfaces and the simulation of evolving spatial environments from the modelling of spatially aware individuals.

In this context, a review of the generic use of intelligent agents as well as in the specific areas of spatial simulation, spatial decision making and in the development of user interfaces was performed. Through this review it was possible to provide an overview of the areas of application of intelligent agents and the analysis of the state of the art of current implementations in intelligent agents with geographic information.

From the review on generic agent issues, it was possible to conclude that this is a research field that is growing fast with the help of developments from several areas, not only from computer science in general and artificial intelligence but also from work developed in the natural sciences.

Intelligent agents are not a pure research field because they are, in their specific nature, interdisciplinary. As Wooldridge (1997) argues, agents are not purely based on one type of existing technology; they use the most useful characteristics of several areas of computer science. In fact, Wooldridge's statement on the intelligence requirements of agents reflects the potential of using agents to solve complex problems: "the only intelligence requirement we generally make of our agents is that they can make an acceptable decision about what action to perform next in their environment, in time for this decision to be useful". By using object-oriented technology, by applying techniques that have been used in distributed computing, client-server applications and artificial intelligence, and by being able to act, while embedded in a dynamic environment, in a timely fashion, agents have become a new paradigm for software development, which can be applied in many different areas.

One of the aims of this work is to show that this is true for several existing research questions in Geographic Information Science (GISc). From the study of the fundamental spatial concepts and the discussion of a key issue of geography, the geographic individual, it is possible to argue that GIS are currently raising some very important cognitive questions for which there are currently very few answers.

However, the point made by several theorists is that geographic space should be handled as a set of geographic objects or entities, describing these in an intuitive fashion. This view supports the notion provided by Egenhofer *et al* (1999) that the most robust way of computationally handling geographic concepts is by using an object-oriented approach. According to Worboys (1995), this can be done even if the available technology is not object-based.

The point to be made in this dissertation is that, the use of agents to represent dynamic individuals located in space or who are spatially-aware not only supports the views presented above, but it provides additional advantages through properties which can exist in agent systems like *reactiveness*, *proactiveness*, *emergence* or *learning*.

Regarding existing agent solutions for the relevant research questions addressed by this dissertation, the following issues arose:

- Very little time and effort has been devoted to creating intelligent agents that may facilitate the use and learning of a GIS user interface. This is an open field where the simple use of hard-wired task facilitators could be useful, let alone intelligent assistants that can improve the interface for every-day use;
- It is clear that the availability of massive geographic datasets reflects a need for structures and processes that will enable the everyday user to access information with precision. The reviewed work on adaptive interface agents does provide a framework for the creation of software assistants that can help the user achieve that. The additional challenge is to enable the assistant to learn

the geographic preferences of the user, both in terms of geographic location and type of geographic information;

- Existing work into spatial simulation and spatial decision making does not involve individuals that consider spatial properties among their learning facilities. Although the presented examples involve individuals that are spatially embedded in an environment and are interacting with others of the same kind, they do not consider their spatial position, that of other individuals and the spatial implications of their actions to improve their performance.

From the issues resulting from the above review, the remaining chapters in this thesis were defined. It is worth saying, at this point, that the order of development of this work is reflected in the order of the chapters. The two main areas of the thesis, spatial interface agents and spatial simulation were implemented in two different periods of time and for each, prototype implementation followed the definition of the methodology. The ordering of the dissertation reflects the description of complete methodological issues before addressing prototype development. The following description of the remaining chapters in this thesis reflects a thematic approach.

Chapter 3 discusses the possibilities for creating adaptive interface agents to facilitate the use of GIS and describes a methodology for the development of such a type of assistant.

Chapter 6 instantiates the described methodology through the description of an intelligent assistant that provides identification and location of specific-purpose spatial information for a user through the search of a metadatabase of available geographic information.

Chapter 5 illustrates how a simple non-adaptive interface assistant, developed in a short period of time can already bring benefits to GIS use.

Chapter 4 discusses the issues of spatial simulation and spatial decision making and presents a methodology for endowing spatially-positioned agents with learning

techniques that take the spatial properties of the environment and of the agents themselves into account in order to improve decisions and performance. Chapter 7 demonstrates the application of this methodology to a car park agent simulation.

Both the agents in chapters 6 and 7 are endowed with learning capabilities but they include previous behavioural experience to improve future action. These are simple learning techniques that enable the developer to decide what are the most important characteristics that the agents should learn.

Ch. 3

Interface agents for spatial knowledge handling

“Why do the old become silent when they should keep speaking, and because of that, the young have to learn everything from the beginning?”

“Porque será que os velhos se calam quando deveriam continuar falando, por isso os novos têm de aprender tudo desde o principio ?”

José Saramago, Baltasar and Blimunda, 1982

Interaction between a user and a computer application is provided by an interface. The user starts by having a task to perform. This task must be realised at the cognitive and social (work) levels. Increasingly often the use of a computer application is an integral part of the task. The domain of knowledge and processes that enable the fulfilment of the task must then be aligned with the information

and activities that the application will provide (Rasmussen *et al*, 1994). Access to these by the user will be provided by the application interface.

These interfaces are currently created according to Human-computer Interaction (HCI) rules following an interaction metaphor. The current dominant interaction metaphor, called *direct manipulation* (Shneiderman, 1998), is no longer accepted as the exclusive model for this interaction (Maes, 1994). Maes argues that direct manipulation requires the user to initiate all tasks explicitly and to monitor all events and that this may slow down untrained user productivity. Besides, the information overload phenomenon must be dealt with by taking the load of selecting and finding (at least part of) the information from the user.

Direct manipulation requires the user to initiate all tasks explicitly and to monitor all events. However, as non-technical users become more common and tasks like acquiring news and information and receiving and sending mail become generally computer-based, it is necessary to make interaction metaphors evolve. According to Maes (1994), one way to do this is by using a complementary style of interaction, called *indirect management* (Kay, 1990), where the user is engaged in a cooperative process in which human and computer agents both initiate communication, monitor events and perform tasks.

Draper (1993) presents the notion of task a being more than the act of execution. The author argues that a task is composed of the following:

- Performing the function;
- Verifying success: almost always, users do not want to achieve a goal, but to know they have indeed achieved it. This is why the user needs some way of evaluating the success of the task;
- Discovering how to perform the function: the first time users need to execute a specific function, they will want to somehow to understand the method that was used. Draper (1993) calls this a knowledge getting task;

- Given the visible presence of the several tools that enable the execution of the task, the users may wish to discover the function, through learning by exploration. This will require visible and comprehensible effects of actions.

Maes (1994) argues that this type of interaction can be implemented using autonomous agents that represent *personal assistants* collaborating with the user in their work environment. The author also argues that the set of tasks or applications an agent can assist a user with is virtually unlimited (Maes, 1994): information filtering, information retrieval, mail management, meeting scheduling, selection of books, movies, music and so forth. In this chapter, the argument presented is that interface agents can prove useful in assisting users that are involved in performing geographically-based tasks⁶. In fact, geographic issues have never been considered as part of the cognitive concepts that may be used in the facilitation of user interfaces.

3.1 Approaches to building interface agents

In Maes (1994), a thorough description of approaches for building interface agents is presented. The author describes existing approaches and proposes a method, which is being used by the Autonomous Agents group at the Media Laboratory in the Massachusetts Institute of Technology (MIT) for the development of learning interface agents. This section, based on Maes (1994), takes Pattie Maes' arguments as a basis for the development of interface agents in this thesis. Maes measures the existing methodologies according to two main problems, which have to be considered when building interface agents:

- *Competence*: how does the agent acquire the knowledge it needs to decide when to help the user, what to help the user with and how to help the user?
- *Trust*: how can we guarantee the user feels comfortable delegating tasks to an agent?

Maes (1994) considers the following approaches for building interface agents:

- Making the end-user program the interface agent (e.g. Lai *et al*, 1988);
- The knowledge-based approach (e.g. Dent *et al*, 1992);
- The machine learning approach (e.g. Kozierok and Maes, 1994).

3.1.1 End-user programming approach

An example of the first approach is Lai *et al* (1988), where users can represent different types of information by defining and modifying templates for various semi-structured objects. The information is then kept in a form that can be intelligently processed by people and their computers.

Users collect this information as objects in customisable folders, and create their personalised displays that summarise relevant information in table or tree formats. Users can also create semiautonomous agents, and specify the rules the agents will use for automatically processing of information in different ways at different times.

Using this method, the authors have created applications for task tracking, intelligent message sorting, database retrieval and hypertext purposes.

The problem with this approach is that it requires too much insight, understanding and effort from the end-user (Maes, 1994). The user has to recognise the opportunity for employing an agent and he/she must go through the work of creating it and endowing it with explicit knowledge (which must be done by using an abstract language). Finally, it is also necessary to update the agent's rules over time (as work habits or interests change). This problem is related to the competence issue mentioned above. Trusting the agent is less of a problem in this approach, since the user will most probably trust his or her programming skills. Nevertheless, programs may behave differently than we expect, even when we trust our skills.

⁶The meaning of geographically-based tasks in the context of this dissertation refers to tasks in which geographic entities are important to a successful execution.

3.1.2 The knowledge-based approach

The second approach relies on providing the agent with extensive domain-specific background knowledge about the application and the user. During execution, the agent uses the knowledge it holds to recognise the intentions of the user and to find opportunities to assist him/her. One of the examples of this type of agent is UCEgo (Chine, 1991), an interface agent that helps users solve problems when using the UNIX operating system. UCEgo has a large knowledge base on how to use UNIX and it can volunteer information or correct the user's misconceptions.

Maes (1994) argues that the problems with this approach are related to both the competence and trust issues. First, this type of agent requires a huge amount of work from the knowledge engineer, with application-specific and domain-specific information (which can only be used for the specific agent and task at hand) being entered into the agent's knowledge base. Once entered, this knowledge is fixed and cannot be changed later on. This means that the agent cannot contemplate any change in the user's preferences or in the task to perform. On the trust issue, it is very difficult to convince a user to trust an agent that is very sophisticated, qualified and autonomous from the start. The user needs to feel in control, even when he/she is being helped. Besides, if someone else has programmed the agent, it may be difficult for the user to understand the way it works and what its limitations are.

3.1.3 The machine learning approach

By comparison Maes (1994) proposes the Media Lab's approach to creating interface agents, which is based on *machine learning* techniques. This approach, also pursued in Dent *et al* (1992) and Mitchell *et al* (1994), is the basis for several projects of the Autonomous Agents Group at the Media Laboratory of the Massachusetts Institute of Technology (MIT), e.g. Kozierok and Maes (1993); Lashkari *et al* (1998); Shardanand and Maes (1994). From these projects resulted several BSc and MSc thesis (Kozierok, 1993; Metral, 1993; Shardanand, 1994). The group is still working in this area with new projects being proposed regularly.

These projects were created to test the hypothesis that under certain conditions, an interface agent can learn the knowledge it needs to assist the user. These conditions are (Maes, 1994):

- The use of the application has to involve a substantial amount of repetitive behaviour;
- This repetitive behaviour is potentially different for different users (that is, it can be personalised).

If the second condition is not met, the knowledge-based approach may prove to be more adequate. If the first condition is not met, a learning agent will not be able to learn, as there are no patterns to recognise in the user's actions.

As noted earlier, the machine learning approach is based on the metaphor of a personal assistant. As a human assistant who has recently entered an organisation, a learning interface agent will need some time to become familiar with the habits and preferences of the user. It can gradually become more helpful and competent as it learns from watching the user work, from instructions he/she provides or from other more experienced interface agents. This approach is also a good solution for the trust problem. As the agent gradually improves its abilities, the user is given time to build up an idea of how the agent learns and performs tasks. Trust develops from that process. Maes also stresses the fact that this particular learning approach allows the agent to give "explanations" for its reasoning and behaviour in terms of past examples. This language is familiar to the user and will improve trust conditions.

The use of the machine learning approach enables the agents to gain competence from studying the user's actions. These agents also perform reasoning and take action while maintaining a low profile in terms of disturbing the user's working environment. This is why this approach was chosen to be used in the work developed in the context of this dissertation. Below, more detail is given in the description of the machine learning approach.

3.1.3.1 Acquiring competence

Specific to this methodology, are the ways the agent uses to acquire competence on the task to perform (see Figure 1). The agent achieves this by (Maes, 1994):

- Continuously “looking over the shoulder “ of the user as the user is performing actions: the agent monitors the activities of the user, and, by keeping track of all of his or her actions over a long period of time, can find regularities and recurrent patterns in these activities. The existence of these patterns triggers the learning process in the agent and enables it to offer to automate user’s tasks;
- Getting direct and indirect user feedback: if the user neglects the agent’s suggestion and takes a different action, he/she is giving indirect feedback to the agent on its performance. The user can also explicitly tell the agent that it should not repeat a specific action;
- Learning from examples given explicitly by the user: the user can train the agent by giving it hypothetical examples of events and situations and by telling the agent what to do in those cases. The interface agent records the actions, tracks relationships among objects and changes its example base to incorporate the example that is shown;

- Asking for advice from agents that assist other users with the same task (and may have built up more experience): if an agent does not know itself what action is appropriate in a certain situation, it can present the situation to other agents with more experience with the same tasks. The agent can also build experience from these suggestions and learn to ask for advice from agents that have suggestions for actions that go along with their user's preferences.

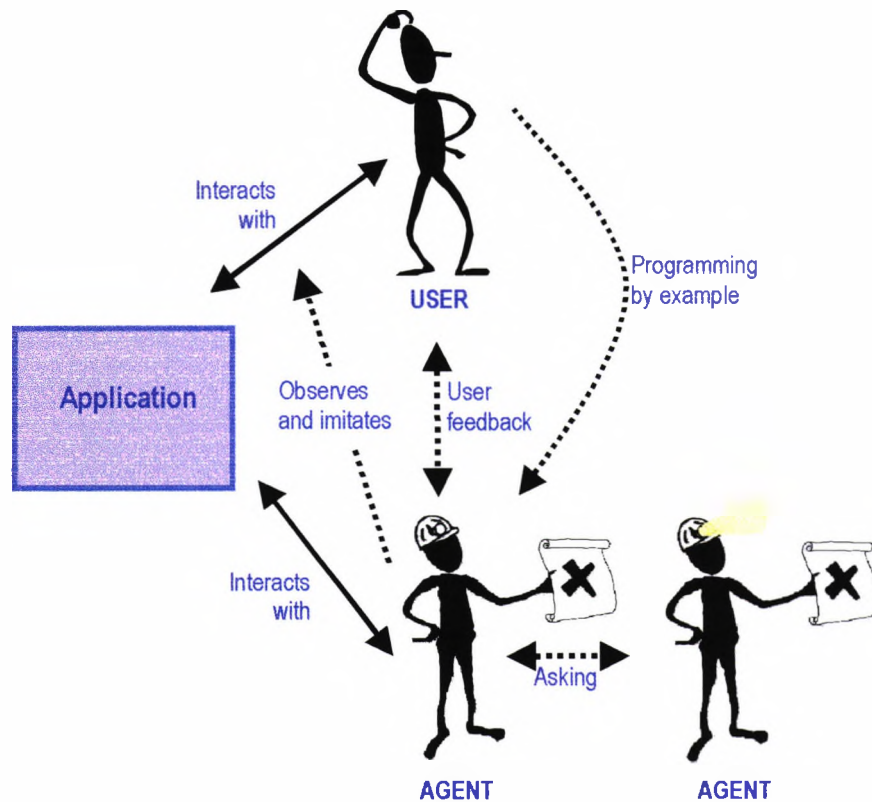


Figure 1 - The agent's learning process (taken from Maes, 1994). The interface agent learns in four different ways: it observes and imitates the user's behaviour, it adapts based on user feedback, it can be trained by the user on the basis of examples and it can ask for advice from other agents assisting other users.

After confronting the approaches for building interface agents, as they have been described above, Pattie Maes associated the following advantages with the methodology proposed by her:

- Less work is necessary from the end-user and application developer, as knowledge builds itself from the experience and learning of the agent;
- The agent can adapt more easily to the user over time and become customised both to individual and organisational preferences and habits;

- The approach helps in transferring information, habits and know-how among the different users of a community.

3.1.3.2 Examples of learning interface agents

A similar approach in the development of learning interface agents is that presented by Dent *et al* (1992) and Mitchell *et al* (1994). Aiming to develop what they call *learning apprentices*, interactive assistants that acquire knowledge through routine use by observing user's actions, these authors have explored the potential of machine learning methods to automatically create and maintain customised knowledge for personal software assistants. As in Maes (1994), the metaphor considered by Mitchell *et al* (1994) is that of a human secretary that starts by carrying out detailed instructions from the user, who has to devote some of his/her time to "training". Over time, the work burden shifts as the human secretary learns enough about the user's preferences and takes over routine aspects of the task. To develop a learning assistant for a specific task, Mitchell *et al* (1994) use the following approach:

1. Provide a convenient interface for the execution of the task;
2. As the system (interface) is used, treat each user interaction as a training example of this user's habits;
3. Learn general regularities from this training data, and use this learned knowledge to increase the services offered by the software assistant.

Mitchell references early work on learning assistants dealing with the domain of VLSI digital logic design (Mitchell *et al*, 1985) and quotes work on a learning apprentice designed to make browsing more efficient through online information sources such as library catalogues (Holte and Drummond, 1994). A prototype system that helps users fill out purchase orders by learning preferences such as which vendors to use for which parts is also mentioned (Nakauchi *et al*, 1991).

Mitchell *et al* (1994) describe CAP, a learning apprentice for calendar management, in detail. It includes an editing and e-mail interface to an online calendar and it

learns user's scheduling preferences through routine use. Several users tested this apprentice, each one getting a different copy of the system. Each copy learned the scheduling preferences of its user and evolved gradually from a passive editing interface to a learning assistant, capable of interacting more intelligently with the user and offloading the work of meeting negotiation from the user.

In the Media Lab work, Kozierek and Maes (1994) and Kozierek (1993) have also presented a meeting scheduling agent, very similar to the one proposed by Mitchell but using different learning mechanisms. These mechanisms were also used to create an interface agent for assisting the user with his/her e-mail. Maxims (Laskari and Maes, 1994) learned how to prioritise, delete, forward, sort and archive mail messages on behalf of the user. These two agents used memory-based learning techniques, based on Stanfill and Waltz (1986). Lashkari and Maes (1994) added the collaborative dimension to this type of agent to solve some problems with the approach. In fact, this type of agent faced the problem of having to learn from scratch, and it took it some time to become useful. Also, its competence would always be limited to actions that it had seen perform. This problem was tackled by allowing it to ask for advice from more experienced agents.

Another project from the autonomous agents group was the news filtering agent (Sheth and Maes, 1993; Sheth, 1993). NewT is an agent that helps the user select articles from a continuous stream of news. It is based on Usenet News and it has been implemented in C++ on a Unix platform. A user can create several "news agents" each one dedicated to a different type of news he/she is interested in, and train them through examples of articles. An agent is initialised through positive and negative examples of articles to be retrieved. Taking those examples, each agent performs a full text analysis to retrieve the words in the text that may be relevant (using the vector-space model for documents in Salton and McGill, 1993). The user can also program the agent explicitly and fill out a set of templates of articles that should be selected. This agent is different from others implemented at the Media Laboratory as it is not meant to automate the user's news interests but rather to

recommend articles to the user about subjects in which he/she has shown interest. Less predictably “interesting” articles must still be found by the user. As stated by the authors, NewT could be improved through the use of social capabilities and by the use of deeper semantic analysis of texts (Maes, 1994).

Recently the trend of research in the Software Agents group at the Media Laboratory has slightly shifted its emphasis. The projects in the group have a new interest in collaborative issues and in the behaviour of societies of agents (see projects Amalthea, Kasbah, Hive, Let’s Browse on the group’s web page for more information, MIT, 1999).

Recent projects following the line of research presented in this section, are:

- Butterfly – an agent that samples thousands of real-time conversational groups and recommends those of interest to the user (Van Dyke *et al*, 1999);
- Yenta – a distributed, privacy-protected, agent-based system that finds clusters of people with common interests (Foner, 1997);
- Footprints – the application of interaction history issues to the problem of social navigation, using information left by other people to help you find your way around (Wexelblat, 1999);
- Expert Finder – the aim of this project is to help people find experts who can assist them with their problems and help leverage knowledge from a community of people (Vivacqua, 1999).

3.1.4 Questioning interface agents

On the subject of the use of direct manipulation as opposed to indirect management, discussed by Maes (1994), Shneiderman (1995) questioned several of the characteristics of agents which are considered to be agent-specific. The author’s evaluation relied on the following components, which he sees as typical of agents: anthropomorphic presentation; adaptive behaviour; accepts vague goal specification; gives you just what you need; works while you do not; and works

where you are not. Shneiderman argues that the first three characteristics, although appealing at first, are in fact counterproductive. The last three are good ideas, but according to the author, can be achieved more effectively with other interface mechanisms.

Shneiderman bases his line of thinking in the fact that the user wants to be in control at all times, and that he/she will refuse any interface that takes that control away. In fact, it is argued here that an effective paradigm involves comprehensible, predictable and controllable interfaces that give users the sense of power, mastery, control and accomplishment. Shneiderman believes that most users want to feel that they have done a good job, not that some machine has done their work for them. He also argues that one should be cautious about the degree of automatic and adaptive behaviour integrated into systems. High accuracy is necessary or a few mistaken choices taken by agents are sufficient to drive the user away.

In another paper (Shneiderman, 1997), the author presents the grander goal of the user interface: a thousand-fold increase in human capabilities. Shneiderman argues that instead of trying to make machines do things that humans can do, researchers should try enable the computer to improve man's capabilities a thousand times.

The idea proposed in the context of this dissertation is that interface agents, by executing tasks on behalf of the user, will not work for them, but will simplify the user's work and, in that way, enable him/her to improve productivity.

As Maes (1994) argues, Shneiderman's direct manipulation metaphor, requires the user to initiate all tasks explicitly and to monitor all events. When the information available to the user arrives in massive quantities every second, direct manipulation will not only be slow, it will be impossible. The user needs to be able to delegate simple filtering tasks or to have his/her interface personally simplified, so that the most common tools are near for fast use. The user does not have to lose

control of the environment in order to have these facilities available, but it may be quite useful to have them close by. Agents are a natural way of achieving this.

3.2 Interface agents that assist users performing geographic tasks

In this section, the assertion that interface agents can help users execute spatial tasks is made. In order to make this assertion, the concept of *geographic task* is defined as a task involving geographic concepts or geographic knowledge.

Draper's notion of task also applies to these concepts, as most of the current geographic tasks will involve several steps or the integration of several types of data. Therefore, the user will certainly want to verify the success of the executed operations and to understand how the results were produced.

This definition is important because it must be clear that a geographic task is not a standard task, and in order to be performed successfully, it must be handled in a geographic/spatial way. This task usually involves the manipulation of geographic/spatial information, or at least the mapping of existing information into a map or some spatial chart. This means that it can be analysed according to two different approaches:

- Geographic data: data that has a spatial reference attached. Under this perspective problems like the location to which the data is associated (a point, a line, a region, etc), the type of data, the integration of several levels (layers) of data or different hierarchies of data, must be considered;
- Spatial processes: How should the data be handled, what are the operations that should be applied on this specific data to solve this type of problems.

The interface of a geographic information system is often complex, difficult to use and manage. This dissertation argues that interface agents can simplify the use of these systems, by frequent execution of repetitive tasks on behalf of the user. The concern with large volumes of information that must be integrated to serve our purposes is also a problem that can be facilitated by an interface agent. The agent

must be capable of learning what types of information are most adequate to perform a specific task and if it can integrate them to ease the user's work.

3.2.1 Previous work

The use of interface agents or intelligent assistants in spatial environments has been, until now, very rare. In an early experiment, Linsey and Raper (1993) developed HyperArc, a fully functional task oriented hypertext GIS interface. HyperArc ran on an Apple Macintosh computer and acted as a client to a host computer running ARC/INFO GIS. The full features of the Apple Macintosh interface were used to perform a range of standard ARC/INFO tasks, including map displays and data queries. The novelty of this type of work, at the time, was that it enabled the creation of tasks and sets of spatial procedures at the users' conceptual level that could be easily customised and executed by an end user, who could have no knowledge of system command language. This work was an early attempt at creating an assistant for the "naïve" GIS user.

Campos *et al* (1996) have also worked in this area, creating a knowledge-based interface agent to help users without a knowledge of ARC/INFO to access and process data stored in ARC/INFO databases. The authors assumed that their user was neither familiar with ARC/INFO, nor with the content or the structure of the ARC/INFO database. They also assumed that the user was not necessarily working on the machine where ARC/INFO is available. The aim of the work was twofold: to use the ARC/INFO documentation files as a source of information about geographic attributes, to partially automate the construction of a domain knowledge base and domain lexicon; and to use an intelligent agent to perform spatial queries in a more natural way, using multimedia language (a combination of natural language and mouse pointing and clicking). An interface between ARC/INFO and a Semantic Network Processing Systems – SNePS (Shapiro et al, 1987) was built. SNePS kept the agent knowledge about data held in ARC/INFO, an automated procedure was built to examine an ARC/INFO workspace and write the relevant information into SNePS. The agent received and processed user's requests in plain English. It

translated this information into sequences of commands that ARC/INFO could understand (plans). If the concepts known to the user were not confirmed by the ARC/INFO database, it interacted with him/her to clarify the misconception. After execution, the agent delivered and presented the results to the user.

This interface agent was knowledge-based and thus did not learn from the user. It was not concerned with capturing the specific preferences of the user in order to execute tasks on his/her behalf. In fact, this is more of a facilitator agent, than an interface agent. It translates the user's requests into something that the ARC/INFO system can understand.

This type of work is similar to that of Etzioni et al (1994), The Internet softbot, a fully implemented AI agent developed at the University of Washington, which uses several Internet tools and the Unix shell to interact with a wide range of Internet resources. It receives requests from the user and it dynamically executes them depending on the transient state of the resources it reaches and on the specific task at hand. It uses Internet facilities such as archie, gopher and netfind as sensors and ftp, telnet and mail as effectors. Like the ARC/INFO agent, it builds plans that will ultimately satisfy the user's request.

The Alexandria Digital Earth project (ADEPT) proposal (ADL, 1999), the follow-up on the Alexandria Digital Library project, is using some of the results from other digital libraries projects in the NSF/DARPA/NASA Digital Libraries initiatives to build on the already available University of California, Santa Barbara Library of geo-referenced information. The purpose of this new phase of the project is to create a distributed digital model of the Earth. A defining characteristic of such a system is its use of the *Earth Metaphor* for organising, using and presenting information at all levels of spatial and temporal resolution. The system will develop digital environments based on the Earth Metaphor, called Iscapes (for Information Landscapes), information service layers in which diverse information resources can be organised to support learning and the creation of knowledge in all disciplines. Iscapes are *virtual Digital Libraries* tailored for specific applications and/or groups

of users. The Iscape is a semantic concept which will be customised to serve a specific purpose according to specific relevant geo-ontology (ADL, 1999a). There is no mention of using agents in the project proposal but there is an explicit reference to a necessity of tailoring and customisation of Iscapes according to user needs.

Another interesting project is an architecture for the definition of agent-based global user interfaces (Tsou and Bittenfield, 1998). The aim of the project is to make available a set of distributed geographic information services by defining the appropriate user tasks (e.g., query, display, data integration and GIS processing functions) through distributed component technology. The concept of the global user interface based on the concept of the global World Wide Web browser is extended to implement GIS on-line services in a platform-independent interface which can access multiple servers and heterogeneous systems. In this architecture agents are used as *information filters* (to limit the user's choices to a reasonable scope) and *information interpreters* (to enable communication between systems resulting from heterogeneous data models). This work is currently under development.

From the above examples, it is clear that work on learning interface agents for spatial tasks has not been addressed. This dissertation aims to open the initial path for that, by demonstrating that these agents are not only useful but needed.

3.2.2 Memory-based framework for developing spatial interface agents

Until now, this chapter has provided information on how interface agents are currently being implemented. The aim of developing interface agents that help users of geographic information systems leads us to the definition of spatial extensions to be applied to any existing successful methodology. Many of the approaches discussed above could have been used as the aim of this work is not to determine which is the best methodology for developing spatial interface agents, but to analyse if any of them can really be useful. The memory-based learning approach (Stanfill and Waltz, 1986; Koziarok and Maes, 1993) has been chosen as

a basis for the design of a methodology in developing spatial interface agents. This approach has been successful in modelling repetitive tasks that handle large volumes of information (Maes, 1994), which is typically true of geographic information systems. This approach also involves learning techniques that can be implemented by development tools associated with existing online mapping systems, as the implemented case-study will demonstrate. However, other approaches could also have been used (e.g. Mitchell et al, 1994).

To ensure that this methodology can be used to develop an interface agent that assists a user with a spatial task, it is necessary to analyse the task at hand, in the following terms:

- Nature of the task;
- Computational application associated with the execution of the task;
- Ontology of the task.

3.2.2.1 Analysis of the spatial task

This analysis must capture information that will help decide whether a learning interface agent will be the right choice for the application at hand.

The analysis of the underlying nature of the task must provide some or all of the following information:

- Definition of the task;
- Aim;
- Type of task;
- Goals and sub goals (if applicable);

The definition of the task will serve as help for the developer, in order to conceptually simplify it. It can be a descriptive text, or a simple flowchart of how it should be performed. It is important to identify if this is a task that can be executed with one request from the user, or if it will involve several requests that will lead to

one final result. That is, if there is one goal, several goals or several sub goals leading to one main objective.

As Maes (1994) specifies, for a successful learning approach to interface agent development, the task must be repetitive. If this is not so, there is no behaviour that the agent can learn by “listening” to the user’s work. Also, there must be potential differences in the repetitive behaviour from user to user. If this is not the case, then all users perform the task in the same manner, and the knowledge-based approach will prove to be a better solution.

If there is going to be an agent assisting the user doing the work, there will most probably be a computational application, which is the tool for performing the task. It is of major importance that this application, if it exists already, be *readable* and *scriptable* (Kozierok, 1993). The application is readable if it is possible to feed the agent with the events that take place as the user works, including requests that are made, results received and data manipulated. The interface agent will keep a record of everything that happens during the user’s work and this must be provided by some inner structure of the application. The scriptable characteristic allows the agent to take actions in the application directly, without having to use its interface. If the existing application does not allow for both of these properties, the implementation of the agent must involve the extension of the application to add these characteristics or, if this is totally impossible, the development of a new application. This has been the case in most of the examples of interface agents presented in this chapter. It may also be the case when the aim is to provide an assistant to a commercial application (like a GIS package). However, if the commercial application includes event facilities and scripting tools, the development of the agent may be facilitated.

The next part of the analysis involves the identification of the ontology of the task. The word ontology has its root in philosophy, where it refers to “the subject of existence” (Gruber, 1993). The Stanford team working in knowledge sharing issues for agents uses the word ontology as a “specification of a conceptualisation (Gruber,

1993). This means that an ontology is a description of the concepts and relationships that can exist in the knowledge sharing of a community of agents. In this sense, ontologies are commitments (agreements) to use a vocabulary when asking questions and making assertions, in a way that is consistent (although not complete) with the underlying theory. An agent commits to an ontology if its observable actions are consistent with the definitions in the ontology. In this work, the word ontology follows the Stanford concept but it has a broader application. Not only will the agent's ontology be the concepts that it will commit to as being known to it, but it will in fact represent the set of concepts that compose its memory.

An interface agent learns how to behave from a database of previous experience that is called a *memory*. This memory will include every situation that has occurred during the agent's life, when the user was working. The situation is translated into the memory of the agent and the action taken by the user, under that situation, is also recorded. This is all the information the agent knows. It is therefore, in totality, its *ontology*. If the agent ever comes to communicate with other similar agents, in order to acquire advice on a new situation, their common concepts will be their communication ontology.

The case of an interface agent that has to learn the specificity of a spatial task will involve learning about spatial processes or spatial data. Therefore, the agent's ontology will probably include spatial operators (intersection, adjacent, buffer) or spatial features (recognising a region as important if it is selected often). This is covered in the detail below.

3.2.2.2 User Model

The design of a learning interface agent is abstract in the sense that its memory will be built from the situations as they occur and from the actions taken by the user. Therefore, the knowledge apprehended by each final working instance of the agent will ultimately depend on whoever becomes its user. This fact has advantages and disadvantages. The advantage is that the user sees the agent learning to do things in the way the user likes it and builds confidence on it. One of the disadvantages is

that the agent will learn slowly, picking knowledge from patterns on the user's behaviour, and gaining confidence on its predictions as they get accepted by the user. The other disadvantage is that the agent also learns the user's bad habits and common mistakes. One way to work around this is to give some initial knowledge to the agent, so that it will be able to execute some simple and obvious tasks from the beginning. This knowledge can even be different for different types of user (e.g. users from the same department can get agents with the same initial knowledge). This introduces the notion of *user modelling* or *user profiling*. The initial agent can be trained into learning the knowledge that is common to people of the same department. It is also possible to hard-wire this knowledge into the initial agent, but that would involve more work into the agent if departmental rules change.

3.2.2.3 Design of a spatial interface agent

If the decision to create an interface agent to aid the user perform the spatial task is made, it is then necessary to design the components that will compose the interface agent, which will be integrated in the spatial system. The design of the interface agent includes the following components:

- The memory;
- The memory manager;
- The agent itself (the reasoning component); The agent interface (or client), if necessary.

Agent Components

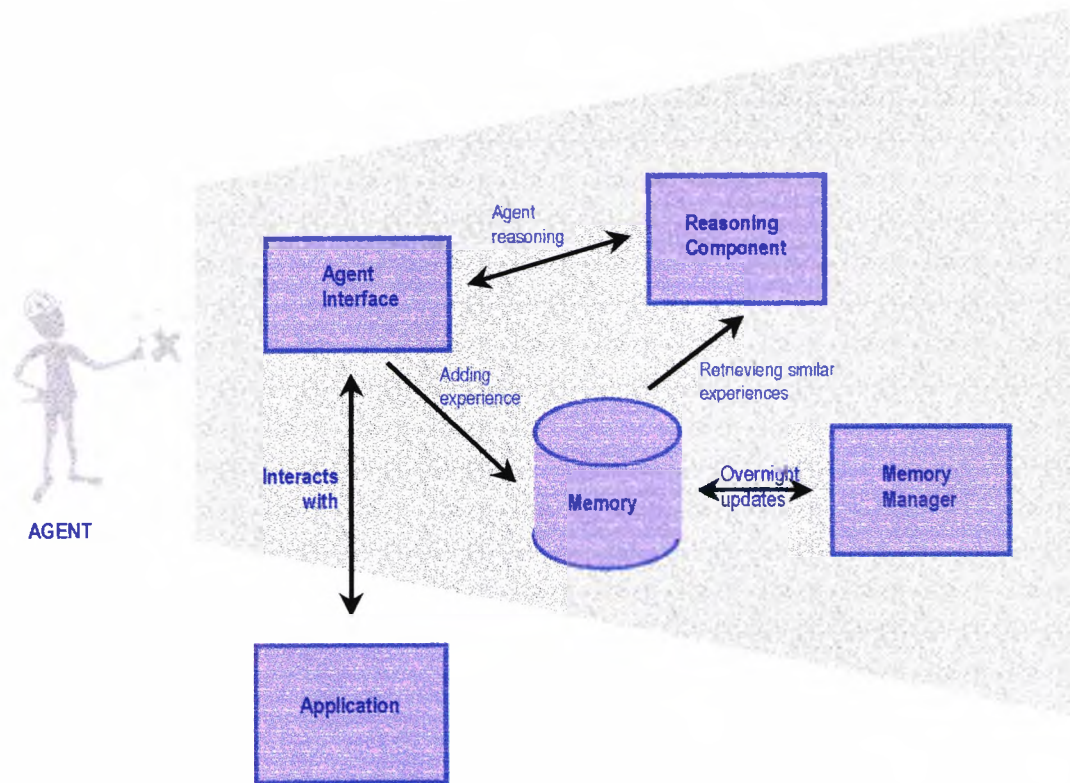


Figure 2 – The decomposition of the agent: the agent interface, the reasoning component, the memory and the memory manager.

The interface agent is an autonomous program that runs continuously and “listens” (through some listening/event mechanism provided by the initial application) to what the user does. Every event that occurs is recorded into the memory and every action that the agent takes for one situation is also recorded. Every time a new situation is presented to the agent, it retrieves the situations stored in the memory and selects a set of situations (probably the n most recent ones), which have led to the same (or similar) action, to be compared with the current one.

All these functions are executed by the agent interface and by the reasoning component. The agent interface is in contact with the user and the application. It records the situations and actions into the memory, so that they can be used later as previous experience of the agent. The reasoning component uses the information kept in the memory to predict new actions when new situations arise. These

predictions (or suggestions) are delivered to the agent interface so that they can be presented to the user.

3.2.2.3.1 Memory-based reasoning

Memory-based reasoning was presented by Craig Stanfill and David Waltz in 1986 (Stanfill and Waltz, 1986) and later improved in Stanfill (1988). Starting in 1993, memory-based reasoning was used in some of the Media Lab's projects on interface agents with the purpose of adding learning facilities to these agents (Kozierok and Maes, 1993); Lashkari *et al*, 1998; Shardanand and Maes, 1994). This is a learning mechanism based on storing information in a database (which we call memory) that describes every situation the user faces while performing a specific task and also the specificity of the action he/she takes when facing that situation.

Each situation is represented by a set of values of various fields, each field representing one feature. One action is also represented by a set of values of various fields, each representing a part of the action taken by the user at that specific situation. Therefore, one *record* in the memory represents one specific situation and the action taken by the user at that situation. This record is composed of a set of *fields*. The set of fields that compose one situation are called *predictors* and the set of associated fields which represent the action taken at the time are called *goals*. A new record representing a new situation (which needs an action taken) will be called a *target* record, because all its goal fields will be empty. The learning process involved in this methodology is one that enables the agent to fill in these fields, after analysing the situation-action records contained in the memory.

For the sake of representation, this is the notation used in this work, based on Stanfill and Waltz (1986), to help the explanation of the methodology:

A *record* will be represented by Greek letters (γ, ρ) and *field* names by italics (n, p) . Specific values will be quoted ('a', 'b'). The letter *u* will be used for unspecific values and λ for empty fields. In this way, field *f* of record ρ will be represented by $\rho.f$. The

set of possible values for a field f is V_f . A non-specific value is represented by an italic letter (v_i). A *memory* M is a set of records.

As said above, a *target* record is a record γ containing some empty fields. The empty fields are called *goals* and the nonempty fields are called *predictors*. The set of goals is written G_γ and the set of predictors is written P_γ . A *feature* is a combination of a field and a value, such as $[f = v]$. The count of the number of items (records) in a full memory is represented by $|M|$. On a restricted memory, this count is represented by $|M_{[f=v]}|$.

According to Stanfill and Waltz (1986), the memory-based reasoning hypothesis is that reasoning may be accomplished by searching a database (memory) of worked problems for the “best match” to the problem at hand. To measure how closely two situations match, it is necessary to develop a metric. A *metric* is a distance measure Δ that satisfies the following properties:

$$\Delta(a, b) \geq 0$$

$$\Delta(a, b) = \Delta(b, a)$$

$$\Delta(a, b) + \Delta(b, c) \geq \Delta(a, c)$$

$$\Delta(a, a) = 0$$

Therefore, the implementation of memory-based reasoning depends on a suitable definition of Δ . The authors present several different metrics, each one trying to reach more accurate results than the previous.

The overlap metric

The overlap metric is the simplest measure of dissimilarity between two records and is based on the number of fields for which they have different values. It is defined in the following way:

Where

$$\Delta(\gamma, \rho) = \sum_{f \in P_\gamma} \delta(\gamma \cdot f, \rho \cdot f)$$

$$\delta(\gamma \cdot f, \rho \cdot f) = \begin{cases} \gamma \cdot f = \rho \cdot f & 0 \\ otherwise & 1 \end{cases}$$

The problem with this metric is that all fields have the same value to the metric. Often, features differ in importance because they differ in the strength with which they constrain the values of goal fields. This is why the *weighted feature metric* was presented.

Weighted feature metric

The different values of a single predictor field may also constrain the values of a goal field in different ways.

The two problems presented above were incorporated into the weighted feature metric by giving different weights to different features. Given predictor field f , the feature's weight $w_f^g(M, \gamma \cdot f)$ is determined by measuring the strength of the constraint imposed by the feature $[f = \gamma \cdot f]$ on the goal field g . The method presented by Stanfill and Waltz (1986) for this metric is to restrict the memory to $M[f = \gamma \cdot f]$, find the frequencies of the various values of g , square these values, sum the results and take the square root. The resulting metric is presented below:

$$\Delta^g(M, \gamma, \rho) = \sum_{f \in P_\gamma} \delta_f^g(M, \gamma \cdot f, \rho \cdot f)$$

$$\delta_f^g(M, \gamma \cdot f, \rho \cdot f) = \begin{cases} \gamma \cdot f = \rho \cdot f & 0 \\ otherwise & w_f^g(M, \gamma \cdot f) \end{cases}$$

$$w_f^g(M, \gamma, f) = \sqrt{\sum_{v \in V_g} \left(\frac{|M[f = \gamma, f][g = v]|}{|M[f = \gamma, f]|} \right)^2}$$

Because these weights depend on the experience stored in the memory, they should be updated every time a new experience is added to the memory. This would be a costly procedure, and the current practice is to create an independent process that re-evaluates the weights overnight, adding the situations and actions added during the day. This process is what we called the *memory manager* in section 3.2.2.3.

This is still not a convenient metric because it is based on the precise equality of the values of the predictor fields. There are cases when different values of predictor fields will be equivalent. It is necessary to insert similarity of values into the metric.

Value differences

What was needed for the metric was a measure of difference between two features. In this way, the penalty given to a field would be the field's weight times this measure of difference.

Given a target γ , a record ρ and a memory M , the difference d_f^g between two predictive features $[f = \gamma, f]$ and $[f = \rho, f]$ measured for the goal field g , is calculated in the following way: Calculate the frequencies of the various possible values of the goal field g in the restricted memories $M[f = \gamma, f]$ and $M[f = \rho, f]$, subtract them, square the results, and sum them over all values of g . The metric is presented below:

$$\Delta_f^g(M, \gamma, \rho) = \sum_{f \in P\rho} \delta_f^g(M, \gamma, f, \rho, f)$$

$$\delta_f^g(M, \gamma, f, \rho, f) = d_f^g(M, \gamma, f, \rho, f) \cdot w_f^g(M, \gamma, f)$$

$$w_f^g(M, \gamma.f) = \sqrt{\sum_{v \in V_g} \left(\frac{|M[f = \gamma.f][g = v]|}{|M[f = \gamma.f]|} \right)^2}$$

$$d_f^g(M, \gamma.f, \rho.f) = \sum_{v \in V_g} \left(\frac{|M[f = \gamma.f][g = v]|}{|M[f = \gamma.f]|} - \frac{|M[f = \rho.f][g = v]|}{|M[f = \delta.f]|} \right)^2$$

Restricting the Database

This metric has further limitations. Different predictors often act together to constrain the values of goals, and the effect of them together may be quite different from taking them separately. The solution to this problem lies in a technique called *restriction*. This is based on applying the memory-based reasoning algorithm not to the entire memory, but to a subset. This subset is obtained by restricting the value of a predictor field (*predictor restriction*) or of a goal field (*goal restriction*) (Stanfill and Waltz, 1986).

Predictor restriction is accomplished by finding the most important field (according to the weight $w_f^g(M, \gamma.f)$) and restricting the memory to those records having the same value in that field as the target record. Goal restriction is accomplished by using the memory-based reasoning algorithm to discover plausible values for the goal field, then restricting the memory to records containing one of those values in their goal field.

Deciding (or Acting)

Once the dissimilarity has been measured between the target and every record in the memory, the decision of what is the most likely value for the goal field must be made.

This is done through the following procedure:

1. Retrieve the records that most closely match the target. This can be done either by setting a threshold or by retrieving the n closest matches;
2. Look at the goal fields of these records. If they are the same, the confidence on the value that they contain is good. If several different values of the goal field are observed, one of them must be chosen. This can be done by assigning, to each record, the reciprocal of its dissimilarity and sum the weight of each observed value of the goal field. The value with the largest weight wins.

There are several ways in which these ideas can be implemented. For examples and more technique details refer to Stanfill and Waltz (1986). In the next section, the concerns relevant to applying this methodology to spatial tasks are addressed, with some possible perspectives for implementation being presented.

3.2.2.3.2 Spatial issues

As stated in the previous section, the problem of applying the memory-based methodology in geographic information systems is related to the issue of finding a metric for spatial features. As a basis of discussion, three types of spatial features were chosen for implementation in this work:

- Location;
- Scale;
- Dataset type.

The problem of identifying which location the user will be interested in, from those available, is a very difficult problem to predict. If the user always picks one specific point location (a town, a country, etc) then the memory-based reasoning algorithm can pick this. The problem arises when the user prefers to work on a region but because the system requires the user to define the region by hand (e.g. drawing a rectangle on the screen), the system can not determine a similarity between the chosen regions. Intersecting the regions and analysing if they have a large intersection area can solve this. This solution must be incorporated into the

memory manager, so that, overnight, the actual frequencies of regions can be updated. Another way to solve this problem is through fuzzy analysis.

The other two problems to solve will require the inclusion of heuristic connections between the various possibilities. Similar scales must be identified as being that in the memory manager, and in the reasoning component. The same must happen when confronting preferences on datasets and when choosing new ones. Further information on this is given in the next chapter, where the implementation of an interface agent for a browser of spatial information is presented.

3.3 Discussion

It is clear that the interface agents created using the memory-based reasoning approach have rational qualities. The other models and architectures discussed in section 4 of chapter 2 do not apply here, as the aim is not to develop an architecture composed of various agents.

Instead, the architecture supporting the agent's performance is based on the work presented by the Software Agents group of the Media Laboratory of the Massachusetts Institute of Technology (MIT). The learning approach for developing interface agents using the indirect management metaphor has been the base for this methodology. The geographic learning extensions will enable an intelligent assistant to identify the user's preferences in terms of geographic locations and information.

This methodology was developed in the context of two of the objectives identified at the beginning of this thesis: to study the potential of intelligent agents for specific-purpose geographic information location (identification of the searched for information) and access and the improvement of spatial information systems interfaces; and to explore the use of simple learning techniques to improve the adaptability of intelligent agents for geographic information..

It is suggested in this dissertation that, the development of geographically aware assistants in this context will enable the user to work with semantically identified

information and processes that represent a preference of the user and will improve productivity and performance on the user's work.

Specifically, the development of geographically aware intelligent assistants should provide:

- Improvements in the use of geographic information system interfaces through growing performance and efficiency in the execution of geographic tasks;
- Facilitation and personalisation of user interfaces through the semantic identification and learning of geographic concepts;
- The identification of user's geographic preferences through the management of previous experiences evolving with time;
- The teaching and helping of users new to specific GIS packages to perform specific tasks.

The prototype assistant for printing and plotting in the Smallworld GIS (presented in chapter 5) and the Spatial Information Facilitator for the World-Wide Web (chapter 6) aim to give two different views of how a user interface may be adapted to the preferences of the user.

The first one is a simple implementation, integrated with the Smallworld GIS framework, which performs a simplification of the printing and plotting task once the user has chosen sufficient attributes for the agent to execute the task. No learning is involved in this work, simply the identification of the task's state-transition diagram.

The Spatial Information Facilitator provides identification and location of specific-purpose spatial information for a user through the search of a metadatabase of available geographic information. It uses memory-based reasoning to evaluate previous working sessions of each user and when it finds patterns of behaviour, it automatically provides the user with the requests most often issued.

Personalisation here is made not only on the operations executed by the user but also on the specificity of the required information.

In each of the chapters, conclusions are drawn that evaluate the methodology, the objectives achieved, those that still need to be reformulated and new ones that have been identified.

Ch. 4

Adaptive Agents for Spatial Simulation

“ Everything should be made as simple as possible, but not simpler.”

-Albert Einstein

The use of multi-agent systems in simulation has evolved, as described in chapter 2, from two different threads of research, with very diverse origins, but which have seemed to integrate as new interdisciplinary work comes into effect. Early researchers in AI focused their work on enabling the computer to perform complex tasks, which were executed by humans, and based their work on the symbolic and complex representation of the knowledge necessary in those tasks. This was the time of expert systems that could give expert advice on specific issues. However, expert systems could not provide answers in real time and did not usually work

directly in the environment where the problems took place. These were the needs of robotics AI researchers, working in systems that involved autonomous robots directly integrated in an environment and taking action in real time. This is how the first reactive agent architectures came about, in works like Brooks (1986, 1990) and Agre and Chapman (1987).

At the same time researchers in the natural sciences were trying to develop tools that could help them simulate the phenomena they were interested in but which could not be studied in a systematic way in the field. This is how Artificial Life (AL) research appeared. The study of a system through the specific study of its elements and interrelations among them led to the development of what are now called *individual-based models* (Lomnick, 1992; Reynolds, 1997; and chapter 2 of this dissertation). Currently, the individual-based approach seems to be conflated with that of agents and specifically reactive agents.

In this chapter, the theory of simulation is analysed according to the view presented by Rasmussen and Barrett (1995). Traditional simulation techniques are compared with the multi-agent approach, and the possibilities of using reactive agents are considered. The specificity of spatial simulation is analysed in a multi-agent context and a framework for the development of agent-based spatial simulations that adapt to changes to the environment through reinforcement learning is presented.

4.1 Simulation

If it is necessary to study phenomena for which no explicit model has been developed, it is common to resort to synthetic methods using computer simulation (Rasmussen and Barrett, 1995). These methods rely on the fact that a simulation is a mechanism that enables the interaction of many state transition models of individual systems and thereby generates system dynamical phenomena. The origin of this approach is described by Lomnicki (1992). If there is a system about which we know very little, so that we are unable to predict and control its behaviour, it is

usual to use the same approach, irrespective of whether it is an individual animal, a cell or a molecule. The approach involves identifying its elements and finding out the properties of these elements and the interrelations among them. This approach is used to predict how the entire system behaves. This is the essence of a fundamental principle of empirical science known as reductionism (Lomnicki, 1992). The author states that scientists who apply reductionist methods do so on the assumption that the properties of the system as a whole are not the simple summation of the properties of its elements, but that there is a major importance in the interrelations among the elements of the system.

According to Rasmussen and Barrett (1995), the dynamical part of a simulation is *implicit* and *constructive*. The relations that realise the properties of interest in the system are not explicitly encoded in the component subsystems that are simulated. They *emerge* and become *observable* as a result of the collective effects of computer interactions among these subsystems. Rasmussen and Barrett (1995) defined a theory of Simulation based on the above formalisable issues and the need to establish an elementary and general concept of the generative concept of emergence. They aimed to separate the dynamics of the simulation mechanism (in itself an iterated system) from the simulated system (represented by the framework provided by the simulation system). The authors wanted to relate the concepts of *emergence* and *simulation*.

In this theory, they start by defining the basis for a general research programme or any application specific programme of investigating or developing control strategies for complex dynamical phenomena using simulation. This programme is effected in four phases:

- Awareness that a simulation is generating dynamical phenomena at a level which is higher than the level from which the elemental interactions are described;

- Availability of methods with which to identify the elements of the underlying system that create the phenomena of interest;
- Development of formulation models of the important underlying subsystems (those that define the elemental subsystems and the element-element or object-object interactions);
- Creation of the framework in which the simulation of the subsystems in interaction is composed and ability to embody the system representation in that framework so that the phenomena of interest can be generated and analysed.

The theory of simulation presented by Rasmussen and Barrett (1995) is based on these four components and is described below.

4.1.1 Theory of simulation

Rasmussen and Barrett (1995) describe *Simulation* as an iterated mapping of a (usually large and complicated) system. The simulated system is usually decomposed to a level where subsystems or system components are individually defined as encapsulated objects that calculate and communicate internal state. The simulation is an iterative system in which the simulated system is represented and its dynamics calculated. The authors find that the interplay between the dynamics of the coordination of the simulation updates and the dynamics calculated in the time series of system states are essential issues.

Therefore they identify four systems, which comprise a simulation:

- Σ_R - a real or natural system in the world that the researcher is interested in;
- $\Sigma_{(S_i \in M)}$ - models of subsystems S_i of this system and rules that define interactions among the subsystems;
- Σ_S - a simulation of Σ_R involving $\Sigma_{(S_i \in M)}$ and some update functional U ;

- Σ_C – a formal (and equivalent physical) computing machine on which Σ_S is implemented.

The formal definitions of these are presented by Rasmussen and Barrett (1995) in the following section:

Objects

The *objects* (elements or subsystems) in a simulation are defined as

$$S_i = S_i(f_i, I_{ij}, x_i, t_i), \quad i \text{ and } j = 1, \dots, n, \quad (1)$$

Where f_i is the representation of the dynamics of the i^{th} object and where $I_{ij=1, \dots, n}$ is composed of the i^{th} object's interaction rules with other objects j . x_i is the state of the i^{th} object, on which interaction and dynamics operate and t_i is the local object time coordinate.

Set of models of the subsystems

S_i is an element in the system $\Sigma_{(S_i \in M)}$, which means that S_i is a *model* of the i^{th} element of the set of modelled system elements $M, I=1, \dots, n$. The algorithmic part of S_i will thus be equivalent to f_i and I_{ij} .

Update functional

An object *update functional* U is the state transition

$$S_i(t_i + \Delta_i) \leftarrow S_i(t_i) \quad i = 1, \dots, n, \quad (2)$$

or

$$S_i(t_i + \Delta_i) = U(S_i(t_i)) \quad i = 1, \dots, n, \quad (3)$$

where U , defines, organises and executes the formal iterative procedure that prescribes the state transition.

Simulation

A *simulation* is the *iteration* of object updates over the entire set of objects

$$\{S_i(t+1)\} \leftarrow \{S_i(t)\} \quad i=1,\dots,n, \quad (4)$$

or

$$\{S_i(t+1)\} = U(\{S_i(t)\}), \quad i=1,\dots,n, \quad (5)$$

A valid update functional U also needs to be able to align all objects at regular intervals or at a given time, perhaps at each update. The dynamical properties of the system are *implicitly* defined by f_i together with I_{ij} , x_i and U . The authors consider that U is the “active” part of Σ_S while f_i , I_{ij} and x_i are its “passive” parts. Thus, the iteration of the dynamics of $\Sigma_{(S^i \in M)}$ using U constitutes a formalisation of Σ_S (the simulation system).

Implementation

Σ_C is the formal, or equivalent physical, implementation of the mechanisms of the iteration procedure that prescribe the interactions, consequent object state transitions and their storage. This system is normally a physically and conceptually digital computer of some kind.

4.1.1.1 Emergence

To study the emergent properties of the simulation, Rasmussen and Barrett (1995) introduce the concept of level of description. As presented in the previous section, n objects or structures are defined ($S^i \in \Sigma_M$) with an update functional U , at some level of description L^1 . At this level, a new function O^1 is introduced by which the objects can be inspected. The iteration of Σ_M using U enables the production of a new structure S^2 over time with the following description:

$$S^2 \leftarrow U\{S_i^1(f_i, I_{ij}^1, x_i, t_i), O^1\} \quad i \text{ and } j = 1, \dots, n. \quad (6)$$

This is called a *second order structure* occurring at level L^2 . At the new higher level, the structure may be subjected to a possible new kind of observer O^2 .

Therefore, The authors define that a property P is *emergent* if

$$P \in O^2(S^2), \text{ but } P \notin O^1(S_i^1) \quad (7)$$

Emergence depends essentially on the observer in use, which may be *internal* or *external* to the system. The property is only called emergent relative to the first level at which it becomes visible to the observer.

This process can be iterated in a *cumulative* way to form *higher order emergent structures* or *hyperstructures*. For example, at order N :

$$S^N \leftarrow U(S_i^{N-1}, O^{N-1}, S_k^{N-2}, O^{N-2}, \dots) \quad (8)$$

It is worth noting that the definition of an observation function is no more – or just as - arbitrary as the objects and their interactions (Rasmussen and Barrett, 1995).

The concept of generalised phenotype from Artificial Life (AL), described in chapter 2, can be related to the Emergence property of simulation, as the behaviours and structures that emerge out of the interactions among the low-level rules called (in AL) the generalised genotype.

Examples

One example of emergent properties in a simulation system is given by the properties of congestion in a traffic system. In this system, a lower level L^1 description of the interactions will include the description of the vehicle-vehicle interactions together with the vehicle-roadway and vehicle-signal interactions (S_i^1 -

S_i^1 interactions). In this example the S_i^1 interactions generate the S^2 phenomena, but the S^2 structures also have a *downward causal effect* on the S_i^1 structures. This means that the traffic jam effect will restrict the dynamics of the vehicles in the simulation. However, according to the authors, some emergent properties do not generate downward causal effects. For example, the joint distribution of heads and tails generated from two independent coin flips is an emergent property of the system, but the distribution does not have any influence on the dynamics of the coins.

4.1.1.2 Simulation and Emergence

According to Rasmussen and Barrett (1995), it is, generally, very difficult to create a direct, a priori, description (model) of the dynamics of the phenomena S^2 of interest in systems consisting of many, interacting, elements with some internal complexity. However, it may be possible to identify the level L^1 , from which the phenomenon of interest emerges and at which it becomes possible, in a direct way, to describe the interactions or the dynamics of the elements or objects that generate S^2 .

To proceed with the authors' description of the theory, let us assume that a formal description of the object-object interactions is possible at level L^1 and that some observation mechanism O^2 exists so that properties of S^2 can be detected and their dynamics followed. Then, at level L^1 , explicit models S_i^1 must exist to describe the dynamics of and the interactions between the n objects where the objects' states depend on each other.

Given this situation, Rasmussen and Barrett (1995) address the definition of a global state dynamics function F^1 , which, they argue, will at least be given implicitly at level L^1 . F^1 is a global function that describes the system wide state changes caused by the object-object interactions described by the set of local f s and l_{ij} s. In fact, the state of the total system given by $\chi^1(t)$ at level L^1 can be obtained through appropriate observational functions O^1 successively applied to each of the objects. The state of the system at moment t is thus given by

$$\chi^1(t) = \{x_1^1(t), \dots, x_n^1(t)\} \quad (9)$$

Which can be computed at any time and may always serve to implicitly provide F^1 at level L^1 . The authors thus state that the description of the dynamics of L^1 is in principle known on the form:

$$\chi^1(t+1) \leftarrow F^1(\chi^1(t)). \quad (10)$$

F^1 is a description function. The *production* of the dynamics of the whole system is given by the update functional U . If it exists, U will organise the update of the interacting set of objects in a consistent way. The dynamics of system Σ_s can thus be *generated* through

$$\{S_1^1(t+1), \dots, S_n^1(t+1)\} = U(\{S_1^1(t), \dots, S_n^1(t)\}) \quad (11)$$

or

$$\chi^1(t+1) = U(F^1(\chi^1(t))). \quad (12)$$

From this, the authors conclude that whenever it is possible to define an update functional U that can organise the interactions of the objects defined at level L^1 through the set of models M , then the L^2 phenomenon of interest S^2 *emerges* and can be observed by applying the observation function O^2 . This is possible even if F^1 is not explicitly known. A recursive application of U to the objects in the S^2 and its dynamics (a property P^2 of S^2) can be followed by a recursive application of O^2 .

From this, Rasmussen and Barrett (1995) redefine simulation in the following way: a simulation is a representational mechanism that is distinguished by its capacity to generate relations that are not explicitly encoded.

Examples

Recalling the above example, S^2 could be a traffic jam described through vehicle-vehicle and vehicle-roadway interactions and P^2 could be the lifetime of a jam. In this example we have

$$S^2 \leftarrow U(\{S_1^1, \dots, S_n^1\}) \quad (13)$$

and

$$P^2 = O^2(S^2) \quad (14)$$

where S^2 in (13) is defined through the implicit (emergent) relations that are generated between the objects on the left-hand side of (11). However, it is important to note that (11) establishes the relation between two states of the dynamics of the simulation at the same level L^1 , while in (13) S^2 results from the complete execution of the simulation and enables the evaluation of new interactions and properties at the higher level L^2 . In fact, formula (13) is a result of (6) in the context of the observer function O^1 .

Studying Emergence of S^2 at L^2

The aim of this process is to be able to directly follow the state dynamics of Σ_R through some Σ_M at level L^2 , requiring that the state variables $\{x_1^2(t), \dots, x_m^2(t)\} = \chi^2(t)$, associated with the state dynamics function F^2 at level L^2 , are known explicitly, so that it is true to write

$$\chi^2(t+1) \leftarrow F^2(\chi^2(t)). \quad (15)$$

This means that the state dynamics can be derived from the current state of the system by applying some F^2 . The knowledge of F^2 would, in principle, also enable some update functional U^2 to produce the dynamics

$$\chi^2(t+1) = U^2(F^2(\chi^2(t))). \quad (16)$$

Rasmussen and Barrett (1995) assume that the system cannot *a priori* be described at level L^2 . However, as it can be described at level L^1 , the dynamics of level L^2 can be *generated* by *simulating* the interactions of the objects S_1^1, \dots, S_n^1 at level L^1 . This means that, the phenomena and relations of interest at level L^2 will *emerge* from the simulation of the interactions of S_i^1 at level L^1 . From this, it is concluded that simulation is a natural method for studying emergence because it is a direct generative way to obtain knowledge of non-explicitly encoded (dynamical) relations and phenomena.

To conclude, science is full of descriptions of systems where we have both an L^1 and an L^2 description (Rasmussen and Barrett, 1995). For example, the Statistical Mechanical (L^1) versus the Thermodynamical (L^2) description of matter; the Lattice Gas Automata for fluid particle dynamics (L^1) versus Navier Stokes equations for macroscopic fluid dynamics (L^2).

4.2 Traditional simulation techniques and their limitations

Traditional techniques of simulation have been based on mathematical and stochastic models, using differential equations, which relate various parameters and describe the dynamics of the systems (Ferber, 1994). These equations provide information on cause-effect relationships as the simulation runs, by relating output variables to input ones. Ecological simulations are one example of this: population size of one specific species can be related to the growth of different species and the number of predations (Ferber, 1994). The following simple formulas defined by Lotka and Volterra express the rate of growth of predator and prey populations (Volterra, 1926):

$$\frac{dN_1}{dt} = r_1 N_1 - P N_1 N_2 \quad \frac{dN_2}{dt} = a P N_1 N_2 - d_2 N_2 \quad (17)$$

Where P is the coefficient of predation, N_1 and N_2 are the prey and predator populations, a represents the efficiency with which predators convert food into offspring, r_1 is the birth rate of prey and d_2 is the death rate of predators.

These equations have been intensely used in the implementation of simulated societies. However, they have shown severe limitations, which Jacques Ferber addresses in a review on the use of reactive agents in simulation (Ferber, 1994):

- *Micro to macro relationships* – All parameters (input and output) are defined at the same level (e.g. a global parameter like population size cannot be related to local ones, like the decision processes of individuals);
- *Complexity and realism of parameters* – The creation of detailed simulations requires the definition of parameters, which will increase the complexity of differential equations. For example, in the Volterra equation above, parameter a , which relates the food taken to the offspring with its growth does not model the actual reality, as offspring are the result of many complex processes and behaviours. To try to include this complexity in a differential equation will involve the creation of awkward parameters whose relation to reality will be difficult to manipulate;
- *Taking behaviours into account* – in differential equations and numerical methods in general, actions are not seen as activities that change the environment. They are seen according to the measurable values which result from them and in terms of their probability of happening;
- *Multi-task behaviours and conditional task switching* – in numerical modelling, actions cannot be considered as proceeding from evaluation decisions whose outcome depends on the conditions of the world (e.g. in numerical modelling, a feeding and hunting process does not describe the behaviour of the predator, only relations between the number of predators and the number of prey in a delimited area);

- *Qualitative information* – numerical simulations cannot represent qualitative data such as the relation between a stimulus and the behaviour of an individual. These relations are beyond the scope of analytical equations and numerical simulations. New computing models and tools are required to enable the capture of the local interactions from which the global behaviour of the population can emerge.

4.3 Individual structures in simulation

In the field of Distributed Artificial Intelligence, cognitive (deliberative) agents are distinguished from reactive agents (chapter 2). Cognitive agents have a symbolic and explicit representation of their environment. They can reason on this environment and predict future events from it. Cognitive agents are driven by intentions, which are explicit goals that conduct their behaviour and enable them to choose between possible actions (Ferber, 1994). In chapter 2, examples of projects that use this approach are given (section 2.4.2.1).

Reactive agents oppose the above approach. They have no representation of their environment and act using a stimulus/response type of behaviour: they respond to the present state of the environment in which they are embedded (Ferber, 1994). Reactive agents follow simple patterns of behaviour that can easily be programmed. Chapter 2 also presents examples of this approach (section 2.4.2.2). The simplicity of this approach has made it widely used in simulation, mainly to represent animal behaviour (Hogeweg and Hesper, 1985; Collins and Jefferson, 1991; Deneubourg *et al*, 1987; Deneubourg and Goss, 1989; Maruichi *et al*, 1987). Work in Robotics when considered as simulations of robots moving around in an environment (Ferber, 1994), can also be quoted (Brooks, 1990; Steels, 1990).

In the following sections a general description of the main characteristics of reactive agents is presented. Based on Ferber (1994), an introduction to the field of multi-agent simulation is presented with examples of several applications using reactive agents being given.

4.3.1 Reactive agents for simulation – characteristics of reactive agents

The volume of research in reactive agents and simulation is already sufficient to enable the definition of their specific characteristics. Jacques Ferber describes these in a seminal review on reactive agents and simulation (Ferber, 1994), which is the basis for the description presented below. These are the major characteristics considered by Ferber (1994):

Cognitive cost and cognitive economy

Reactive agents are designed according to a behaviour-based model of activity, as opposed to the symbol manipulation model used for cognitive agents. The concept of *cognitive cost*, the complexity of the overall architecture needed to achieve a task, can be used to reinforce this distinction in the two approaches (Ferber, 1994). Cognitive agents have an internal representation of the world, which must be adequate to the world itself. To relate the internal representation held by the agents with the world is considered a complex task. Cognitive agents have to support a complex architecture and therefore, their cognitive cost is high.

On the other hand, reactive agents are simple, easy to understand and do not use an internal representation of the world. Not only is their cognitive cost low, they tend to what is called *cognitive economy*, the property of being able to perform complex actions with simple architectures (Ferber, 1994). That is why reactive-type agents are used in this study.

Situation

Reactive agents are situated: They do not take past events into account and cannot foresee the future. Their actions are based on what happens now, on how they sense and distinguish situations in the world and on how they react to that. It is, therefore, impossible for them to plan ahead what they will do.

This can be considered their weakness. However, it is also their strength. They do not have to revise their world model every time it is changed in an unexpected way.

Self-sufficiency

Because of their complexity, cognitive agents are often considered self-sufficient: they can work alone or in cooperation with the user or a few agents. Reactive agents, on the contrary, need companionship: they cannot work in an isolated way and they usually execute their tasks in groups.

Robustness and fault tolerance

Two of the main group properties of reactive agent systems are concerned with robustness. A group of reactive agents can complete tasks even when one of them breaks down. The loss of one agent does not prohibit the completion of the whole task because the allocation of roles is achieved locally through the perception of environmental needs.

Flexibility and adaptability

Robustness and fault tolerance lead to another two major properties of reactive agent systems: flexibility and adaptability. They can manage their resources abilities in unpredictable worlds, and complete the tasks they are engaged in, even in the case of partial failures of the system.

4.3.1.1 Feedback

Ferber (1994) argues that, the emergence of functionality and of stable states is a consequence of the combined forces of the different feedback mechanisms to the agents and to the system as a whole. Ferber refers to positive versus negative feedback and to local versus global feedback. *Positive feedback* tends to create diversities among agents whereas *negative feedback* regulates societies, imposing a conservative force upon their social structures. Both positive and negative feedback may be local or global. *Local feedback* is built into the system by the agent designer and is part of the primitive constructs of agents. *Global feedback* is the result of interactions between agents, its action is not explicitly specified at the agent level.

Positive feedback at the agent level can be implemented as part of a reinforcement process (which will be described in more detail below), for instance, making an

already specialised agent more inclined to perform one specific task. The effect of this is to strengthen differentiations and create disparities among agents. Negative feedback resulting in emergent functionality in the agents' interaction may be the distribution of roles in the society. Whereas local feedback loops are deterministic because they have been implemented as such by a programmer, global feedback is not always deterministically predictable. It often results from autocatalytic processes which, in turn, have resulted from the interactions in the system.

4.3.2 Multi-agent simulation

According to Ferber (1994), the multi-agent simulation model is based on the idea that programs exhibit *behaviours*⁷ that can be entirely described by their internal mechanisms, the program instructions. By relating an individual to a program, it is possible to create an artificial world populated with interacting computational entities. This artificial world is a simulation of a real world, which is populated with the actual entities that have their artificial counterparts in the simulation. As described in section 4.1, a real system Σ_R is simulated using several elements $\Sigma_{(St \in M)}$, some of which are agents. The creation of the simulation Σ_S , an artificial world is only possible thanks to the existence of computational mechanisms Σ_C (Rasmussen and Barrett, 1995). A simulation can therefore be achieved by transposing the population of a real biosystem to its artificial counterpart where each individual organism is separately represented as a computing process (an agent). In this artificial system, the behaviour of an agent during all stages of life is programmed with all the required details.

The primary use of multi-agent simulations is the representation of complex situations in which individuals have complex and different behaviours, to study the global situations that arise as emergent structures resulting from the interaction processes (Ferber, 1994). As mentioned in section 4.2 above, one of the major characteristics of multi-agent simulation is the possibility of considering not only

quantitative (e.g. numerical parameters), but also qualitative (e.g. individual behaviours) properties of the system in the model. Compare this to traditional techniques of simulation in which it is only possible to relate the systems properties to quantitative parameters. As the author argues, this type of simulation enables the construction of a microworld that works as a research laboratory, where particular hypothesis can be explored, and experiments can be repeated and controlled.

In a multi-agent simulation, each individual (or group of individuals) is analogically represented as a computational agent and the behaviour of this agent is a consequence of its observation and interactions⁸ with other agents. The model used is not a set of equations as in mathematical models but a set of the following entities (Ferber, 1994; Russell and Norvig, 1995⁹):

- *Agents* – the set of all the simulated individuals, which are defined by their ability to perceive specific types of communications, their skills in performing various actions, their deliberation model (if it exists) and their capability to relate perception with action;
- *Objects* – the set of all represented passive entities that do not react to stimuli;
- *Environment* – the topological space where agents and objects are located, where they can move and act upon;
- *Communications* – the set of all communication categories such as voice, written materials, media, scent and signs (Ferber, 1994).

The author states that multi-agent simulation can be used for the following purposes:

⁷Ferber (1994) defines *behaviour* as the set of actions an agent performs in response to its environmental conditions, its internal state and its drives.

⁸In Ferber (1994), *interaction* means communications, stimuli influences or direct actions of other agents.

⁹In section 2.1 of chapter 2 a discussion of the fundamental elements of agents systems is presented.

- *To test hypothesis* about the emergence of social structures from the behaviours of each individual and its interactions, by experimenting about the minimal conditions necessary at a lower level for the phenomena of interest to become observable at the next or any higher level;
- *To build theories* that contribute toward the development of a general understanding of ethological¹⁰, sociological and psycho-sociological systems, by relating behaviours to structural and organisational structures;
- *To integrate different partial interdisciplinary theories* into a general framework, by providing tools that enable the integration of disjointed studies.

Ferber (1994) concludes that multi-agent simulation and numerical analysis are not incompatible. However, they should be used at different levels and only combined in that way. Multi-agent models are used as analogical mappings of a real system. From the simulations they provide it is possible to derive global parameters, which can be studied and incorporated into a mathematical model. Therefore, numerical data and statistics are not eliminated from these simulations, rather they are used as evaluation procedures to compare simulation results with “real” world observed data. Mathematical models are used at the macro-level whereas multi-agent simulation models enable the crossing of the “Micro-macro Bridge” through the emergence of global configurations from the local agent interactions (Ferber, 1994).

4.4 Spatial Simulation

Most types of simulation involve some spatial context. The most typical example is the fact that generally a simulation is representing an environment that exists somewhere in space. It may be an ecosystem, a city or an urban area. The environment involved will have a spatial mapping, position or location. The objects embedded in the environment will be spatially located. The agents in the simulation

¹⁰*Ethology* is the study of animal behaviour within a dynamic environment (Franklin, 1995, pp.13-14).

will move spatially. However, most important is the fact that the emergent phenomenon of interest to the researcher will probably have a spatial representation and its evolution may have (will probably have) spatial consequences and spatial properties. The phenomenon itself will probably evolve spatially.

Basic individual-based models (IBM), like cellular automata, are normally associated with a two dimensional or three dimensional grid where the phenomena of interest are represented and have the capability to evolve. More recent IBM's, have shifted their focus from the cell to the individual, but the spatial area of interest is still important (Reynolds, 1997).

In this dissertation, however, the focus is not on the importance of spatial issues of a simulation but rather on the possibility of extending existing learning methods to enabling agents to learn spatial concepts and properties.

The following section is concerned with the development of a framework of adaptive agents that use reinforcement learning while executing a spatial task in a spatial environment. The reinforcement process will enable them to recognise that concepts like distance, location and direction are of major importance in the fulfilment of these tasks and change their behaviour accordingly.

There are characteristics associated with spatial simulations, which are a consequence of the spatial properties of the represented phenomena. For instance, Rasmussen and Barrett (1995), in their theory of simulation refer to the *downward causal effect* of emergent properties in a simulation system (section 4.1.1.1 of this chapter). They give an example of a simulation studying the properties of congestion in a traffic system. In this example the interactions which exist at level L^1 (vehicle-vehicle, vehicle-roadway and vehicle-signal) lead to the generation of a level L^2 phenomenon, which is the congestion of the traffic. Because this phenomenon has a spatial mapping, and because the Level L^1 interactions also involve spatial interactions, the traffic jam effect will constrain the behaviour of the elements at level L^1 , that is, their movement. On the other hand, the phenomena of

the joint distribution of heads and tails generated by two independent coin flips (emergent property of the system) does not have any influence on the dynamics of the coins. This phenomenon is not conditioned by spatial location and therefore, there are no spatial constraints imposed at different levels of the simulation.

The above statement is not asserting that the downward causal effect is directly connected with emergent spatial phenomena. The issue here is that the existence of a spatial mapping of an emergent phenomenon will generally involve a downward causal effect on lower level elements whose interactions are of a spatial nature.

The spatial structure of the environment in which an agent lives is also of major importance to the organisation of the agent society (system), because spatial differences are transformed into organisational structures and a way of socially/economically differentiating agents. According to (Ferber, 1994), spatial relations provide major opportunities and constraints to self-organisation. Propagation of stimuli, as well as reciprocal influences in the environment decrease in strength as a function of the distance between agents and between agents and objects. Because of this, the behaviours of agents are strongly dictated by their relative positions in a topological structure.

Another characteristic typical of a spatial simulation is the spatio-temporal issue. In a system where several agents are moving and acting it is necessary to define a method to make the system evolve. If the computer system is of a parallel nature, this task is facilitated as each agent may be associated with a different processor. For the most part, though, the parallel characteristic must also be simulated. This fact raises the problem of associating the actions and movements of the agent with their temporal consequences. The fact that an agent decides to move towards some object is limited by the fact that, on its way there, the object may be moved. At each step taken, the agent must confirm that the object is still there and that it is still worthwhile to go to it. If the distance between the agent and the object is larger than the one between another agent and the same object, it may decide that it

should not make the effort to reach it. Therefore, the velocities of movement of the different agents may also be relevant.

All these issues have been considered in this work, and a framework for the development of adaptive agents that learn spatial concepts has been developed, and is presented below.

4.5 Framework for developing adaptive agent-based spatial simulations

The methodology here presented is based on the framework used by Russell and Norvig (1995) for the development of rational agents. They begin by defining agents as “anything that can be viewed as perceiving its environment through sensors and acting upon the environment through effectors.”

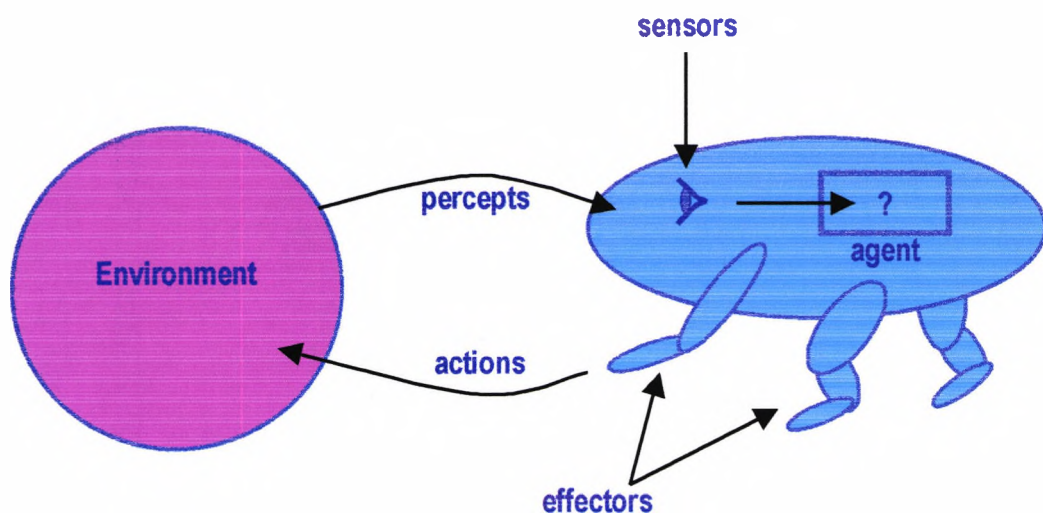


Figure 3 - An agent structure in Russell and Norvig's framework

As can be seen in Figure 3, the agent exists in an *environment* and it gets information from this environment through its *sensors*. It acts on its environment through actuators or *effectors*. The information about the environment is received through structures called *percepts*.

Russell and Norvig (1995) seek the development of a rational agent – one that does the right thing. The right action is the one that will cause the agent to be most

successful. It is then necessary to resolve how and when to evaluate the agent's success.

For the *how*, the authors rely on what they call a *performance measure*, the criteria that determine how successful an agent is. It is obvious that there is not one fixed measure suitable for all agents, and that this measure must be objective and imposed by some authority external to the agent (Russell and Norvig, 1995, page 32). The designer, as an outside observer, can be the one to establish a standard of what it means to be successful in an environment and use it to measure the performance of agents. As to the *when*, it is important that the measure is not momentary but that it represents the agent's performance in the long run.

Russell and Norvig (1995), argue that rational action at a given time depends on the following:

- The performance measure, which defines the degree of success;
- Everything that the agent has perceived until that time (the authors call this the complete perceptual history or *percept sequence*);
- What the agent knows about the environment;
- The actions that the agent can perform.

From these rationality factors, Russell and Norvig define *ideal rational agent* in the following way. "For each possible percept sequence, an ideal rational agent should do whatever action is expected to maximise its performance measure, on the basis of the evidence provided by the percept sequence and whatever built-in knowledge the agent has."

This is the framework that the Russell and Norvig use as a basis for the issues they present and develop in their book and which has been used as a template in this dissertation.

4.5.1 Learning

When an agent is acting in an environment which is not completely known, the only way it can acquire the knowledge it needs is by learning. Learning provides the agent with autonomy. It also provides a good way to build high-performance systems-by giving a learning system experience in the specific application domain (Russell and Norvig, 1995, page 524). This is why it is of major importance to endow agents in a simulation environment with machine learning¹¹ structures.

Russell and Norvig devote several chapters of their book “Artificial Intelligence: A Modern Approach” to learning agents (Russell and Norvig, 1995, chapters 18, 19,20 and 21). The approach they presented into reinforcement learning was used as the basis for the framework developed in this dissertation for adaptive agents in spatial simulations.

Russell and Norvig (1995) divide a learning agent into the following components:

- *The performance element* – responsible for selecting external actions. This element is what is considered to be a complete non-learning agent: it receives information from the environment through sensors and decides on actions to take;
- *The learning element* – responsible for making improvements. It takes some knowledge about the environment and some feedback on how the agent is doing, and determines how the performance element should be modified to do better in the future;
- *The critic* – designed to tell the learning element how well the agent is doing by employing a fixed standard of performance (performance measure as discussed above). This standard should be conceptually outside the agent. It cannot be controlled by it;

¹¹*Machine learning* is, according to Russell and Norvig (1995), page 524, the sub field of Artificial Intelligence concerned with programs that learn from experience.

- *The problem generator* – responsible for suggesting actions that will lead to new and informative experiences. If the performance element had control, the agent would keep carrying out the actions that are the best, given the experiences it has. In this way, the agent would never learn anything. If the agent explores new actions, even if they prove to be sub-optimal in the short run, it might discover much better ones for the long run.

The design of the learning element depends very much on the design of the performance element, and the learning element has the responsibility of improving the efficiency of the performance element every time a new experience has been added to the agent.

4.5.1.1 Reinforcement learning

Reinforcement learning is a learning approach in which agents can learn without being provided any examples, starting with no model of the environment and no concept of the utility of its actions (Russell and Norvig, 1995).

A reinforcement learning agent learns by receiving rewards that serve as feedback on the success of its actions. The agent learns its function from the classification of that success. This classification of success serves as a feedback (positive or negative) to the agent. The difficulty is that, in some cases, the agent only receives feedback at the end of an entire process (e.g. a chess playing agent that only receives the reward at the end of a game, either “you win” or “you lose”). The agent is never told what the right actions are, nor which rewards are due to which actions. It must be capable of determining, at the end of a failed process that has failed, where and what was its mistake.

In other systems, it is possible to give feedback of the success of the agent at each step. Using Russell and Norvig’s framework, as presented above (agents as functions from percepts to actions) a reward can be provided as a percept, but the agent must be “hardwired” to recognise that percept as a reward, rather than just as another sensory input. In very complex systems, reinforcement learning may be

the only feasible way to train a program to perform at high level (Russell and Norvig, 1995, page 599). The authors see reinforcement learning as a restatement of the entire AI problem. An agent in an environment gets percepts, maps some of them to positive or negative utilities, and then has to decide what action to take.

A reinforcement learning agent can vary in the following characteristics (Russell and Norvig, 1995, page 599):

- The environment can be accessible or inaccessible. In an accessible environment, states can be identified with percepts, whereas in an inaccessible environment, the agent must maintain some internal state to try to keep track of the changes;
- The agent may be executing with some knowledge of the environment and the effects of its actions; if not it will have to learn this model as well as utility¹² information;
- Rewards may be received only in terminal states, or in any state;
- Rewards can be components of the actual utility (points for a ping-pong agent or escudos for a betting agent) that the agent is trying to maximise, or they can be hints as to the actual utility ("nice move" or "bad dog");
- The agent can be a passive learner or an active learner. A *passive learner* simply watches the world going by, and tries to learn the utility of being in various states. An *active learner* must also act using the learned information, and can use its problem generator to suggest explorations of unknown portions of the environment.

When it comes to design, a reinforcement learning agent can learn a utility function on states (or state histories) and use it to select actions that maximise the expected

¹²According to Russell and Norvig (1995, page 44), a utility function is a function that maps a state of an agent (or a sequence of states, if we are measuring the utility of an agent over the long run) onto a real number. This describes the associated degree of usefulness and effectiveness of the state (or the sequence of states reached by the agent until that specific moment).

utility of their outcomes. Or, the reinforcement learning agent can learn an action-value function that provides the expected utility of taking a given action in a given state. The latter type of reinforcement learning is called *Q-learning*.

Reinforcement learning is very useful in simulation agents not only because it enables them to learn while executing but also to start with a very simple model of the environment and build from that. There is no training stage before the agent starts acting. It may behave awkwardly in the beginning, but it will soon catch up with the world it is in. These agents are truly adaptive as they may receive rewards on their performance at each step of execution, and use this evaluation to adapt to the conditions and to the needs of the task to perform.

An active learner

An adaptive agent that is executing in a simulation system interacting with other similar agents will be an active learner. It must consider what actions to take, what their outcomes may be and how they will affect the rewards received (Russell and Norvig, 1995, page 607). This agent is embedded in an environment to which it has access, but that is not completely known to it. It will have some knowledge of the environment, but, because there are other agents there, the environment is constantly evolving, and it must be aware of these changes. This means that the agent must be constantly adapting its behaviour to the situation it is in. Therefore, the agent will need feedback (reward) at each step of the simulation. This reward should be very clearly specified, so that it can be used in the overall utility of the activity of the agent.

Using Q-learning

As discussed above, an agent can either learn a utility function or an action-value function (Russell and Norvig, 1995). In the first option this function reflects the utility of reaching a certain state (or the utility of performing a certain state history) for the completion of its task. This value is then used to select actions that maximise the expected utility of their outcomes. This type of agent must have a

model of the environment in order to make decisions, because it must know the states to which actions will lead, and compute the utility of states from that. As the agent takes actions, it will receive rewards in the form of numeric values. These will represent feedback on the agent's action (or, if it is the case, on its performance until the moment). This reward will enable the agent to review the success of its actions so far and change its utility function to reflect this view.

In Q-learning, an agent can be seen as the association of every state that it may reach with every action that it may decide to take on that state (an *action-state* pair). Each action-state pair will have a numeric value associated with it, which will map the utility, for the agent (and to complete its task), of taking that action once it has reached that particular state. This type of agent does not need to have a model of the environment (Russell and Norvig, 1995, page 600). As long as it knows the legal moves available to it, it can compare their values directly without having to consider their outcomes. These learners can be slightly simpler in design than utility ones. However, because they do not know where their actions lead, they cannot look ahead. This may not represent a problem if the result of the actions of the agents is quite simple, and if the complexity of the system is given by the interactions between the different agents.

Q-learning relies on a two dimensional matrix defined as $Q(a,i)$, where a is the action and i is the state of the agent. If all the agents are learning by themselves, they will have their own Q matrix. The values in this matrix represent the utility, for the specific agent, of taking action a at state i . In the beginning of execution, all the values may be the same, probably with a medium number associated. The utilities will then evolve as the agents learn the differences between taking different actions. At each step, each agent will also receive a reward for the state it is in, and this reward will add to the utility and will help update the specific value at the Q matrix.

Updating the Q matrix

The Q matrix is updated by the following formula:

$$Q(a,i) \leftarrow Q(a,i) + \alpha(R(i) + \max_{a'} Q(a',j) - Q(a,i)) \quad (18)$$

The update is realised through the following components:

- Temporal difference learning: the utility of taking action a at state i should consider the expected utilities of the possible actions to take in the following state. If this is not the case, the presented utility should be updated. The α parameter will represent the learning rate of the agent;
- Select the maximum utility for the next action: if this was a passive learner, every possible sequence of actions could be tested by the agent, in order to learn the utilities of each complete sequence. However, because this is an active learner, the agent must act with incomplete information. Therefore, the current utility must be updated to include the maximum expected utility from all the possible actions after the current one.

Exploration function

In order to decide which action to take it is not enough to simply choose the action-state pair with the highest utility value associated. If an agent always did that it would never learn anything. An action has two kinds of outcome (Russell and Norvig, 1995, page 609):

It gains rewards on the current sequence and it affects the percepts received. Hence the ability of the agent to learn – and receive rewards in future sequences.

An agent must make a trade-off between its immediate good – as reflected in its current utility estimates – and its long-term well being. According to Russell and Norvig (1995, page 609), there are two approaches to be considered by an agent in making a decision. In the “wacky” approach, the agent acts randomly, in the hope that it will eventually explore the entire environment. The “greedy” agents acts to maximise its utility using current estimates. According to the tests performed by the authors, a “wacky” agent learns good utility estimates for all the states but it never gets to use them. The “greedy” agent often finds one of the good sequences

but it then sticks to it and never learns the utilities of other states. In the end, the ideal approach is one that is more “wacky” when the agent has little idea of the environment and more “greedy” when its estimates of the utilities are close to being correct. This type of approach can be implemented with the help of the following type of function:

$$f(u, n) = \begin{cases} R^+ & \text{if } n < N_e \\ u & \text{otherwise} \end{cases}$$

This is called the *exploration function*. It determines how greed (preference for high values of u) is traded off against curiosity (preference for low values of n). R^+ is an optimistic estimate of the best possible reward obtainable in any state and N_e is a fixed parameter. The effect will be of making the agent try each action-state pair at least N_e times.

Therefore, the Q-learning agent using all of the above properties can be implemented in the following way:

```

function Q-Learning-Agent( $e$ ) returns an action
static:  $Q$ , a table of action values
 $N$ , a table of state-action frequencies
 $a$ , the last action taken
 $i$ , the previous state visited
 $r$ , the reward received in state  $i$ 
 $j \leftarrow \text{STATE}[e]$ 
if  $i$  is non-null then
     $N[a, i] \leftarrow N[a, i] + 1$ 
     $Q[a, i] \leftarrow Q[a, i] + \alpha (r + \max_{a'} Q[a', j] - Q[a, i])$ 
if  $\text{TERMINAL?}[e]$  then
     $i \leftarrow \text{null}$ 
else
     $i \leftarrow j$ 
     $r \leftarrow \text{REWARD}[e]$ 
     $a \leftarrow \arg \max_{a'} f(Q[a', j], N[a', j])$ 
return  $a$ 

```

Figure 4 – Q-learning algorithm for an active learner agent with exploration function

4.5.2 Analysis of the spatial simulation

For the creation of a spatial simulation, the developer should start by identifying the components of the simulation, as follows:

- Environment – the world where the simulation is going to take place, its shape, its configuration, all information and functions associated with it;
- Agents – the active objects or creatures in the simulation. The agents will be (as noted by Russell and Norvig, 1995) the objects in the simulation which will react to stimuli. These agents will be provided with the capacity of moving (changing location), calculating and changing direction, and changing the other elements in the simulation and environment (through interactions);
- Other elements – any other objects that may exist in the environment but are not represented by agents. Their positions, their possibilities of movement and change must be considered;
- Interactions – between the agents, between the agents and the environment, between the agents and the elements. These interactions should be analysed according to their causes (why the agents decide to interact), the objectives (as to the agent's tasks) and their consequences to the environment.

It is also important, if it is the case, to know (or at least to have studied the possibilities) of what are the emergent phenomena the simulation is seeking to study. These phenomena should have been analysed previously in the real environment. The success of the simulation as an approximation to the real world depends on being able to identify its faults by getting information on its possibilities. The analysis of the spatial consequences of the phenomena is of major importance. Not only its spatial mapping and evolution but also how lower level spatial interactions may affect the spatial properties of higher level phenomena.

4.5.3 Spatial agent design

The design of the agent in this study is based on the Q-learning framework presented above. Several agents are moving in the environment, with the same or similar objectives, taking actions and relying on a reward function and on the Q matrix to give them feedback on those actions.

One of these agents, in a spatial environment, with a task that involves spatial learning (e.g. finding the best position to be according to several parameters) should have spatial components included in their reward function. There are several possibilities for the learning of the agent. The Q matrix is shared by all of them (they are all the same and they all learn the same thing) or each one has its own Q matrix (they are in different positions, they have different objectives, they learn different things). Because these agents will always be different, even if only in their spatial positions, the experience becomes richer if each one has its own matrix.

The agents can also be parameterised according to their own capabilities. They may have different objectives, velocities and even preferences (one agent may have a favourite position or direction, and rewards should be higher if their actions lead to those preferences).

Therefore, in this framework the reward function is not simply the “hardwire” of the value of percepts but the conjunction of spatial parameters that will evaluate the agent’s spatial success. The numeric result of the reward function is an emergent evaluation of the spatial behaviour of the agent.

4.6 Discussion

For the type of agents that concern this methodology, the following definition (taken from chapter 2) could be chosen: “An intelligent agent is generally regarded as an autonomous decision-making system, which senses and acts in some environment.” (Wooldridge, 1997).

First of all, it is necessary to evaluate the models that best reflect the agents that will be built in the context of this methodology. Following Russell and Norvig’s

definition (section 4.5), an ideal rational agent carries out whatever action is expected to maximise its performance measure, on the basis of the evidence provided by percepts and by the knowledge held by the agent (Russell and Norvig, 1995). According to this definition, the agents to be developed in the described methodology are rational agents. However, this is a type of economic rationality. The decisions taken by the agents will be the ones that have a maximised utility value. Also, because this quantitative value changes as agents go through experiences (that is they *learn*) they also qualify to be adaptive agents. Finally, they also have interactive properties, for the decisions taken partly result from the actions of agents.

Once evaluated according to the existing types of agency models, it is relevant to classify the agent architecture used. The first reaction is to qualify these agents as reactive ones. However, it is important to thoroughly evaluate this opinion. Following Ferber's list of properties for reactive agents (Ferber, 1994), it is possible to take the following conclusions:

- Cognitive cost: it is true that these agents do not have an internal representation of the world they are embedded in. However, they can have access to attributes and properties that inform them on their own state and on the global state of the environment. This type of information can be included in the calculus of the utility values;
- Situation: it is clear that Q-learning agents do not plan ahead, as they have not model to follow. However, their past experience is taken into account in the utility function, thus enforcing their adaptive nature. Although they do not explicitly foresee the future, they do become more prepared to it, thanks to their learning facilities;
- Self-sufficiency: this is the most difficult property to analyse. In fact, they are not self-sufficient, as they interact with other agents to evaluate the state of the

environment. However, they do have a higher level of independence than simple reactive agents, because of their adaptive capabilities;

- Robustness and fault tolerance: this is clearly true. Each agent in the simulation is trying to perform a task. The execution of this task may involve the state of other agents but the decision is taken independently. The disappearance of one agent will not affect the action of any other, as these agents are not trying to reach a common goal;
- Flexibility and adaptability: these are also properties of the agents in this methodology, achieved through two characteristics that have already been mentioned: learning and robustness.

The conclusion taken in this dissertation is that these are reactive agents. They may include some complex characteristics, but they do not handle symbolic knowledge. The information they have access to is purely either quantitative or behavioural.

This methodology was developed with the objective of evaluating the potential for the use of intelligent agents in the simulation of evolving spatial environments from the modelling of spatially aware individuals. Furthermore, it aims to explore the use of simple learning techniques to improve the adaptability of these agents, in the context of spatial information.

From the implementation of spatial simulation using this methodology, some possibilities are put forward:

- The extension of learning methods to enable agents to learn spatial concepts and properties;
- The recognition of the importance of spatial properties in the adaptive nature of spatially-aware individuals in simulation environments;
- To create reinforcement structures that take spatial properties of an environment as measures of success;

- To enable agents to evaluate downward causal effects of emergent phenomena as global measures of success;
- To enable agents to use the spatial/temporal consequences of taking an action as a step-by-step evaluation of their current performance;
- To provide agents with learning structures that will enable them to recognise a specific set of attribute values as a geographic (spatial) individual.

In chapter 7, this methodology will be applied to a car park simulation. Cars are reinforcement learning agents with a spatial reward function, which privileges their preferences for parking, the position of other cars going towards the same parking spot and their different speeds. The simulation will show that, with their learning of these spatial properties and parameters, cars can learn to park much more rapidly after a few uses of the car park. Conclusions on the application of the methodology will be drawn on the next chapter.

Ch. 5

First Case-study: An assistant for printing and plotting for Smallworld GIS

The first case study was developed during an internship in Smallworld Ltd., Cambridge, England. This work was carried out in the spring of 1996, when the candidate spent three months working in this company. The aim of this internship was to study the Smallworld GIS and identify opportunities for the implementation of spatial interface assistants as part of the GIS itself.

This internship consisted of three weeks of training in the Smallworld GIS version 2 tools, six weeks of research inside the company and 3 weeks of development. This

chapter describes the work achieved during the internship, which focussed on the opportunities found for developing interface agents to facilitate the use of GIS tools and specifically, the use of the drafting and plotting tool.

5.1 Smallworld GIS

Smallworld GIS follows an object-oriented methodology not only in data management but also in application development. Smallworld architecture is completely integrated in one graphical environment where the development of GIS applications as well as database management can be executed. The key element of this is the GIS programming language: Magik. Magik programming language is an extremely powerful hybrid of the procedural and object-oriented approaches with which it is possible to control and change the whole GIS.

The architecture of the GIS includes an event-handling component that enables any agent system to listen to everything that happens inside it. Also, at any moment, it is possible to know the last event that occurred and thus to know the state in which the system is in. The agent system presented below is developed using the facilities that this architecture makes available.

5.2 The agent architecture

After three weeks of training at Smallworld in Cambridge, and after about a month of research into the way people used the software, an idea of a possible interface assistant architecture for the GIS started to take form from user comments and feedback. The choice of the part of the GIS to facilitate and the analysis of the task to implement was mostly empirical, based on the preferences presented by the Smallworld team and on the possibilities of the GIS package itself. However, the experience of developing this application was invaluable for the design of later projects.

The basic idea behind this architecture was to create a set of agents that would guide new users in the execution of tasks in Smallworld GIS. This would involve the development of one different agent for each specific task. The transitions from one

task to the other and the requests from the user would be handled by one overall entity: the agent controller.

In this way, the agent architecture is composed of the following components:

- The agent controller – the entity that communicates with the several task agents in order to update the possibilities of help, suggestion and execution of tasks given to the user. It also receives requests from the user which are then delivered to the current task agent;
- The task agents – several of these should exist, one for each assisted task provided by the GIS. Each task agent is notified of the events occurring by the agent controller, reviews the state of the environment, re-evaluates the possibilities and finally provides possible help, suggestions and actions to take.

From this basic architecture, one initial task had to be chosen for assistance. After several interviews with Smallworld users, the drafting and plotting tool was chosen. It was selected because some users said that it was quite difficult to use and because it was quite simple to understand in terms of the Smallworld implementation and thus the assistant could be easily implemented. Moreover, the creation of a drawing or plot involved a series of sequential steps, which could easily be mimicked by a simple assistant, based on a state transition diagram.

5.3 The drafting and plotting assistant

To find out how Smallworld users generated drawings and plots using the existing tool, it was necessary to watch the workers at the company use it. However, this phase of work was very quick because drawing and plotting is a sequential (repetitive) task where the customisable part lies mostly in configuring parameters. The need for facilitation comes, not from the possibility of personalisation, but from that fact that, because the task is quite long-winded to perform, it is quite easy for users to get lost in the chain of actions and forget about fundamental parts or do them out of the necessary order (which invalidates successful execution).

To create a drawing or a plot in Smallworld GIS 2 it is necessary to create, at least, one drawing area (a visual geometric entity) and associate it with a drawing function. These two must be compatible (e.g., an area chosen from the GIS main window must be inserted into a rectangle or a square, not a line). The problem is that, as can be seen in Figure 8, if several of these components are to be inserted to the drawing, the task becomes quite repetitive. Moreover, the user must remember to execute four operations, all of them fundamental, in each step of the loop, or the element will not become part of the drawing. This is why an interface assistant can be really useful.

The agent controller user interface for Smallworld GIS 2 is presented in Figure 5. It includes a menu with six functions. These functions are presented to the user through icons that may be enabled or disabled, depending on the last communication with the current task agent. The controller functions are the following:

- Help - The Smallworld GIS manuals have all been converted into HTML form. Therefore, it is possible, at any moment, to access the specific HTML page related to the part of the current tool that is being used. This icon is always enabled during the life of the agent controller;
- Suggest - The enabling of this icon by the controller means that the current task agent knows what the state of the work is and that it can suggest further actions to the user. If the user decides to click on this icon, directions will be provided;
- Perform - The enabling of this icon means, not only that the task agent knows the state of the work but also that it holds all the information to carry the execution of the initiated function through to the end of the process. If the icon is used, the current task function will be executed using the parameters that have already been entered by the user;
- Enable - Enables the controller to listen to events occurring in the GIS;

- Disable - Disables the controller. After the use of this icon the controller is “asleep” and will not communicate either with the GIS or with the current agent. The Enable icon will “awake” it and put it to work again;
- Quit - kills the controller process.

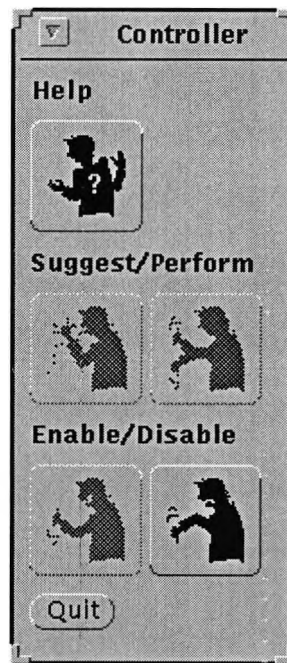


Figure 5 – The agent controller user interface

The task agent receives event information from the controller and uses that information to review its state and re-assess the possibilities ahead. Afterwards, it will communicate to the controller its availability either to suggest or perform further actions.

For each tool, a specific task agent should be developed to enable the use of that tool to be assisted. In this case, the drafting agent was developed, as a class inheriting from the task agent class, which implements assistance for the drafting and plotting tool of the GIS.

The development of this agent depended on the event handling structures that Magik makes available and which are accessed by the agent controller. At each step of a drafting task, the drafting agent evaluates that state of the task, the event that occurred and consults a state-transition array to retrieve the next action to suggest.

Then it communicates with the controller to enable the suggest icon. There is a specific state at which the drafting agent is capable of finishing the task by itself with the attributes that have been inserted by the user. This is when the 'perform' icon will become enabled.

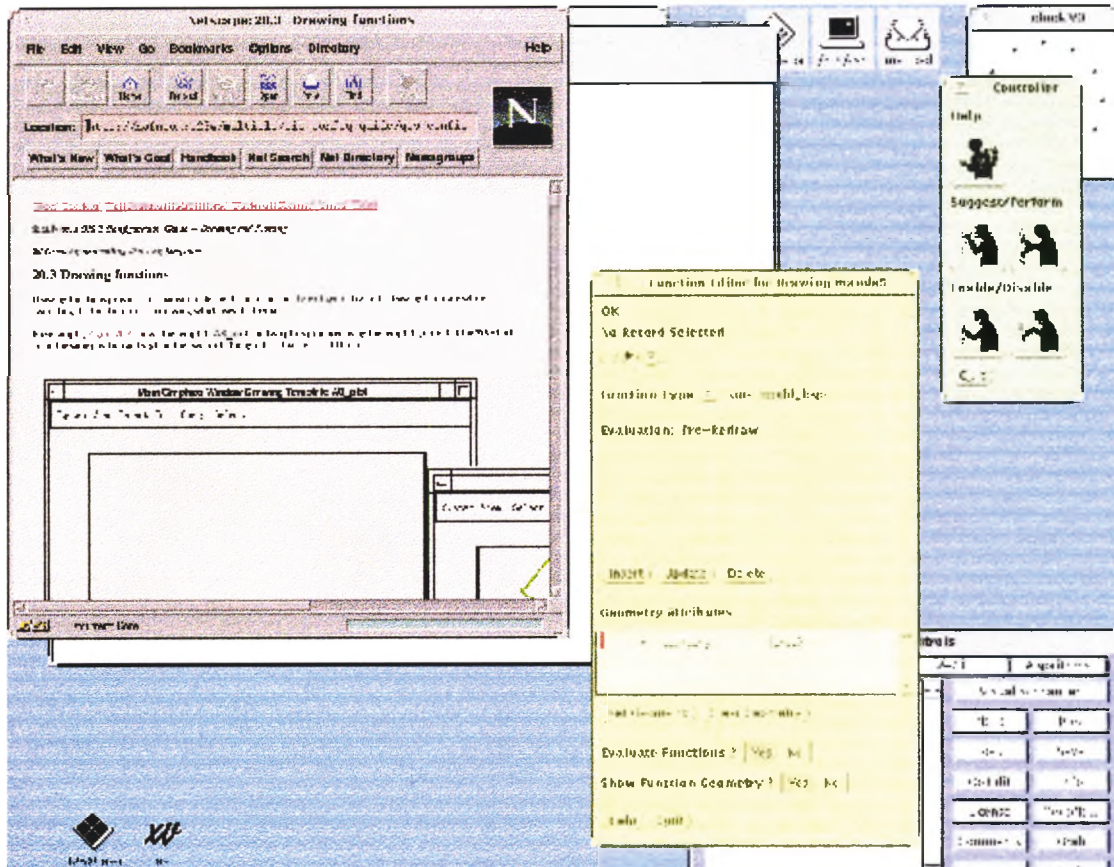


Figure 6 - The drafting agent calls a specific HTML page that provides information on the current state of the drafting task

The same is true for the help icon. At each step of the execution of the drafting task, the clicking of this icon will make the agent evaluate the current state of the task and open the right help page, in order to enable the user to continue working.

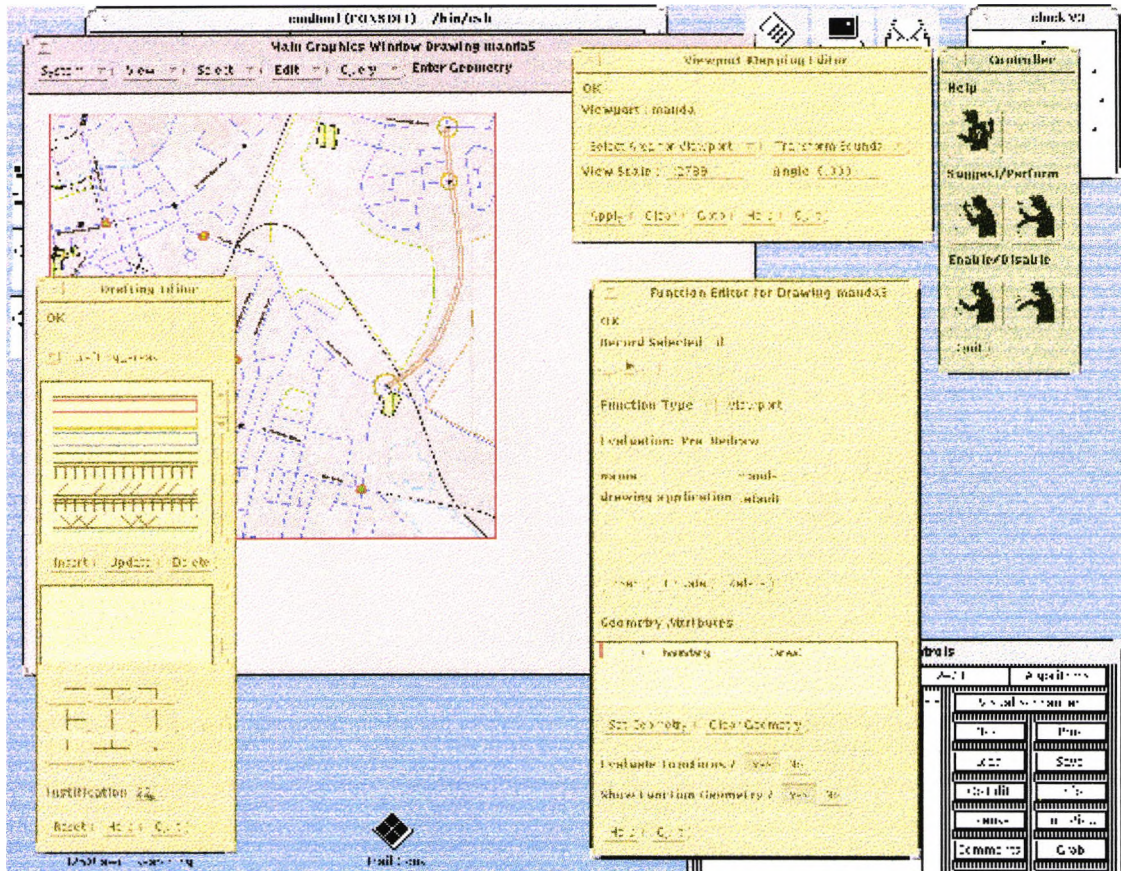


Figure 7 - The agent finishes the execution of the task on behalf of the user (once it has sufficient information to do it)

5.3.1 The drafting and plotting task

To create a drawing or a plot in Smallworld GIS 2 it is necessary to add a drawing area (a visual geometric entity) and associate it with a drawing function. These two must be compatible (e.g., an area chosen from the GIS main window must be inserted into a rectangle or a square, not a line).

The drawing functions considered to be fundamental to create a drawing or a plot were the creation of a viewport (and rectangular area from a GIS window) or the creation of a customised (in terms of proportions) Smallworld logo. If one of these functions have been created since the birth of the agent, then it assumes that the

fundamental operations have been executed and it provides the possibility of generating the drawing as it is, using default values for all other parameters. However, this is not compulsory. The user may add as many drawing functions as he/she sees fit and may also create text or graphics inside the drawing.

If the user decides to add a viewport or a smallworld logo to the drawing, the agent puts the suggest procedure in motion. As said above, to add one of these drawing functions, it is necessary to create a geometric entity and associate it with the function. This can be achieved in several ways:

- The user explicitly creates a geometry by creating a graphic trail and by converting it into a geometry. Then, he/she chooses the drawing functions to be applied to that geometry in the drawing. If the geometry is compatible with the drawing function (e.g. in the case of a viewport the geometry must contain a rectangle), the integration can be executed;
- The user has created a trail but has not inserted it into the system as a geometry. When the user decides to add a specific drawing function to the drawing, the agent realises the geometry is missing and suggests that the user inserts the trail into the system;
- The user decides to create the drawing function but there is no trail or geometry to integrate it with. The agent can then suggest the creation of the trail;
- The user creates the trail and asks the agent what to do next. The agent suggests the insertion of a geometry according to the trail and opens the drawing function editor, so that the user can choose the function to integrate with the created geometry.

The sequence in which the agents suggestions and actions are implemented are based on a state transition diagram, described in the following section.

5.3.2 State Transitions

The conceptual states identified in the drawing sequence are the following:

- State a – initial state;
- State b – the drawing function editor has been activated. From this the user may decide either to create a trail, geometry or drawing function;
- State c – the user has created a trail but no geometry has been inserted and no drawing function has been chosen;
- State d – the trail has been inserted into the system as a geometric entity;
- State e – the drawing function has been chosen, and the geometry too. The integration of the two components can be checked and inserted into the drawing;
- State f – the drawing function has been chosen but now trail or geometry exists to integrate it with;
- State g – the drawing function has been chosen and a trail has been created;
- State h – the drawing function has successfully been added to the drawing structure;
- State i – the drawing or plot has successfully been created;
- State x – error state. Integration of the components has been unsuccessful because they are not compatible.

The state transition diagram is presented in Figure 8. If the user tries to execute a transition out of the order presented in the figure the result will eventually reach the error state x (these wrong transitions are not explicitly represented).

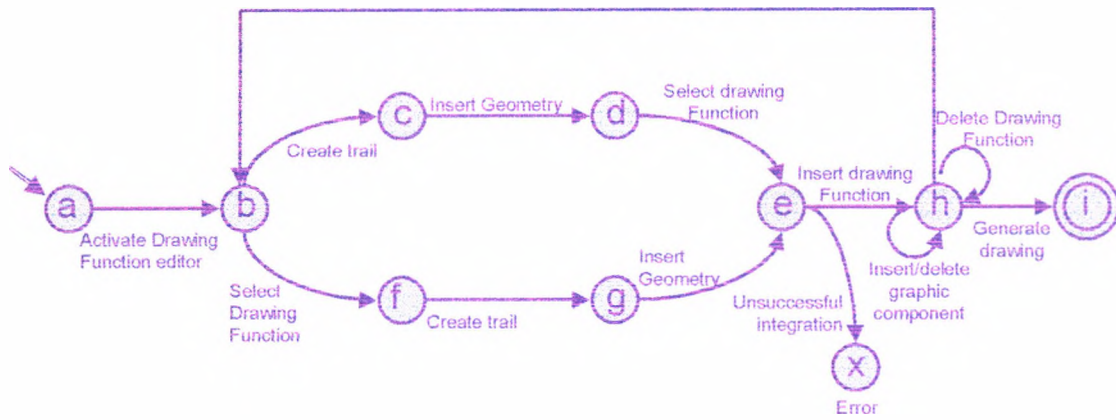


Figure 8 – Conceptual State Transition Diagram

5.3.3 Suggestions and actions

If the user requests a suggestion, the agent responds differently depending on the state:

- State b – the agent suggests the creation of a trail before the selection of a drawing function;
- State c – Once the trail has been created, the agent suggests its insertion as a geometry;
- State d – Once the geometry exists, the agent suggests the selection of the drawing function to be created;
- State e – at this state the agent suggests the creation of the drawing function from the created components. From this moment on, the agent is also capable of creating the drawing or plot using default values as long as the components are compatible;
- State f – the drawing function has been selected, so the agent suggests the creation of a trail;
- State g – at this point the agent suggests the conversion of the trail into a geometric feature;

- State h – the drawing function has been successfully created. The agent suggests the generation of the entire drawing or plot. The agent can perform this at this point.

Depending on the state at which the task is, the agent can either suggest the next step to take, or perform the rest of the task and provide the resulting drawing or plot.

In Figure 7, the Netscape window (in rose) with manual information referring to the drawing function editor (in brown) is called when, after the activation of the editor, the user requests help from the agent controller (in green). The agent displays the information available for that state of the environment.

In Figure 7, the drawing window (in rose) reflects the drawing resulting from the choices made by the user in the viewport, drafting and drawing function editors (in yellow). This screen capture was taken after a request for the generation of a drawing from the introduced information (perform action).

5.4 Discussion

The reaction to the implementation of this assistant at Smallworld was very positive as judged by the reaction to a demonstration in a seminar to users of the system at Smallworld. With very little programming effort it was possible to create a structure that not only provides help at each step of the execution of the task, but also can suggest the next step to take at that state. Once enough information has been inserted, the agent can even offer to finish the task on behalf of the user.

This was a very positive experience. However, there were some immediate limitations. The implementation of the drafting agent was hardwired to the sequential nature of the task itself. It would work only for this tool. The possibility of implementing a more general assistant that could learn the best way to perform a specific task, from the user's actions, was the next required improvement for this implementation.

Also, the drawing and plotting tool was one of the simplest tools available in the GIS. Therefore, the development of assistants for more complex tasks, those where execution would not have such a sequential profile, would also be a more challenging extension. However, the development did determine the kind of 'prototypical work situations' (Rasmussen et al. 1994) which users found themselves in when using Smallworld. These insights would allow re-design of the system interface or the scope for a truly intelligent agent approach.

As presented in the next chapter, the use of simple learning techniques can enable the agent to suggest the next best action for a specific user, depending on the personal way he/she worked with the GIS.

Ch. 6

Case-study: Intelligent assistant for spatial information access

The idea for the creation of an intelligent assistant to aid in the manipulation of a (in this case) geographic information access facilitator emerged from collaboration with the application development team of the Portuguese National Geographic Information Infrastructure (SNIG: <http://snig.cnig.pt>).

The SNIG was one of the first national geographic information infrastructures to be created and implemented within the European Union (Gouveia, 1998). It was created by the Portuguese Government in 1990 at the same time as its coordinating body, The Portuguese Centre for Geographic Information (CNIG:

<<http://www.cnig.pt>>). The Internet version of the system was launched in 1995 with the aim of joining all the Portuguese producers of Geographic information (the SNIG nodes) on the Net. The intention of the SNIG is to be the heart of geographic data distribution and accessibility in Portugal (Gouveia, 1998). The SNIG includes the following services:

- Access to every node's home page;
- Query and access to a CEN/TC 287 geographic metadata¹³ standard compliant metadatabase, which includes information on all of the cartographic and alphanumeric geo-referenced data made available by SNIG's geographic information (GI) producers;
- Query and Access to the GI available through the SNIG's network;
- Online ordering of some commercially available GI (e.g., 1:25 000 maps produced by the Geographic Institute of the Army), the first step for the implementation of a mechanism to perform commercial transactions online using the ATM system.

Presently, more than one hundred institutions (115 in September of 1998) have joined the SNIG, several of them with online databases and/or downloadable cartographic data available from the system (e.g., the patrimony inventory from the General Directorate of Buildings and Monuments – DGEMN and the Environmental Atlas from the General Directorate of the Environment – DGA).

As described above, one of the services of this infrastructure is to direct users to information about the data they are looking for and, if possible, to provide access to that data.

Currently, other projects with the objective of building frameworks to offer a wider and easier access to geographic information users are under way. The following

three examples are the results of some of those efforts. The National Geospatial Data Clearinghouse (NGDC) coordinated by the Federal Geographic Data Committee (USA) allows for the search of digital geographic data, image processing systems, and other modelling software over a collection of more than 100 spatial servers. MEGRIN (<http://www.megrin.org/>) is an organisation representing 19 National Mapping Agencies (NMAs). Its objective is to provide access to NMAs data and, by doing so, to meet the increasing demand for pan-European digital data. The European Spatial Metadata Infrastructure project (ESMI, 1999) aims to create a European spatial metadata framework by providing mechanisms to link GI users with (meta)data services and providers using the Internet.

Each of these structures implement search processes to allow the users to find the data they are looking for. The NGDC uses a query applet that enables, among other possibilities, the graphical definition of spatial and temporal coverage of the query. ESMI searching facility is currently under construction and will provide a user interface with different levels of complexity. MEGRIN's search facility is currently being reviewed with the objective of improving its CEN/TC 287 compliant metadatabase.

By analysing several National Geographic Information Infrastructures available on the World Wide Web (WWW), it is possible to conclude that these have a very real need to create metadata structures that will enable the comprehensive classification of their data. The existence of such metadata will not only provide information on the data produced but also enable the maintenance of the data structure (Rodrigues, 1998). Moreover, if compliant with existing metadata standards, they will ease sharing and integration with other information infrastructures (Rodrigues, 1998).

¹³Metadata consist of information that characterises data. It documents the production of information by answering the following questions (FGDC, 1999; Metadata FAQ): What is the data produced ? Who produced it ? When was it produced ? Where, Why and How ?

The Distributed Geolibraries report (MSC, 1999), which presents the findings of the Workshop on Distributed Geolibraries: Spatial Information Resources, convened by the Mapping Science Committee (MSC) of the USA National Research Council in June of 1998, aims to follow up some of the ideas described above, with the objective of creating *Distributed Geolibraries*. As described in the report, “A geolibrary is a digital library filled with geoinformation—information associated with a distinct area or footprint on the Earth’s surface—and for which the primary search mechanism is *place*¹⁴. A geolibrary is distributed if its users, services, metadata, and information assets can be integrated among many distinct locations.”

The MSC’s vision for digital libraries includes:

- Distributed search and access to information. Retrieval and integration with other information. Manipulation and analysis of the results;
- Integration of WWW browsing functions with those of Geographic Information Systems (GIS);
- Enabling of collaborative work and capturing of results;
- Enabling work located in the field.

The report also considers that metadata will be of extreme importance to distributed geolibraries, as their services will certainly be refined with more sophisticated tools (cataloguing, indexing and abstracting tools) designed to assist in search, evaluation and use. Moreover, a distributed geolibrary will offer something that is not possible in the traditional library, the ability to search based on geographic location. The use of geo-spatial metadata standards allows catalogues to be constructed using well-defined content.

¹⁴The term *place* is used throughout the report to refer to a location of interest on or near the Earth’s surface.

As described above, several interfaces have been or are being built to search data in Geographic Information (GI) infrastructures. Those interfaces which base their search on metadata structures, by enabling the user to select the specific characteristics of the required data, are quite relevant to the context of this dissertation. To provide the user with a form that will enable a search on all the specific characteristics of geographic data involves the building and manipulation of a very complex interface, especially when the metadata structure complies with some kind of standard. The need for a simplification in the use of this type of interface is clear and the idea of using simple learning mechanisms to do this has a very strong appeal.

The prototype presented in this chapter is not based on the operational version of SNIG's metadata access interface. A simple interface was developed specifically for this purpose. The fundamental operations are available and the relevant functionality can be demonstrated through this prototype. Its simplicity facilitated the development of the intelligent assistant without a specific concern with the actual interface's specificity. However, the reasoning implemented in this context can be easily transferred to the operational SNIG metadata interface.

6.1 A Spatial Information Facilitator on the World Wide Web

This prototype involved the development of several applications, using different programming languages and connected in various ways. The components presented in Figure 2 of chapter 3 had to be adapted to the possibilities of the existing technology, concerning the specificity of the handled information.

It was first necessary to create an online mapping application that would serve as a simple spatial interface to the area referenced by the metadata. This was accomplished by taking an example application provided in ESRI MapObjects 1.2 and extending it to fulfil the needs of the prototype. This application included a Java Applet to be used as the client inserted in a WWW Browser, and a Microsoft Visual Basic application as the system's metadata map server component. Both of

the programming languages used, although not the best choice for the development of agent and learning procedures, were required because of the facilities included with the ESRI packages to manipulate geographic information on the Internet.

6.1.1 Online Mapping with ESRI's Mapobjects and MapObjects Internet Map server

ESRI's MapObjects and MapObjects Internet Map Server¹⁵ were used to develop the Spatial Information Facilitator. An evaluation of the major existing commercial tools used for this purpose was not carried out for this dissertation as the final application was only to be used as a basis for the learning of the assistant. Of major importance was the possibility of recording every action the user took when in contact with the facilitator. For a better understanding of the resulting agent system, a succinct description of how these tools work is given.

The MapObjects product includes an ActiveX control called the Map control and a set of over thirty OLE automation objects which enable the integration of mapping functions in industry standard programming environments such as Microsoft Visual Basic, Delphi, Power Builder, Microsoft Access and others (ESRI, 1996b).

The MapObjects Internet Map Server (IMS) is a product for web authors who wish to make dynamic maps available from their web sites (ESRI, 1996a). This product consists of a web server extension and programmable objects that extend MapObjects. Also included is source code that can be compiled to create an ActiveX control or a Java Applet which can be customised to create specific active content for client web pages.

Client code can either be created in HTML, Java or ActiveX. In any case, the client application communicates with a possibly remote HTTP server to request a MapObjects service. This request is passed to an extension that then communicates with the server-side mapping application that has been built in one of the ActiveX

¹⁵ESRI's MapObjects and MapObjects Internet Map Server are based on Microsoft's OLE/COM technology.

enabled development environments. This application will include one Map Control to provide the mapping capabilities and one Web Link control to provide the communication facilities for the mapping functionality. The architecture of this type of system is portrayed in Figure 9. More information about these tools and others associated with them can be found in ESRI's guides (ESRI, 1996; ESRI, 1996a; ESRI, 1996b).

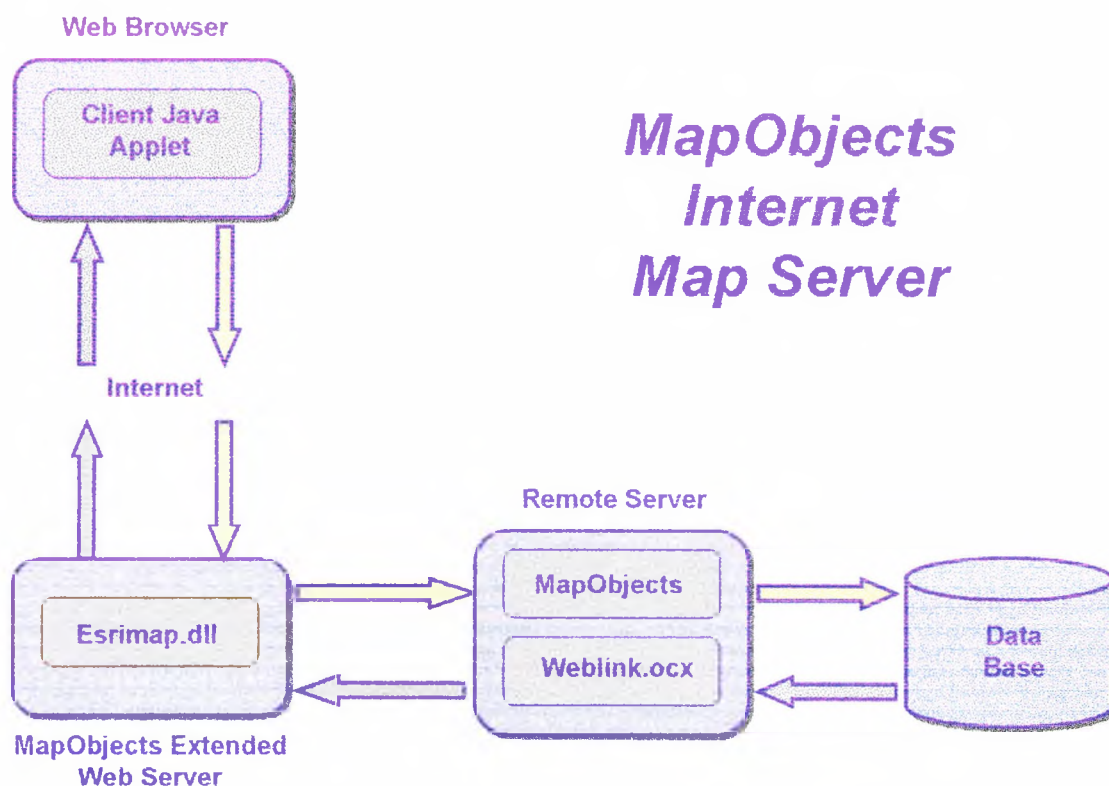


Figure 9–MapObjects IMS Architecture

This architecture has been extended to include the components depicted in Figure 2 of Chapter 3. However, the integration of these two architectures proved to be difficult and some changes had to be included in order to build a successful system. The result is presented later in Figure 19.

6.1.2 Using metadata to search for fitting information

Consistent, reliable means to share geographic data among all users could result in significant savings for data collection, enhanced use of data, and better decision

making (NSDI, 1998). GI Infrastructures can be viewed as consisting of nodes that enable that sharing of data and become, at the same time, a centre of communication between organisations that would otherwise be isolated. To minimise the time spent in transferring, manipulating and updating the information, available databases should have their structures published in a clear and well-defined way. This is why metadata is becoming not only popular but a necessity. In this context, the metadata structure can become a means for searching for information, if it is publicly available as the basis for a searching tool. This prototype uses simple non-standard metadata to search for geographic data. Standard geographic metadata was not available for integration in this work at the time of development, as the SNIG's CEN/TC 287 metadatabase was still under development. Thus, the metadata information used includes geographic region and type of spatial data and was created for the implementation of this prototype. However, this work could be extended to the different attributes of spatial data considered in a metadatabase as it uses the generic structures now present in the standard.

6.1.3 Metadata map server

The metadata map server, developed in Visual Basic 5.0, facilitates the identification of available data according to location and type of information. The server executes the following operations, according to client requests:

- Change the geographic region visualised by the client user according to request;
- Select a geographic region for later retrieval of information;
- Retrieve information on the data identified inside the selected region and (optionally) which belongs to a specific theme.

These facilities were implemented through the MapObjects tools.

6.1.4 Client requests

MapObjects IMS includes Java libraries that allow web requests to be handed to the Visual Basic Map server. Objects like Map and Extent instances (geographic region) can be created in the applet context and sent to the map server as part of client requests.

The client Java applet prototype includes a map window and a set of tools that enable the manipulation of that window (pan, zoom in, zoom out and full extent). Besides these manipulation tools (which were part of ESRI's java applet example that was extended to create this prototype), there are the tools specific to the application, which are the following (These tools are part of the Java applet presented in Figure 10):

- *Select relevant region* – this request will need a rectangle defined by the user which will determine the region of the map he/she is interested in accessing. Once this rectangle has been defined, the server can, later on, search for relevant information related to that region;
- *Enlarge relevant region* – The selected region can be enlarged using this operation;
- *Constrain relevant region* – The selected region can be constrained using this operation;
- *Unselect relevant region* – If the selection was a mistake, it is possible to remove the selection made;
- *Retrieval of data about the selected region* – This operation uses the selected region to search for metadata on relevant information inside the selected area.

- Access to the assistant window – This operation calls the assistant window, enabling the user to ask for suggestion on next requests to the map server, or to ask for the assistant to take action on his/her behalf.

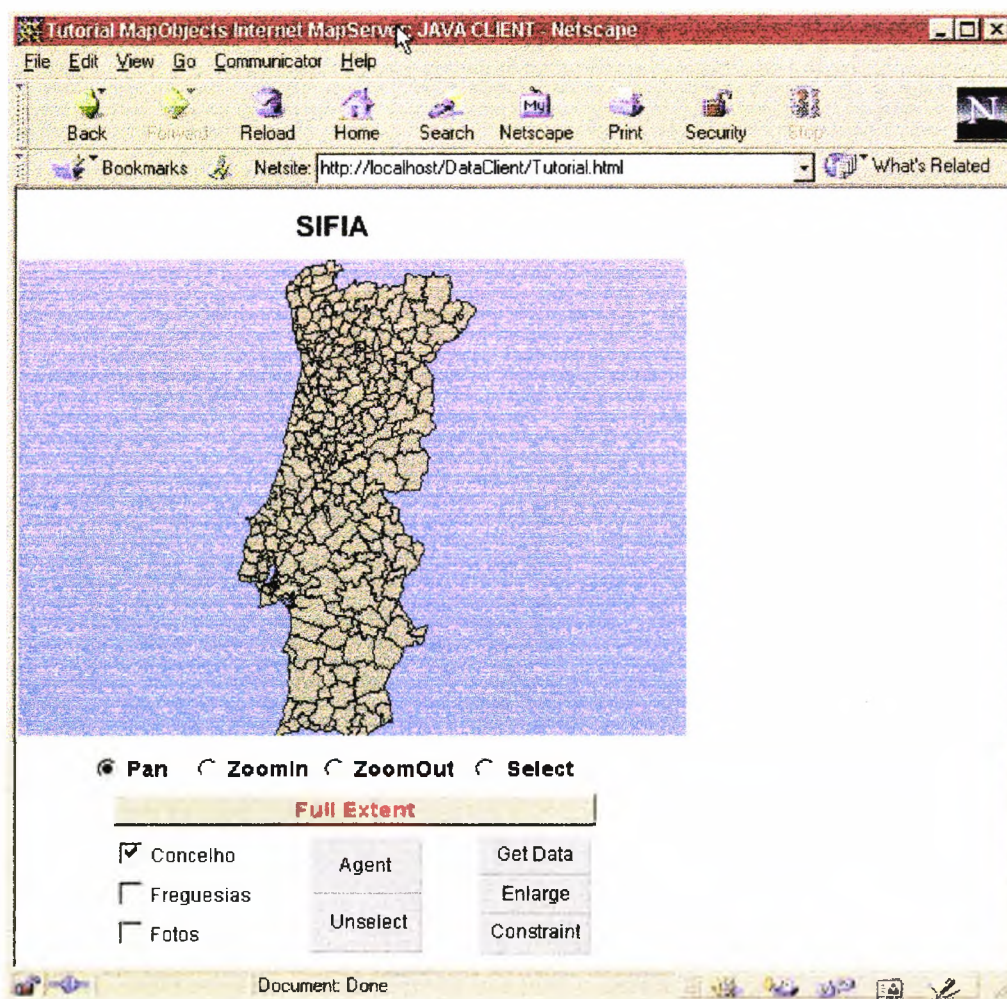


Figure 10 – The client interface-a java applet which communicates with the map server

When a retrieval request is sent to the map server it is possible to configure the request on the specific characteristics of the information that may be relevant to the user. The prototype enables the user to choose the type of information required by theme.

This means that each request sent to the server is based on the following values:

- The command the server should execute - *cmd*

- a region that the user is currently visualising – *cr*. This region is a set of four coordinates (*left, right, top and bottom*);
- the indication of whether or not a selection for retrieval has been defined – *sel*. This is a boolean value which, if true, should be followed by the description of the selected region;
- a region that the user has selected – *scr*. This region is also a set of coordinates (*sleft, sright, stop and sbottom*);
- The indication of the themes in which the user is interested – *themes*;
- The indication of the level at which the user is working – *levels*. This means that at each request, the system should be able to, from the base data available, determine if the request was concerned with the national, regional or local level.

The communication between the client applet and the map server is *stateless*. This means that the state at which the server is currently (e.g. the region that is currently selected) is not considered as current if it is not mentioned in the next request. The system was implemented in this way to account for the possibility of the application being used by several users at the same time.

Before moving on to the implementation of Memory-Based Reasoning, it is important to note that this facilitator was fully implemented in the context of this work and, although the code developed is not an issue of the dissertation, it is made available for inspection in Appendix B.

6.2 Using Memory-based reasoning to personalise the use of the Spatial Information Facilitator

This prototype is based on the methodology described in chapter 3 which is in itself based on work presented in Kozierok and Maes (1993) and Stanfill and Waltz (1986). Memory-based reasoning (Stanfill and Waltz, 1986), which is the basis for most of the work done on intelligent assistants by the Autonomous Agents Group of the Massachusetts Institute of Technology (MIT), is also behind the reasoning

methodology used in this prototype. The aim of the agent design here is to create profiles of query use of this facilitator in order to help users rapidly jump to situations that they aim to reach during the execution of the application.

To implement this it was necessary to create two new components in the system described in 6.2.2 and extend the ones that already existed. Because of the limitations of developing mapping applications for the WWW it was not possible to create the assistant as a modular entity, but its functionality had to be distributed between the client Java Applet and the Visual Basic Metadata Map Server. This happened for the following reasons:

- Because the client part was implemented to be used through a “standard” WWW browser, it was not possible to create, at the time and as part of the client, persistent structures that would handle the memory information. This led to the next best option, to associate the memory with the server. This created an additional difficulty: The personal information of use to the client application had to be kept in a common server database, and not where it would be natural and easier, associated with the java applet;
- The need for the availability of mapping operations led to the use of an industry “standard”, in this case, Visual Basic. This choice presented several difficulties in the implementation of the reasoning facility and the manipulation of the memory. The lack of flexibility of Visual Basic in matters of polymorphism and integration with tools outside of the Microsoft universe resulted in repetitive code for the manipulation of similar but different entities. The memory had to be implemented in a relational database which would be accessible and easy to handle by Visual Basic and the consequence of this was a large number of similar tables storing the same information for different entities of the systems.

The problems described above did not represent an obstacle for the implementation of the system. However, in this description of the prototype, the concern will be to

describe the modular set of entities and operations that would make sense in an ideal implementation. The actual code is listed in Appendix C.

6.2.1 Memory-based information structure

For Stanfill and Waltz (1986), the memory-based reasoning hypothesis is that reasoning may be accomplished by searching a database (memory) of worked problems for the “best match” to the problem at hand. To measure how closely two situations match, it is necessary to develop a metric (Chapter 3). The metric developed for use in this work, also described in chapter 3, extends the Value Differences metric to compare spatial regions. This metric was implemented using two simple algorithms, presented in Figure 11. It was not the subject of a thorough study, as it is not the aim of this work to define it. The aim was to show that the use of a well-built metric can enable the assistant to learn “spatial” experience from the system memory.

The memory-based reasoning methodology is based on the definition of predictor and goal fields. The aim is to provide the user with the possible next request from the one he/she has just formulated, using the experience kept in the memory. In this way, the values sent in the previous request are considered predictor fields, whereas the possible values for the next action are the goal fields. Thus, the memory is composed of pairs of requests: the first one and the one that followed it in execution.

6.2.1.1 Mapping between the methodology definitions and the Memory database structure

The algorithm described by Stanfill and Waltz (1986) describes a set of formulae that, from the existing experience stored by the agent, can calculate the application’s closest action to take in the present situation (see the description of the algorithm in chapter 3):

a) The measure of dissimilarity between two records specific to one goal field:

$$\Delta^g(M, \gamma, \rho) = \sum_{f \in P\rho} \delta_f^g(M, \gamma, f, \rho, f)$$

b) To calculate the measure of dissimilarity it is necessary to calculate the penalty to each predictor field which is the result of the product of the weight given to the specific predictor field in predicting the goal field value and the difference between the two records being compared in terms of the possible values of the goal field:

$$\delta_f^g(M, \gamma.f, \rho.f) = d_f^g(M, \gamma.f, \rho.f) \cdot w_f^g(M, \gamma.f)$$

$$d_f^g(M, \gamma.f, \rho.f) = \sum_{v \in V_g} \left(\frac{|M[f = \gamma.f][g = v]|}{|M[f = \gamma.f]|} - \frac{|M[f = \rho.f][g = v]|}{|M[f = \rho.f]|} \right)^2$$

$$w_f^g(M, \gamma.f) = \sqrt{\sum_{v \in V_g} \left(\frac{|M[f = \gamma.f][g = v]|}{|M[f = \gamma.f]|} \right)^2}$$

The above formulas represent values that had to be calculated and to exist in the relational database tables that form the memory. To create these tables the following values had to be calculated:

- the frequency of every value of every predictor field in the memory, needed for the calculus of w_f^g and d_f^g above (e.g. *frequency_cmd*);
- the frequency of the association of every value for the goal fields and predictor fields, in pairs, also needed for the calculus of w_f^g and d_f^g above (e.g. *frequency_cmd_gcr*);
- The weight of every value of a predictor field when considered in the context of one goal field, that is w_f^g (e.g. *w_cr_gcmd*);

- The distance between two different values of every predictor field, considered in the context of one goal field, that is d_f^g (e.g. *d_gcmd_cmd*);
- The distance between a target record¹⁶ and every record in the memory, in the context of one goal field, that is the final measure of dissimilarity between two records mentioned in a) above (e.g. *delta_gcmd*).

The implementation of the Stanfill and Waltz (1986) methodology included two alterations:

- The spatial extension of the metric through the development of the functions described below (Figure 11), which enable the comparison of two geographic areas. The development of metric was not extensively studied and the only concern was to be able to evaluate if two regions were the same or not. Because the regions were defined with the help of a mouse, it was almost impossible to generate the same region twice. Thus, the function returns true if the area of the intersection of the two initial regions is larger than fifty percent of any of the initial regions' areas.
- From the sets presented above, only the last one is calculated at run-time. For performance reasons, the others are calculated every night, from the experience contained in the memory. In Stanfill and Waltz (1986) these sets were calculated every time a new experience was added to the memory. As the volume of the memory grew the time spent on this process became extremely long. The decision taken to re-calculate these values only once a day (during the evening) followed a recommendation given Kozierok and Maes (1993) when developing a learning interface agent for scheduling meetings.

```
Private Function equal_extents(ext1 As MapObjects.Rectangle, ext2 As
MapObjects.Rectangle) As Boolean
    Dim area1 As Double
    Dim area2 As Double
    Dim area3 As Double
```

¹⁶ A request which needs the agent to predict the next action and, therefore, has no goal fields values.

```

Dim intersect As New MapObjects.Rectangle
area1 = area_extent(ext1)
area2 = area_extent(ext2)
If ((area1 < area2 * 0.5) Or (area2 < area1 * 0.5)) Then
    equal_extents = False
Else

    intersect.Left = ext1.Left
    intersect.Right = ext1.Right
    intersect.Top = ext1.Top
    intersect.Bottom = ext1.Bottom
    intersect.intersect ext2
    area3 = area_extent(intersect)
    If (area3 < (area1 * 0.5)) And (area3 < (area2 * 0.5)) Then
        equal_extents = False
    Else
        equal_extents = True
    End If
End If
End Function

Private Function area_extent(ext As MapObjects.Rectangle) As Double

    area_extent = Abs(ext.Right - ext.Left) * Abs(ext.Top -
ext.Bottom)

End Function

```

Figure 11 – The Visual Basic functions that implement the metric for the spatial region. The region in ext1 is placed in the intersect structure before executing the intersection with ext2. The result of the intersection is placed in the same intersect structure

6.2.1.2 Memory implementation

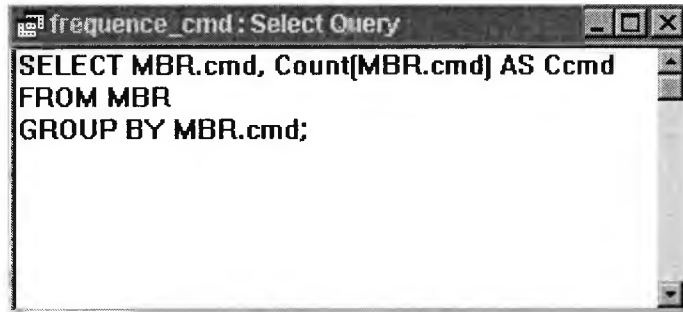
The Memory database includes the following types of tables and views:

- *MBR* – the basic table that includes all of the requests of one user described in terms of the request values presented in 6.1.4 (Figure 12);
- *frequency_XXX* – defined for each predictor field and containing the frequencies of the values of each predictor field (e.g. Figure 13);
- *frequency_XXX_gXXX* – where XXX is every predictor field and gXXX is every goal field. This table contains the frequency of every combination of values of one predictor field with one goal field (e.g. Figure 14);
- *w_XXX_gXXX* – for each association of predictor field and goal field. Each of these tables will include a set of values, calculated from the frequencies in the tables above, that describe the weight of every value of a predictor field when considered in the context of one goal field (e.g., Figure 16);

- *d_gXXX_XXX* – Each of these tables holds the differences between the two records being compared in terms of the possible values of the goal field (e.g., Figure 17);
- *delta_gXXX* – The values in these tables are calculated at runtime and compare the last request sent by the user with every relevant previous request in the database. The MBR table is restricted using the predictor restriction method described in chapter 3. The result is the delta value which represents the quantitative distance between each record in MBR and the current request, in terms of the specific goal field which the table refers to (e.g.: Figure 18).

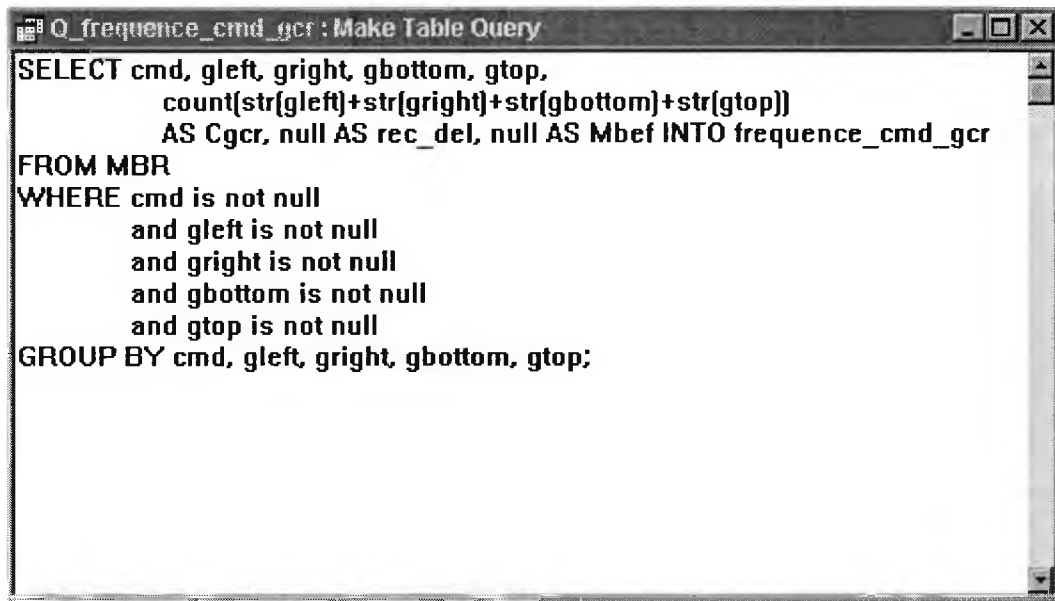
	Field Name	Data Type
?	user	Text
?	date	Date/Time
?	time	Date/Time
?	number	Number
	cmd	Text
	left	Number
	right	Number
	bottom	Number
	top	Number
	sel	Yes/No
	sleft	Number
	sright	Number
	sbottom	Number
	stop	Number
	levels	Text
	scales	Text
	n_records ret	Number
	n_request	Number
	gcmd	Text
	gleft	Number
	gright	Number
	gbottom	Number
	gtop	Number
	gsel	Yes/No
	gsleft	Number
	gsright	Number
	gsbottom	Number
	gstop	Number
	glevels	Text
	gscales	Text
	gn_records ret	Number

Figure 12 – Design of memory table MBR, which contains all of requests ever sent by a user to the metadata map server



```
frequence_cmd : Select Query
SELECT MBR.cmd, Count(MBR.cmd) AS Ccmd
FROM MBR
GROUP BY MBR.cmd;
```

Figure 13 – SQL query that generates the values in view `frequence_cmd`



```
Q_frequence_cmd_gcr : Make Table Query
SELECT cmd, gleft, gright, gbottom, gtop,
       count(str(gleft)+str(gright)+str(gbottom)+str(gtop))
       AS Cgcr, null AS rec_del, null AS Mbef INTO frequence_cmd_gcr
FROM MBR
WHERE cmd is not null
      and gleft is not null
      and gright is not null
      and gbottom is not null
      and gtop is not null
GROUP BY cmd, gleft, gright, gbottom, gtop;
```

Figure 14 – SQL query that generates the first version of the table `frequence_cmd_gcr`. This table has to go through further processes so that the existence of the frequencies of similar regions not identified at runtime may be evaluated

frequence_cmd_gcr : Table						
	cmd	gleft	gright	gbottom	gtop	Cgcr
▶	Gif	-148400,8	416331,27	78304,636	480543,05	12
	Gif	-49782,531	277868,62	120046,36	353420,53	9
	Gif	41981,404	417581,51	235784,77	503311,26	3
	Retr	-185738	618738	10000	583000	2
	Retr	-148400,8	416331,27	78304,636	480543,05	5
	Retr	-49782,531	277868,62	120046,36	353420,53	2
	Retr	93172,185	442284,77	273678,09	522338,47	2
	Sel	-185738,41	618738,41	10000	583000	4
	Sel	-148400,8	416331,27	78304,636	480543,05	9
	Sel	-49782,531	277868,62	120046,36	353420,53	8
	Sel	41981,404	417581,51	235784,77	503311,26	1
*						

Record: 14 | 1 | of 11

Figure 15 - Resulting frequence_cmd_gcr after process of identification of similar regions

w_cr_gcmd : Table					
	left	right	top	bottom	weight
▶	-185738,41	618738,41	583000	10000	0,7777777778
	-148400,8	416331,27	480543,05	78304,636	0,6412487817
	-49782,531	277868,62	353420,53	120046,36	0,614135677
	41981,404	417581,51	503311,26	235784,77	0,7071067812
*	0	0	0	0	0

Record: 14 | 1 | of 4

Figure 16 – Example of one of the weights table. The weight of each predictor region is measured according to the goal field cmd

d_gcmd_cmd : Table			
	cmd1	cmd2	d
▶	Gif	Retr	1,0559859963
	Gif	Sel	0,4874598255
	Retr	Gif	1,0559859963
	Retr	Sel	0,6239669421
	Sel	Gif	0,4874598255
	Sel	Retr	0,6239669421
*			0

Record: 1 of 6

Figure 17 – Structure and values in d_gcmd_cmd table. The d value represents the comparison of two records in terms of the values of the predictor field gcmd and of the records' values for the predictor field cmd

delta_gcmd : Table						
	user	date	time	number	gcmd	delta
	user1	04-02-1996	15:11:26	4	Retr	0
	user1	04-02-1996	15:13:50	2	Sel	0
	user1	04-02-1996	15:10:36	5	Retr	0
	user1	04-02-1996	16:00:36	4	Retr	0,016678681
	user1	04-02-1996	16:01:00	4	Retr	0,016678681
	user1	04-02-1996	16:01:26	4	Retr	0,016678681
	user1	04-02-1996	16:02:01	1	Gif	0,087985546
	user1	04-02-1996	14:19:23	4	Gif	0,087985546
	user1	04-02-1996	14:20:21	7	Retr	0,087985546
	user1	04-02-1996	14:21:18	13	Sel	0,104664227
	user1	04-02-1996	14:21:23	15	Sel	0,121342908
	user1	04-02-1996	16:03:39	6	Gif	0,138021589
	user1	04-02-1996	14:21:32	17	Retr	0,138021589
	user1	04-02-1996	16:02:23	1	Gif	0,138021589
	user1	04-02-1996	16:02:31	4	Retr	0,138021589
	user1	04-02-1996	16:03:22	4	Sel	0,138021589
	user1	04-02-1996	16:04:25	10	Sel	0,154700270
	user1	04-02-1996	16:04:31	12	Retr	0,171378951
	user1	04-02-1996	16:05:28	4	Retr	0,171378951
	user1	04-02-1996	16:06:07	4	Gif	0,171378951
	user1	04-02-1996	16:07:11	4	Gif	0,242685816
▶	user1	04-02-1996	16:07:40	9	Retr	0,242685816

Record: 22 of 31

Figure 18 – Measure of distance between each record in the memory (after predictor restriction) and the last request made by the user, in terms of the predictor field gcmd. As can be seen, the assistant recommends the retrieval of information.

In each of the tables presented above the comparison of regions had to be re-evaluated using the functions presented in Figure 11. This means that the frequency of requests on one specific region had to be calculated using these functions. This is where the methodology presented by Stanfill and Waltz (1986) had to be extended, as there was no metric for comparing regions.

The code created for this matter is included in Appendix A in every procedure with a name beginning with `rec_frequence`. There were other procedures where the same function had to be used, in fact, every time it was necessary to compare two regions.

6.2.1.3 Memory Manager's overnight update

As stated above, only the last set in the above list is calculated at run-time. This means that if substantial changes are inserted into one user's behaviour over the period of one day, they will only be noticed on the following day once these are included in the sets of general weights and differences.

6.2.1.4 Action Prediction

Another two requests have been added to the system in the context of the interface assistant: suggestion and action. The former asks the agent to gather information to advise the user on the next action to take. The latter enables the user to give the agent permission to execute this next action. For each of these requests, the agent selects the predictor field value with the largest weight and restricts the memory to the requests where that predictor field equals that value. This restricted memory includes all the requests that will be compared with the current predictor values. The agent will then generate the *delta_gcmd*, *delta_gcr*, *delta_gsel* and *delta_gscr*, which will contain the distances between the last request sent by the user and all the requests contained in this restricted memory, in the contexts of the different goal fields.

6.2.2 SIFIA - The Spatial Information Facilitator Interface Assistant

As stated in section 6.2 the architecture for the system presented in the methodology chapter (Chapter 3) had to be adapted to the technological limitations of WWW mapping. The result was the architecture described below, where the assistant component had to be divided between the client applet and the map server.

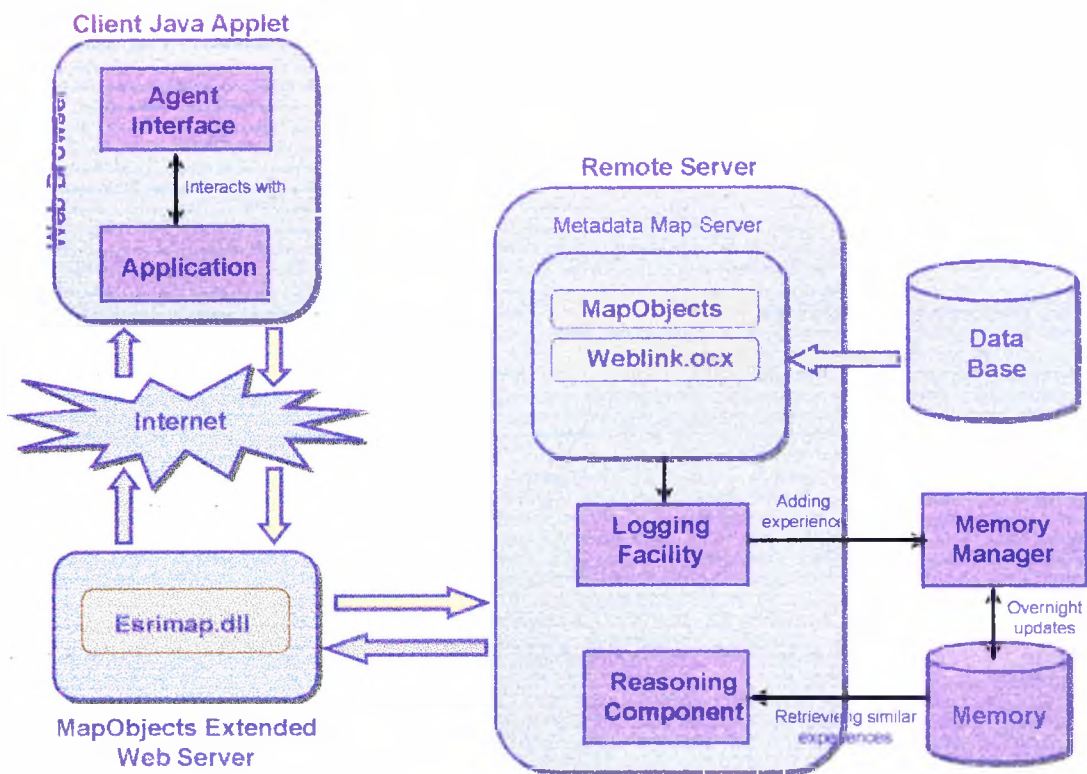


Figure 19 - The Implemented architecture: basis for SIFIA (The Spatial Information Facilitator Interface Assistant)

The Spatial Information Facilitator Interface Assistant (SIFIA) was implemented using the architecture presented above (Figure 19). This architecture contains the following components (already described in chapter 3):

- *Agent Interface* - This component is part of the Java applet and receives indications from the user on whether suggestion or action is required on his/her behalf. It then communicates with the server in order to determine the next action to suggest or take;

- *Memory* – The memory contains the acquired experience information the agent has gathered. It also contains daily updated information on the weights and distances of the features in the memory;
- *Memory Manager* – The process that daily updates the weights and distances of the features which are part of the memory as well as select the experience that has become too old to be considered;
- *Logging Component* – This component receives information about every request sent by the client and stores it in the memory. It is responsible for adding experience to the memory;
- *Reasoning Component* – The reasoning component takes the reasoning request from the client (suggest or take action) and the experience in the memory and orders all the relevant requests in the memory by its distance to the current request. The best match is chosen and sent back to the user. If the user has asked for a suggestion, he/she then decides whether to take it and ask for the execution of the selected operation. If the request was already to take the action, this is done without delay.

The visual interface of the application is presented in Figure 10 (the Client Java Applet interface, which reflects the operations the client can perform), and in Figure 20, the assistant's interface, which handles the requests for suggestion or action. The text window presents the possible action to take, once its selection has been made.

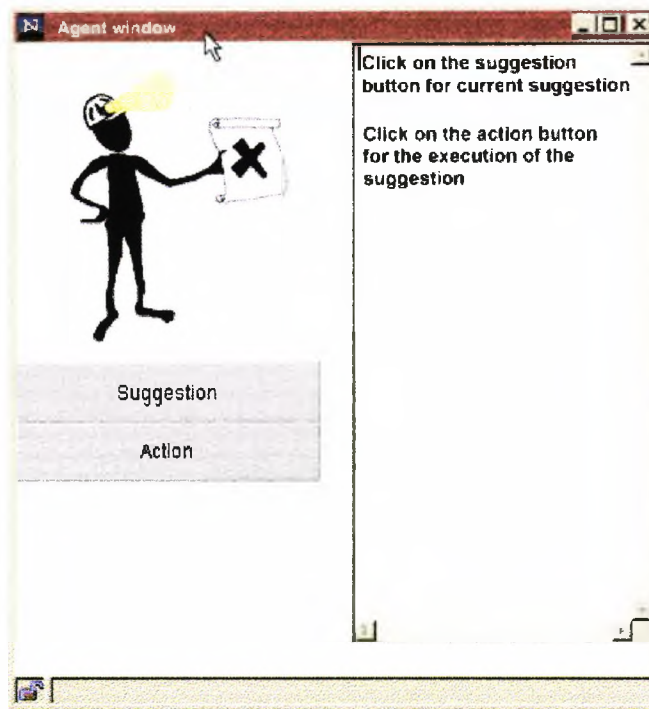


Figure 20 – The Interface Assistant Window

6.3 Recommendation example

One example of the recommendation given by the assistant is illustrated in Figure 18, Figure 20, Figure 21, Figure 22 and Figure 23. One user was instructed to make several requests to the server and to systematically insert selections of the area around Lisbon, and to follow each of these requests with a further request for retrieval of information on the available data.

Once the last selection was made, which is shown in red in Figure 21, the agent classified, as most highly rated regions, the ones also shown in the same figure in blue. It also highly rated the further request for retrieval (in Figure 18). In fact 6 of the highest rated command suggestions are of retrieval and with the lowest measures of dissimilarity (as can also be seen in Figure 18).

Figure 23 shows the complete process of the recommendation generation. First, the user selects a region and it is compared with the requests held by the memory, using predictor restriction. Then the values in the weights tables w_{XXX_gXXX} and the intermediary tables d_{gXXX_XXX} are combined to generate the δ_{gXXX} table which contains the measures of dissimilarity for every relevant request

contained in MBR. Finally, the 10 best results of the delta values are taken from the tables and combined to generate the final request and best fit region (as in Figure 21).

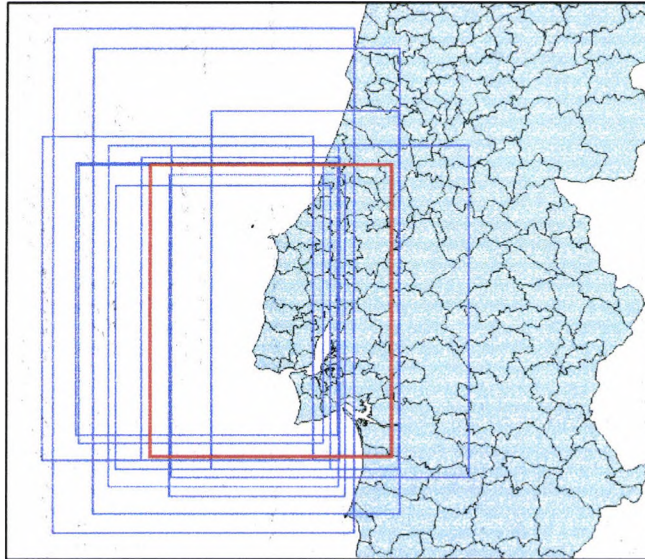


Figure 21 – In blue, examples of requests selected by the agent after the user has selected the region in red.

user	date	time	number	gsleft	gsright	gsbottom	gstop	delta
user1	04-02-1996	15:11:26	4	-7701,5482	164391,96	110660,12	372833,82	0
user1	04-02-1996	15:10:36	5	36770,742	202457,81	130410,19	318247,94	0
user1	04-02-1996	16:00:36	4	18960,702	130116,83	140111,74	309727,76	0,151155156
user1	04-02-1996	16:01:00	4	371,61747	130478,38	125663,02	318043,1	0,151155156
user1	04-02-1996	16:01:26	4	34398,283	133663,67	120008,66	301664,31	0,151155156
user1	04-02-1996	14:19:23	4	-30458	139256	99486	384296	0,151155156
user1	04-02-1996	14:20:21	7	-17778	129177	154757	308217	0,151155156
user1	04-02-1996	14:21:32	17	-37261	115469	141042	323176	0,302310311
user1	04-02-1996	16:02:31	4	59088,976	164033,85	135514,17	338166,33	0,302310311
user1	04-02-1996	16:03:39	6	-15793,562	85783,825	191756,17	307381,5	0,302310311
user1	04-02-1996	16:06:07	4	-86133,766	131634,38	90624,852	376320,68	0,453465467
user1	04-02-1996	16:05:28	4	5297,1909	126055,41	135482,75	295557,61	0,453465467
user1	04-02-1996	16:04:31	12	83905,322	229184,61	159457,23	283871,94	0,453465467
user1	04-02-1996	16:07:11	4	224912,78	338539,05	380717,32	484463,05	0,631136578
user1	04-02-1996	16:07:40	9	-36199,509	184037,71	101500,07	319161,41	0,631136578
user1	04-02-1996	16:25:59	24	24584,832	161093,58	141947,57	307708,19	0,782291734
user1	04-02-1996	16:16:58	4	-56062,009	143932,61	103450,82	333074,27	0,782291734
user1	04-02-1996	16:24:47	6	24584,832	135498,19	135853,43	296738,74	0,782291734
user1	04-02-1996	16:25:09	9	29460,144	142811,16	139509,91	311364,68	0,782291734
user1	04-02-1996	16:25:19	12	34335,457	144029,99	133415,77	306489,36	0,782291734
user1	04-02-1996	16:25:27	15	28241,316	154999,44	140728,74	311364,68	0,782291734
user1	04-02-1996	16:25:40	18	22147,176	152561,78	146822,88	304051,71	0,782291734
user1	04-02-1996	16:25:53	21	27022,488	151342,96	139509,91	312583,51	0,782291734
*				0	0	0	0	0

Figure 22 – The highest rated regions after one spatial selection. Their spatial mapping is presented in Figure 21.

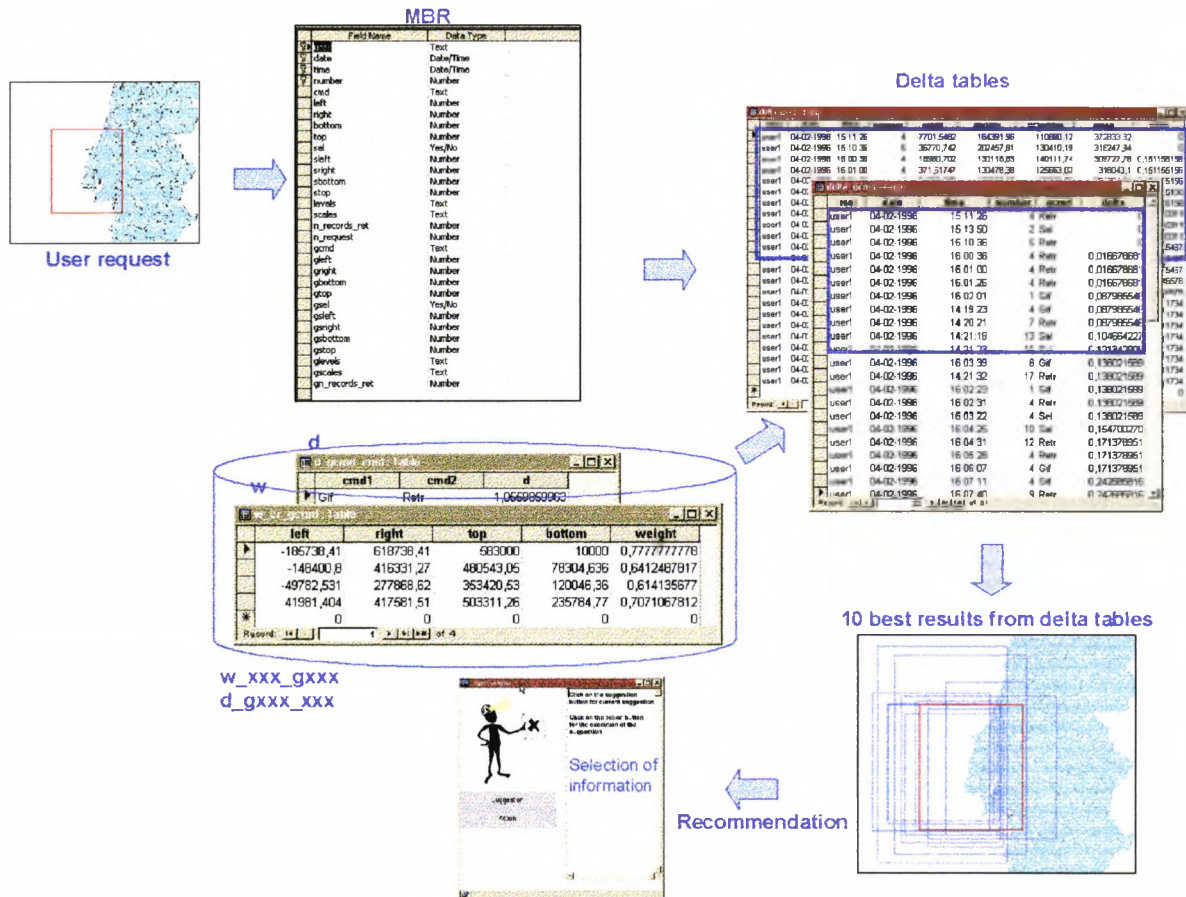


Figure 23 – Generation of information for recommendation

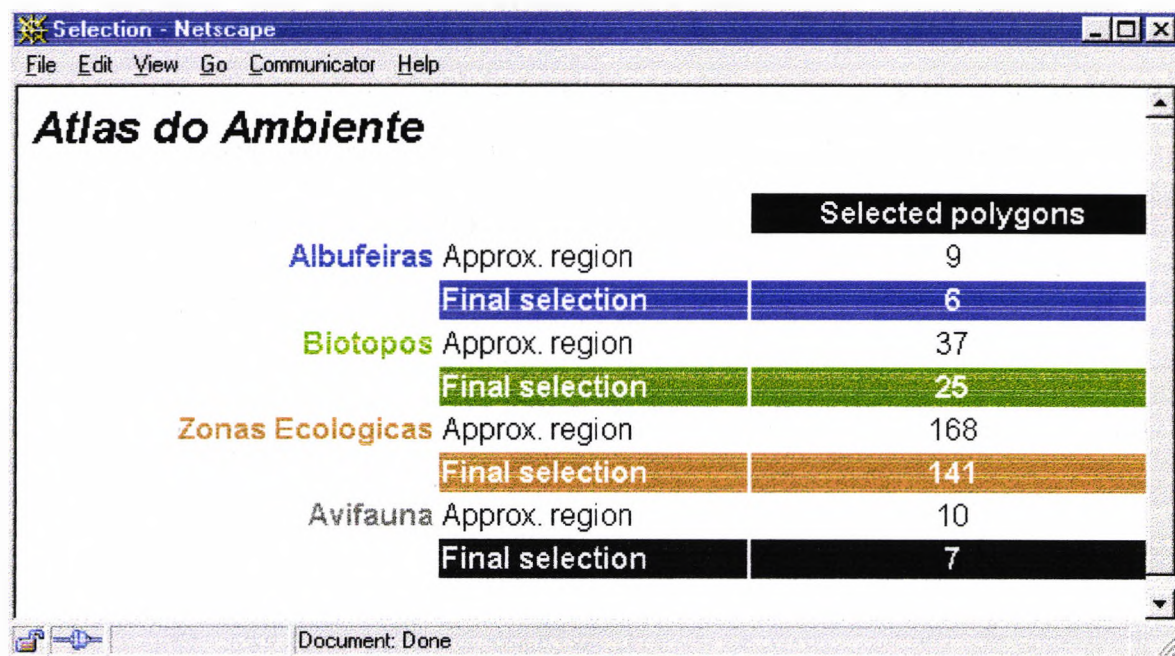


Figure 24- Metadata results for the example. This figure shows the display of the number of entities selected by the sum of relevant regions and by the region itself

6.4 User testing

Four user profiles were defined to test the possibilities of the developed prototype and the relevant metadata to those profiles were added to the interface of the java applet. These profiles were schematically defined on purpose, with the aim of creating simple but significant tests and with the hope of finding the prototypical work situations (as described in chapter 3) relevant for each of the profiles from the actual use of the system.

6.4.1 User Profiles

The following user profiles were defined:

- Public user (traveller/tourist) - this profile should show the preferences of the public user. The metadata thought relevant for this profile included road networks, gas stations, hotels, restaurants, heritage, natural parks, and topography. The spatial preferences of this profile are not predictable as a traveller may search for information all over the country and at different scales;
- Architect/urban planner - this profile should belong to a professional working in urban planning. Their spatial preferences are usually focussed on local information, mostly in the area of the municipality ("concelho"-county) the user

works at. The metadata preferences should include the Municipal Plan (which includes zoning, planning constraints and spatially referenced regulations about both), elevation, hydrography and possibly information about facilities located in the municipality in question;

- Central Administration – a professional working at a central government administration in Portugal will be concerned with metadata available at the district or municipality (“concelho”) level. The specific theme information should include roads network, hydrography and altimetry;
- Business professional – this profile will include users who handle relevant information for the evaluation of possible locations for the building of a commercial/industrial site. In this case the professional could be interested in information at the national, municipal or even at a lower level (called Freguesias in Portugal). The typical business user might search for census information (or geo-demographics), postcodes or street axes.

Twelve users were asked to use the system by adopting the above user profiles (three users per profile). They were asked to perform 20 searches of information according to these profiles and to ask the assistant to make suggestions on the 21st query.

These tests were not meant to be exhaustive, as this was a prototype implementation, but they aimed to show that the assistant could recognise a spatial pattern in the user’s use of the system, even if the user was not aware of it.

To recognise the level at which the user was working a string describing it was included as part of the requests inserted into the memory. If an identification of local user was identified the information inserted would be either “Freguesia” or “Concelho” (County) while the regional level would be described by the string “Districto” (District) and the national level by “Nacional” (National).

The twelve users were asked to represent the four profiles presented above by inserting the profile's patterns of use into the system. Below, the patterns suggested to the users are described.

Public type users

User 1 was asked to generate 20 different requests to the system in which there was no spatial pattern and no particular level. However, the metadata selections for this user were composed of different sets which included hotels, roads network and gas stations;

User 2 was asked to mostly select the city of Lisbon's Metropolitan Area, and metadata information which included restaurants and hotels.

User 3 reflects a preference on different geographic data at the "concelho" level and a metadata preference of tourist information.

Urban Planner type users

User 4 is a local planner who mostly selects a specific "concelho" and is concerned with zoning and constraint information.

User 5 is also a local planner who, besides asking for the planning information also requests environmental information for the "concelho".

User 6 is someone working with two neighbour "concelhos" in a water management project and therefore often selects information about those three "concelhos".

Central Administration type users

Users 7, 8 and 9 should prefer regional and district information which may include administrative boundaries, roads network, hydrography and altimetry.

Business Professional type users

User 10 was asked to insert a local pattern and search for postcodes and street axes for that area.

User 11 and 12 were asked to insert two areal patterns (at different levels) each: National and municipal.

In Figure 25 the results of the tests are presented.

6.4.2 Tests Results

Before describing the assistant's reaction to the users' requests after learning the preference patterns, it is important to note that some heuristics were introduced into the SIFIA system to enable it to deliver the normally preferred type of information for different profiles. The heuristics are the following:

- The recognition of a preference on the selection of the complete area of Portugal will provide the assistant the option of suggesting the retrieval of the administrative boundaries of the country at the various existing levels;
- If a pattern of frequent area selection that comprises and almost corresponds to a Portuguese "Districto" (District) is recognised the assistant will suggest the retrieval of information which is often useful for Central administrations, that is administrative boundaries, roads network, hydrography and altimetry. This pattern will be recognised even if the user selects different districts at each time of use. The pattern here is the recognition of interest in district information and not a specific district. The same happens in the next definition;
- A pattern containing the complete area of a concelho and whose area value is quite similar to the area of the next selection of the same concelho will allow the assistant to suggest the retrieval of local planning information.

After the execution of 20 requests to build the assistant's experience user 1 entered the system again and selected a region rectangle. The assistant, with a memory composed of metadata preferences of roads network, hotels and restaurants, suggests that the users ask for the download of the same information for the area now chosen. It does this with a confidence of 68%. The confidence value takes into

account the fact that the user gives no pattern for the geographical area and has selected other types of information in some of the submitted requests.

User 2, who also belongs to the public profile, has enabled the assistant to build up experience on the preferred geographic area, Lisbon's metropolitan area. The user has, in most of the submitted requests, showed a preference for information on restaurants and hotels. This has been done on a high percentage of the requests and, therefore, the result has a level of confidence of 78%.

The third user for this profile has selected different regions, which have mostly been identified with a *concelho's* (county) complete region. The user has also shown a pattern of interest for tourist information. Because the user has most of the time, selected a region which almost corresponds to a *concelho* region, the assistant uses the heuristic information and offers the delivery of not only tourist but also local planning information. The confidence value is lower because there are actually two profiles recognised here: the planner and the public user.

On the second profile, which should reflect the work of an urban planning professional, the same type of use appears. User 4 has a pattern of choosing an area that corresponds to a *concelho*. This pattern leads to a suggestion of retrieval of local planning information by the assistant, which is useful for this user.

The assistant behaves in a similar way towards user 5 who has generated the same pattern but with an interest in environmental information. The assistant recognises these two interests and suggests the retrieval of local planning and environmental information.

Finally, urban planner user 6, someone working in a *concelho* and collaborating with professionals in other two municipalities has searched for information regarding the sum of the regions of the three *concelhos*. Although the geographic pattern has been recognised, the fact that it is the sum of the areas of three *concelhos* has not. Therefore, instead of suggesting the retrieval of complete planning information for the three *concelhos* the assistant suggests the retrieval of

the personal preference of the user, zoning and constraints (both part of planning information).

The pattern presented by user 7 has been divided between requests for information at the district level and “concelho”. The result is that a request of the user for a district area with a specific interest in administrative boundaries will only generate solutions whose confidence is lower than 50%. Therefore, the assistant does not provide a suggestion at this specific moment.

	User last request				Agent's suggestion				
	command	Selected region	levels of use	requested themes	command	Selected region	levels of use	suggested themes	confidence
Public Profile									
user1	region selection	region1			retrieve info	region1		road network, hotels, restaurantes	0,68
user2	region selection	region2 (Lisbon metropolitan area)			retrieve info	region2		restaurants, hotels	0,78
user3	region selection	region3	concelho (county)	tourist information	retrieve info	region3	concelho (county)	local planning info, tourist info	0,54
Urban Planner									
user4	region selection	region4	concelho (county)		retrieve info	region4	concelho (county)	local planning info	0,73
user5	region selection	region5	concelho (county)	environmental info	retrieve info	region5	concelho (county)	local planning and environmental info	0,63
user6	retrieve	region6	3 concelhos (counties)	constraints	retrieve info	region6		zoning, constraints	0,54
Central Administration									
user7	region selection	region7	Districto (District)+ concelho (county)	administrative boundaries	No suggestion (prediction lower than 50%)				
user8	region selection	region8	Districto (District)		retrieve info	region8	Nacional (National)	administrative boundaries, roads networks	0,65
user9	region selection	region9			retrieve info	region9		administrative boundaries, hydrography, altimetry	0,75
Business Profile									
user10	region selection	region10			retrieve info	region10	freguesia (local)	postcodes, street axes	0,68
user11	region selection	region1	concelho (county)	geo-demographic info	No suggestion (prediction lower than 50%)				
user12	region selection	Portugal	Nacional (National)	administrative boundaries	retrieve info	Portugal	Nacional (National)	postcodes, street axes, administrative boundaries, hydrography, altimetry	0,53

Figure 25-Testing results. The tests evaluate the behaviour of 12 users identified with 3 group profiles.

The experience of user 8 was very similar but in this case the request for an area that corresponded to a district generated a suggestion of retrieval of information at that level such as administrative boundaries, roads networks, hydrography and elevation, as the central administration profile suggests. The confidence level is of 65% because the specific user has also submitted requests for municipal (“concelho”) information.

User 9 inserted a pattern that has not been identified (in most of the cases) as a district or a “concelho”. However, the user has previously requested central administration information, which has created a pattern of use for that specific area with a preference for administrative boundaries, hydrography and elevation.

For the final profile, information for business, user 10 has shown a preference for a local area and has previously searched for postcodes and street axes. After a request for a similar local area, the assistant suggests the retrieval of the same theme information for that area, as has been done before.

However, user 11, who has submitted requests of the same type of information, does not receive a suggestion, because the geographic selection has included regions of different levels. The confidence levels of the various possible suggestions are, thus, under 50%.

Finally, user 12 who has shown a preference for national information (requesting data for the whole of Portugal), after a final request for the administrative boundaries with no particular geographic selection (thus requesting national information), receives a suggestion for retrieving, not only the administrative boundaries, but also the rest of the central administration related data. Moreover, the data personally chosen by the user, for that area, postcodes and street axes are also suggested for retrieval.

6.4.3 Interpretation of results

From the results presented above and given in full in

Figure 25 it is now possible to draw an interpretation. These tests were useful not only to show how the system actually works (and what are the actions resulting from the learning) and to realise the cases in which it can be successful, but also to evaluate its faults, which can be improved in later implementations.

In fact, the user profiles drawn from general knowledge of what the information needs of four groups of users are have provided some insight in what these preferences actually mean in the context of a memory-based system depending on individual patterns of use. Moreover, the personal preferences of a user, added to the group's profile can deliver some unexpected results that must be accounted for.

This is the case for the public user who represents the most general profile that receives a suggestion for retrieval of planning information because the system has detected a pattern of use focussed on a municipal area. This may be considered wrong, but it may also be considered right. There must be some reason why a traveller always picks an area similar to a "concelho". The assistant actually decides to suggest a preference that the user may not yet know.

The recognition by the assistant of a geographic pattern provides two advantages to the user. The preferred information is reached more rapidly, new information becomes available and new information belonging to the pattern will not have to be requested by the user.

The use of the heuristics may fail in some cases but may show the users a preference they did not realise they had. The problems of wrongly assuming a pattern may be solved by providing the user with a feedback mechanism on the assistant's suggestions.

Another weakness of the system is the incapacity of the assistant to recognise a pattern of several "concelhos" shown in the suggestions made to user 6, or the central administration profile in the behaviour of user 9. This problem can be solved by inspecting the selected region in terms of the lowest relevant hierarchical geographic level. In this way the area could be identified as a set of concelhos.

Another possibility would be to analyse the user's theme preferences which showed a preference for zoning and constraints, typical of urban planning. However, the latter solution does not involve a spatial evaluation of the preferences.

6.5 Discussion

As stated in chapter 1, the aim of this dissertation is to analyse research issues in geographic information science in the areas of spatial information handling and spatial reasoning; to study the potential of spatial intelligent agents in this area of research; and to explore the use of simple learning techniques to improve the adaptability of spatial intelligent agents.

Specifically in this chapter, the objective was to study the potential of using agent-assisted applications for the location and access of geographic information. They were intended to be used with the specific purpose of geographic information location and to improve geographic information system interfaces. In this context, it was considered of major importance to explore simple learning techniques to improve the adaptability of spatial intelligent agents (in this case, an interface assistant in a geographic information system).

From the tests carried out with this implementation, the following conclusion can be drawn: it is possible to build up information use experience through a spatial information facilitator that leads to the management of patterns of a spatial and non-spatial nature. These patterns can include (depending on the use that is made of the facilitator) the type of command most often used; the preferred visualised area; the preferred region (or preferred geographic unit) and the preferred type of information (in this case, classified by theme). However, the tests presented in the previous section concentrated on the latter three.

The above findings will improve the use of this type of information in the following cases:

- a spatial pattern of selection for retrieval recognised by the assistant will allow the agent to select and retrieve new information that may become available for

the preferred region. This means that the user will not have to look for new information becoming available in their region of interest;

- the same observation can be made for information which is included as part of the preferred layer type (theme) selected by the user;
- the identification of the preferred geographic unit facilitates the correspondence with a specific profile of use and thus, the suggestion of retrieval of relevant information for that user type;
- it is possible to build profiles of use for specific users who may wish to do that, automatically zooming into the preferred areas and retrieving relevant information. Moreover, these configurations may become part of their personal interface to the system. The experience kept in the memory is entered as part of one specific user's previous experiences. If different users enter the system and log onto different user profiles, their experiences are kept separate and the calculus of suggestions is personal.

The potential recognised above could be realised in the Portuguese National Geographic Information Infrastructure – the SNIG. The complexity of the metadata information available at present is a limitation in the building of a user search interface. However, given the possibilities of the tools presented in this chapter, it also represents a potential for profiling and personalisation where relevant. Preferences of file format, or spatial reference system are normally long lasting, while other attributes may vary with time.

The potential of the work described also illustrates its limitations. There is much work to be done, especially in the following:

- improving the system in itself: the implementation is very simple and its integration in a production environment will involve the rethinking of many of the choices taken in terms of architecture and tools used;

- problems of performance have not been addressed. It is necessary to evaluate a similar system available on the web with several users accessing information and producing reasoning at the same time;
- the metric used for handling spatial regions must be improved. Goodchild (1998) suggests that the use of fuzzy regions and fuzzy queries in the handling of geographic locations can be useful for locating and accessing information in spatial digital libraries. Its use for the same purpose in the context of geographic information infrastructures may be a solution;
- the limitations of the tools used in online mapping also limited the resulting implementation. As new versions of online mapping software become available, the possibilities of overriding these limitations grow. The analysis of these new tools for this purpose is of major importance;
- learning can be improved if patterns are translated into rules and if feedback from users can be integrated in the formulae calculating the value of each action in the system; the weights of predictor fields may be re-evaluated according to feedback from the user (Kozierok and Maes, 1994);
- Finally, a modular implementation of the assistant would be much more natural to implement and easy to maintain. The possibility of storing the memory locally must be pursued.

These findings might also be taken to go further than a falsification of the premise that agent systems cannot reason. The results are indicative of the ability of the agent to detect patterns of searching within area hierarchies – and this may make a stronger case for the use of spatial agents within GIS interfaces to geolibraries.

Ch. 7

Car Park Agent Simulation

The car park agent simulation was conceived as a testing environment for the methodology presented in chapter 4. The idea was to create a simulation, which would involve a spatial environment with elements positioned in it. Some of these elements would be adaptive agents endowed with reinforcement learning capabilities, which would enable them to improve their capacity to take actions with spatial implications.

This simulation is therefore composed of a car park, which can be of several shapes, with one or more entrances and one or more exits. The car agents enter the car park through one of the entrances and leave it by one of the exits. In each

execution of the simulation each car enters to park in the car park only once. However, they may or may not store every parking experience they have in the form of action-state pairs of the Q matrix (associated with Q learning as described in the chapter 4 section 4.5.1.1).

The aim of the simulation is to park all of the cars, according to their preferences, as quickly as possible. These preferences may include: a preference for parking quickly, near the car exit, or near specific positions in the car park, which may be associated with exits for people (e.g. specific exits for a shopping centre).

As will be demonstrated, the agents start by choosing car spaces and, as they get feedback by way of the reward function adapted to their preferences, start to go directly to parking spaces that are better for them.

This type of simulation was chosen because it involves several decisions taken by the agents, all of them of a spatial and temporal nature. As will be seen below, the learning algorithm used enables the agents to react conditioned by the concepts of relative location, distance, direction and speed, all of which have spatio-temporal dimension. It will also enable them to review their current decision, at each step, as the environment changes.

Finally, the car agents moving and acting on the environment also interact with each other by way of their movements and realisation of the position of others. Although the agents are all algorithmically similar, they may have different configurations and preferences, which enables a rich testing of the simulation.

7.1 Simulation description

In this section, a description of the potential of the simulation is given, with the objective of demonstrating how complex the simulation could be. It is not a description of all the potential options, but an exploration of how flexible the methodology could be from the point of view of the potential of the agent architecture accomplished.

This simulation is composed of the following;

- One car park, with a defined shape, containing several parking spaces, one or more entrances and one or more exits;
- The cars, adaptive agents, searching for a empty parking space that fulfils their preferences and learning the best way to search for that space;
- The parking spaces, static objects which may be full or empty;
- The entrances, one or more, which the cars use to enter the simulation;
- The exits, one or more, which the cars use to exit the simulation.

7.1.1 Environment – car park

The environment of the simulation is a car park. The car park has been defined as an entity with properties that are configurable such as its shape, the number of spaces, entrances and exits. At each step of the simulation it is possible to know: if the car park is full or empty; how many cars are inside the car park; how many cars have parked; how many are looking for a parking space; and, how many are moving in order to leave the car park.

The car park uses four lists (vectors): the list of car spaces, the list of entrances, the list of exits and the list of processes (threads), each associated with the execution of a car agent. Each of the elements in these lists is defined as a specific object. The access to the list enables the access to these objects.

The creation of a car park is accomplished by the definition of:

- The shape of the car park, as a polygon;
- Its name;
- The maximum number of cars in the simulation;
- A list of car spaces, where each space is defined as a specific object;
- A counter of full spaces, initialised to 0 (zero);

- Full park flag, initialised as false;
- A list of entrances, where each entrance is a point object;
- A list of exits, where each exit is a point object;
- A list of processes (threads) for the parallel execution of the car agents.

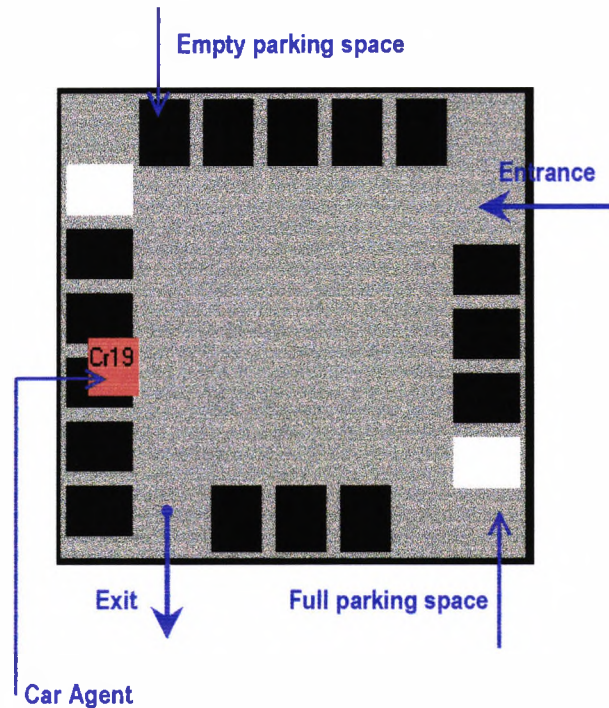


Figure 26 – An example of the car park environment: here there is one marked entrance and one marked exit and 18 parking spaces. As a car reaches a parking space (like car 19) just before parking, it asks permission to the parking space. A red car is looking for a place to park, while a blue car is moving towards the exit, after parking.

The global state of the car park evolves from the following events:

- Adding a new process to the list of threads in the car park, which means that a new car has entered the park;
- When a car parks, its state changes and the number of parked cars is raised by one; if the number of parked cars equals the number of cars then the state of the car park changes to full;

- When a car leaves the space where it has parked, the state of the former changes and the number of parked cars is reduced by one.

While the simulation runs, the car park can provide several percepts for the car agents, providing information on the activity of the simulation:

- The distance between two points of the simulation (which may represent either spaces or the location of cars);
- How many cars are in the car park;
- How many cars are leaving the car park;
- How many cars are parked;
- How many cars are looking for a space to park.

7.1.2 Agents – Cars

The agents in the simulation are the cars trying to park. They have been created as entities (objects), with the primary aim of parking according to their own preferences. Each car is given its own process (thread) which exists as long as the agent is in the simulation environment. Each car has the following information:

- Name;
- Time of creation;
- Location in the simulation at the time creation;
- Current location;
- Speed of movement;
- The importance given to the distance between the chosen parking space and one of the exits (which may be specific);
- The importance given to the time the agent takes to park;
- The period of time during which the car will stay in the car space;

- Information on whether the car is inside the car park;
- Information on whether the car is parked;
- Information on whether the car has left the space where it parked and is leaving the car park.

This is the information needed to create a new car and associate it with a new thread.

The car changes its state through the following functions:

- Birth: the car is in a waiting state until its time of creation comes up. Then it is allowed to move and enter the car park;
- Moving towards a specific point: given the location point which the car agent aims to reach, it moves towards it by calculating a direction taken from its current location and the aim location;
- Park: the agent attempts to park in a specific space: the agent asks the parking space if it is empty and if it can park in it. More than one car may be trying to park in the same place, so parking spaces are managed as synchronised resources;
- Leaving the parking space: once the period of time to stay in the parking space runs out, the agent leaves it and starts moving towards its specific exit;
- Leaving the car park: once the agent has reached the exit that it aimed for, it leaves the car park and exits the simulation.

7.1.3 Passive Elements

The passive elements in the simulation are objects that inhabit the simulation, suffer interactions from the agents, but do not initiate actions, nor react to stimuli. In this simulation, there exist three types of elements, which have not been modelled through agents:

- Parking spaces;

- Entrances;
- Exits.

Parking spaces are the physical aim of car agents. Their execution is carried out with the aim of reaching the empty space that suits them best. Therefore, cars interact directly with them.

Entrances are the first target of interaction by the car agents, as they enter the simulation. They enable their entrance in the simulation. The simulation may have one or more entrances and the developer defines their location.

Exits are the ultimate target of interaction from the car agents. They are not an aim of the agents in themselves, because the task of the agent is fulfilled once it has parked. Afterwards, when the agent's parking time limit runs out they start a movement towards one of the exits (their favourite) and interact with it, just before they leave the car park.

It is worth noting that this simulation is implemented not in a grid environment composed of discrete locations, but in a continuous vector environment. Each agent embedded in the simulation environment can move to any location in the environment and object entities belong to classes defined in the context of the simulation.

From the above, it is clear that the interactions of the cars with these elements are of major importance in the simulation and must thus be modelled. They provide information that adds to the experience of the agents and helps the process of adaptation.

In the following section these elements are thoroughly analysed.

7.1.3.1 Parking spaces

As said above, finding *parking spaces* is the physical aim of the car agents. All the cars execute with the aim of reaching the empty space that best suits them.

Although they have a previously defined visual area, they are defined as points (they are extended in a defined way from that point).

Parking spaces have the following properties:

- Coordinate x (inheriting from the point object) ;
- Coordinate y (inheriting from the point object) ;
- Width of the parking space;
- Height of the parking space;
- Place - Rectangle defined using the four properties above (to be used to draw the parking space on the visual interface);
- Information on whether the space is full or not;
- Link to the car that is parked in the space, if it is full.

Although the Parking space is not a very complicated object, its definition is fundamental for the execution of the car agents and for implementing the learning process.

The parking space changes state (full and empty) and therefore, influences the environment, by the interaction of the car agent with it. As the agent reaches the specific parking space it aimed for, it sends a message to it, asking for permission to park there. This has been implemented in this way to ensure that two cars do not park in the same space at the same time. Therefore, the action of parking in the space (as an object-oriented method of the parking space) is a synchronised method, which means that only one object can call it at a time. The whole action of parking is implemented as a complete transaction, which either runs completely, or does not run at all. In this process, the state of the parking space changes from empty to full and a link to the car that parks there is created.

When the car asks the space to leave it, the change of state is reversed. It is important to note that, in this case, there is no need for the implementation of a

synchronised method, as only one car will try to exit a parking space: the one that is parked there. Figure 27 and Figure 28 present the code used for the parking procedure. Both methods are synchronised to ensure that only one car will park in each parking space at one time.

```

public synchronized boolean park(Space spa)
{
    if (spa.park(this))
    {
        /* if the parking space allows the car to park
        then the state of the car changes */
        this.parked=true;
        return true;
    }
    else
    {
        return false;
    }
}

```

Figure 27 – Parking function of the car agent. The car will only change its state if parking has been successful near the parking space.

```

public synchronized boolean park(Car c)
{
    /* this is a synchronized method to ensure that only
    one car is able to park in each parking space at
    one specific time */
    if (!this.full)
    {
        this.full=true;
        this.parked_car=c;
        return true;
    }
    return false;
}

```

Figure 28 – Parking function of the parking space. This is a synchronised method, to ensure that only one car tries to park at each parking space at a time.

The parking space also adds functionality to the simulation by implementing the following functions, which are also used as percepts by the car agents:

- The calculation of the distance between the parking space object instance and a given point in the car park. This functionality is needed and used by several of the objects in the simulation, especially the agents. However, it may need different implementations for different configurations of the car park;

```

public long distanceTo(Point p)
{
    /* This method will be changed as the configuration of
    the car park changes */

    long d =(long) Math.abs (Math.sqrt(((Math.abs(this.x-
    p.x))*(Math.abs(this.x-p.x)))+(Math.abs(this.y-p.y))*(Math.abs(this.y-p.y)))));
    return d;
}

```

Figure 29 – Distance function used in the car park simulation. This needs to be changed if the configuration and the method of movement changes in the environment.

- Given the current location of one car agent, the parking space searches the car park for another car, which may be nearer to it than the first car. This is a very important function in the simulation because its result may influence the reward given to the agent at that step and the utility of the next action-state pair. As presented in Figure 30, in function `betterCar`, if the parking space realises that another car looking for a car space is nearer to it than the car being considered, then the function evaluates to true. This means that this fact will be considered in the calculus of the reward function at the next step in the car's execution, and the reward value given to the car for pursuing parking in that space will be lowered. The car that is better positioned to park may, in fact, not be interested in parking there. However, this information is not given to the car, as in a car park, car drivers do not know other drivers' preferences.

7.1.3.2 Entrances and Exits

The set of entrances in a car park is defined by a list (vector) of points, which define the spatial location of the entrance. This list may be composed of one or more points and is part of the definition of the car park. It is specifically defined by the developer when configuring the car park. The set of exits is treated in exactly the same way (in the presented example, the car park has only one entrance and one exit).

```

public boolean betterCar(CarPark cp, Point location)
{
    int i;
    Vector ct;
    Car c;
    long d1,d2=0;

    ct=cp.car_threads;
    /* the space searches in the list of all existing car
    threads */
    for (i=0;i<ct.size();i=i+1)
    {
        c= ((CarThread)ct.elementAt(i)).owner;
        if (!(c.exiting) && (c.in_car_park) && (!c.parked))
        {
            /* from the cars that are inside the car park and
            are not exiting-are looking for a space to park */
            d1=this.distanceTo(location);
            d2=this.distanceTo(c.current_location);
            if (d2 < d1)
            {
                /* if any of the cars are nearer to the space
                than the car being considered, the function
                returns true */
                return true;
            }
        }
    }
    return false;
}

```

Figure 30 – Parking space function betterCar. This function analyses if there are any car agents better position to park in it. The result of this function will influence the reward function.

7.1.4 Time and space

A car thread is owned by one car agent and drives the entire behaviour of the car. Associated with each thread is also one learning object (of the Learning class) that uses each step taken by the agent to add experience to the car Q-matrix.

For each car thread, a new simulation step is executed every five seconds. However, there is no control structure to divide computing time by the different threads. That is left to the programming language's thread control. This fact provides the simulation with a time division that gives a very "real" impression, as the division of time to each car is different at each run and is even unfair at times. Each car thread sleeps for five seconds and then tries to get a time slice to execute a new simulation step. This does not mean that it will run every five seconds, but that it will try to run every five seconds. By then, the car's objective (its preferred parking space) may have become full. This will make the agent review its objectives and choose another direction.

Some examples have shown that one car may have been waiting to move forward for several steps while others keep moving (due to the *ad hoc* division of time between car threads). This capacity to take (or not) the opportunity to move could be considered, in a real environment, as the driving experience of the driver of the car.

The division of time between car threads may give more opportunities of reaching a parking space to one specific car agent than to another. Thus, once a car that has been still for a while gets a time slot to act, it reviews its state and decides on the next action, its previous objective (a specific parking space) may have become full, or other cars may be nearer to it. The agent is thus obliged to review its decision and even take a new objective parking space. This is why it is considered that, in this simulation (as well as in reality), time and space are directly connected. Not moving for a while (that is, losing time instead of moving towards the objective) does degrade the spatial performance of the agents. The run method of the car thread is presented in Appendix D as part of the CarThread class.

Visualisation of the simulation environment is made every 2 seconds and is a snapshot of the environment at the time of visualisation. The simulation does not stop to be visualised and no control is allowed over the state of the system at the time of visualisation (this process is described below).

7.1.5 Changing the environment and interacting

In each of the components described above the functions that make the system evolve, are described. The fundamental functionality is summarised here:

- Adding car threads to the car park (one car agent enters the car park). This function involves the interaction of the new car agent with its preferred entrance;
- The awakening of a car to enter the car park;
- The movement of a car agent towards a specific point in the car park;

- A car agent is able to park in a preferred parking space. This function involves the interaction of the car agent with the parking space;
- A car agent leaves the parking space where it has parked, involving a further agent-space interaction;
- A car agent reaches its preferred exit and leaves the car park simulation.

All these functions are the fundamental events that enable the simulation system to evolve and fulfil its aim. To take the action, car agents also use percepts, which provide them with information on the way the system has changed and about its present state. This is described below.

7.1.6 Percepts

According to Russell and Norvig (1995, pg. 32), percepts are information structures, defined in the simulation and provided by sensors, which enable agents to acquire knowledge on the state and evolution of the system they are embedded in. In the above description of the components of the simulation, these have been noted and are described here:

- Information on the distance to or between relevant elements, either car agents, parking spaces, entrances or exits. These are basically used to achieve a relative knowledge about the utility of the agent's position;
- Information on the number of car agents in the car park, looking for a parking space, parked and exiting the car park. These percepts are used to provide global information on the current percentage of use of the car park;
- Information on the relative position of car agents which may be competing for the same parking space.

Most of this information is used in the agents' learning process. In the next section, the implementation of Q-learning for this simulation is addressed and the use of the above percepts for the definition of a reward function with spatial considerations is described.

7.2 Learning

The learning process used in this methodology has been extensively described in chapter 4. However, it is necessary to show how the ideas behind the theoretical description of Reinforcement Learning (and specifically Q-learning), presented in chapter 20 of Russell and Norvig (1995), were transformed in order to be implemented as the learning process of a car agent acting spatially.

Learning is implemented as an object associated with the car thread. However, if that is the case it could be associated with the car park and provide the agents with one common action-state utility structure. This may be useful when all the agents have the same configuration and can be easily implemented. It will certainly be less expensive to the computer, in terms of memory resources. The learning object holds several state properties, which are important in the learning process. These properties are divided in terms of the previous action taken by the car agent, and the next one. The learning properties are described below (in brackets are the names of the properties in the code in Appendix D):

- Initial direction – first parking space the car agent decides to park in (initial_direction);
- Expected number of steps¹⁷ to reach the initial direction (initial expected number of steps to achieve goal) (steps_initial_direction);
- Current car location (j_location);
- Reviewed direction at the previous step – location of the target parking space after using the learning tool, at the previous step(j_direction);
- Car location at previous step (i_location);
- Initial direction at previous step – location of the target parking space, before using the learning tool, at the previous step (i_direction);

¹⁷A step is the movement a car takes at each time period assigned to the car thread.

- Target space – location of the target parking space for the action to be taken at the current step (target_space);
- Number of previous target spaces for the current car, at the current experience of execution (n_previous_target);
- Number of steps taken until the current step (n_previous_steps);
- Number of steps needed to reach the current target (n_steps_target);
- The value of the reward for the current step, that is a value that quantitative measures the utility of being at the present state at that specific point in time (current_reward);
- Type of action taken (space, centre or beginner). The action taken is classified as beginner when the agent has just entered the simulation and has not yet taken any decision on its preferred parking space. After the first decision, the type of action is either space or centre. The centre type of action is only taken when all the spaces in the car park are full, and the car agent has to wait until one becomes available. The choice of the position to wait at also goes through a reinforcement process (action).

From these properties, it was possible to define a reward function, which takes the spatial state of the simulation and of the car agent into account. This reward function is then used to update the Q-matrix values of the agent, incorporating the evolution of the environment. In the following sections, these functions are described.

7.2.1 Q matrix

The Q-matrix associated with Q-learning is composed by the utility values of each action-state pair in the simulation. The double index of this matrix is, therefore, composed of a pair of identifiers: the identifier of the state of the agent and the identifier of the possible action to be taken at that state. In this case the state is

identified by the number of the parking space that the agent will be moving to. The actions are either the maintenance of the objective, or the choice of a new objective.

$Q(i,j)$ means the utility value of deciding to go to parking space j when the previous objective was to go to parking space i . In Figure 31 is an example of a Q matrix in a car park of 18 places. The 19th state is the centre state, taken when the car park is full. The matrix presented shows a new car park agent that has initially entered a full car park. As parked cars start to leave the car park, parking spaces become empty and the utility of leaving the centre position and moving on to the new empty space is raised.

7.2.2 Spatial reward

Figure 34 presents the reward function of the Learning class. The specificity of the method is explained below.

The reward function for the car agent is defined in two parts: when the car park is full, and therefore the agent cannot immediately park (action=centre); and when there are available parking spaces and the agent has to make a decision on which one is best (action=space).

If, in the previous step of the simulation the action taken was to move towards an empty parking space (Figure 32), then the value of the reward includes the distance between the agent and the parking space. Of course, this value can be parameterised according to the agent's preferences. Besides this, the existence of another car agent between this one and the selected parking space is also taken into account.

If the car agent is currently moving towards a waiting location, the reward is a combination of the following values:

- The relationship between the number of cars in the car park as opposed to the number of existing car threads;
- The relationship between the number of parked cars and the maximum number of cars in the simulation;
- The relationship between the number of cars which are looking for a parking space and the number of parked cars;
- The relationship between the number of cars exiting the car park and the number of parked cars;
- The distance between the car agent and the entrance it used to enter the car park.

```
if (action==centre)
{
  entrance=(Point)(cp.entrances.elementAt(0));
  dist=distanceBetween(j_location,entrance) /32 ;
  if (dist !=0)
  {
    pos=1/dist;
  }
  else
  {
    dist=1;
  }
  icp=cp.carsInCarPark();
  pa=cp.carsParked();
  cen=cp.carsEntering();
  cex=cp.carsExiting();
  if (icp < 0.7)
  {
    r=0.002;
  }
  else
  {
    r=icp*pa*cen*(1-cex)*pos;
  }
}
```

Figure 33 - Part of the reward function referring to agents that were previously moving towards an awaiting position (action=centre).

These values are only used to calculate the reward value if the number of cars inside the car park is higher than 70 percent of the number of car threads. This

value was chosen because at state of the environment, it is quite difficult for agents to park. The existence of many other agents in the car park, all of them looking for a free space, raises the probability of every agent entering the car park being at a worse position to park than some other agent. If the percentage of use is lower than that, it will not be very difficult for an agent to find a parking space, once they start becoming available. Therefore, the reward value given to the agent will be very low, so that as soon as a parking space becomes available, it will start moving towards it.

The calculus used to generate the reward value is the result of an empirical evaluation of the behaviour of parking cars. The performance of the learning algorithm could be improved from a careful statistical analysis of the behaviour of cars in car parks. However, they are sufficiently successful to provide interesting results, and to recommend further use of the methodology, which was the objective of the experiment.

As can be seen, the reward function is a combination of the values of global parameters of the simulation (e.g. number of cars in the simulation - icp , number of parked cars- pa) with the results of local agent functions, like calculating its distance to other elements. This type of adaptation cannot be achieved when using numerical approaches in simulation. It may be done with other types of individual-based models approaches like complex types of Cellular Automata but not in such a natural way as this. Moreover, all of the local agent functions used in the reward are of a spatial nature, which enables the agent to use its location and that of others in the simulation to assess its performance. This is a new finding as it has not previously been implemented in the literature.

```

public double reward(CarPark cp) throws IOException
{
    Point entrance;
    double pos=0, icp, pa, cen, cex;
    double r=0, location_direction=0, direction_exit=0, dist=0;
    Point exit=(Point) (cp.exits.elementAt(0));
    if (action==space)
    {
        location_direction=distanceBetween((Point)j_location,
            (Point)j_direction)/32;
        direction_exit=distanceBetween((Point)j_direction,
            exit)/32;
        if (location_direction!=0)
        {
            r=1/location_direction;
        }
        else
        {
            r=1;
        }
        if (((Space) j_direction).betterCar(cp, j_location))
        {
            r=r*0.3;
        }
        else
        {
            r=r*0.7;
        }
    }
    if (action==centre)
    {
        entrance=(Point) (cp.entrances.elementAt(0));
        dist=distanceBetween(j_location, entrance) /32 ;
        if (dist !=0)
        {
            pos=1/dist;
        }
        else
        {
            dist=1;
        }
        icp=cp.carsInCarPark();
        pa=cp.carsParked();
        cen=cp.carsEntering();
        cex=cp.carsExiting();
        if (icp < 0.7)
        {
            r=0.002;
        }
        else
        {
            r=icp*pa*cen*(1-cex)*pos;
        }
    }
    return r;
}

```

Figure 34 – Reward function algorithm

The Q-learning function is implemented using the characteristics and the algorithm described in chapter 3. The fundamental perspective of the Q-learning algorithm is

that it updates the values from the previous simulation step to the new one.

Specifically:

- The variables used in the learning and in the calculus of the reward function are updated;
- The new reward value is calculated;
- The utility values in the Q-matrix are updated;
- The action to take in the next step is found from the new utility values in the updated Q-matrix.

For reasons of extension, the Q-learning function is not included here. However, the complete algorithm is included in Appendix D.

Another important feature to take into account, which is part of reinforcement learning techniques, is that, while the agent's experience is limited, the algorithm declines to use the previous experiences and makes the agent take action at random. This is what Russell and Norvig (1995, pg. 612) call the *exploration function*. The exploration function allows the agent to go around the environment taking various types of actions until it becomes experienced enough to start making learned decisions. In this case, the agent searches for the parking spaces that it has not used before and makes the decision to park there, without considering previous experience. This can be seen in part of the `max_action` function of the Learning class:

```

if (this.littleExperience(cp))
{
    if ((action==beginner) || ((action==space) && (target_space.full)))
    {
        i_less_used=less_used_space(cp);
        if (i_less_used != 18)
        {
            target_space=(Space)cp.spaces.elementAt(i_less_used);
            action=space;
            return action;
        }
        else
        {
            action=centre;
            target_space=null;
            return action;
        }
    }
}

```

Figure 35 – part of the max_action function showing the implementation of exploration in this simulation.

7.3 Implementation issues

The simulation was implemented in Microsoft J++. Java was the language used because it is an object-oriented language with multi-threading capabilities that can be event driven and includes exception handling. The fact that it is an easy to use language that enables rapid prototyping was also taken into consideration (Rodrigues and Raper, 1999).

Several modules have been implemented as part of the structure that controls and configures the elements of the car park simulation. These modules are:

- The *administration module*, which enables the update of all the configurable values in the simulation;
- The *statistical module*, which helps establish the potential of the learning facilities by measuring different parameters for different types of car agents and for different simulations;
- The *control module* which serves as a type of remote control on the simulation;
- The *simulation window*, accessible from every module, which allows the visualisation of what is going on in the simulation environment.

The *administration module* is the configuration tool enabling the user to change configurable elements in the simulation. Several shapes for the car park are available, each one with a different number of parking spaces, entrances and exits. The number of car agents is configurable for all types of car parks, although every configuration can be created with default values. The preferences of the agents can be configurable one by one. As this may prove to be rather slow, the definition of subsets of agents with identical preferences is possible. To configure an agent means to define the weights of their preferences on their speed of parking and closeness to a specific entrance or exit. Once the configuration is made, it is possible to store it for further use. The learning experience of the configuration can also be stored. At each execution of the configuration it is also possible to decide if the knowledge from previous experience should be used or not and whether the new experience should also be integrated.

The *statistical module* enables the calculation of several values that will evaluate the evolution of the performance of the simulation as the utility information resulting from Q-learning is used. For each execution of one specific configuration of the simulation, the statistical module presents:

- For one specific agent, in one configuration:
 - Number of target parking spaces considered as objectives by the agent;
 - Number of steps taken by the agent until it reaches the parked state;
- For all the agents in one simulation:
 - Mean number of target parking spaces considered as objectives;
 - Mean number of steps taken by the agents until parked state.

The *control module* enables the control of the simulation, while it is already running. At any time the user can:

- Pause and continue the simulation;

- Add new car agents during execution;
- Identification of car agent by clicking on it.

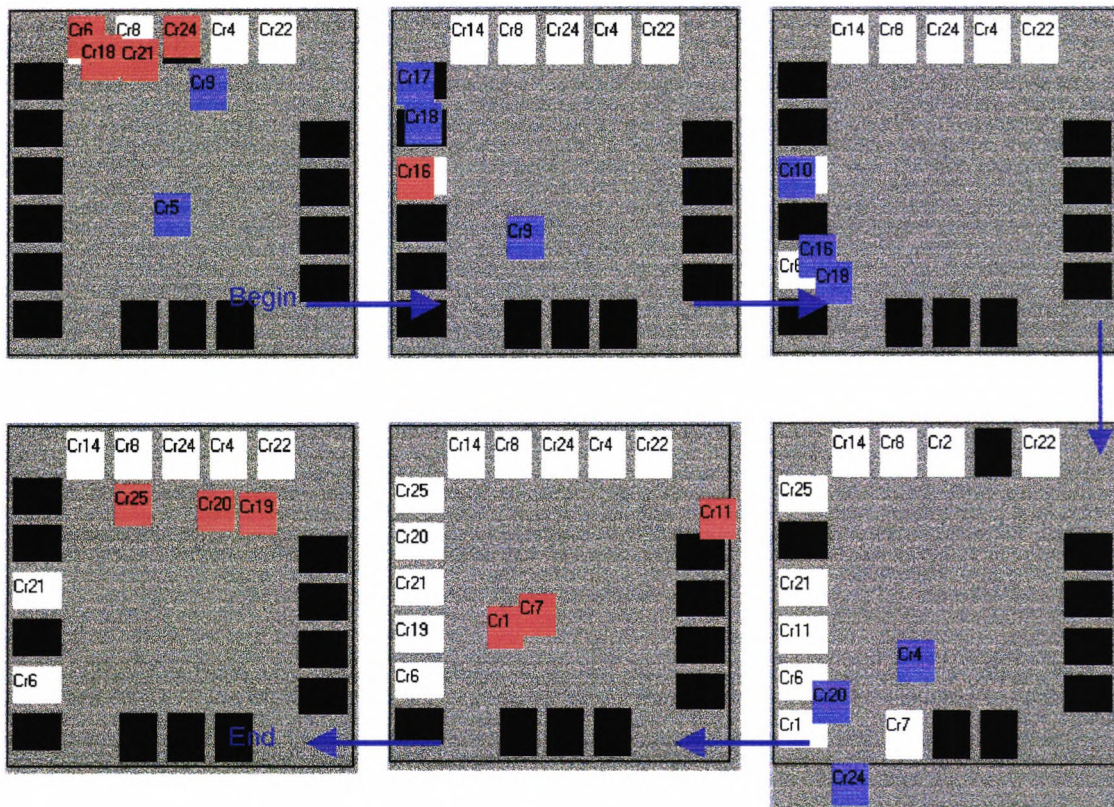


Figure 36 – Example of the simulation with agents that do not hold any previous knowledge. As the placing is done iteratively, they search for parking spaces in a series. The read cars are looking for a space to park, while blue cars are leaving the car park.

Finally, the simulation window provides a visualisation medium of moments in the execution of the simulation. At regular intervals, the visualisation window is redrawn, reflecting the changes in the simulation global state since it was last drawn. Each car is represented by a rectangle with the agent's identification number on. Different types of car agents have different colours, reflecting their different preferences.

7.3.1 Running the simulation

The examples of running simulations presented in Figure 36 and Figure 37 were taken from the car park configuration presented in Figure 26: a square shaped car park with 18 parking spaces and 25 car agents entering the car park some time during the execution of the simulation.

The car park has one entrance (on the top right side) and one exit (on the bottom left). Each car's configuration includes a preference for parking as soon as possible (that is, taking as few steps as possible) and two randomly generated times: the time at which the car enters the car park after the simulation has started and the time period the car stays in its parking space once it has parked. Both of these time periods are less than one minute.

In Figure 36 and Figure 37 some examples of the running of the simulation are presented. In Figure 36, the agents are all new (they have had no simulation experience). In Figure 37, they are trying to park as fast as possible using their previous experience.

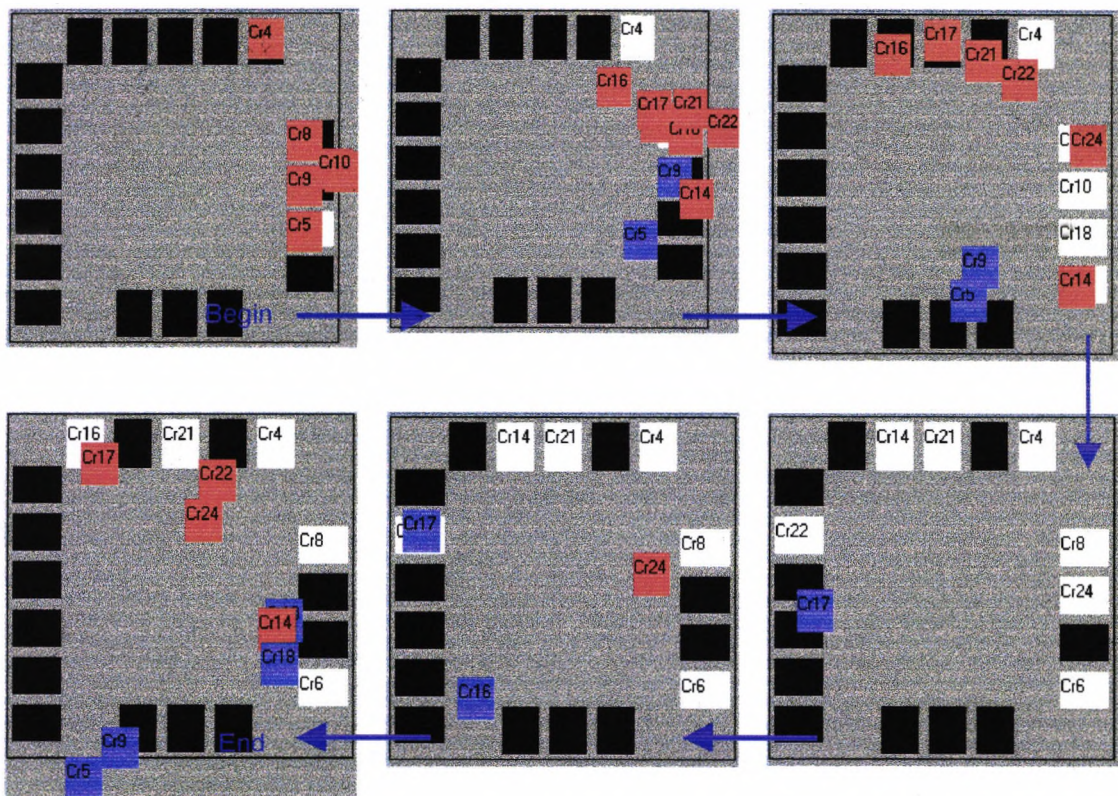


Figure 37 – Example of the simulation with experienced agents whose primary concern is to park as fast as possible. These cars have had twenty previous experiences of parking, with random configurations each time.

One flaw of this simulation is the lacking of the concept of queue of cars in this implementation. There is also no process for preventing cars from moving to the same position during the execution of one step. The only mechanism implemented in this concept is the one that prevents several cars from parking in the same parking space.

7.3.1.1 Testing results

Eight sets of car configurations were recorded and tested using three different experience profiles: no previous learning experience, 20 randomly generated learning experiences, 20 learning experiences with the same configuration.

The results of these running examples were measured in the following way:

- the number of target parking spaces that each agent has considered as its objective. This value shows the success of the agent's decision as its need to change objective shows a failure in the evaluation of the best parking space to move to;
- the number of steps taken inside the car park before parking. This value provides the evaluation of the agent's performance in terms of its preference parameter. In this simulation the agent wishes to park as soon as possible so the number of steps taken before parking evaluates the success of using learning to minimise that parameter.

In Figure 39 the tests results for running the eight configurations using the three learning profiles are presented in terms of the mean values described above for all the agents in the simulation. Figure 40 below shows individual values for the first configuration presented in Figure 39.

Test Number	No learning		20 random runs		20 equal runs	
	Nr of targets	Nr of steps	Nr of targets	Nr of steps	Nr of targets	Nr of steps
1	2.96	6.6	1.36	4.48	1.4	5
2	3.15	6.8	1.43	4.7	1.5	5.2
3	2.81	6.21	1.28	4.56	1.72	5.31
4	2.01	6.03	1.31	4.37	1.52	4.77
5	3.4	6.72	1.4	5.2	1.73	5.4
6	2.85	6.43	1.31	4.09	1.33	4.34
7	3.32	6.22	1.46	4.51	1.68	5.31
8	3.03	6.62	1.38	4.61	1.49	5.04
Mean	2.94	6.45	1.37	4.57	1.55	5.05
Mean improvement			53.07%	29.23%	47.43%	21.81%

Figure 39 - Comparison of three different experience profiles over 8 different simulation configurations. For each run the number of targets considered by each agent is taken into account, as well as the number of steps taken by the agent before parking.

Car agents 1 st configuration	No Learning		20 random runs		20 equal runs	
	Nr of targets	Nr of steps	Nr of targets	Nr of steps	Nr of targets	Nr of steps
1	2	8	2	6	1	3
2	4	9	1	1	1	2
3	2	4	1	2	1	2
4	1	3	1	1	1	2
5	1	3	1	5	2	8
6	3	4	1	7	1	2
7	3	9	2	4	1	5
8	4	6	1	4	5	11
9	2	4	1	2	1	2
10	4	7	1	2	2	9
11	6	13	1	8	2	8
12	2	8	1	4	2	5
13	1	3	1	2	1	3
14	3	8	1	7	1	7
15	2	3	1	2	1	2
16	2	3	1	3	1	3
17	4	10	1	3	1	3
18	3	7	2	4	1	2
19	4	11	4	11	1	7
20	3	9	1	4	1	4
21	2	3	2	9	2	8
22	5	7	1	2	2	12
23	2	3	1	8	1	3
24	4	10	2	5	1	6
25	5	10	2	6	1	6
Means	2.96	6.6	1.36	4.48	1.4	5

Figure 40 - Individual agent values for the 25 agents in the first tested configuration and whose results are presented in Figure 39. The values presented reflect the number of parking spaces considered as targets by the agents and the number of steps taken by each agent until they reach a parked state.

As can be seen from Figure 39 and Figure 40, the improvement between the number of targets considered by the same agent, before it has had some learning experience and after it has gone through 20 random configuration experiences is of a mean 53.07%. The average number of targets considered by each non-experienced agent is of 2.94 while the same agent after 20 random configuration experiences will consider 1.37 target spaces on average.

The number of steps taken by the agent to reach the final objective parking space also has an average decrease of 29.23% from non-experienced cars to experienced ones (cars which have learned through 20 random configuration experience runs). The average number of steps taken by a non-experienced car to reach its final parking space, over all the executed simulations is 6.45 while the experienced car will take a mean 4.57 number of steps.

When using the same car configuration over the 20 learning runs the rate of learning is slower, because the exploration factor is also lower. However, there is a considerable improvement in the behaviour of the agents with the number of objective parking spaces considered falling 47.43% while the number of steps taken by the agent before parking decreases 21,81%.

The results show that a reward function, which considers local spatial properties of the entities in the simulation as well as global ones, can be used to improve the performance of the car agents over a number of 20 simulation runs. They also show that the agents will learn more rapidly if they are given different configuration experiences every time they enter the car park, as opposed to executing the same configuration for each agent over and over again.

As can be seen in Figure 40, the individual performance of most of the car agents shows a general improvement in terms of the number of changes in the possible targets for parking. However, the number of steps taken to achieve a parking position rises in some cases. This is due to changes in the parking opportunities available to each agent as a result of the learning that has taken place. The agents

have become capable of making a more “wise” decision, from the utility information that has become available to them and the car park as a whole is working in a different fashion as the parking of each agent becomes more efficient. In any case, the sum of the number of steps taken by all the agents before parking is lower, which means that the global task of parking has become more efficient.

The use of the betterCar function (presented in section 7.2.2) has provided the reward function with a measure of the value of the position of other cars in the car park in terms of the current possible target parking space. This means that each agent is given information on the possible targets of other agents and reviews its decision using that information. The practical result of this is that when several agents enter the car park at the same time they all move in different directions, in order to try to reach a space that no other agent is trying to reach.

In conclusion, with learning, the agents’ behaviour becomes sub-optimal. Each agent tries to reach the best parking space that is reachable and available to them. Their performance may not be the best of all but it is explainable and it is a reaction to the possibilities they have at each step.

7.4 Emergent properties

Considering the theory of simulation presented in chapter 4, the entities that are part of the simulation constitute its active and passive elements (as presented in formula (4) of chapter 4): cars (active entities, that is agents), parking spaces, entrances and exits. These elements have been defined at the lower level of the simulation, which will be called L^1 .

The performance measures considered in this work and presented in the previous section have contributed to an evaluation of the simulation as a whole and therefore, constitute the basis for an external observer function at a higher level L^2 (O^2). Although these evaluation parameters have been defined inside the code of the simulation itself, they are considered external because they are not used in the system as feedback for later execution and they have been defined after the

creation of the simulation system with the aim of evaluating the performance of the agents and the effectiveness of learning.

The values presented in the previous section as parameters to measure the improvements in the simulations enabled the evaluation of several emergent properties whose values have improved through learning.

The most obvious emergent property is the level of use of the car park. As the car agents enter the simulation and start looking for parking spaces, the level of use of the car park (of the number of full parking spaces) is raised. This means that there will be fewer empty parking spaces available. Therefore, the state of each parking space and the position of car agents looking for a space to park, local properties at level L^1 , will create a new property P^2_1 at level L^2 which is the percentage of use of the car park at any one moment t , which obeys to formula (7) in chapter 4.

The aim of the learning procedure used in the simulation is to improve the performance of the agents and, globally, the effectiveness of parking in the car park. The utility values available to the agents reduce the total number of steps they take before parking. By using these values, an agent starts by selecting its favourite parking space and move towards it immediately. If this space becomes full, the agent's decision is reviewed for the current state of the car park. If another agent is better positioned to park on that space, its utility value is lowered.

The learning mechanism aims to help the agent find the current best available parking space. The agent decision must be taken in real time, with the available information, and on the existing conditions.

The number of targets considered by each agent before parking is a performance measure which enables the evaluation of the performance of a specific agent in different simulation runs with different experience profiles. It also enables the evaluation of the complete set of agents globally.

The improvement reflected through learning is evaluated in terms of the performance of the whole system, and enables the measure of another emergent

property, P^2_2 , which is the lowering of the mean number of target changes in a simulation run.

The other emergent property, P^2_3 , is the mean number of steps an agent has to take inside the car park before parking and which is improved through the knowledge that each agent acquires. This improvement could be reflected in a more rapid execution of the simulation, as the sum of steps taken by the agent is lowered as learning takes place. However, the evidence taken from the tests described in the previous section is inconclusive on that matter, due to the different states that the same simulation goes through with different experience profiles.

The final emergent property, P^2_4 , is a result of evaluating P^2_2 and P^2_3 in conjunction and is a qualitative property of the car park, resulting from the application of the learning experience by each agent. This is the effectiveness of parking in the car park, improved through learning the utility of moving towards a specific parking space. Each agent improves its performance through learning (as can be seen in Figure 40) as the conjunction of the global performance measures at each simulation run improves. This means that in the simulation, agents tend to globally (in the majority cases) make the right decision (a change in target becomes increasingly rare). Moreover, the global number of steps taken in each simulation is lower, which means that the agents tend to take less time (tend to take a lower number of steps) to park.

7.5 Discussion

As stated in chapter 1, the aim of this thesis is to analyse the potential of research in intelligent agents in geographic information science and to explore the use of simple learning techniques to improve the adaptability of spatial intelligent agents. This simulation was implemented in the context of the following objectives of the dissertation:

5. To study the potential of intelligent agents for the simulation of evolving spatial environments from the modelling of spatially-aware individuals;

6. To explore the use of simple learning techniques to improve the adaptability of spatial intelligent agents in simulation.

As stated in section 3 of chapter 3, an agent-based simulation has several advantages over the traditional numerical techniques of simulation. In this case, the use of global properties associated with local ones in the definition of the reward function (as described in 7.2.2 of this chapter), enables the agents behaviour to integrate the results of its interaction with other agents and the numerical values emerging from the global state of the simulation.

Car agents react to changes in the environment by reviewing their objective parking space and changing their current direction. The agents' decisions are the result of the evaluation of the utility of taking a specific action at the specific state of the environment. In this type of simulation, they are not event-driven, but that factor could easily be integrated. In fact, this methodology could be adapted for resource allocation in systems where there is no central control, e.g.:

- Equipment networks, with very fragile equipment at every location where technical engineers populate the network in order to fix malfunctioning equipment;
- Migration fluxes of populations looking for a specific characteristic in the landscape;
- Offer-demand systems where parking spaces are resources and agents are entities, which need these resources.

In fact, the economic factor of using a resource could be associated with the cost of moving towards the resource, and in this way, the system could be transformed into a market simulation.

One other possibility is its actual use in cars that often park at well known car parks. In Lisbon, Portugal, a car park database could include some popular car parks like Amoreiras Shopping Centre, Colombo Shopping Centre, Jumbo de

Alfragide. The use of different configuration parameters at different car parks could help build utility databases for each car. Sensors could endow the cars with car park use updated information like, how many cars are inside the car park, and how many parking spaces are empty. Of course, the realisation of this application would have to involve car manufacturers, car park administrations and the car owners themselves.

It is important to stress the truly adaptive nature of the simulation, through the step by step reviewing of the objective of each agent and the constant updating of the utilities of every action-state pair of the q-learning matrix. The performance of these utilities and of the spatial reward function are not addressed here, as there may be room for improvement through a statistical study of the importance of every parameter in the learning process.

There could also be some improvement in the Q-matrix, as the choice of states versus actions may not have been the optimal. Instead of deciding whether to go to parking space j when already going to parking space i the agent's state could be evaluated in a more concentrated fashion. The state at which the car agent is could be evaluated in terms of its spatial location (area inside the environment), instead of reflecting the importance of the parking space to which the agent is going. This means that the i index of the matrix could reflect simply a coarse spatial division of the environment, e.g. a quadrant.

It is also important to analyse the increase in the complexity of the implementation, with the corresponding increase in the number of agents or elements in the simulation.

7.5.1 Increase of complexity

The problem of using individual-based models in simulation is the exploding complexity of handling state transition rules of a growing number of individuals, as well as the possibility of having each one interacting with all the others.

In this methodology, a multi-reactive agent system (and therefore an individual-based simulation) where each agent has an associated q-learning procedure, the choices of actions are made from the utility of taking an action at a specific state. This value is kept in a matrix, and the process of accessing its values is not dependent on the number of agents or spaces in the simulation. The volume of required memory does increase with the number of parking spaces, but if this becomes difficult to handle, an indexed database table can be used to store the utility values. In this way, the memory problem is resolved and the access time is a numeric constant.

Because actions are taken thanks to the utility information held by the agent, and because every action is possible from every state, there is no need for transition rules. The complexity of the decision about which action to take is thus a function of the complexity of the q-learning algorithm. This algorithm is only dependent on the number of agents who are trying to find out if another car agent is in a better position to reach a certain parking space than the current one. However, this is a quick search because it depends on the numerical analysis of the agents' locations. Other than that, the time needed to run the algorithm does not vary.

The reason why the complexity of the algorithm is not an important issue is because there is no need for direct communication or interaction between the agents. The relative positions of other agents that may be interested in the same parking space is sufficient to achieve their main objective: to park.

7.5.2 Conclusions from the implementation

In chapter 4 some possibilities for the results of applying this methodology were put forward. The issues were:

- *the extension of learning methods to enable agents to learn spatial concepts and properties;*
- *the recognition of the importance of spatial properties in the adaptive nature of spatially-aware individuals in simulation environments;*

- *To creation of reinforcement structures that take spatial properties of an environment as measures of success.*

These three issues have been considered in the development of the spatial reward function. This function takes into account the location of the agent, of the parking space and of other agents in the car park. The decision taken by each agent is based on these spatial variables. The success of the learning procedure has been evaluated in section 7.3.1.1 (considering as the most important factors in this measure), the number of times the agent changes target parking space and the number of steps taken by the agent inside the car park before parking.

- *To enable agents to evaluate downward causal effects of emergent phenomena as global measures of success.*

The performance of the agents depends on the number of steps they take before parking. Their improved performance has the emergent property of making the overall use of the park become more effective and thus has a downward causal effect of enabling cars to also park more effectively, making their final choice of objective more rapidly (in the agent's local time). However, this realisation has not been considered in the development of the simulation and is not provided to each agent as feedback.

- *To enable agents to use the spatial/temporal consequences of taking an action as a step-by-step evaluation of their current performance.*

This possibility has become reality and has added to the effectiveness of the simulation in terms of fairness. The division of execution time slots among the agents is not fair, as time is given to each of them at random. If an agent stands still for a number of slots, its re-evaluation of the state of the environment may reflect many changes, including the choice of a new target parking space. Time and space are thus interconnected.

- *To provide agents with learning structures that will enable them to recognise a specific set of attribute values as a geographic (spatial) individual.*

The recognition of an individual as having a location and a direction has been successfully implemented. However, much more work needs to be done on implementing spatial semantic notions in simulations. This is a metric space and not a topologic one and improvement could be achieved from the implementation of concepts like neighbourhood and nearness.

Overall the results from this simulation suggest concrete potential for spatial agent reasoning and make a strong case for the effectiveness of this kind of simulation.

Ch. 8

Conclusions and further developments

The aims of this dissertation have been to explore the potential of research in geographic information science (GISc) in the following areas: spatial information handling and reasoning in current information systems; the design and behaviour of spatial assistants and spatial intelligent agents; and, the use of simple learning techniques to improve the adaptability of spatial intelligent agents.

From the above statement, three major objectives were drawn. These objectives guided the work described in this dissertation and the conclusions drawn in this chapter.

The first main objective was to review the state-of-the-art in GISc and spatial data handling with respect to:

- the identification of and access to special interest geographic information through the use of metadata structures, by a non-expert user with very specific needs;
- the integration of these processes in adaptable user interfaces for geographic information systems;
- at a different level of research, the development of systems that can provide the simulation of spatial processes resulting from the individual execution of spatial tasks.

The second objective was to study the potential for using spatial intelligent agents in the following areas, in the context of the research issues described above:

- specific-purpose geographic information location (identification of the searched information) and access;
- the improvement of geographic information systems interfaces;
- the simulation of evolving spatial environments from the modelling of spatially-aware individuals.

Finally, the third objective was the exploration of simple learning techniques to improve the adaptability of spatial intelligent agents in the context addressed by the work.

These objectives were fulfilled through the analysis of current research work in the relevant areas and the development of three prototypes that addressed the research questions involved.

In the introductory chapter of this thesis two claims are made:

- That the work in this thesis enables the falsification of the negative hypothesis about the potential of agent systems, which asserts that any

simple program can be considered an agent if evaluated according to agent-oriented concepts. It is argued that this general claim has been falsified specifically in the use of intelligent agents for spatial reasoning;

- That simple learning added to spatially-aware agents will add to the knowledge they hold of the domain they are embedded in and will enable them to improve their performance. These are stronger claims about the potential of spatial agents. These claims are supported by the tests results provided in chapters 6 and 7.

In this chapter, the results of this work are analysed according to the above claims and conclusions are drawn from the achievements and the limitations in the results. These conclusions address the larger questions in the dissertation and the specific technical questions.

8.1 Evaluation of research area from literature review

From the analysis of the state-of-the-art of GISc research, in the issues relevant to this dissertation, some key considerations emerged which are presented below:

8.1.1 Spatial Simulation

Existing work into spatial simulation is mostly done in the natural sciences, where researchers need to create tools to simulate the complex environments they study. The evaluation of the role played by space in these simulations has been limited.

Simulation has evolved from numerical techniques to location-based models to individual-based models. The line that separates these from reactive agents is very thin and, depending from the original training of the developers, it is possible to have very similar implementations with different research fields.

It was concluded that the field could gain from the development of agent-based simulations that would treat spatial properties in a systematic way, and would explicitly include these in a learning procedure. The use of simple learning algorithms in this type of simulation could help develop a methodology for creating

spatial simulations where spatial attributes could be part of the performance measure of the actors and of the simulated system in itself.

Simulations where different types of spatial entities existed in a spatial environment that changes dynamically and where local behaviour evolves at each step from changes in space and time were also very rare.

8.1.2 Spatial Decision Making

Research into the use of agents for spatial decision making uses interactive as well as social models of agency, where properties like collaboration and negotiation are of major importance. However, it was felt that decision making with spatial implications, where spatial location or spatially referenced information must be considered in the decision making process, could gain from the use of adaptive agents. The process of decision-making can only benefit from the evaluation of previous decision-making experiences at similar situations. This is where agent-based systems using learning techniques can be useful.

8.1.3 Interface agents for Geographic Information Systems (GIS)

Finally, the evaluation of literature concerning the development of interface agents for GIS was not very effective. Very little work has been done into finding ways of facilitating complicated GIS interfaces using agents. Despite this, the complex nature of GIS packages and the generality of their functionality is almost certainly in need of facilitation and personalisation techniques.

8.2 Specific findings of the research

The next stage in the research was to try and find answers for the questions that resulted from the study of the state-of-the-art in the area. The work presented includes the development of three prototypes, where the problems of this thesis were addressed. These prototypes were:

- One experiment to simplify the execution of the drafting and plotting tool in Smallworld GIS version 2. This first prototype did not use any intelligence or

learning to personalise the use of the GIS interface. Rather, it used a state transition model to assess the work that had already been done, and determine when and if the agent could finish the task for the user, with the information available;

- The second interface agent is an intelligent assistant that uses memory-based reasoning to help a user access and retrieve information on data he/she requires. The system listens to requests made by the user to a metadata map server, to determine his/her preferences on visualisation and retrieval of data. These requests are stored as a set of experiences and used later to determine the next best action to take at a specific state;
- Finally, the third prototype is a spatial simulation of a car park, where agents are cars with parking preferences. These cars use Q-learning to learn the best way to park (use previous parking experience to search for new places to park).

At this stage and thanks to the test results presented in chapters 6 and 7, it is possible to conclude that the first claim of the thesis, that this work enables the falsifying of the negative hypothesis in agents systems, specifically in the use of intelligent agents for spatial reasoning, is justified. The successful development of the prototypes, which are concerned with solving spatial problems and have been implemented using an agent-oriented methodology is sufficient to defend the claim. The prototypes work and the agents in the systems learn spatially in the terms defined here. Spatial learning is realised in the car park simulation through the comparison of the performance measures' values before and after learning has taken place in the simulation. In the case of SIFIA, learning is realised through the identification of patterns of use that enable the matching of the users' behaviour with group profiles defined specifically for the purpose of the tests.

The second claim made in the dissertation, that simple learning added to spatially-aware agents will add to the knowledge they hold of the domain they are embedded

in and will enable them to improve their performance, is also justified by the tests results.

The learning techniques used by the car agents enabled them to learn to improve their performance in such a way that this evolution in their behaviour can be identified in the emergent properties in the simulation, through an improvement in the effectiveness of parking in the car park.

In SIFIA, learning has provided the assistant with experience of the patterns of use of the system by different users. Improvement is achieved because users that show a specific preference for certain types of geographic information, can reach that information more rapidly. New information, identified as being part of those preferences, which becomes available between two uses of SIFIA is automatically delivered to the user on the next request that verifies the preference. If the behaviour of the user enables his/her identification as part of a group profile, information that may be relevant to that group is automatically suggested for retrieval. This latter result, enabled by the use of heuristic information in the profiles, enables the user to have access to information that he/she may not have realised as useful. In some cases, this suggestion may in fact be wrong, but the implementation of a feedback mechanism on the suggestions given by the assistant may solve this problem.

From the above, some further conclusions on SIFIA can be made:

- Interface agents can facilitate the use of geographic information system interfaces, through the simplification of tasks that are identified as being part of a specific pattern of use;
- The use of learning associated with interface agents can improve facilitation and it will also enable personalisation, if the evidence shown in patterns of use can be added to the specifications of applications, during runtime;
- This potential can be realised in searching and accessing spatial information when the personalisation of requests includes attributes like spatial regions,

geographic units, and thematic data. Access to a database of spatial information using a specific user's profile will enable the automatic delivery of new information (as it becomes available) to satisfy that profile, the more rapid access to data that is part of a pattern of use, and the automatic suggestion of delivery of thematic data relevant to a group profile.

The above conclusions have led to the realisation of the potential in future implementations described below:

- Management of previous experiences can handle preferences that evolve with time, by giving a stronger weight to more recent requests;
- Memories resulting from the use of applications by people in different areas of activity can be used to create initial professional profiles. This means that the memory created by someone working in, for example, local planning, can be used to form an initial profile of preferences for local planners. This profile can then be updated by each specific planner, with later use of the application;
- Some changes into the agent can also enable it to use its memory to teach and aid users in performing specific tasks in the application;
- Agent-based approaches to interface design can help generate the 'prototypical work situations' that Rasmussen et al. (1994) have referred to. In fact, the evidence in the tests results of chapter 6 have shown some mistakes in the matching of users and profiles which could be fed back to the system and enable the profiles to evolve from there.

The above suggestions identify the potential of this type of interface assistant, used in the context of an information system. However, this work also has its defaults and limitations:

- The definition of a metric for the measurement of equality in spatial attributes is very important. The application of results from work into fuzzy regions and queries may prove very useful (Goodchild, 1998);

- The process for the identification of the next best action to take by the user is executed at each request for advice. As the memory grows, this process may become very slow. The solution may be to create rules from the most accurate situations presented by the memory (Stanfill, 1988; Kozierok and Maes, 1993);
- The solution of the above problem may depend on the evolution of the online mapping tools available. In fact, this work has been implemented using development tools which were not the best fit for agent implementation but which enabled the easy manipulation of geographic information. The creation of rules from this memory would be quite difficult to implement using these tools;
- The modularity of the implementation has suffered from the fact that, at the time of implementation, java applets embedded in "standard" web browsers could not write on the local host disk and more research needs to be done for the best implementation of this type of problem, specifically with the use of cookies;
- A fundamental problem in using learning agents as assistants is that agents learn what the user does. If the user makes mistakes, that is what the agent will learn. This is why the use of initial profiles generated by an expert user can be of help in teaching new users and preventing them from making mistakes. The problem that this raises is the feeling of losing control over his/her work that the user may experience. This is why the balance between initial knowledge versus the learning of the agent must be carefully addressed.

Further conclusions can also be drawn from the car park simulation. The car park prototype was implemented with the aim of studying the potential of simulating evolving spatial environments from the modelling of spatially aware individuals. From this implementation and from the testing results, it was possible to conclude:

- Agent-based simulation provides a natural way to study the emergent properties of a community from the behaviour of local actors. It enables the

association of local and global characteristics of the simulation, which may determine its outcome and emergent properties;

- This type of implementation is truly adaptive, enabling the dynamic review, at each step, of the agent's decisions, determined by the spatial-temporal changes in the environment;
- The simulation is implemented as the parallel execution of several agents. There is no sequential control over the execution and therefore the order by which the agents act is decided at random by the computer;
- Another important factor is that this type of simulation can execute some reasoning locally, make decisions and take actions in real time, and still learn something from the experience;
- The performance of the agent is assessed by a reward function which includes spatial properties in its calculus;
- Each agent (individual) in the simulation can have a different configuration in execution, or simply a different behaviour;
- The learning technique used enables the developer to change the reward function, and thus the evaluation of the performance of the agent, according to the attributes that are more important or relevant.

Some limitations were also identified in this type of implementation:

- The definition of the reward function may be improved through some statistical study of the importance of every parameter in the learning process;
- The performance of the simulation is good because decisions are taken using the measurement of the utility of taking a specific action at a specific state. However, this is only possible because there is very little communication between the agents themselves. If it was necessary for each agent to communicate with every other one, performance would suffer. Some work must be put into studying this problem.

The work developed in the context of this dissertation led to the considerations presented above. From these findings it was possible to draw some major conclusions about the initial research questions. After this, it will be possible to draw further objectives for development.

8.3 Key Conclusions

Worboys (1995) argues that research is needed on the applications of newer computational paradigms, interface metaphors and approaches to metadata handling. This dissertation has contributed to these three issues, although the presented implementations can be considered quite simple.

The use of interface assistants (IA) can help lower the complexity of handling spatial (geographic) information and help generate “prototypical work situations”. The search and retrieval of data can be simplified by the use of agents in applications like SIFIA (chapter 6), which are becoming available in many existing spatial data infrastructures (Rodrigues, 1998). Moreover, the execution of spatial tasks on the data may become simpler and more direct through the use of IA (chapter 5).

Wooldridge (1997) defines the intelligence requirement of agents in the following way: “the only intelligence requirement we generally make of our agents is that they can make an acceptable decision about what action to perform next in their environment, in time for this decision to be useful”. This is what fundamentally distinguishes agents from other intelligent systems: they may not always make the right decisions but these are acceptable according to the information they hold and should enable them to act in real-time or inside a useful time period.

Wooldridge provides us with one of the fundamental reasons for using agents to reason spatially in dynamical systems (although this is a good reason to use agents in other areas, too). This property provides a simulation of a changing environment with actors that can use spatial attributes to make real-time decisions and that can also learn in the process. Moreover, agent-based simulations can involve

heterogeneous agents learning about making decisions in an environment (which can be populated with several types of passive entities) in a truly parallel fashion. Finally, actions taken can be reviewed and changed at any moment, as the characteristics of the environment evolve and as other agents present themselves in the position of outrunning the current one. The use of learning in a spatial context can improve the performance of agents individually and the effectiveness of executing a global spatial task becomes an emergent property of the simulation.

The use of interface assistants may facilitate the use of spatial information systems interfaces by personalising or restricting general interfaces to the user's necessities. This type of agent may also be adapted to prevent the user from carrying out operations that should not be used at a certain stage, or to restrict the user's choices according to his/her profile.

The spatio-temporal issues addressed in the use of agents are related to the way in which an agent-based system evolves depending on the changes in the spatial environment through time (chapter 7). The continuously parallel nature of this type of system may involve situations where the objective of a certain agent may become impossible because another one has reached it first. Therefore, the reviewing of objectives at each relevant period of time is not only useful but necessary.

Finally, agents can also be used in simulations that use GIS data as input or that generate output for GIS. It should be noted that nothing related to this has been attempted in this dissertation. However, the opportunity exists and is already in the agenda of researchers in this area (Guerrin *et al*, 1998). Another useful possibility would be to use the event driven facilities associated with agents to trigger the execution of spatial models in the base spatial environment.

8.4 Further developments of the research

Several directions for research in the area of spatial intelligent agents have already been suggested in the previous section. Finally, further suggestions about how the results of this dissertation could be used in the future can be made.

In the area of interface assistants, the next step would be to implement SIFIA in a production system. This system could either be a national spatial data infrastructure or a geographic digital library.

It would be necessary to develop metrics for the comparison of spatial attributes and to change the agent component of the system in order to make it more modular. This would involve the evaluation of new tools for online mapping, new versions of existing ones and the study of the possibility using java security features to enable an applet to use locally resident memory (or any other persistent memory structures) .

Another area of interest would be the creation and manipulation of spatial ontologies to facilitate the definition of spatial entities and the transparent sharing of these definitions.

In the study of spatial reasoning in dynamic systems, it would be interesting to apply the developed methodology in other examples, specifically those with a higher level of interaction between agents. This will probably involve the study of current research into negotiation, coordination and collaboration among agents, which has not been an issue in this dissertation. The issue of performance would be of major importance in this context.

For these new examples, it would be important to study other learning techniques, as opposed to the results presented by reinforcement learning.

Finally, the association of an agent-based simulation with raster-based spatial models (probably cellular automata), integrated in an event-driven fashion (as described in the previous section), would be another potentially interesting project. The execution of spatial models triggered by events on the agents' state or on the environment's attribute values could generate new information which could then be used as input into the environment, and lead to further evolution of the system.

References and Bibliography

- Aangeenbrug, R. T. (1982), The future of Geographical Information Systems, *Computer Graphics News*, 2:2:4.
- Aangeenbrug, R. T. (1991), A critique of GIS, In D. J. Maguire, M. F. Goodchild and D. W. Rhind (Eds.) *Geographical Information Systems: Principles and applications*, Longman Scientific & Technical, Essex, England, 101-107, 1991.
- ADL (1998), *Alexandria Digital Library*, University of California, Santa Barbara, CA, USA, <<http://alexandria.sdc.ucsb.edu/>>, May 1998.
- ADL (1999), *Alexandria Digital Earth Prototype Project*, University of California, Santa Barbara, CA, USA, <<http://www.alexandria.ucsb.edu/adept/10.html>>, 1999.
- ADL (1999a), *The Alexandria Digital Earth Modeling System (ADEPT): Towards a distributed digital model of the earth in support of learning*, Project Proposal, University of California, Santa Barbara, CA, USA, <<http://www.alexandria.ucsb.edu/adept>>, 1999.
- Adler, R.M. and Cottman, B.H. (1989), A Development Framework for Distributed Artificial Intelligence, *Proceedings of the Fifth Conference on Artificial Intelligence Applications*, IEEE, 1989.
- Agha, G. (1986), *ACTORS: a model of concurrent computation in distributed systems*, MIT Press, Cambridge, MA, USA, 1986.
- Agre, P. and Chapman, D. (1987), *PENGI: An implementation of a theory of activity*, Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87), 268-272, Seattle, WA, USA, 1987.

- Ambros-Ingerson, J. and Steel, S. (1988), *Integrating planning, execution and monitoring*, Proceedings of the Seventh National Conference on Artificial Intelligence (AAAI-88), 83-88, St. Paul, MN, USA, 1988.
- Baeijs, C., Demazeau, Y. and Alvares, L. (1996), SIGMA: Application of multi-agent systems to cartographic generalization, In *Proceedings of the Seventh European Workshop on Modelling Autonomous Agents in a Multi-Agent World*, Eindhoven, January, 1996.
- Bond, A. and Gasser, L. (Eds.)(1988), *Readings in Distributed Artificial Intelligence*, Morgan Kaufmann Publishers, California, USA, 1988.
- Bratman, M. E., Israel, D. J. and Pollack, M. E. (1988), *Plans and resource-bounded practical reasoning*, Computational Intelligence, 4:349-355, 1988.
- Brooks, R. A. (1986), *A robust layered control system for a mobile robot*, IEEE Journal of Robotics and Automation, 2:1:14-23, 1986.
- Brooks, R. A. (1990), Elephants don't play chess, In P. Maes (Ed.) *Designing Autonomous Agents*, MIT Press, Cambridge, MA, USA, 1990.
- Campos, D., Naumov, A. Y. and Shapiro S. C. (1996), Building an interface agent for ARC/INFO, *1996 ESRI User Conference Proceedings*, Palm Springs, California, USA, May 1996.
- Carter, J. R. (1989), On defining the geographic information system, In W. J. Ripple (Ed.) *Fundamentals of Geographics Information Systems: a compendium*, ASPRS/ACSM, Falls Church Virginia, 3-7, 1989.
- Castelfranchi, C. (1995), Commitments: From individual intentions to groups and organizations, *Proceedings of the International Conference on Multiagent Systems*, 41-48, 1995.
- Chapman, D. (1987), *Planning for conjunctive goals*, Artificial Intelligence, 32:333-378, 1987.

- Chapman, G. P. (1977), *Human and Environmental Systems: A Geographer's Appraisal*, Academic Press, London, UK, 1977.
- Chin, D. (1991), Intelligent interfaces as agents, In J. Sullivan and S. Tyler (Eds.) *Intelligent user interfaces*, ACM Press, New York, NY, USA, 1991.
- Chrisman, N. R. (1975), Topological information systems for geographic representations, *Proceedings of The Second International Symposium on Computer-Assisted Cartography (Auto-Carto-2)*, Falls Church: ASPRS/ACSM, 366-351, 1975.
- Chrisman, N. R. (1978), Concepts of space as a guide to cartographic data structures, In G. Dutton (Ed.) *Proceedings of the First International Advanced Study Symposium on Topological Data Structures for Geographic Information Systems*, Harvard Laboratory for Computer Graphics and Spatial Analysis, Cambridge, MA, USA, 1978.
- CISE (1999), *Digital Libraries Initiative: A community of researchers and agencies working together to bring the world's knowledge to your desktop*, Special Projects Program, Information and Intelligent Systems Division, Directorate for Computer and Information Science Engineering, National Science Foundation, <<http://www.dli2.nsf.gov/>>, June 1999.
- CISE (1999a), *Digital Libraries Initiative: Funded Projects*, Special Projects Program, Information and Intelligent Systems Division, Directorate for Computer and Information Science Engineering, National Science Foundation, <<http://www.dli2.nsf.gov/projects.html>>, July 1999.
- Clarke, K. C. (1990), *Analytic and Computer Cartography*, Prentice Hall, Englewood Cliffs, New Jersey, 1990.
- Cohen, P. R., Greenberg, M. L., Hart, D. M., Howe and A. E. (1989) , Trial by Fire: Understanding the design requirements for agents in complex environments, *AI Magazine*, 10:3:32-48, 1989.

- Collins, R. J. and Jefferson, D. R. (1991), *Ant Farm: Toward Simulated Evolution*, In C. Langton (Ed.) *Artificial Life*, Addison-Wesley, MA, USA, 1991.
- Davis, R. (1980), Report on the Workshop on Distributed AI, *SIGART Newsletter*, 73:42-52, October 1980.
- Davis, R. (1982), Report on the Second Workshop on Distributed AI, *SIGART Newsletter*, 80:13-23, April 1982.
- Davis, R. and Smith, R. G. (1983), Negotiation as a metaphor for distributed problem solving, *Artificial Intelligence*, 20:63-109, 1983.
- DeAngelis, D. L. and Gross, L. J. (1992) (Eds.), *Individual-based models and approaches in Ecology: Populations, Communities and Ecosystems*, Chapman & Hall, New York, USA, 1992.
- Deneubourg, J.-L. and Goss, S. (1989), Collective patterns and decision making, In *Ethology, Ecology and Evolution*, 1: 295-311, 1989.
- Deneubourg, J.-L., Goss, S., Pasteels, J. M., Fresneau, D. and Lachaud, J.-P. (1987), Self-organisation mechanisms in ant societies (II): learning in foraging and division of labour, In J. M. Pasteels and J.-L. Deneubourg (Eds.) *From Individual Characteristics to Collective Organisation in Social Insects, Experientia Supplementum*, Birkhäuser, Bâle, 54:177-196, 1987.
- Dent, L., Boticario, J., McDermott, J., Mitchell, T. and Zabowski, D. (1992), A personal learning apprentice, *Proceedings of AAAI'92*, AAAI Press/The MIT Press, 96-103, 1992.
- Draper, S. W. (1993), The notion of Task in HCI, In S. Ashlund, K. Mullet, A. Henderson, E. Hollnagel and T. White (Eds.) *Interchi'93 Adjunct Proceedings*, ACM, USA, 207-208, 1993.
- Egenhofer, M. J., Glasgow, J., Günther, O., Herring, J. R. and Peuquet, D. (1999), Progress in Computational Methods for Representing Geographic

Concepts, *International Journal of Geographic Information Science: Special Issue on the Varenius Project*, to appear, 1999.

ESMI 1999), *the European Spatial Metadata Infrastructure*, <<http://www.geodan.nl/esmidev/index.html/>>, 1999.

ESRI, Inc. (1996), *Building Applications with MapObjects*, Environmental Systems Research Institute, Inc., Redlands, CA, USA, 1996.

ESRI, Inc. (1996a), *Getting Started with MapObjects Internet Map Server*, Environmental Systems Research Institute, Inc., Redlands, CA, USA, 1996.

ESRI, Inc. (1996b), *MapObjects Programmer's Reference*, Environmental Systems Research Institute, Inc., Redlands, CA, USA, 1996.

Etzioni, O., Lesh, N. and Segal, R. (1994), Building softbots for UNIX, In O. Etzioni (Ed.) *Software Agents – Papers from the 1994 Spring Symposium (Technical Report SS-94-03)*, AAAI Press, 9-16, 1994.

Etzioni, O. and Weld, D. (1994), A softbot-based interface to the Internet, In *Communications of the ACM*, 37:7:72-76, July 1994.

Fehling, M. and Erman, L. (1983), Report on the Third Annual Workshop on Distributed Artificial Intelligence, *SIGART Newsletter*, 84:3-12, April 1983.

Ferber, J. (1994), Simulating with Reactive Agents, In J. Stender, and E. Hillebrand (Eds.), *Many Agent Simulation and Artificial Life*, IOS Press, Amsterdam, The Netherlands, 1994.

Ferber, J. (1994a), Reactive Multi-Agent Systems: Principles and applications, In N. Jennings (Ed.) *Fundamentals of Distributed Artificial Intelligence*, North-Holland, Amsterdam, The Netherlands, 1994.

Ferguson, I. A. (1992), *TouringMachines: An Architecture for Dynamic, Rational, Mobile Agents*, PhD Thesis, Clare Hall, University of Cambridge, UK, November 1992.

- Ferrand, N. (1995), Multi-Reactive-Agents Paradigm for Spatial Modelling, *Proceedings of the GISDATA Workshop on Spatial Modelling*, Stockholm, June 1995.
- Ferrand, N. (1996), Modelling and Supporting Multi-Actor Spatial Planning Using Multi-Agents Systems, *Proceedings of the Third NCGIA Conference on GIS and Environmental Modelling*, Santa Fe, USA, January 1996.
- FGDC (1999), *The Federal Geographic Data Committee*, <<http://www.fgdc.gov/index.html>>, 1999.
- Fisher, P. F. and Langford, M. (1996), Modelling sensitivity to accuracy in classified imagery: a study of areal interpolation by dasymetric mapping, In *Professional Geographer*, 48:3:299-309, 1996.
- Fisher, P. F. and Wood, J. (1999), What is a mountain? or The Englishman who went up a Boolean Geographical concept but realised it was fuzzy, *Geography*, 1999.
- Foner, L. (1997), Yenta: A Multi-Agent, Referral-Based Matchmaking System, *Proceedings of the First International Conference on Autonomous Agents (Agents'97)*, Marina del Rey, CA, USA, February, 1997.
- Fonseca, A. (1998), *The use of Multimedia Spatial Data Handling for Environmental Impact Assessment*, Ph. D. Dissertation, Faculty of Science and Technology, New University of Lisbon, Lisbon, Portugal, 1998.
- Franklin, S. (1995), *Artificial Minds*, MIT Press, Cambridge, MA, USA, 1995.
- Franklin, S. and Graesser, A. (1996), Is it an Agent, or just a Program?: Taxonomy for Autonomous Agents, *Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages*, Springer-Verlag, Germany, 1996.
- Gaede, V. and Günther, O. (1998), Multidimensional Access methods, *ACM Computing Surveys*, 30, 1998.

- Genesereth, M. R. and Ketchpel, S. P. (1994), Software agents, In *Communications of the ACM*, 37:7:48-53, July 1994.
- Georgeff, M. P. and Lansky, A. L. (1987), Reactive reasoning and planning, *Proceedings of the Sixth National Conference on Artificial Intelligence (AAAI-87)*, 677-682, Seattle, WA, USA, 1987.
- Georgeff, M. P. and Ingrand, F. F. (1989), Decision-Making in an embedded reasoning system, *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence (IJCAI-89)*, 972-978, Detroit, MI, USA, 1989.
- Gilbert, N. (1995), Simulation: an emergent perspective, *Conference on New Technologies in the Social Sciences*, Bournemouth, UK, <<http://www.soc.surrey.ac.uk/research/simsoc/tutorial.html>>, October 27-29, 1995.
- Goodchild, M. F. (1992), Geographical information science, *International Journal of Geographical Information Systems*, 6:1:31-45, 1992.
- Goodchild, M. F. (1998), Fuzzy Spatial Queries in Digital Spatial Data Libraries, *Proceeding of the IEEE International Conference on Fuzzy Systems*, Anchorage, USA, May 4-9, 1998.
- Goodchild, M. F. (1998a), The Geolibrary, In S. Carver (Ed.) *Innovations in GIS 5*, Taylor & Francis, London, UK, 1998.
- Goodchild, M. F., Egenhofer, M. J., Kemp, K. K., Mark, D. M. and Sheppard, E. (1999), Whither Geographic Information Science? The Varenus Project, *International Journal of Geographic Information Science: Special Issue on the Varenus Project*, to appear, 1999.
- Gouveia, C. (1998), National Geographic Information Infrastructures: The Portuguese Experience, *Proceedings of the Workshop on Challenges and*

Future Developments of GI Infrastructures: The Portuguese Experience, GISPlanet'98, Lisbon, Portugal, 11-15, September 1998.

Gruber, T. (1993), Toward Principles for the Design of Ontologies Used for Knowledge Sharing, *Presented at the Padua workshop on Formal Ontology*, March, 1993.

Guerrin, F., Courdier, R., Calderoni, S., Paillat, J.-M., Soulié, J.-C. and Vally, J.-D. (1998), Biomass: un modèle multi-agents pour aider à la gestion négociée d'effluents d'élevage, *Colloque SMAGET, Modèles et Systèmes Multi-Agents pour la Gestion de l'Environnement et des Territoires*, Cemagref, Clermont-Ferrand, France, 6-8 October, 1998.

Haddadi, A. (1994), A hybrid architecture for multi-agent systems, In S. M. Deen (Ed.) *Proceedings of the 1993 Workshop on Cooperating Knowledge Based Systems (CKBS-93)*, 13-26, DAKE Centre, University of Keele, UK, 1994.

Haddadi, A. (1995), Towards a pragmatic theory of interactions, *Proceedings of the International Conference on Multiagent Systems*, AAAI Press, 133-139, 1995.

Haddawy, P. (1996), Believing change and changing belief, *IEEE Transaction on Systems, Man and Cybernetics: Special Issue on Higher-Order Uncertainty*, 26:5, 1996.

Harvey, D. W. (1969), *Explanation in Geography*, Edward Arnold, London, UK, 1969.

Hayes-Roth, B. (1995), An Architecture for Adaptive Intelligent Systems, *Artificial Intelligence: Special Issue on Agents and Interactivity*, 72:329-365, 1995.

Hayes-Roth, F., Waterman, D. A., Lenat, D. B. (1983) (Eds.), *Building Expert Systems*, Addison-Wesley, Reading, MA, USA, 1983.

Hern, L.E. (1988), On distributed artificial intelligence, *The Knowledge Engineering Review*, 3:1:21-57, 1988.

- Hogeweg, P. and Hesper, B. (1985), SocioInformatic Processes: MIRROR Modeling Methodology, *Journal of Theoretical Biology*, 113:311-330, 1985.
- Hewitt, C. E. (1985), The Challenge of Open Systems, *Byte*, 10:4:223-242, April 1985.
- Hewitt, C. E. (1986), Offices are Open Systems, *ACM Transactions on Office Information Systems*, 4:3:271-287, 1986.
- Holte, R. and Drummond, C. (1994), A Learning Apprentice For Browsing, *Proceedings of AAAI Spring Symposium on Software Agents*, 1994.
- Horvitz, E. and Rutledge, G. (1991), Time-dependent utility and action under certainty, *Proceedings of the 7th Conference on Uncertainty in Artificial Intelligence*, 151-158, 1991.
- Huhns, M.N. (1987), *Distributed Artificial Intelligence*, Morgan Kaufmann, Los Altos, CA, 1987.
- Huhns, M. N. and Bridgeland, D. M. (1991), Multi-Agent Truth Maintenance, *IEEE Transactions on Systems, Man and Cybernetics*, 21:6:1437-1445, 1991.
- Huhns, M. N. and Singh, M. P. (1998), Agents and Multiagent Systems: Themes, Approaches, and Challenges, In M. N. Huhns and M.P. Singh (Eds.) *Readings in Agents*, Morgan Kaufmann Publishers, Inc, San Francisco, California, 1-23, 1998.
- Huhns, M. N. and Singh, M. P. (1998a) (Eds.), *Readings in Agents*, Morgan Kaufmann Publishers, Inc, San Francisco, California, 1998.
- Jennings, N. R. (1992), On being responsible, In E. Werner and Y. Demazeau (Eds.) *Decentralized AI 3 - Proceedings of the Third European Workshop on Modelling Autonomous Agents in Multi-Agent Worlds (MAAMAW-91)*, 32-42, Rotterdam, The Netherlands, 1992.

- Jennings, N. R. (1993), Specification and implementation of a belief desire joint-intention architecture for collaborative problem solving, *Journal of Intelligent and Cooperative Information Systems*, 2:3:289-318, 1993.
- Jennings, N.R. (1995), Controlling cooperative problem solving in industrial multi-agent systems using joint intentions, *Artificial Intelligence*, 74:2, 1995.
- Jennings, N. R., Corera, J., Laresgoiti, I., Mamdani, E. H., Perriolat, F., Skarek, P. and Varga, L. Z. (1996), Using ARCHON to develop real-world DAI applications for electricity transportation management and particle accelerator control, *IEEE Expert*, 11:6:64-70, 1996.
- Kaelbling, L. P. and Rosenchein, S. J. (1990), Action and planning in embedded agents, In P. Maes (Ed.) *Designing Autonomous Agents*, 35-48, MIT Press, Cambridge, MA, USA, 1990.
- Kay, A. (1990), User Interface: A personal view, In B. Laurel (Ed.) *The Art of Human-Computer Interface Design*, Addison-Wesley, Reading, Massachusetts, USA, 1990.
- Kautz, H., Selman, B., Coen, M., Ketchpel, S. and Ramming C. (1994), An experiment in the design of software agents, *Proceedings of the National Conference on Artificial Intelligence*, 438-443, 1994.
- Kewley, R. H. Jr. and Embrechts, M. J. (1998), Fuzzy-Genetic Decision Optimization for Positioning of Military Combat Units, *Proceedings of the IEEE World Congress on Computational Intelligence*, <<http://www.rpi.edu/~kewler/fgdo/fgdo.html>>, 1998.
- Kozierok, R. (1993), *A learning approach to knowledge acquisition for intelligent interface agents*, MSc Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA, May 1993.

- Kozierok, R. and Maes, P. (1993), A learning interface agent for scheduling meetings, *Proceedings of the ACM-SIGCHI International Workshop on Intelligent user interfaces*, Florida, USA, January 1993.
- Lai, K.-Y., Malone, T. W., and Yu, K.-C. (1988), Object Lens: A "spreadsheet" for cooperative work, *ACM Transactions on Office Information Systems*, 6:4:332-353, October 1988.
- Langton, C. G. (1989), Artificial Life, In C.G. Langton (Ed) *Artificial Life: The Proceedings of an Interdisciplinary Workshop on The Synthesis and Simulation of Living Systems*, 1-47, Santa Fe Institute Studies in the Sciences of Complexity, Addison-Wesley Publishing Company, Redwood City, CA, USA, 1989.
- Lashkari, Y., Metral, M. and Maes, P. (1998), Collaborative interface agents, In M. Huhns and M. Singh (Eds.) *Readings in Agents*, Morgan Kaufmann Publishers Inc., San Francisco, California, USA, 1998.
- Laurini, R. and Thompson, D. (1992), *Fundamental of Spatial Information Systems*, Academic Press, London, UK, 1992.
- Lesser, V.R. and Corkill, D.D. (1988), Functionally Accurate, Cooperative Distributed Systems, In A. Bond, L., Gasser (Eds.), *Readings in Distributed Artificial Intelligence*, Morgan Kaufman Publishers, California, USA, 1988.
- Lesser, V. R. and Corkill, D. D. (1987), Distributed Problem Solving, In S. C. Shapiro (Ed.) *Encyclopedia of Artificial Intelligence*, 245-251, John Wiley and Sons, New York, USA, 1987.
- Lieberman, H. (1998), Integrating User Interface Agents with Conventional Applications, *Knowledge-Based Systems Journal*, Elsevier, 11:1:15-24, September 1998.

- Lieberman, H. (1994), Powers of ten thousand: navigating in large information spaces, *Proceedings of the conference on User Interface Software*, Marina Del Rey, California, November 1994.
- Lieberman, H. (1994), A user interface for knowledge acquisition from video, In *Proceedings of the Conference for American Association for Artificial Intelligence (AAAI'94)*, Seattle, USA, 31 July - 4 August 1994.
- Lindenmayer, A. and Prusinkiewicz, P. (1989), Developmental Models of Multicellular Organisms: A Computer Graphics Perspective, In C.G. Langton (Ed.) *Artificial Life: The Proceedings of an Interdisciplinary Workshop on The Synthesis and Simulation of Living Systems*, 221-250, Santa Fe Institute Studies in the Sciences of Complexity, Addison-Wesley Publishing Company, Redwood City, CA, USA, 1989.
- Linsey, T. K. and Raper, J. F. (1993), A task-oriented hypertext GIS interface, *International Journal of Geographical Information Systems*, 7:5:435-52, 1993.
- Lomnicki, A. (1992), Population Ecology from the Individual Perspective, In D. L. DeAngelis and L. J. Gross (Eds.) *Individual-based models and approaches in Ecology: Populations, Communities and Ecosystems*, 3-17, Chapman & Hall, New York, USA, 1992.
- Maguire, D. J. (1991), An overview and definition of GIS, In D. J. Maguire, M. F. Goodchild and D. W. Rhind (Eds.) *Geographical Information Systems: Principles and applications*, Longman Scientific & Technical, Essex, England, 101-107, 1991.
- Maes, P. (1990), Situated agents can have goals, In P. Maes (Ed.) *Designing autonomous agents*, 49-70, MIT Press, Cambridge, MA, USA, 1990.
- Maes, P. (1994), Agents that Reduce Work and Information Overload, *Communications of the ACM*, 37:7:31-40, 1994.

- Maes, P. (1994a), Modeling Adaptive Autonomous Agents, *Artificial Life*, 1:135-162, 1994.
- Maes, P. (1995), Artificial Life Meets Entertainment: Life like Autonomous Agents, *Communications of the ACM*, 38:11:108-114, 1995.
- Mark, D. M., Freksa, C., Hirtle, S. C., Lloyd, R. and Tversky, B. (1999), Cognitive Models of Geographic Space, *International Journal of Geographic Information Science: Special Issue on the Varenius Project*, to appear, 1999.
- Maruichi, T., Uchiki, T. and Tokoro, M. (1987), Behaviour Simulation based on Knowledge Objects, In *Proceedings of ECOOP'87*, 1987.
- MEGRIN (1999), MEGRIN Organisation Web Page, <<http://www.megrin.org>>, 1999.
- Metral, M (1993), *Design of a generic learning interface agent*, Bsc Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA, May 1993.
- Minar, N., Burkart, R., Langton, C. and Askenazi, M. (1996), *The Swarm Simulation System: A Toolkit for Building Multi-Agent Simulations*, <<http://www.santafe.edu/projects/swarm/overview.ps>>, June 1996.
- Minsky, M.(1985), *The Society of Mind*, Simon & Schuster, USA, 1985.
- MIT, 1999, *MIT Media Lab: Software Agents Group: Projects*, <<http://agents.www.media.mit.edu/groups/agents/projects/>>, 1999.
- Mitchell, T., Caruana, R., Freitag, D., McDermott, J. and Zabowski, D. (1994), Experience with a learning personal assistant, *Communications of the ACM*, 37:7:81-91, July 1994.
- Mitchell, T. M., Mahadevan, S. and Steinberg, L. (1985), LEAP: A learning apprentice for VLSI design, *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, 1985.

MSC (1999), *Distributed Geolibraries Spatial Information Resources: Summary of a workshop*, Mapping Science Committee, Board on Earth Sciences and Resources, Commission on Geosciences, Environment and Resources, National Research Council, The National Academy of Sciences, USA, <<http://www.nap.edu/html/geolibraries/>>, 1999.

MSC (1999a), *Distributed Geolibraries Spatial Information Resources: Summary of a workshop, Appendix D: Example Prototypes*, Mapping Science Committee, Board on Earth Sciences and Resources, Commission on Geosciences, Environment and Resources, National Research Council, The National Academy of Sciences, USA, <http://www.nap.edu/html/geolibraries/app_d.html>, 1999.

Müller, J. P., Pischel, M. and Thiel, M. (1995), Modelling reactive behaviour in vertically layered agent architectures, In M. Wooldridge, N. R. Jennings (Eds.), In *Intelligent Agents – Proceedings of the 1994 Workshop on Agent Theories, Architectures, and Languages*, 1995.

Nakauchi, Y., Okada, T and Anzai, Y. (1991), Groupware that learns, *Proceedings of the IEEE Pacific Rim Communications, Computers and Signal Processing Conference*, IEEE, New York, USA, 1991.

NCGIA (1995), *Advancing Geographic Information Science*, Project Proposal to the National Science Foundation, National Centre for Geographic Information and Analysis, University of California, Santa Barbara, USA, <<http://ncgia.ucsb.edu/secure/secB.html>>, November 1995.

Newell, A. and Simon, H. A. (1976), Computer Science as empirical enquiry, *Communications of the ACM*, 19:113-126, 1976.

Nunes, J. (1991), Geographic space as a set of concrete geographical entities, In D. M. Mark and A. U. Frank (Eds.) *Cognitive and Linguistic Aspects of Geographic Space*, 63: 9-33, NATO ASI Series, Series D-Behavioural and

- Social Sciences, Kluwer Academic Publishers, Dordrecht, The Netherlands, 1991.
- Nwana, H.S. (1996), Software Agents: An Overview, *Knowledge Engineering Review*, 11:3:205-244, 1996.
- O'Brien, P. and Wiegand, M. (1996), Agents of Change in Business Process Management, *British Telecommunications Journal*, 14:4, 1996.
- Pollack, M. E. and Ringuette, M. (1990), Introducing the Tileworld: Experimentally evaluating agent architectures, *Proceedings of the Eighth National Conference on Artificial Intelligence (AAAI-90)*, 183-189, Boston, MA, USA, 1990.
- Plewe, B. (1997), *GIS Online: Information retrieval, mapping and the Internet*, Onword Press, Santa Fe, NM, USA, 1997.
- Rao, A. S. and Georgeff, M. P. (1995), BDI Agents: From Theory to Practice, *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, San Francisco, USA, June 1995.
- Raper, J. (1996), Progress towards spatial multimedia, In M. Craglia and H. Couclelis (Eds.) *Geographic Information Research: Bridging the Atlantic*, Taylor & Francis, London, UK, 512-530, 1996.
- Raper, J. F. and Rhind, D.W. (1990), UGIX (A): the design of a spatial language interface for a topological vector GIS, *Proceedings of the Fourth International Symposium on Spatial Data Handling*, International Geographical Union, Zurich, 1:405-412, 1990.
- Rasmussen, J., Pejtersen, A. M. and Goodstein, L. P. (1994), *Cognitive Systems Engineering*, John Wiley & Sons, New York, NY, USA, 1994.
- Rasmussen, S. and Barrett, C. L. (1995), Elements of a Theory of Simulation, In *Proceedings of ECAL'95*, Lecture Notes in Computer Science, Springer-Verlag, Berlin, Germany, 1995.

- Reynolds, C. W. (1987), Flocks, Herds, and Schools: A Distributed Behavioural Model, *Proceedings of SIGGRAPH'87*, Computer Graphics, 21:4:25-34. 1987.
- Reynolds, C. W. (1997), Individual-Based Models: an annotated list of links, <<http://hmt.com/cwr/ibm.html>>, November 1997.
- Rodrigues, A. (1998), The Importance of Metadata within GI Infrastructures, *Proceedings of the Workshop on Challenges and Future Developments of GI Infrastructures: The Portuguese Experience*, GISPlanet'98, Lisbon, Portugal, 26-30, September 1998.
- Rodrigues, A. and Raper, J. (1998), Defining Spatial Agents, In A. S. Câmara and J. Raper (Eds.) *Spatial Multimedia and Virtual Reality*, Research Monographs, Taylor & Francis, UK, 111-129, 1998.
- Rodrigues, A., Grueau, C., Raper, J. and Neves, N. (1998), Environmental Planning using Spatial Agents, In S. Carver (Ed.) *Innovations in GIS 5*, Taylor & Francis, London, UK, 1998.
- Rosenchein, J. S. and Zlotkin, G. (1994), Designing conventions for automated negotiation, *AI Magazine*, 29-46, 1994.
- Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F. and Lorensen, W. (1991), *Object-Oriented Modeling and Design*, Englewood Cliffs, Nj, Prentice-Hall, 1991.
- Russell, S. and Norvig, P. (1995), *Artificial Intelligence: A Modern Approach*, Prentice Hall Series in Artificial Intelligence, New Jersey, USA, 1995.
- Salton, G. and McGill, M. (1993), *Introduction to modern information retrieval*, McGraw-Hill, New York, NY, USA, 1993.
- Shardanand, U. (1994), *Social information filtering for music recommendation*, BSc and MEng Thesis, Department of Electrical Engineering and Computer

Science, Massachusetts Institute of Technology, Cambridge, MA, USA, September 1994.

Shardanand, U. and Maes, P. (1995), Social Information Filtering: Algorithms for Automating 'Word of Mouth', *Proceedings of CHI '95*, ACM Press, 1995.

Shneiderman, B. (1988), Direct Manipulation: A step beyond programming languages, *IEEE Computer*, 16:8:57-69, August 1988.

Shneiderman, B. (1995), Looking for the bright side of user interface agents, *ACM Interactions*, 2:1:13-15, January 1995.

Shneiderman, B. (1997), A Grander Goal: A Thousand-Fold Increase in Human Capabilities, *Educom Review*, 32:6:4-10, November/December 1997.

Shneiderman, B. (1997a), Designing Information-Abundant Websites: Issues and Recommendations, *International Journal of Human-Computer Studies: Special Issue on Human-Computer Interaction & the World-Wide Web*, <<http://www.cs.umd.edu/projects/hcil/members/bshneiderman/ijhcs/ijhcs.htm>>, 1997.

Shen, W.-M. (1993), *Autonomous Learning from the Environment*, Computer Science Press and W. H. Freeman, New York, USA, 1993.

Sheppard, E., Couclelis, H., Graham, S., Harrington, J. W., Onsrud, H. (1999), Geographies of Information Society, *International Journal of Geographic Information Science: Special Issue on the Varenus Project*, to appear, 1999.

Sheth, B. (1994), *A learning approach to personalised information filtering*, SM Thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, USA, February 1994.

Sheth, B. and Maes, P. (1993), Evolving agents for personalised information filtering, *Proceedings of the Ninth Conference on Artificial Intelligence for Applications*, IEEE Computer Society Press, 1993.

- Singh, M. P. (1997), Commitments among autonomous agents in information-rich environments, *Proceedings of the 8th European Workshop on Modelling Autonomous Agents in a Multi-Agent World (MAAMAW)*, 141-155, 1997.
- Slothower, R. L., Schwarz, P. A. and Johnston, K. M. (1996), Some Guidelines for Implementing Spatially Explicit, Individual-Based Ecological Models within Location-based Raster GIS, *Proceedings of The Third International Conference/Workshop on Integrating GIS and Environmental Modelling*, Santa Fe, New Mexico, USA, January 21-25, 1996.
- Smith, R. G. and Davis, R. (1981), Frameworks for Cooperation in Distributed Problem Solving, *IEEE Transactions on Systems, Man and Cybernetics*, SMC-11:1:61-70, 1981.
- SNIG (1998), The Geographic Metadata Page at the SNIG, <<http://snig.cnig.pt/snig/english/metadata.htm>>, 1998.
- Stanfill, C. (1988), Learning to read: A Memory-Based Model, *Proceedings of the 1988 DARPA Workshop on Case-Based Reasoning*, 1988.
- Stanfill, C. and Waltz, D. (1986), Toward Memory-based Reasoning, *Communications of the ACM*, 29:12:1213-1228, December 1986.
- Steels, L. (1990), Cooperation between distributed agents through self organization, In Y. Demazeau and J.-P. Müller (Eds.) *Decentralized AI – Proceedings of the First European Workshop on Modelling Autonomous Agents in Multi-Agent Worlds (MAAMAW-89)*, 175-196, Elsevier Science Publishers B. B., 1990.
- Stone, P. and Veloso, M. (1997), Multiagent Systems: A Survey from a Machine Learning Perspective, <<http://cs.cmu.edu/afs/cs/usr/pstone/public/papers/96ieee-survey/survey.ps.Z>>, February 1997.
- Sullivan, J. W. and Tyler, S. W. (1991), Eds., *Intelligent User Interfaces*, ACM Press, New York, USA, 1991.

- Sycara, K. (1995), Intelligent Agents and the Information Revolution, *Proceedings of The UNICOM Seminar on Intelligent Agents and their Business Applications*, London, UK, 143-159, 8-9 November 1996.
- Titmuss, R, Winter, C. S. and Crabtree, B. (1996), Agents, Mobility and Multimedia Information, *Proceedings of the First International Conference on the Practical Application of Intelligent Agents and Multi-Agent Technology (PAAM '96)*, London, UK, 693-708, 22-24 April 1996.
- Toomey, C. H. et al (1994), Software Agents for the dissemination of Remote Terrestrial Sensing Data, *Proceedings of the Third International Symposium on Artificial Intelligence, Robotics and Automation for Space (i-SAIRAS 94)*, NASA Jet Propulsion Laboratory, Pasadena, California, October 1994.
- Touret, A. (1995), Agripa: un modèle de calcul de courbes isochrones fondé sur un Système Multi-Agent, *Revue internationale de géomatique: numéro spécial aide a la décision spatiale*, 5:3-4:299-314, Editons Hermès, Paris, France, 1995.
- Tsou, M.-H. and Battenfield, B. P. (1998), An Agent-based, Global User Interface for Distributed Geographic Information Services, *Proceedings of Spatial Data Handling '98*, Vancouver, Canada, 603-12, 1998.
- Van Dyke, N. W., Lieberman, H. and Maes, P. (1999), Butterfly: A Conversation-Finding Agent for Internet Relay Chat, *Proceedings of the 1999 International Conference on Intelligent User Interfaces*, <<http://agents.www.media.mit.edu/groups/agents/publications/butterfly-iui99/paper.pdf>>, January 1999.
- Vigneron, V. (1995), Accessibilité routière au sud de l'agglomération grenobloise en liaison avec la construction de l'autoroute A51, *Revue internationale de géomatique: numéro spécial aide a la décision spatiale*, 5:3-4:283:297, Editons Hermès, Paris, France, 1995.

- Vivacqua, A. S. (1999), Agents for Expertise Location, *Proceedings of the 1999 AAAI Spring Symposium on Intelligent Agents in Cyberspace*, Technical Report SS-99-03, Stanford, CA, USA, March 1999.
- Volterra, V. (1926), Variation and fluctuations of the number of individuals of animal species living together, *Animal Ecology*, McGraw-Hill, 1926.
- Weisbuch, G., Gutowitz, H. and Nguyen, G. D. (1994), *Information Contagion and the Economics of Pollution*, Santa Fe Series Working Paper, 94-04-018, The Santa Fe Institute, Santa Fe, New Mexico, USA, <<http://www.lps.ens.fr/~weisbuch/car/car.html>>, 1994.
- Weiß, G. (1997) (Ed.), *Distributed Artificial Intelligence Meets Machine Learning: Learning in Multi-Agent Environments*, Lecture Notes in Artificial Intelligence, 1221, Springer-Verlag, Berlin, Germany, 1997.
- Wellman, M. P. (1995), A computational market model for distributed configuration design, *AI EDAM*, 9: 125-133, 1995.
- Wexelblat, A. (1999), *History-Rich Tools for Social Navigation*, <<http://wex.www.media.mit.edu/people/wex/Footprints2/fp-v2.html>>, 1999.
- Wolfram, S. (1994) *Two-Dimensional Cellular Automata*, Cellular Automata and Complexity: Collected Papers, Addison-Wesley Publishing Company, 1994.
- Wood, S. (1993), *Planning and Decision Making in Dynamic Domains*, Ellis Horwood, 1993.
- Wooldridge, M. (1997), Agent-based software engineering, *IEE Proceedings on Software Engineering: Special issue on Agent-based systems*, 144:1: 26-37, 1997.
- Wooldridge, M. and Jennings, N. R. (1995), Agent Theories, Architecture, and Languages: A Survey, *Intelligent Agents: Proceedings of The ECAI-94*

Workshop on Agent Theories, Architectures, and Languages, Amsterdam, The Netherlands, 1-39, 1995.

Worboys, M. F. (1994), A Unified Model of Spatial and Temporal Information, *Computer Journal*, 37:1:26-34, 1994.

Worboys, M. F. (1995), *GIS: A Computing Perspective*, Taylor & Francis, London, UK, 1995.

Worboys, M. F. (1996), Adding the temporal dimension to GIS, *Proceedings of the Second GIS Workshop*, Genova, Italy, 1996.

Worboys, M. F. (1996a), A generic model for spatio-bitemporal geographic information, In R. G. Golledge and M. Egenhofer (Eds.) *Spatial and Temporal Reasoning in Geographic Information Systems*, in press, 1996.

Worboys, M. F. (1999), Object-oriented modelling of changes and events for dynamic spatial systems, In A. Frank, J. F. Raper and J.-P. Cheylan, *Life and motion of Socio-economic Units*, GISDATA Series, Taylor and Francis, England, in press, 1999.

Appendix A: Magik code for the printing and plotting assistant

Agent Controller Class

```
_pragma(classify_level=restricted, topic={agent})
##
## agent_controller - controlling agent for all task agents
## opens agent menu
##
def_slotted_exemplar(:agent_controller,
                    {
                        {:grs, _unset}, # graphics system
                        {:menu_items, _unset}, # hash table of menu
items
                        {:images, _unset}, # icon images for buttons
                        {:functions_on, _unset}, # functions enabled
                        {:agents, _unset}, # existing task agents
                        {:last_event, _unset}, # last_event
                        {:current_task, _unset},
                        {:current_agent, _unset}
                    },
                    {:model}
                )
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
agent_controller.define_slot_access(
    :agents,
    ##
    ## List of active task agents
    :writable)
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
agent_controller.define_slot_access(
    :last_event,
    ##
    ## Last occurring event
    :writable)
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
agent_controller.define_slot_access(
    :current_task,
    ##
    ## current task
    :writable)
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
agent_controller.define_slot_access(
    :current_agent,
    ##
    ## current active agent
    :writable)
$
```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method agent_controller.title
    ##
    ## GUI title string
    ##
    >> _self.message(:agent_controller_title, "Controller")
_endmethod

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method agent_controller.new(a_grs)
    ##
    ## Create a new instance of the agent controller.
    ##
    >> _clone.init(a_grs)
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_private _method agent_controller.open_on(a_grs)

    ## Same as new except in addition the editor is activated
    ## to realise its display

    new << _self.new(a_grs)
    new.activate()
    >> new

_endmethod
$

## INIT:

_pragma(classify_level=restricted, topic={agent}, usage={external})
_private _method agent_controller.init(a_grs)
    ##
    ## Initialisation for the new method
    _super.init()
    .grs << a_grs
    .menu_items << hash_table.new()
    .images << hash_table.new()
    .functions_on << hash_table.new()
    .agents << hash_table.new()
    >> _self
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method agent_controller.activate_in(a_frame)
    ##
    ## create panel
    ##
    _local buttons_panel << panel.new( a_frame )

    buttons_panel.start_row()

    ## help text
    .menu_items[:help_text] <<
label_item.new(buttons_panel, "Help")

    buttons_panel.start_row()

```

```

    ## help button
    _local_icons << system.getenv("SW_AGENT_ICONS")
    .images[:help] <<
raster_image.new_from_file(icons+"/help.xbm")
    .functions_on[:help] << _true
    .menu_items[:help] <<
image_button_item.new_safe(buttons_panel, .images[:help], _self,
:agent_help|()|, :visibility, _true)

    buttons_panel.start_row()

    ## suggest/perform text
    .menu_items[:sp_text] <<
label_item.new(buttons_panel, "Suggest/Perform")

    buttons_panel.start_row()

    ## suggest and perform buttons
    .images[:suggest] <<
raster_image.new_from_file(icons+"/sugoff.xbm")
    .functions_on[:suggest] << _false
    .menu_items[:suggest] <<
image_button_item.new_safe(buttons_panel, .images[:suggest], _self,
:agent_suggest|()|, :visibility, _true)
    .images[:perform] <<
raster_image.new_from_file(icons+"/peroff.xbm")
    .functions_on[:perform] << _false
    .menu_items[:perform] <<
image_button_item.new_safe(buttons_panel, .images[:perform], _self,
:agent_perform|()|, :visibility, _true)

    buttons_panel.start_row()

    ## enable/disable text
    .menu_items[:ed_text] <<
label_item.new(buttons_panel, "Enable/Disable")

    buttons_panel.start_row()

    ##enable and disable buttons
    .images[:enable] <<
raster_image.new_from_file(icons+"/eoff.xbm")
    .functions_on[:enable] << _false
    .menu_items[:enable] <<
image_button_item.new_safe(buttons_panel, .images[:enable], _self,
:agent_enable|()|, :visibility, _true)
    .images[:disable] <<
raster_image.new_from_file(icons+"/disable.xbm")
    .functions_on[:disable] << _true
    .menu_items[:disable] <<
image_button_item.new_safe(buttons_panel, .images[:disable], _self,
:agent_disable|()|, :visibility, _true)

    buttons_panel.start_row()

    ## quit button
    .menu_items[:quit] <<
button_item.new_safe(buttons_panel, "Quit", _self, :suspend|()|, :visibi
lity, _true)
    _endmethod

```

```

$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method agent_controller.event_occurred(topic_list,class_name,
method_name, args)
  ##
  ## Event occurred - send it to the proper agent
  ##
  _if class_name _is :agent_controller _orif
_self.agent_disabled?() _orif
    method_name _is :|perform_transaction()|
    _then _return
    _endif

    _if .current_agent _is _unset
    _then
      .current_agent <<
drafting_agent.new_with_event(.grs,_self,gis_program_manager.applica
tions[:drawing],topic_list,class_name, method_name, args)
    _else
      ## This is for the drafting_agent specifically
      .current_agent.event_occurred(topic_list, class_name,
method_name, args)
    _endif
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method agent_controller.agent_help()

  ##
  ##
  .current_agent.agent_help()

_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method agent_controller.agent_suggest()
  ##
  ##
  _if _not _self.agent_disabled?()
  _then
    _if _self.suggest_enabled?()
    _then .current_agent.agent_suggest()
    _endif
  _endif

_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method agent_controller.agent_perform()
  ##
  ##
  _if _not _self.agent_disabled?()
  _then
    _if _self.perform_enabled?()
    _then .current_agent.agent_perform()
    _endif
  _endif

```

```

_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method agent_controller.agent_enable()

    ##
    ##
    _local icons << system.getenv("SW_AGENT_ICONS")
    .menu_items[:enable].image_file << icons+"/eoff.xbm"
    .functions_on[:enable] << _false
    .menu_items[:disable].image_file << icons+"/disable.xbm"
    .functions_on[:disable] << _true
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method agent_controller.agent_disable()

    ##
    ##
    _local icons << system.getenv("SW_AGENT_ICONS")
    .menu_items[:enable].image_file << icons+"/enable.xbm"
    .functions_on[:enable] << _true
    .menu_items[:disable].image_file << icons+"/doff.xbm"
    .functions_on[:disable] << _false
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method agent_controller.agent_quit()

    ##
    ##
    _super.quit()
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method agent_controller.enable_suggest_button()

    ##
    ##
    _local icons << system.getenv("SW_AGENT_ICONS")
    .menu_items[:suggest].image_file <<
icons+"/suggest.xbm"
    .functions_on[:suggest] << _true
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method agent_controller.disable_suggest_button()

    ##
    ##
    _local icons << system.getenv("SW_AGENT_ICONS")
    .menu_items[:suggest].image_file << icons+"/sugoff.xbm"
    .functions_on[:suggest] << _false

```

```

_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method agent_controller.enable_perform_button()

    ##
    ##
    _local icons << system.getenv("SW_AGENT_ICONS")
    .menu_items[:perform].image_file <<
icons+"/perform.xbm"
    .functions_on[:perform] << _true
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method agent_controller.disable_perform_button()

    ##
    ##
    _local icons << system.getenv("SW_AGENT_ICONS")
    .menu_items[:perform].image_file <<
icons+"/peroff.xbm"
    .functions_on[:perform] << _false
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method agent_controller.perform_enabled?()

    ##
    ##
    _return .functions_on[:perform]
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method agent_controller.suggest_enabled?()

    ##
    ##
    _return .functions_on[:suggest]
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method agent_controller.agent_disabled?()

    ##
    ##
    _return .functions_on[:enable]
_endmethod
$

```

Task Agent Class

```
_pragma(classify_level=restricted, topic={agent})
  ##
  ## Task Agent
  ##
def_slotted_exemplar(:task_agent,
{
  {:grs, _unset},
  {:parent, _unset}, ## agent_controller of this agent
  {:last_topic_list, _unset},
  {:last_class_name, _unset},
  {:last_method, _unset},
  {:last_args, _unset},
  {:task_history, _unset}, ## is not being used yet
  {:next_suggestion, _unset},
  {:performable_task, _unset}
})
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
task_agent.define_slot_access(
  :grs,
  ##
  ##
  ##
  :writable)
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
task_agent.define_slot_access(
  :parent,
  ##
  ##
  ##
  :readable)
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
task_agent.define_slot_access(
  :last_topic_list,
  ##
  ##
  ##
  :writable)
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
task_agent.define_slot_access(
  :last_class_name,
  ##
  ##
  ##
  :writable)
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
task_agent.define_slot_access(
  :last_method,
```



```

    ##
    ##
    ##
    :writable)
$

__pragma(classify_level=restricted, topic={agent},usage={external})
task_agent.define_slot_access(
    :last_args,
    ##
    ##
    ##
    :writable)
$

__pragma(classify_level=restricted, topic={agent},usage={external})
task_agent.define_slot_access(
    :task_history,
    ##
    ##
    ##
    :writable)
$

__pragma(classify_level=restricted, topic={agent},usage={external})
task_agent.define_slot_access(
    :next_suggestion,
    ##
    ##
    ##
    :writable)
$

__pragma(classify_level=restricted, topic={agent},usage={external})
task_agent.define_slot_access(
    :performable_task,
    ##
    ##
    ##
    :writable)
$

__pragma(classify_level=restricted, topic={agent}, usage={external})
_method task_agent.new(a_grs,parent)
    ##
    ## New general agent
    ##
    >> _clone.init(a_grs,parent)
_endmethod
$

__pragma(classify_level=restricted, topic={agent}, usage={external})
_method task_agent.init(a_grs,parent)
    ##
    ##
    ##
    .grs << a_grs
    .parent << parent
    >> _self

```

```

_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method task_agent.new_with_event(a_grs,parent, topic_list,
class_name, method_name, args)
  ##
  ## create a task agent with an event that already occurred
  ##
  new << task_agent.new(a_grs,parent)
  new.event_occurred(topic_list, class_name, method_name, args)
  >> new

_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method task_agent.event_occurred(topic_list, class_name,
method_name, args)
  ##
  ##
  .last_topic_list << topic_list
  .last_class_name << class_name
  .last_method << method_name
  .last_args << args

  _self.review_status()
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method task_agent.review_status()

_endmethod
$

```

Drafting Agent Class

```
_pragma(classify_level=restricted, topic={agent})
    ##
    ## Drafting Agent
    ##

def_slotted_exemplar(:drafting_agent,
{
    {:last_trail, _unset},
    {:last_geometry, _unset},
    {:drawing_application, _unset},
    {:drawing_function, _unset},
    {:drawing_function_inserted?, _false}
},
{:task_agent}
)
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
drafting_agent.define_slot_access(
    :last_trail,
    ##
    ##
    ##
    :writable)
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
drafting_agent.define_slot_access(
    :drawing_function_inserted?,
    ##
    ##
    ##
    :writable)
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
drafting_agent.define_slot_access(
    :last_geometry,
    ##
    ##
    ##
    :writable)
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
drafting_agent.define_slot_access(
    :drawing_application,
    ##
    ##
    ##
    :writable)
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
drafting_agent.define_slot_access(
    :drawing_function,
    ##
    ##
```

```

    ##
    :writable)
$

__pragma(classify_level=restricted, topic={agent}, usage={external})
drafting_agent.define_shared_constant(:drawing_help,
    ##
    ## Constant that will provide web page
to call for help
    ##

    __block t << hash_table.new()

        t[:create_trail] <<
"http://hafnium:1234/multiple/gis_user_guide/gis_user_guide-43.html"
        t[:choosing_drawing_function] <<
"http://hafnium:1234/multiple/gis_config_guide/gis_config_guide-
166.html"
            t[:selecting_drawing_function] <<
"http://hafnium:1234/multiple/gis_config_guide/gis_config_guide-
166.html"

            t[:insert_drawing_function] <<
"http://hafnium:1234/multiple/gis_config_guide/gis_config_guide-
166.html"
                t[:create_drafting_geometry] <<
"http://hafnium:1234/multiple/gis_user_guide/gis_user_guide-49.html"
                t[:insert_drafting_geometry] <<
"http://hafnium:1234/multiple/gis_user_guide/gis_user_guide-49.html"
                t[:change_trail] <<
"http://hafnium:1234/multiple/gis_user_guide/gis_user_guide-43.html"
                t[:set_geometry] <<
"http://hafnium:1234/multiple/gis_config_guide/gis_config_guide-
166.html"

                t[:create_box] <<
"http://hafnium:1234/multiple/gis_user_guide/gis_user_guide-48.html"
                t[:click_box_button] <<
"http://hafnium:1234/multiple/gis_user_guide/gis_user_guide-48.html"
                t[:delete_geometry] <<
"http://hafnium:1234/multiple/gis_user_guide/gis_user_guide-49.html"
            >> t

    __endblock, :writable)
$

__pragma(classify_level=restricted, topic={agent}, usage={external})
drafting_agent.define_shared_constant(:drawing_events,
    ##
    ## constant to match to occurred
events
    ##
    ## structure: topic - metadata topic
    ## class - class that is
generating event
    ## method - method called
by class
    ## These should be matched against
each event do
    ## generate suggestions

```

```

##
_block h << dual_key_a_table.new()

h[:graphics_system, :|activate_drawing_function_editor()|] <<
:|selecting_drawing_function()|

h[:drawing_function_editor, :|set_function_name_list_index()|] <<
  {{10,
:|combine_smallworldlogo_with_last_trail_or_geometry()|, :smallworld_
logo}},
  {5, :|combine_viewport_with_last_trail_or_geometry()|, :viewport
}
}
h[:drawing_function_editor, :|insert()|] <<
:|dfinsert_next_step()|

h[:graphics_system, :|activate_trail_construction_panel()|] <<
:|draw_box()|
h[:graphics_system, :|trail_box()|] <<
:|combine_box_trail_with_drawing_function()|

h[:drawing_function_editor, :|activate_geometry_editor()|] <<
:|combine_geometry_definition_with_last_trail()|
h[:drafting_editor, :|insert()|] <<
:|combine_new_geometry_with_status()|
h[:drawing_function_editor, :|set_geometry()|] <<
:|suggestions_after_setting_geometry()|
h[:graphics_system, :|activate_drawing_manager()|]
<< _unset
h[:drawing_manager, :|int!display_drawing()|] <<
_unset

h[:graphics_system, :|activate_viewport_mapping_editor()|] <<
:|combine_viewport_with_view()|
h[:viewport_mapping_editor, :|apply()|] << _unset
h[:graphics_system, :|refresh()|] << _unset
h[:graphics_system, :|trail_clear()|] <<

:|no_trail()|
h[:drawing_function_editor, :|suspend()|] <<
:|no_drawing_function_editor()|
h[:logical_mouse, :|trail_event()|] <<

:|update_trail()|
h[:drawing_function_editor, :|delete()|] <<
:|drawing_function_deleted()|
>> h

_endblock, :writable)

$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.init(a_grs,parent)
##
##
##
.last_geometry << _unset
_if gis_program_manager.applications[:drawing] _isnt _unset
_then

```

```

        .drawing_application <<
gis_program_manager.applications[:drawing]
    _endif
    >> _super.init(a_grs,parent)
_endif
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.new_with_event(a_grs,parent,_optional
drawing_application,topic_list, class_name, method_name, args)
    ##
    ## creating new drafting agents with information from last
event
    ##
    new << drafting_agent.new(a_grs,parent)
    _if drawing_application _isnt _unset
    _then .drawing_application << drawing_application
    _endif

    new.event_occurred(topic_list,class_name, method_name, args)
    >> new
_endif
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.find_trail()
    _if .drawing_application _isnt _unset
    _then
        _if .drawing_application.gtrail.current > 1_andif
            .last_method ~= :|trail_clear()|
        _then
            _if .drawing_application.gtrail ~= .last_trail
            _then
                .last_trail << .drawing_application.gtrail
            _endif
        _else
            .last_trail << _unset
        _endif
    _else
        .last_trail << _unset
    _endif

    _endif
_endif
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method
drafting_agent.pick_drawing_function(args,drawing_functions_list)
    _for i _over drawing_functions_list.elements()
    _loop
        _if i[1]=args[1]
        _then
            _self.perform(i[2])
            _return i[3]
        _endif
    _endloop
_endif
$

_pragma(classify_level=restricted, topic={agent}, usage={external})

```

```

_method drafting_agent.is_drawing_function_event?(method_name)
  _return method_name _is :|set_function_name_list_index(|
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.no_drawing_function_editor()
  ##
  ##
  ##
  .drawing_function << _unset
  .drawing_function_inserted? << _false
  .performable_task << _unset
  .parent.disable_perform_button()
  .next_suggestion << _unset
  .parent.disable_suggest_button()

_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.combine_new_geometry_with_status()
  ##
  ##
  ##
  .last_trail << _unset
  .last_geometry << .drawing_application.current_geometry
  _if .drawing_function _isnt _unset
  _then
    _if _self.drawing_function_needs_2_point_trail?
    _then
      _if .last_geometry.class_name _is :simple_area
      _then
        _if .drawing_function_inserted?
        _then
          .next_suggestion << :set_geometry
          .parent.enable_suggest_button()
        _else
          .next_suggestion <<
:insert_drawing_function
          .parent.enable_suggest_button()
        _endif
      _else
        .next_suggestion << :create_trail
        .parent.enable_suggest_button()
      _endif
    _endif
  _else
    .next_suggestion << :choosing_drawing_function
    .parent.enable_suggest_button()
  _endif
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.drawing_function_needs_2_point_trail?
  ##
  ##
  ##
  _return .drawing_function _is :smallworld_logo _orif
.drawing_function _is :viewport

```

```

_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.drawing_function_deleted()
  ##
  ##
  ##
  .drawing_function_inserted? << _false
  .next_suggestion << :insert_drawing_function
  .parent.enable_suggest_button()
_endmethod

```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.update_trail()
  ##
  ##
  ##
  _if .last_geometry _isnt _unset
  _then
    _if .drawing_application.gtrail.current = 2 _orif
    .drawing_application.gtrail.current = 5
    _then
      _if .drawing_function _isnt _unset
      _then
        _if
        _self.drawing_function_needs_2_point_trail?
        _then
          .last_geometry << _unset
          .last_trail <<
        .drawing_application.gtrail
        _if .drawing_function_inserted?
        _then
          .next_suggestion <<
        :set_geometry
          .parent.enable_suggest_button()
        _else
          .next_suggestion <<
        :insert_drawing_function
          .parent.enable_suggest_button()
        _endif
      _endif
    _else
      .next_suggestion <<
    :choosing_drawing_function
      .parent.enable_suggest_button()
    _endif
  _endif
  _else
    _if .drawing_application.gtrail.current > 0
    _then .last_trail << .drawing_application.gtrail
    _endif
    _if .drawing_function _isnt _unset
    _then
      _if .drawing_function_inserted?
      _then

```



```

        _if
        _self.drawing_function_needs_2_point_trail?
        _then
            _if .last_trail.current = 2
            _then
                .next_suggestion << :create_box
                .parent.enable_suggest_button()
            _else
                _if .last_trail.current = 5
                _then
                    .next_suggestion <<
: create_drafting_geometry
                .parent.enable_suggest_button()
            _else
                .next_suggestion <<
: delete_geometry
                .parent.enable_suggest_button()
            _endif
        _endif
        _endif
        _endif
        _else
            .next_suggestion << :choosing_drawing_function
            .parent.enable_suggest_button()
        _endif
    _endif
endmethod
$

```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.dinsert_next_step()
    ##
    ##
    ##
    .drawing_function_inserted? << _true
    .drawing_function <<
    .drawing_application.sub_menus[:drawing_function_editor].current_reco
rd.function_name.as_symbol()
    _if .last_trail_isnt_unset
    _then
        _if _self.drawing_function_needs_2_point_trail?
        _then
            _if .last_trail.current = 2
            _then
                .next_suggestion << :create_box
                .parent.enable_suggest_button()
            _else

```

```

        _if .last_trail.current = 5
        _then
            .next_suggestion <<
:create_drafting_geometry
            .parent.enable_suggest_button()
        _else
            .next_suggestion << :change_trail
            .parent.enable_suggest_button()
            .performable_task << _unset
            .parent.disable_perform_button()
        _endif
    _endif
    _endif

    _endif
    _else
        _if .last_geometry _isnt _unset
        _then
            _if _self.drawing_function_needs_2_point_trail?
            _then
                _if .last_geometry.class_name _is
:simple_area
                _then
                    .next_suggestion << :set_geometry
                    .parent.enable_suggest_button()
                _else
                    .next_suggestion << :create_trail
                    .parent.enable_suggest_button()
                    .performable_task << _unset
                    .parent.disable_perform_button()
                _endif
            _endif
        _endif
        _else
            .next_suggestion << :create_trail
            .parent.enable_suggest_button()
        _endif
    _endif
    _endif
_endmethod
$

```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.suggest_choosing_drawing_function()
    ##
    ##
    ##
    .next_suggestion << :choosing_drawing_function
    .drawing_function << _unset
    .drawing_function_inserted? << _false
    .parent.enable_suggest_button()
_endmethod
$

```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.selecting_drawing_function()
    ##
    ##
    ##
    .next_suggestion << :selecting_drawing_function
    .drawing_function << _unset

```

```

        .drawing_function_inserted? << _false
        .parent.enable_suggest_button()
    _endmethod
$

    _pragma(classify_level=restricted, topic={agent}, usage={external})
    _method
drafting_agent.combine_smallworldlogo_with_last_trail_or_geometry()
    ##
    ## Event - The drawing function smallworld logo has been
chosen
    ## Requisites - There is a trail or a geometry
    ## suggest - Create a trail or a geometry
    ## perform - if requisites exist - create smallworld logo
    ##
    .drawing_function << :smallworld_logo
    .drawing_function_inserted? << _false
    _if .last_geometry_isnt_unset
    _then
        .next_suggestion << :insert_drawing_function
        .parent.enable_suggest_button()
        .performable_task << :smallworld_logo_with_geometry
        .parent.enable_perform_button()
    _elif .last_trail_isnt_unset
    _then
        .next_suggestion << :insert_drawing_function
        .parent.enable_suggest_button()
        .performable_task << :smallworld_logo_with_trail
        .parent.enable_perform_button()
    _else
        .next_suggestion << :create_trail
        .parent.enable_suggest_button()
        .performable_task << _unset
        .parent.disable_perform_button()
    _endif
    _endmethod
$

    _pragma(classify_level=restricted, topic={agent}, usage={external})
    _method
drafting_agent.combine_viewport_with_last_trail_or_geometry()
    ##
    ## Event - The drawing function viewport has been created
    ## requisites - there is a trail or a geometry (and the
drawing
    ## must the parameters inserted)
    ## suggest - insert a trail or geometry, if that exists,
choose
    ## the view that will be in the viewport
    ## perform - create a viewport and a viewport mapping using
    ## the bounds of the main view
    ##
    .drawing_function << :viewport
    .drawing_function_inserted? << _false
    _if .last_geometry_isnt_unset
    _then
        .performable_task << :viewport_with_geometry
        .parent.enable_perform_button()
    _elif .last_trail_isnt_unset

```

```

        _then
            .performable_task << :viewport_with_trail
            .parent.enable_perform_button()
        _endif
    _endmethod
$
    _pragma(classify_level=restricted, topic={agent}, usage={external})
    _method drafting_agent.combine_viewport_with_view()
        ##
        ## Event - the viewport mapping editor has been called
        ## requisites - there is a viewport
        ## suggest - select a view from the graphics system
        ##
    _endmethod
$

    _pragma(classify_level=restricted, topic={agent}, usage={external})
    _method drafting_agent.draw_box()
        ##
        ## Event - The trail constructions menu has been called
        ## requisites - there must be a trail
        ## suggest - if there is no trail create one, if there is a
        ## trail and a drawing function has been chosen, create a box
        ## perform - create a box from the trail
        ##
        _if .drawing_function _isnt _unset
            _then
                _if _self.drawing_function_needs_2_point_trail?
                    _then
                        _if .last_trail _isnt _unset
                            _then
                                _if .last_trail.current = 2
                                    _then
                                        .next_suggestion << :click_box_button
                                        .parent.enable_suggest_button()
                                    _else
                                        .next_suggestion << :change_trail
                                        .parent.enable_suggest_button()
                                    _endif
                                _else
                                    _if .last_geometry _isnt _unset
                                        _then
                                            _if .last_geometry.class_name _is
: simple_area
                                                _then
: delete_geometry
                                                    .next_suggestion <<
                                                        .parent.enable_suggest_button()
                                                    _else
: create_trail
                                                        .next_suggestion <<
                                                            .parent.enable_suggest_button()
                                                        _endif
                                                    _else
                                                        .next_suggestion << :create_trail
                                                        .parent.enable_suggest_button()
                                                    _endif
                                                _endif
                                            _endif
                                        _endif
                                    _else
                                        .next_suggestion << :create_trail
                                        .parent.enable_suggest_button()
                                    _endif
                                _endif
                            _endif
                        _endif
                    _endif
                _endif
            _endif
        _endif
    _endmethod
$

```

```

        .next_suggestion << :create_trail
        .parent.enable_suggest_button()
    _endif
_endif
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.combine_box_trail_with_drawing_function()
    ##
    ## Event - A trail box has been created
    ## suggest - if a drawing_function has been chosen, and if it
    ## needs as box geometry, create one
    ## perform - create the drafting_area
    ##
    .last_trail << .drawing_application.gtrail
    .last_geometry << _unset
    _if .drawing_function _isnt _unset
    _then
        _if _self.drawing_function_needs_2_point_trail?
        _then
            .next_suggestion << :create_drafting_geometry
            .parent.enable_suggest_button()
        _endif
    _else
        .next_suggestion << :choosing_drawing_function
        .parent.enable_suggest_button()
    _endif
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.combine_geometry_definition_with_last_trail()
    _if .last_trail _isnt _unset
    _then
        .next_suggestion << :insert_drafting_geometry
        .parent.enable_suggest_button()
    _endif
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.suggestions_after_setting_geometry()
    _if .drawing_function = :smallworld_logo
    _then
        .performable_task << _unset
        .parent.disable_perform_button()
        .next_suggestion << _unset
        .parent.disable_suggest_button()
        .drawing_function << _unset
        .drawing_function_inserted? << _false
        .last_geometry << _unset
    _endif
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.no_trail()
    ##
    ##
    ##

```

```

        .last_trail << _unset
        _if .drawing_application.current_geometry _isnt _unset
        _then
            .last_geometry << .drawing_application.current_geometry
            _if .drawing_function _isnt _unset
            _then
                _if .last_geometry.class_name _is :simple_area
                _then
                    _if
                    _self.drawing_function_needs_2_point_trail?
                    _then
                        _if .drawing_function_inserted?
                        _then
                            .next_suggestion <<
:
:set_geometry
                            .parent.enable_suggest_button()
                        _else
                            .next_suggestion <<
:
:insert_drawing_function
                            .parent.enable_suggest_button()
                        _endif
                    _endif
                _else
                    .next_suggestion << :delete_geometry
                    .parent.enable_suggest_button()
                _endif
            _else
                .next_suggestion << :choosing_drawing_function
                .parent.enable_suggest_button()
            _endif
        _else
            _if .drawing_function _isnt _unset
            _then
                _if .drawing_function_inserted?
                _then
                    .next_suggestion <<
:
:create_trail
                    .parent.enable_suggest_button()
                _else
                    .next_suggestion <<
:
:insert_drawing_function
                    .parent.enable_suggest_button()
                _endif
            _else
                .next_suggestion << :choosing_drawing_function
                .parent.enable_suggest_button()
            _endif
        _endif
    _endmethod
$
    _pragma(classify_level=restricted, topic={agent}, usage={external})
    _method drafting_agent.enable_perform_drawing_function()
        ##
        ##
        ##
        _if .drawing_function = :smallworld_logo
        _then

```

```

        _self.combine_smallworldlogo_with_last_trail_or_geometry()
        _elif .drawing_function = :viewport
        _then
            _self.combine_viewport_with_last_trail_or_geometry()
        _endif
    _endmethod
$

    _pragma(classify_level=restricted, topic={agent}, usage={external})
    _method drafting_agent.review_status(_optional no_event?)
        ##
        ## the review_status will have to :
        ##     - check if there is a geometry defined since
        ##     the agent was born or if there is a usable
trail
        ##     - check that a drawing function has been
        ##     chosen since the agent was born
        ##     -compute possible suggestions or actions by
        ##     matching last event with drawing_events
        ##     constant and with the agent's status
        ##
        _if gis_program_manager.applications[:drawing] _is _unset
        _then
            .drawing_application << _unset
            _return
            _elif .drawing_application _is _unset
            _then
                .drawing_application <<
gis_program_manager.applications[:drawing]
        _endif
        ##
        ## disable suggestions to inspect the current state of the
agent
        _if no_event? _is _unset
        _then
            .next_suggestion << _unset
            .parent.disable_suggest_button()
        _endif

        ##
        ## If there is no trail and no geometry defined the the agent
        ## cannot perform any tasks
        ##
        _if .last_geometry _is _unset _andif .last_trail _is _unset
        _then
            .performable_task << unset
            .parent.disable_perform_button()
        _endif

        event << _self.drawing_events[.last_class_name, .last_method]
        _if no_event? _isnt _unset _andif no_event?
        _then
        _else
            _if event _isnt _unset
            _then
                _if _self.is_drawing_function_event?(.last_method)
                _then

```

```

        .drawing_function <<
    _self.pick_drawing_function(.last_args,event)
        _else
            _self.perform(event)
            _return
        _endif
    _endif

    _if .drawing_function _isnt _unset
        _then
            _if (.last_trail _isnt _unset _orif
.last_geometry _isnt _unset)
                _then
                    _self.enable_perform_drawing_function()
                _else
                    .next_suggestion << :create_trail
                    .parent.enable_suggest_button()
                _endif
            _elif .last_trail _isnt _unset _orif .last_geometry
_isnt _unset
                _then
                    .performable_task << _unset
                    .parent.disable_perform_button()
                    _self.suggest_choosing_drawing_function()
                _endif
            _endif
        _endif
    _endmethod
$

```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.agent_perform()
    ##
    ## According with .performable_task the agent will perform the
    ## most likely task until de end
    _if .performable_task = :smallworld_logo_with_trail
        _then
            _self.perform_smallworld_logo_with_trail()
        _elif .performable_task = :viewport_with_trail
        _then
            _self.perform_viewport_with_trail()
        _endif
    _endmethod
$

```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.agent_suggest()
    ##
    ##
    ##
    _if .next_suggestion = :create_trail
        _then
            _self.suggest_create_trail()
        _elif .next_suggestion = :choosing_drawing_function
        _then
            _self.suggest_drawing_function()
        _elif .next_suggestion = :insert_drawing_function
        _then _self.suggest_insert_drawing_function()
    _endif

```



```

        _elif .next_suggestion =
:create_drafting_geometry
        _then
            _self.suggest_creating_geometry()
            _elif .next_suggestion = :change_trail
            _then
                _self.suggest_changing_trail()
            _elif .next_suggestion =
:set_geometry
            _then
                _self.suggest_setting_geometry()
        _elif .next_suggestion =
:create_box
            _then
                _self.suggest_calling_trail_menu()
            _elif
.next_suggestion = :click_box_button
            _then
                _self.suggest_click_box_button()
            _elif
.next_suggestion = :delete_geometry
            _then
                _self.suggest_delete_geometry()
            _elif
.next_suggestion = :selecting_drawing_function
            _then
                _self.suggest_selecting_drawing_function()
        _elif .next_suggestion = :insert_drafting_geometry
        _then
            _self.suggest_insert_drafting_geometry()
        _endif
        ##.parent.disable_suggest_button()
        ##.next_suggestion << unset
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.agent_help()
    ##
    ##
    ##
    _if .next_suggestion _isnt unset
    _then
        net << "netscape_1.2
"+_self.drawing_help[.next_suggestion]+" &"
        pid << system.start_command(net)
    _endif
_endmethod
$

```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.suggest_calling_trail_menu()
    ##
    ##
    ##
    .parent.show_alert("You should turn the trail into box. Call
the Trail Constructions Menu from the Edit Menu. Or click on Help")
    ## self.review_status(_true)
_endmethod
$

```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.suggest_click_box_button()
    ##
    ##
    ##
    .parent.show_alert("To create an area simply click on the box
icon. Or Click on Help")
    ## self.review_status(_true)
_endmethod
$

```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.suggest_delete_geometry()
    ##
    ##
    ##
    .parent.show_alert("You should delete the existing geometry.
Click on Help")
    ## self.review_status(_true)
_endmethod
$

```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.suggest_create_trail()
    ##
    ##
    ##
    .parent.show_alert("You have defined the drawing_function. You
should create a trail. Click help")
    ## self.review_status(_true)
_endmethod
$

```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.suggest_insert_changing_trail()
    ##
    ##
    ##
    .parent.show_alert("The trail is not adequate for the chosen
drawing function. Draw another trail. There is help on the Agent
Controller.")
    ## self.review_status(_true)
_endmethod
$

```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.suggest_insert_drawing_function()
    ##
    ##
    ##
    .parent.show_alert("Click on the insert button at the drawing
function editor. There is help on the Agent Controller.")
    ## self.review_status(_true)
_endmethod
$

```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.suggest_setting_geometry()
    ##
    ##
    ##
    .parent.show_alert("Now you should link the geometry to the
drawing function. Click help")
    ##_self.review_status(_true)
_endmethod
$

```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.suggest_selecting_drawing_function()
    ##
    ##
    ##
    .parent.show_alert("Select the drawing function fro the choice
item on the drawing function editor. Click help")
    ##_self.review_status(_true)
_endmethod
$

```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.suggest_drawing_function()
    ##
    ##
    ##
    .parent.show_alert("Select the <Edit Drawing Functions> option
from the Edit Menu to create a drawing function. Click help")
    ##_self.review_status(_true)
_endmethod
$

```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.suggest_creating_geometry()
    ##
    ##
    ##
    .parent.show_alert("Open the drafting geometry editor by
clicking on the Geometry Attribute at the Drawing Function Editor.
Click help")
    ##_self.review_status(_true)
_endmethod
$

```

```

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.suggest_insert_drafting_geometry()
    ##
    ##
    ##
    .parent.show_alert("To insert geometry click on the insert
button at the Drafting Geometry Editor. Click help")
    ##_self.review_status(_true)
_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.perform_smallworld_logo_with_trail()
    ##
    ## executed when perform button was clicked and
    ## :performable_task is :smallworld_logo_with_trail
    ##
    _if .drawing_application _isnt _unset _andif
.drawing_application.sub_menus[:drawing_function_editor] _isnt
_unset
    _then
        dfe
<<.drawing_application.sub_menus[:drawing_function_editor]
        dfe.insert()
        current_record << dfe.current_record
        dfe.activate_geometry_editor()
        de << .drawing_application.sub_menus[:drafting_editor]
        _if .last_geometry_is _unset
        _then
            .drawing_application.trail_box()
        _endif
    ##
    ## This will make sense later on
    ##
    ##_self.review_status()
    .last_trail << .drawing_application.gtrail

    de.perform_transaction("Insert", :|insert()|, simple_vector.new(
20))
        current_geometry <<
.drawing_application.current_geometry
        .drawing_application.set_selection(current_geometry)
        dfe.set_function_geometry_list_index(1)
        dfe.changed(:|set_function_geometry_list_index()|, 1)
        dfe.set_geometry()
        .drawing_application.refresh()

    ##
    ##end task
    .last_trail << _unset
    .last_geometry << _unset
    .performable_task << _unset
    .next_suggestion << _unset
    .drawing_function << _unset
    .parent.disable_perform_button()
    .parent.disable_suggest_button()
    _self.review_status(_true)
_endif

```

```

_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.perform_viewport_with_trail()
  ##
  ## executed when perform button was clicked and
  ## :performable_task is :viewport_with_trail
  ##
  _if .drawing_application_isnt_unset_andif
.drawing_application.sub_menus[:drawing_function_editor] _isnt
_unset
  _then
    dfe
<<.drawing_application.sub_menus[:drawing_function_editor]
    dfe.insert()
    current_record << dfe.current_record
    dfe.activate_geometry_editor()
    de << .drawing_application.sub_menus[:drafting_editor]
    _if .last_geometry_is_unset
    _then
      .drawing_application.trail_box()
    _endif
    ##
    ## This will make sense later on
    ##
    ##_self.review_status()
    .last_trail << .drawing_application.gtrail

    de.perform_transaction("Insert", :|insert()|, simple_vector.new(
20))
    current_geometry <<
.drawing_application.current_geometry
    .drawing_application.set_selection(current_geometry)
    dfe.set_function_geometry_list_index(1)
    dfe.changed(:|set_function_geometry_list_index()|, 1)
    dfe.set_geometry()
    current_geometry <<
.drawing_application.current_geometry
    .drawing_application.refresh()
    .drawing_application.activate_viewport_mapping_editor()
    vme <<
.drawing_application.sub_menus[:viewport_mapping_editor]
    .drawing_application.changed(:select, current_geometry)
    vme.changed(.grs, :select, current_geometry)
    vme.sys!slot(:viewport_function) << dfe.current_record
    vme.changed()
    ##vme.int!by_window_bounds(.grs.main_view)
    vme.by_window_bounds()
    vme.apply()

    ##
    ##end task
    .last_trail << _unset
    .last_geometry << _unset
    .performable_task << _unset
    .next_suggestion << _unset
    .drawing_function << unset
    .parent.disable_perform_button()
    .parent.disable_suggest_button()

```

```
        _self.review_status(_true)
    _endif

_endmethod
$

_pragma(classify_level=restricted, topic={agent}, usage={external})
_method drafting_agent.end_task()

    ## before another event comes in ...
    .last_trail << _unset
    .last_geometry << _unset
    .performable_task << _unset
    .parent.disable_perform_button()
    .parent.disable_suggest_button()
    .next_suggestion << _unset
    _self.review_status()
_endmethod
$
```

Appendix B: SIFIA Code

```
Option Explicit
Private Declare Function GetTempPath Lib "kernel32" Alias
"GetTempPathA" _
    (ByVal nBufferLength As Long, ByVal lpBuffer As String) As Long

Dim g_tmpPath As String 'system folder for temp files
Dim g_counter As Long 'counter for emp file name generator
Dim g_backgroundSym As New Symbol 'to draw the ocean
Dim g_outlineSym As New Symbol 'to make a border around the
map
Dim rec As MapObjects.Recordset
Dim rec1 As MapObjects.Recordset
Dim rec2 As MapObjects.Recordset

im user, cmd, levels, scales, gcmd, glevels, gscales, lleft, lright,
lbottom
Dim ltop, sleft, sright, sbottom, sstop, gleft, gright, gbottom,
gtop
Dim gsleft, gsright, gsbottom, gstop
Dim sel, gsel, n_records_ret, n_request, gn_records_ret, sdate,
stime, number

'current last action for suggestion
Dim tcmd As String
Dim tsel As Boolean
Dim tleft As Double
Dim tright As Double
Dim tbottom As Double
Dim ttop As Double
Dim hoje As Date
Dim tsleft As Double
Dim tsright As Double
Dim tsbottom As Double
Dim tstop As Double
Dim max_date As Date

'predictor field and it maximum weight for memory restriction
Dim maxweight As Double
Dim pf As String
Dim lastsel As Boolean

Sub DrawSelection(recs As MapObjects.Recordset, color)
    'draw the features in a recordset
    Dim sym As New MapObjects.Symbol
    sym.SymbolType = moFillSymbol
    sym.Style = moSolidFill
    sym.color = color
    If Not recs Is Nothing Then
        Map.DrawShape recs, sym
    End If
End Sub
```

```

Private Sub DoSelect(ext As MapObjects.Rectangle, ByVal values As
Object, ByVal arguments As Object)
    Dim n_ext As New MapObjects.Rectangle
    Dim sel As New MapObjects.Rectangle

    Set sel = ExtractSelect(values, arguments)
    Set rec = Map.Layers("portugal").SearchShape(sel,
moCentroidInPolygon, "")
    Set rec1 = rec
    Map.Extent = ext
    Map.Refresh
    CreateMap ext

End Sub

Private Sub DoGetData(ext As MapObjects.Rectangle, ByVal values As
Object, ByVal arguments As Object)
    If values(arguments.Find("Sel")) = "true" Then
        Set sel = ExtractSelect(values, arguments)
        Set rec = Map.Layers("portugal").SearchShape(sel,
moCentroidInPolygon, "")
        Set rec1 = rec
        Map.Extent = ext
        Map.Refresh
        'RetrieveInfo ext, values, arguments
        WebLink.WriteResponseHeader "Content-type: text/plain" &
vbCrLf & vbCrLf
        WebLink.WriteString "Data retrieved to you local directory"
    Else
        Set rec = Nothing
        Set rec1 = Nothing
        WebLink.WriteResponseHeader "Content-type: text/plain" &
vbCrLf & vbCrLf
        WebLink.WriteString "No selection to retrieve"
    End If

End Sub

Private Sub DoUnselect(ext As MapObjects.Rectangle, ByVal values As
Object, ByVal arguments As Object)

    Set rec = Nothing
    Set rec1 = Nothing
    Map.Extent = ext
    Map.Refresh
    CreateMap ext

End Sub

Private Sub DoGif(ext As MapObjects.Rectangle, ByVal values As
Object, ByVal arguments As Object)
    Dim n_ext As New MapObjects.Rectangle
    Dim sel As New MapObjects.Rectangle

    If values(arguments.Find("Sel")) = "true" Then
        Set sel = ExtractSelect(values, arguments)
        Set rec = Map.Layers("portugal").SearchShape(sel,
moCentroidInPolygon, "")
        Set rec1 = rec
    Else

```



```

        Set rec = Nothing
        Set rec1 = Nothing

    End If

    Map.Extent = ext
    Map.Refresh
    CreateMap ext

End Sub

Private Sub DoEnl(ext As MapObjects.Rectangle, ByVal values As
Object, ByVal arguments As Object)
    Dim n_ext As New MapObjects.Rectangle
    Dim sel As New MapObjects.Rectangle

    If values(arguments.Find("Sel")) = "true" Then
        Set sel = ExtractSelect(values, arguments)
        sel.ScaleRectangle 1.5
        Set rec = Map.Layers("portugal").SearchShape(sel,
moCentroidInPolygon, "")
        Set rec1 = rec
    Else
        Set rec = Nothing
        Set rec1 = Nothing

    End If

    Map.Extent = ext
    Map.Refresh
    CreateMap ext

End Sub

Private Sub DoConst(ext As MapObjects.Rectangle, ByVal values As
Object, ByVal arguments As Object)
    Dim n_ext As New MapObjects.Rectangle
    Dim sel As New MapObjects.Rectangle

    If values(arguments.Find("Sel")) = "true" Then
        Set sel = ExtractSelect(values, arguments)
        sel.ScaleRectangle 0.5
        Set rec = Map.Layers("portugal").SearchShape(sel,
moCentroidInPolygon, "")
        Set rec1 = rec
    Else
        Set rec = Nothing
        Set rec1 = Nothing

    End If

    Map.Extent = ext
    Map.Refresh
    CreateMap ext

End Sub

Private Sub ChangeExtent(ext As MapObjects.Rectangle)
    'Set Map.Extent.Left = ext.Left
    'Set Map.Extent.Bottom = ext.Bottom
    'Set Map.Extent.Right = ext.Right

```

```

        'Set Map.Extent.Top = ext.Top
        Set Map.Extent = ext
End Sub
Private Sub CreateError(message As String)
    'specify what type of data we are going to send
    WebLink.WriteResponseHeader "Content-type: text/plain" & vbCrLf
    & vbCrLf
    WebLink.WriteString "Tutorial ERROR: " & message
End Sub

Private Sub CreateHTML(ext As MapObjects.Rectangle)
    'specify what type of data we are going to send
    WebLink.WriteResponseHeader "Content-type: text/html" & vbCrLf &
    vbCrLf

    'write a bunch of prettyr header stuff
    WebLink.WriteString "<HTML> <HEAD> <BODY BGCOLOR=""#ffffff"">" &
    vbCrLf
    WebLink.WriteString "<TITLE>Tutorial </TITLE> </HEAD>" & vbCrLf
    WebLink.WriteString "<BODY><H2>Tutorial</H2><P>" & vbCrLf
    WebLink.WriteString "This Map was created by the MapObjects " &
    -
    "Internet Map Server. <P>" & vbCrLf

    'create an HTML form
    WebLink.WriteString "<FORM ACTION=""esrimap.dll"">" _
    & vbCrLf
    'put the context information in as hidden variables
    WebLink.WriteString "<INPUT TYPE=""hidden"" NAME=""name""
Value=""Tutorial"">" _
    & vbCrLf
    WebLink.WriteString "<INPUT TYPE=""hidden"" NAME=""Cmd""
Value=""ZoomIn"">" _
    & vbCrLf
    WebLink.WriteString "<INPUT TYPE=""hidden"" NAME=""Left""
Value=""& _
    ext.Left & "">" & vbCrLf
    WebLink.WriteString "<INPUT TYPE=""hidden"" NAME=""Bottom""
Value=""& _
    ext.Bottom & "">" & vbCrLf
    WebLink.WriteString "<INPUT TYPE=""hidden"" NAME=""Right""
Value=""& _
    ext.Right & "">" & vbCrLf
    WebLink.WriteString "<INPUT TYPE=""hidden"" NAME=""Top""
Value=""& _
    ext.Top & "">" & vbCrLf

    'write the reference to the image
    Dim imgURL As String
    imgURL = "esrimap.dll?name=Tutorial&Cmd=Gif"
    imgURL = imgURL & "&Left=" & ext.Left & "&Bottom=" & ext.Bottom
    & _
    "&Right=" & ext.Right & "&Top=" & ext.Top
    WebLink.WriteString "<CENTER><INPUT TYPE=image NAME=click
SRC=""& imgURL _
    & ""><BR>"

    'write a hyperlink to the full extent

```

```

        WebLink.WriteString "<P><P><A HREF=""esrimap.dll?" & _
                               "name=Tutorial&Cmd=Map"" >Return to Full
Extent</A>"
        WebLink.WriteString "</CENTER></FORM></BODY></HTML>"

```

```
End Sub
```

```

Private Sub CreateMap(ext As MapObjects.Rectangle)
    'generate temporary filenames for the bmp and gif files

    Dim bmpFile As String, gifFile As String
    bmpFile = g_tmpPath & WebLink.MapPort & "MO" & g_counter &
".bmp"
    gifFile = g_tmpPath & WebLink.MapPort & "MO" & g_counter &
".gif"

    'specify what type of data we are going to send
    WebLink.WriteResponseHeader "Content-type: image/gif" & vbCrLf &
vbCrLf

    'Set Map.Extent = ext                'set the extent of the
map
    Set Map.Extent = ext
    Map.Refresh
    'ChangeExtent (ext)
    Map.ExportMap moExportBMP, bmpFile, 1 'create a BMP file

    WebLink.Bmp2Gif bmpFile, 1 'convert to gif
    WebLink.WriteFile gifFile 'send the GIF file to the client

    'clean up
    Kill gifFile
    Kill bmpFile
    g_counter = g_counter + 1 'increment the counter for the next
request
End Sub

```

```

Private Function ExtractExtent(values As Object, arguments As
Object) As MapObjects.Rectangle
    On Error GoTo ErrorExit:

    'extract the extent from the argument values if it'd there
    If values.Count > 3 Then
        Dim ext As New MapObjects.Rectangle
        Dim inteiro As String
        Dim d As Double
        ext.Left = Val(values(arguments.Find("left")))
        ext.Bottom = Val(values(arguments.Find("bottom")))
        ext.Right = Val(values(arguments.Find("right")))
        ext.Top = Val(values(arguments.Find("top")))
        Set ExtractExtent = ext
    Else
        'use the full extent by default
        Set ExtractExtent = Map.FullExtent
    End If

    'Normal exit
    Exit Function

```

```

        'Error Exit
ErrorExit:
    Set ExtractExtent = Map.FullExtent
End Function

Private Function ExtractSelect(values As Object, arguments As
Object) As MapObjects.Rectangle
    On Error GoTo ErrorExit:

        'extract the extent from the argument values if it'd there
    If values.Count > 3 Then
        Dim ext As New MapObjects.Rectangle
        Dim inteiro As String
        Dim d As Double
        ext.Left = Val(values(arguments.Find("sleft")))
        ext.Bottom = Val(values(arguments.Find("sbottom")))
        ext.Right = Val(values(arguments.Find("sright")))
        ext.Top = Val(values(arguments.Find("stop")))
        Set ExtractSelect = ext
    Else
        'use the full extent by default
        Set ExtractSelect = Map.FullExtent
    End If

    'Normal exit
    Exit Function

    'Error Exit
ErrorExit:
    Set ExtractSelect = Map.FullExtent
End Function

Private Sub DoZoomIn(ext As MapObjects.Rectangle, x As Long, y As
Long)
    Dim pt As MapObjects.Point
    Set pt = ConvertClick(ext, x, y)

    'shrink the extent and center it on the click location
    Dim ctr As MapObjects.Point
    Set ctr = ext.Center
    ext.ScaleRectangle 0.5
    ext.Offset pt.x - ctr.x, pt.y - ctr.y

    CreateHTML ext
End Sub

Private Sub DoSuggest(ext As MapObjects.Rectangle, ByVal values As
Object, ByVal arguments As Object)

    texto.text = CalculateNextAction(ext, values, arguments)
    'WebLink.WriteResponseHeader "Content-type: text/plain" & vbCrLf
& vbCrLf
    'WebLink.WriteString texto.Text
End Sub

Private Function ConvertClick(ext As MapObjects.Rectangle, x As
Long, y As Long) _
    As MapObjects.Point
    Set Map.Extent = ext 'update the extent

```

```

        'convert the click location to control coordinates
        x = ScaleX(x, vbPixels, vbTwips)
        y = ScaleY(y, vbPixels, vbTwips)

        'convert the control coordinates to map coordinates
        Set ConvertClick = Map.ToMapPoint(x, y)

End Function

Private Sub DoDist(ext As MapObjects.Rectangle, sel As Rectangle)
Dim dist1 As MapObjects.Recordset
Set dist1 = Map.Layers("portugal").SearchByDistance(sel, 10000, "")
Set rec = dist1
Set Map.Extent = ext
End Sub

Private Sub Form_Load()

    lastsel = False
    Initialize_everything True, True
    WebLink.Start

    'Do not allow zooms below an extent width of 3.4 longitudinal
degrees
    Map.MinWidth = 3.4

    'setup the g_tmpPath variable for temp files
    Dim returnLen As Integer
    g_tmpPath = String(255, 0)
    returnLen = GetTempPath(Len(g_tmpPath), g_tmpPath)
    g_tmpPath = Left(g_tmpPath, returnLen)

    'initialize the symbols
    g_backgroundSym.color = 16761220    'and ocean blue
    g_outlineSym.Style = moTransparentFill

    'setup the label renderer for the freguesias
    Dim labelRndr As New LabelRenderer
    labelRndr.Field = "FREGUESIA"
    labelRndr.Symbol(0).Height = 1000
    Set Map.Layers("Admin2").Renderer = labelRndr

    'Initialize variables for learning
    Initialize_experience

End Sub

Private Sub Form_Unload(Cancel As Integer)
    WebLink.Stop
End Sub

Private Sub Map_AfterLayerDraw(ByVal index As Integer, ByVal
canceled As Boolean, ByVal hDC As Stdole.OLE_HANDLE)
    Call DrawSelection(rec, moYellow)
End Sub

```

```

Private Sub Map_AfterTrackingLayerDraw(ByVal hDC As
Stdole.OLE_HANDLE)
    'draw an outline around the map
    Map.DrawShape Map.Extent, g_outlineSym

End Sub

Private Sub Map_BeforeLayerDraw(ByVal index As Integer, ByVal hDC As
Stdole.OLE_HANDLE)
    If index = Map.Layers.Count - 1 Then
        Dim ext As MapObjects.Rectangle
        Set ext = Map.Extent
        'draw the map bacground before the first layer is drawn
        Map.DrawShape Map.Extent, g_backgroundSym

        'set the visibility of the map layers based on the width of
the
        'current extent
        Dim width As Double
        width = ext.width

        Map.Layers(0).Visible = width <= 100000    'freguesias
        'Map.Layers(1).Visible = width <= 5        'concelhos

        'make the country layer invisible if the current extent
        'is completely contained by the extent of the states layer
        'Dim stateExt As MapObjects.Rectangle
        'Set stateExt = Map.Layers(2).Extent

        'Map.Layers(4).Visible = (ext.Left < stateExt.Left) Or _
                                (ext.Right > stateExt.Right) Or _
                                (ext.Bottom < stateExt.Bottom) Or _
                                (ext.Top > stateExt.Top)

    End If
End Sub

Private Sub WebLink_Request(ByVal arguments As Object, ByVal values
As Object)

    Dim Teste As Boolean
    Dim ext As MapObjects.Rectangle
    Set ext = ExtractExtent(values, arguments)

    Teste = False
    'if there are any missing parameters, exit this sub.
    On Error GoTo ErrorExit:

    If values(arguments.Find("Cmd")) = "Map" Then
        CreateHTML ext
    ElseIf values(arguments.Find("Cmd")) = "Gif" Then
        Teste = TesteGif(values, arguments)
        DoGif ext, values, arguments
    ElseIf values(arguments.Find("Cmd")) = "ZoomIn" Then
        DoZoomIn ext, values(arguments.Find("click.x")), _
            values(arguments.Find("click.y"))
    ElseIf values(arguments.Find("Cmd")) = "Sel" Then
        DoSelect ext, values, arguments
    ElseIf values(arguments.Find("Cmd")) = "Enl" Then
        DoEnl ext, values, arguments
    ElseIf values(arguments.Find("Cmd")) = "Const" Then

```

```

        DoConst ext, values, arguments
    ElseIf values(arguments.Find("Cmd")) = "Retr" Then
        DoGetData ext, values, arguments
    ElseIf values(arguments.Find("Cmd")) = "Sug" Then
        DoSuggest ext, values, arguments
    Else
        CreateError "Invalid Cmd value."
    End If

    If Not Teste And Not (values(arguments.Find("Cmd")) = "Sug") And
Not wasLastSel(values(arguments.Find("Cmd"))) Then
        AddNewX arguments, values
        Teste = False
    End If

    ' Normal exit point.
    Exit Sub

    'Abnormal exit point
ErrorExit:
        'CreateError "Invalid argument."
        Debug.Print Error(Err.number) ' Print error to Debug
window.

End Sub

Function wasLastSel(cmd As String) As Boolean

    If Not lastsel Then
        If gcmd = "Sel" And cmd = "Gif" Then
            lastsel = True
            wasLastSel = True
        Else
            wasLastSel = False
        End If
    Else
        lastsel = False
        wasLastSel = False
    End If

End Function

Sub Initialize_everything(predictors As Boolean, goals As Boolean)
    If predictors Then
        sdate = Null
        stime = Null
        user = Null
        cmd = Null
        lleft = Null
        lright = Null
        lbottom = Null
        ltop = Null
        sel = Null
        sleft = Null
        sright = Null
        sbottom = Null
        sstop = Null
        levels = Null
        scales = Null
    End If
End Sub

```

```

    n_records_ret = Null
    n_request = Null

End If

If goals Then
    gcmd = Null
    gleft = Null
    gright = Null
    gbottom = Null
    gtop = Null
    gsel = Null
    gsleft = Null
    gsright = Null
    gsbottom = Null
    gstop = Null
    glevels = Null
    gscale = Null
    gn_records_ret = Null
    number = Null
End If
End Sub

Function TesteGif(ByVal values As Object, ByVal arguments As Object)
As Boolean

    If (values(arguments.Find("Cmd")) = "Gif") And _
        (values(arguments.Find("left")) = gleft) And _
        (values(arguments.Find("right")) = gright) And _
        (values(arguments.Find("top")) = gtop) And _
        (values(arguments.Find("bottom")) = gbottom) Then

        If (arguments.Find("Sel") <> -1) Then

            If (values(arguments.Find("Sel")) = "true") Then

                TesteGif = (Val(values(arguments.Find("sleft")))
= gsleft) And _
                (values(arguments.Find("sright")) = gsright) And
                (values(arguments.Find("stop")) = gstop) And _
                (values(arguments.Find("sbottom")) = gsbottom)
            Else
                TesteGif = True
            End If
        Else
            TesteGif = True
        End If
    Else
        TesteGif = False
    End If
End Function

Sub GetDBData(ByVal arguments As Object, ByVal values As Object)
'falta tratar gn_records_ret

If IsNull(gcmd) Then
    user = "user1"
    sdate = Date
    stime = Time
    number = values(arguments.Find("number"))

```



```

If arguments.Find("Cmd") <> -1 Then
    gcmd = values(arguments.Find("Cmd"))
End If
If arguments.Find("left") <> -1 Then
    gleft = values(arguments.Find("left"))
End If
If arguments.Find("right") <> -1 Then
    gright = values(arguments.Find("right"))
End If
If arguments.Find("bottom") <> -1 Then
    gbottom = values(arguments.Find("bottom"))
End If
If arguments.Find("top") <> -1 Then
    gtop = values(arguments.Find("top"))
End If
If arguments.Find("Sel") <> -1 Then
    If (values(arguments.Find("Sel"))) = "true" Then
        gsel = True
    Else
        gsel = False
    End If
End If
If arguments.Find("sleft") <> -1 Then
    gsleft = values(arguments.Find("sleft"))
End If
If arguments.Find("sright") <> -1 Then
    gsright = values(arguments.Find("sright"))
End If
If arguments.Find("sbottom") <> -1 Then
    gsbottom = values(arguments.Find("sbottom"))
End If
If arguments.Find("stop") <> -1 Then
    gstop = values(arguments.Find("stop"))
End If
If arguments.Find("levels") <> -1 Then
    glevels = values(arguments.Find("levels"))
End If
If arguments.Find("scales") <> -1 Then
    gscales = values(arguments.Find("scales"))
End If

Else
    Initialize_everything True, False
    user = "user1"
    sdate = Date
    stime = Time
    cmd = gcmd
    lleft = gleft
    lright = gright
    lbottom = gbottom
    ltop = gtop
    sel = gsel
    sleft = gsleft
    sright = gsright
    sbottom = gsbottom
    sstop = gstop
    levels = glevels
    scales = gscales
    n_records_ret = gn_records_ret

    Initialize_everything False, True

```

```

    If arguments.Find("number") <> -1 Then
        n_request = values(arguments.Find("number"))
    End If
    If arguments.Find("Cmd") <> -1 Then
        gcmd = values(arguments.Find("Cmd"))
    End If
    If arguments.Find("left") <> -1 Then
        gleft = values(arguments.Find("left"))
    End If
    If arguments.Find("right") <> -1 Then
        gright = values(arguments.Find("right"))
    End If
    If arguments.Find("bottom") <> -1 Then
        gbottom = values(arguments.Find("bottom"))
    End If
    If arguments.Find("top") <> -1 Then
        gtop = values(arguments.Find("top"))
    End If
    If arguments.Find("Sel") <> -1 Then
        If (values(arguments.Find("Sel"))) = "true" Then
            gsel = True
        Else
            gsel = False
        End If
    End If
    If arguments.Find("sleft") <> -1 Then
        gsleft = values(arguments.Find("sleft"))
    End If
    If arguments.Find("sright") <> -1 Then
        gsright = values(arguments.Find("sright"))
    End If
    If arguments.Find("sbottom") <> -1 Then
        gsbottom = values(arguments.Find("sbottom"))
    End If
    If arguments.Find("stop") <> -1 Then
        gstop = values(arguments.Find("stop"))
    End If
    If arguments.Find("levels") <> -1 Then
        glevels = values(arguments.Find("levels"))
    End If
    If arguments.Find("scales") <> -1 Then
        gscales = values(arguments.Find("scales"))
    End If
End If
End Sub

Function IsDBData(ByVal arguments As Object, ByVal values As Object)
As Boolean

    Dim cgleft As Double
    Dim cgright As Double
    Dim cgbottom As Double
    Dim cgtop As Double
    Dim ccmd As String
    Dim cgsel
    Dim cgsleft As Double
    Dim cgsright As Double
    Dim cgbottom As Double
    Dim cgstop As Double

    Dim g As New MapObjects.Rectangle

```

```

Dim cg As New MapObjects.Rectangle
Dim gs As New MapObjects.Rectangle
Dim cgs As New MapObjects.Rectangle

g.Left = gleft
g.Right = gright
g.Bottom = gbottom
g.Top = gtop
If gsel Then
    gs.Left = gsleft
    gs.Right = gsright
    gs.Bottom = gs.Bottom
    gs.Top = gstop
Else
End If

    If arguments.Find("Cmd") <> -1 Then
        cgcmd = values(arguments.Find("Cmd"))
    End If

    If arguments.Find("left") <> -1 Then
        cgleft = values(arguments.Find("left"))
    End If
    If arguments.Find("right") <> -1 Then
        cgright = values(arguments.Find("right"))
    End If
    If arguments.Find("bottom") <> -1 Then
        cgbottom = values(arguments.Find("bottom"))
    End If
    If arguments.Find("top") <> -1 Then
        cgtop = values(arguments.Find("top"))
    End If

cg.Left = cgleft
cg.Right = cgright
cg.Bottom = cgbottom
cg.Top = cgtop

If arguments.Find("Sel") <> -1 Then
    If (values(arguments.Find("Sel")) = "true") Then
        cgssel = True
    Else
        cgssel = False
    End If
End If
If arguments.Find("sleft") <> -1 Then
    cgssleft = values(arguments.Find("sleft"))
End If
If arguments.Find("sright") <> -1 Then
    cgssright = values(arguments.Find("sright"))
End If
If arguments.Find("sbottom") <> -1 Then
    cgssbottom = values(arguments.Find("sbottom"))
End If
If arguments.Find("stop") <> -1 Then
    cgstop = values(arguments.Find("stop"))
End If

    If cgssel Then
        cgs.Left = cgssleft
        cgs.Right = cgssright

```

```

        cgs.Bottom = cgsbottom
        cgs.Top = cgstop
    Else
    End If

If (gcmd = "Sel" Or gcmd = "Gif") And cgcmd = "Gif" Then

    If really_equal_extents(g, cg) Then
        If cgssel Then
            If really_equal_extents(gs, cgs) Then
                IsDBData = False
            Else
                IsDBData = True
            End If
        Else
            If Not gsel Then
                IsDBData = False
            Else
                IsDBData = True
            End If
        End If
    Else
        IsDBData = True
    End If
Else
    IsDBData = True
End If

End Function

Sub AddNewX(ByVal arguments As Object, ByVal values As Object)

    'Dim dbsExperience As Database
    'Dim rstExp As Recordset
    'Dim t As Variant
    'Dim td As TableDef

    'Set dbsExperience = DBEngine.Workspaces(0).OpenDatabase _
        ("C:\Projects\Webagent\experience\experience.mdb")
    'Set td = dbsExperience.TableDefs(0)
    'Set rstExp = td.OpenRecordset()
    'Set rstExp = dbsExperience.OpenRecordset("MBR", dbOpenDynamic)
    'Set rstExp = dbsExperience.OpenRecordset( _
        "SELECT * FROM MBR", dbOpenDynamic)

    'Dim myDB As Database, myRS As Recordset
    ' set the database to BIBLIO.MDB
    'Set myDB = DBEngine.Workspaces(0).OpenDatabase(sDir &
"\biblio.mdb")
    ' Set the recordset to Publishers table.
    'Set myRS = myDB.OpenRecordset("Publishers", dbOpenDynaset)

    'be careful about first experience

    GetDBData arguments, values

```

```

If Not IsNull(cmd) Then

    AddExperience
End If
'rstExp.Close
'dbsExperience.Close

End Sub

Function AddExperience()

' Adds a new record to a Recordset using the data passed
' by the calling procedure. The new record is then made
' the current record.
With MBR.Recordset
    .AddNew
    !user = user
    !cmd = cmd
    If Not IsNull(lleft) Then
        !Left = Val(lleft)
    Else
        !Left = Null
    End If
    If Not IsNull(lright) Then
        !Right = Val(lright)
    Else
        !Right = Null
    End If
    If Not IsNull(lbottom) Then
        !Bottom = Val(lbottom)
    Else
        !Bottom = Null
    End If
    If Not IsNull(ltop) Then
        !Top = Val(ltop)
    Else
        !Top = Null
    End If
    !sel = sel
    If Not IsNull(sleft) Then
        !sleft = Val(sleft)
    Else
        !sleft = Null
    End If
    If Not IsNull(srigh) Then
        !srigh = Val(srigh)
    Else
        !srigh = Null
    End If
    If Not IsNull(sbottom) Then
        !sbottom = Val(sbottom)
    Else
        !sbottom = Null
    End If
    If Not IsNull(sstop) Then
        !Stop = Val(sstop)
    Else
        !Stop = Null
    End If
    !levels = levels
    !scales = scales

```

```

!n_records_ret = n_records_ret
!n_request = n_request
!gcmd = gcmd
If Not IsNull(gleft) Then
    !gleft = Val(gleft)
Else
    !gleft = Null
End If
If Not IsNull(gright) Then
    !gright = Val(gright)
Else
    !gright = Null
End If
If Not IsNull(gbottom) Then
    !gbottom = Val(gbottom)
Else
    !gbottom = Null
End If
If Not IsNull(gtop) Then
    !gtop = Val(gtop)
Else
    !gtop = Null
End If
!gsel = gsel
If Not IsNull(gsleft) Then
    !gsleft = Val(gsleft)
Else
    !gsleft = Null
End If
If Not IsNull(gsrigh) Then
    !gsrigh = Val(gsrigh)
Else
    !gsrigh = Null
End If
If Not IsNull(gstop) Then
    !gstop = Val(gstop)
Else
    !gstop = Null
End If
If Not IsNull(gsbottom) Then
    !gsbottom = Val(gsbottom)
Else
    !gsbottom = Null
End If
!glevels = glevels
!gscales = gscales
!gn_records_ret = gn_records_ret
!Date = sdate
!Time = stime
!number = n_request
.Update
.Bookmark = .LastModified
End With

```

End Function

Sub Initialize_experience()

```

cmd = Null
gcmd = Null

```

End Sub

Private Sub GetPreviousAction(ext As MapObjects.Rectangle, ByVal values As Object, ByVal arguments As Object)

```
Dim qd As QueryDef
Dim db As Database
Dim rs As Recordset
Dim dt As Date
Dim tm As Date
```

'get the previous action to consider, (previous goal fields, or state in the current request)

```
datac.RecordSource = "Q_maxdate"
datac.Refresh
dt = datac.Recordset!max_date
max_date = dt
hoje = Date
If dt = hoje Then
```

```
Set db =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
Set qd = db.CreateQueryDef("Q_maxtime", "select max(time) as
max_time from MBR where str(date)='" & dt & "'")
datac.RecordSource = "Q_maxtime"
datac.Refresh
tm = datac.Recordset!max_time
qd.Close
db.QueryDefs.Delete ("Q_maxtime")
Set qd = db.CreateQueryDef("Q_lastrecord", "select * from
MBR where str(date)='" & dt & "' and str(time)='" & tm & "'")
datac.RecordSource = "Q_lastrecord"
datac.Refresh
```

```
With datac.Recordset
tcmd = !gcmd
tleft = !gleft
tright = !gright
tbottom = !gbottom
ttop = !gtop
tsel = !gsel
If tsel Then
tsleft = !gsleft
tsright = !gsright
tsbottom = !gsbottom
tstop = !gstop
Else
'tsleft = Null
'tsright = Null
'tsbottom = Null
'tstop = Null
End If
End With
```

```
qd.Close
db.QueryDefs.Delete ("Q_lastrecord")
```

Else

```

    tcmd = values(arguments.Find("Cmd"))
    If (values(arguments.Find("Sel")) = "true") Then
        tsel = True
    Else
        tsel = False
    End If
    tleft = values(arguments.Find("left"))
    tright = values(arguments.Find("right"))
    tbottom = values(arguments.Find("bottom"))
    ttop = values(arguments.Find("top"))
    If tsel Then
        tsleft = values(arguments.Find("sleft"))
        tsright = values(arguments.Find("sright"))
        tsbottom = values(arguments.Find("sbottom"))
        tstop = values(arguments.Find("stop"))
    Else
        'tsleft = Null
        'tsright = Null
        'tsbottom = Null
        'tstop = Null
    End If
End If

End Sub

Private Sub GetHighestPFfield()

    Dim db As Database
    Dim qd As QueryDef

    If tcmd <> "Sug" Then
        pf = "cmd"

        Set db =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
        Set qd = db.CreateQueryDef("Q_weight", "select weight from
w_cmd_gcmd where cmd= '" & tcmd & "'")
        datac.RecordSource = "Q_weight"
        datac.Refresh
        maxweight = datac.Recordset!Weight
        qd.Close
        db.QueryDefs.Delete ("Q_weight")

        Set qd = db.CreateQueryDef("Q_weight", "select weight from
w_cmd_gcr where cmd= '" & tcmd & "'")
        datac.RecordSource = "Q_weight"
        datac.Refresh
        If datac.Recordset!Weight > maxweight Then
            maxweight = datac.Recordset!Weight
        Else
            End If
        qd.Close
        db.QueryDefs.Delete ("Q_weight")

        Set qd = db.CreateQueryDef("Q_weight", "select weight from
w_cmd_gsel where cmd= '" & tcmd & "'")
        datac.RecordSource = "Q_weight"
        datac.Refresh
        If datac.Recordset!Weight > maxweight Then
            maxweight = datac.Recordset!Weight
        Else

```



```

End If
qd.Close
db.QueryDefs.Delete ("Q_weight")

If tsel Then

    Set qd = db.CreateQueryDef("Q_weight", "select weight
from w_cmd_gscr where cmd= '" & tcmd & "'")
    datac.RecordSource = "Q_weight"
    datac.Refresh
    If datac.Recordset!Weight > maxweight Then
        maxweight = datac.Recordset!Weight
    Else
        End If
    qd.Close
    db.QueryDefs.Delete ("Q_weight")
Else
    End If
db.Close

Else
End If

Dim s As String

Set db =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
s = "select weight from w_cr_gcmd where left - (" & Str(tleft) & "
") < 1 and right - (" & Str(tright) & ") < 1 and bottom - (" &
Str(tbottom) & ") < 1 and top - (" & Str(ttop) & ") < 1 and weight >
(" & Str(maxweight) & " )"
Set qd = db.CreateQueryDef("Q_weight", "select left, right,
bottom, top, weight from w_cr_gcmd where left - (" & Str(tleft) & ")
< 1 and right - (" & Str(tright) & ") < 1 and bottom - (" &
Str(tbottom) & ") < 1 and top - (" & Str(ttop) & ") < 1 and weight >
(" & Str(maxweight) & " )"
    datac.RecordSource = "Q_weight"
    datac.Refresh
    GetcrMaxweight
    qd.Close
    db.QueryDefs.Delete ("Q_weight")

Set qd = db.CreateQueryDef("Q_weight", "select left, right,
bottom, top, weight from w_cr_gcr where left - (" & Str(tleft) & ") <
1 and right - (" & Str(tright) & ") < 1 and bottom - (" &
Str(tbottom) & ") < 1 and top - (" & Str(ttop) & ") < 1 and weight >
(" & Str(maxweight) & " )"
    datac.RecordSource = "Q_weight"
    datac.Refresh
    GetcrMaxweight
    qd.Close
    db.QueryDefs.Delete ("Q_weight")

Set qd = db.CreateQueryDef("Q_weight", "select left, right,
bottom, top, weight from w_cr_gsel where left - (" & Str(tleft) & ")
< 1 and right - (" & Str(tright) & ") < 1 and bottom - (" &
Str(tbottom) & ") < 1 and top - (" & Str(ttop) & ") < 1 and weight >
(" & Str(maxweight) & " )"
    datac.RecordSource = "Q_weight"
    datac.Refresh
    GetcrMaxweight
    qd.Close

```

```

db.QueryDefs.Delete ("Q_weight")

If tsel Then

    Set qd = db.CreateQueryDef("Q_weight", "select left, right,
bottom, top,weight from w_cr_gscr where left - (" & Str(tleft) & ")
< 1 and right - (" & Str(tright) & ") < 1 and bottom - (" &
Str(tbottom) & ") < 1 and top - (" & Str(ttop) & ") < 1 and weight >
(" & Str(maxweight) & " )")
    datac.RecordSource = "Q_weight"
    datac.Refresh
    GetcrMaxweight
    qd.Close
    db.QueryDefs.Delete ("Q_weight")
Else
End If

Dim textsel As String
If tsel Then
    textsel = "Yes"
Else
    textsel = "No"
End If

Set qd = db.CreateQueryDef("Q_weight", "select weight from
w_sel_gcmd where sel=" & textsel)
datac.RecordSource = "Q_weight"
datac.Refresh
If datac.Recordset!Weight > maxweight Then
    maxweight = datac.Recordset!Weight
    pf = "sel"
Else
End If
qd.Close
db.QueryDefs.Delete ("Q_weight")

Set qd = db.CreateQueryDef("Q_weight", "select weight from
w_sel_gcr where sel=" & textsel)
datac.RecordSource = "Q_weight"
datac.Refresh
If datac.Recordset!Weight > maxweight Then
    pf = "sel"
    maxweight = datac.Recordset!Weight
Else
End If
qd.Close
db.QueryDefs.Delete ("Q_weight")

Set qd = db.CreateQueryDef("Q_weight", "select weight from
w_sel_gsel where sel=" & textsel)
datac.RecordSource = "Q_weight"
datac.Refresh
If datac.Recordset!Weight > maxweight Then
    pf = "sel"
    maxweight = datac.Recordset!Weight
Else
End If
qd.Close
db.QueryDefs.Delete ("Q_weight")

```

```

If tsel Then

    Set qd = db.CreateQueryDef("Q_weight", "select weight from
w_sel_gscr where sel=" & textsel)
    datac.RecordSource = "Q_weight"
    datac.Refresh
    If datac.Recordset!Weight > maxweight Then
        pf = "sel"
        maxweight = datac.Recordset!Weight
    Else
    End If
    qd.Close
    db.QueryDefs.Delete ("Q_weight")
Else
End If

    Set qd = db.CreateQueryDef("Q_weight", "select sleft, sright,
sbottom, stop,weight from w_scr_gcmd where sleft - (" & Str(tsleft)
& ") < 1 and sright - (" & Str(tsright) & ") < 1 and sbottom - (" &
Str(sbottom) & ") < 1 and stop - (" & Str(tstop) & ") < 1 and
weight > (" & Str(maxweight) & " )")
    datac.RecordSource = "Q_weight"
    datac.Refresh
    GetscrMaxweight
    qd.Close
    db.QueryDefs.Delete ("Q_weight")

    Set qd = db.CreateQueryDef("Q_weight", "select sleft, sright,
sbottom, stop,weight from w_scr_gcr where sleft - (" & Str(tsleft) &
") < 1 and sright - (" & Str(tsright) & ") < 1 and sbottom - (" &
Str(sbottom) & ") < 1 and stop - (" & Str(tstop) & ") < 1 and
weight > (" & Str(maxweight) & " )")
    datac.RecordSource = "Q_weight"
    datac.Refresh

    GetscrMaxweight
    qd.Close
    db.QueryDefs.Delete ("Q_weight")

    Set qd = db.CreateQueryDef("Q_weight", "select sleft, sright,
sbottom, stop,weight from w_scr_gsel where sleft - (" & Str(tsleft)
& ") < 1 and sright - (" & Str(tsright) & ") < 1 and sbottom - (" &
Str(sbottom) & ") < 1 and stop - (" & Str(tstop) & ") < 1 and
weight > (" & Str(maxweight) & " )")
    datac.RecordSource = "Q_weight"
    datac.Refresh
    GetscrMaxweight
    qd.Close
    db.QueryDefs.Delete ("Q_weight")

If tsel Then

    Set qd = db.CreateQueryDef("Q_weight", "select sleft,
sright, sbottom, stop,weight from w_scr_gscr where sleft - (" &
Str(tsleft) & ") < 1 and sright - (" & Str(tsright) & ") < 1 and
sbottom - (" & Str(sbottom) & ") < 1 and stop - (" & Str(tstop) &
") < 1 and weight > (" & Str(maxweight) & " )")
    datac.RecordSource = "Q_weight"
    datac.Refresh
    GetscrMaxweight
    qd.Close

```

```

        db.QueryDefs.Delete ("Q_weight")
    Else
    End If
    db.Close

End Sub

Private Sub GetcrMaxweight()

    Dim curr_req_ext As New MapObjects.Rectangle
    Dim weight_ext As New MapObjects.Rectangle

    curr_req_ext.Left = tleft
    curr_req_ext.Right = tright
    curr_req_ext.Bottom = tbottom
    curr_req_ext.Top = ttop

    With dataac.Recordset
        If Not .EOF Then
            .MoveFirst
        Else
        End If
        While Not .EOF
            If !Weight > maxweight Then

                weight_ext.Left = !Left
                weight_ext.Right = !Right
                weight_ext.Bottom = !Bottom
                weight_ext.Top = !Top

                If equal_extents(curr_req_ext, weight_ext) Then
                    maxweight = !Weight
                    pf = "cr"
                Else
                End If
            Else
            End If
            .MoveNext
        Wend
    End With

End Sub

Private Sub GetscrMaxweight()

    Dim curr_req_ext As New MapObjects.Rectangle
    Dim weight_ext As New MapObjects.Rectangle

    curr_req_ext.Left = tsleft
    curr_req_ext.Right = tsright
    curr_req_ext.Bottom = tsbottom
    curr_req_ext.Top = tstop

    With dataac.Recordset
        If Not .EOF Then
            .MoveFirst
        Else
        End If
        While Not .EOF

```

```

        If !Weight > maxweight Then

            weight_ext.Left = !sleft
            weight_ext.Right = !sright
            weight_ext.Bottom = !sbottom
            weight_ext.Top = !Stop

            If equal_extents(curr_req_ext, weight_ext) Then
                maxweight = !Weight
                pf = "scr"
            Else
                End If
            Else
                End If
            .MoveNext
        Wend

    End With

End Sub
Private Function RestrictMemory(maxweight As Double, pf As String)
As String

    Dim db As Database
    Dim qd As QueryDef
    Dim tb As Table
    Dim text As New MapObjects.Rectangle
    Dim cext As New MapObjects.Rectangle

    Set db =
OpenDatabase("C:\projects\webagent\experience\experience.mdb")
    If pf = "cmd" Then
        Set qd = db.CreateQueryDef("Q_MBR", "select* from MBR where
cmd = '" & tcmd & "'")

    Else
        If pf = "sel" Then
            If sel Then
                Set qd = db.CreateQueryDef("Q_MBR", "select * from MBR
where sel=Yes")
            Else
                Set qd = db.CreateQueryDef("Q_MBR", "select * from MBR
where sel=No")
            End If
        Else
            If pf = "cr" Then
                text.Left = tleft
                text.Right = tright
                text.Bottom = tbottom
                text.Top = ttop

                Set qd = db.CreateQueryDef("Q_MBR", "select * into
CMBR from MBR")
                qd.Execute
                datac.RecordSource = "CMBR"
                datac.Refresh
                With datac.Recordset
                    If Not .EOF Then

```

```

        .MoveFirst
    Else
    End If
    While Not .EOF
        cext.Left = !Left
        cext.Right = !Right
        cext.Bottom = !Bottom
        cext.Top = !Top
        If Not equal_extents(text, cext) Then
            BeginTrans
            .Edit
            .Delete

            CommitTrans
        Else

            End If
            .MoveNext
        Wend
        db.QueryDefs.Delete ("Q_MBR")
        Set qd = db.CreateQueryDef("Q_MBR", "select * from
CMBR")
        End With
    Else
        text.Left = tsleft
        text.Right = tsright
        text.Bottom = tsbottom
        text.Top = tstop

        Set qd = db.CreateQueryDef("Q_MBR", "select * into
CMBR from MBR")
        qd.Execute
        datac.RecordSource = "CMBR"
        datac.Refresh
        With datac.Recordset
            If Not .EOF Then
                .MoveFirst
            Else
            End If
            While Not .EOF
                cext.Left = !sleft
                cext.Right = !sright
                cext.Bottom = !sbottom
                cext.Top = !Stop
                If Not equal_extents(text, cext) Then
                    BeginTrans
                    .Edit
                    .Delete
                    CommitTrans
                Else
                    .MoveNext
                End If
            Wend
        End With
        db.QueryDefs.Delete ("Q_MBR")
        Set qd = db.CreateQueryDef("Q_MBR", "select * from
CMBR")

        End If
    End If

```

```

End If
db.Close
RestrictMemory = "Q_MBR"

End Function

Private Sub delta_gcmd(qdstring As String)

Dim db As Database
Dim qd As QueryDef
Dim cqd As QueryDef
Dim lambda As Double
Dim temp1 As Double
'Not necessary to test if the cmd was suggest because
'it is done before the call

lambda = 0
temp1 = 0

Set db =
OpenDatabase("C:\projects\webagent\experience\experience.mdb")
db.Execute "delete * from delta_gcmd"
delta.RecordSource = "delta_gcmd"
delta.Refresh
datac.RecordSource = qdstring
datac.Refresh

With datac.Recordset

If Not .EOF Then
.MoveFirst
Else
End If
While Not .EOF
If tcmd <> !cmd Then
Set cqd = db.CreateQueryDef("cqd", "select d from
d_gcmd_cmd where cmd1='" & tcmd & "'" and cmd2 = '" & !cmd & "'")
temp.RecordSource = "cqd"
temp.Refresh
If Not temp.Recordset.EOF Then
temp1 = temp.Recordset!d
Else
End If

db.QueryDefs.Delete ("cqd")
Set cqd = db.CreateQueryDef("cqd", "select weight
from w_cmd_gcmd where cmd='" & tcmd & "'")
temp.RecordSource = "cqd"
temp.Refresh
If Not temp.Recordset.EOF Then
temp1 = temp1 * temp.Recordset!Weight
Else
End If
lambda = lambda + temp1
temp1 = 0
db.QueryDefs.Delete ("cqd")
Else
End If
lambda = lambda + lambda_for_cr("d_gcmd_cr",
"w_cr_gcmd")

```

```

                If (!sel = True And Not tsel) Or (!sel = False And tsel)
Then
                If tsel Then
                    Set cqd = db.CreateQueryDef("cqd", "select d
from d_gcmd_sel where sel1= Yes and sel2=No")
                Else
                    Set cqd = db.CreateQueryDef("cqd", "select d
from d_gcmd_sel where sel1= No and sel2=Yes")
                End If
                temp.RecordSource = "cqd"
                temp.Refresh
                If Not temp.Recordset.EOF Then
                    temp1 = temp.Recordset!d
                Else
                    End If

                db.QueryDefs.Delete ("cqd")

                If tsel Then
                    Set cqd = db.CreateQueryDef("cqd", "select
weight from w_sel_gcmd where sel= Yes")
                Else
                    Set cqd = db.CreateQueryDef("cqd", "select
weight from w_sel_gcmd where sel= No")
                End If
                temp.RecordSource = "cqd"
                temp.Refresh
                If Not temp.Recordset.EOF Then
                    temp1 = temp1 * temp.Recordset!Weight
                Else
                    End If
                lambda = lambda + temp1
                temp1 = 0
                db.QueryDefs.Delete ("cqd")
            Else
                End If
            If tsel Then
                lambda = lambda + lambda_for_scr("d_gcmd_scr",
"w_scr_gcmd")
            Else
                End If

            BeginTrans
                delta.Recordset.AddNew
                delta.Recordset!user = !user
                delta.Recordset!Date = !Date
                delta.Recordset!Time = !Time
                delta.Recordset!number = !number
                delta.Recordset!gcmd = !gcmd
                delta.Recordset!delta = lambda
                delta.Recordset.Update
            CommitTrans
            .MoveNext

        Wend
    End With
db.Close

```



```

End Sub

Private Function lambda_for_cr(d_table As String, w_table As String)
As Double

    Dim text As New MapObjects.Rectangle
    Dim ctext As New MapObjects.Rectangle
    Dim dtext1 As New MapObjects.Rectangle
    Dim dtext2 As New MapObjects.Rectangle

    Dim sair As Boolean
    Dim temp2 As Double
    temp2 = 0
    text.Left = tleft
    text.Right = tright
    text.Bottom = tbottom
    text.Top = ttop
    ctext.Left = datac.Recordset!Left
    ctext.Right = datac.Recordset!Right
    ctext.Bottom = datac.Recordset!Bottom
    ctext.Top = datac.Recordset!Top
    'adicionar que se estas duas sao iguais entao a distancia e
zero!!!!
    If equal_extents(text, ctext) Then
        lambda_for_cr = 0
    Else

        sair = False

        temp.RecordSource = d_table
        temp.Refresh
        If Not temp.Recordset.EOF Then
            temp.Recordset.MoveFirst
            While (Not temp.Recordset.EOF) And (Not sair)
                dtext1.Left = temp.Recordset!Left1
                dtext1.Right = temp.Recordset!Right1
                dtext1.Bottom = temp.Recordset!Bottom1
                dtext1.Top = temp.Recordset!Top1

                dtext2.Left = temp.Recordset!Left2
                dtext2.Right = temp.Recordset!Right2
                dtext2.Bottom = temp.Recordset!Bottom2
                dtext2.Top = temp.Recordset!Top2

                If equal_extents(text, dtext1) And
equal_extents(ctext, dtext2) Then
                    temp2 = temp.Recordset!d
                    sair = True
                Else
                    temp.Recordset.MoveNext
                End If
            Wend

            temp.RecordSource = w_table
            temp.Refresh
            sair = False
            If Not temp.Recordset.EOF Then
                temp.Recordset.MoveFirst
                While (Not temp.Recordset.EOF) And Not sair
                    ctext.Left = temp.Recordset!Left
                    ctext.Right = temp.Recordset!Right

```

```

        ctext.Bottom = temp.Recordset!Bottom
        ctext.Top = temp.Recordset!Top
        If equal_extents(text, ctext) Then
            temp2 = temp2 * temp.Recordset!Weight
            sair = True
        Else
            temp.Recordset.MoveNext
        End If

    Wend
Else
    End If

Else
    End If

    lambda_for_cr = temp2
End If
End Function

Private Function lambda_for_scr(d_table As String, w_table As
String) As Double

    Dim text As New MapObjects.Rectangle
    Dim ctext As New MapObjects.Rectangle
    Dim dtext1 As New MapObjects.Rectangle
    Dim dtext2 As New MapObjects.Rectangle

    Dim sair As Boolean
    Dim temp2 As Double
    If tsel Then

        temp2 = 0
        text.Left = tsleft
        text.Right = tsright
        text.Bottom = tsbottom
        text.Top = tstop
        If datac.Recordset!sel = True Then
            ctext.Left = datac.Recordset!sleft
            ctext.Right = datac.Recordset!sright
            ctext.Bottom = datac.Recordset!sbottom
            ctext.Top = datac.Recordset!stop
            If equal_extents(text, ctext) Then
                lambda_for_scr = 0
            Else

                sair = False

                temp.RecordSource = d_table
                temp.Refresh
                If Not temp.Recordset.EOF Then
                    temp.Recordset.MoveFirst
                    While (Not temp.Recordset.EOF) And (Not sair)
                        dtext1.Left = temp.Recordset!sleft1
                        dtext1.Right = temp.Recordset!sright1
                        dtext1.Bottom = temp.Recordset!sbottom1
                        dtext1.Top = temp.Recordset!stop1

                        dtext2.Left = temp.Recordset!sleft2
                        dtext2.Right = temp.Recordset!sright2
                    
```

```

        dtext2.Bottom = temp.Recordset!sbottom2
        dtext2.Top = temp.Recordset!stop2

        If equal_extents(text, dtext1) And
equal_extents(ctext, dtext2) Then
            temp2 = temp.Recordset!d
            sair = True
        Else
            temp.Recordset.MoveNext
        End If
    Wend

    temp.RecordSource = w_table
    temp.Refresh
    sair = False
    If Not temp.Recordset.EOF Then
        temp.Recordset.MoveFirst
        While (Not temp.Recordset.EOF) And Not sair
            ctext.Left = temp.Recordset!sleft
            ctext.Right = temp.Recordset!sright
            ctext.Bottom = temp.Recordset!sbottom
            ctext.Top = temp.Recordset!Stop
            If equal_extents(text, ctext) Then
                temp2 = temp2 * temp.Recordset!Weight
                sair = True
            Else
                temp.Recordset.MoveNext
            End If
        Wend
    Else
        End If

    Else
        End If

        lambda_for_scr = temp2
        End If
    Else
        lambda_for_scr = 0
    End If
Else
    lambda_for_scr = 0
End If
End Function

```

```
Private Sub delta_gsel(qdstring As String)
```

```

    Dim db As Database
    Dim qd As QueryDef
    Dim cq As QueryDef
    Dim lambda As Double
    Dim temp1 As Double
    'Not necessary to test if the cmd was suggest because
    'it is done before the call

    lambda = 0
    temp1 = 0

```

```

Set db =
OpenDatabase("C:\projects\webagent\experience\experience.mdb")
db.Execute "delete * from delta_gsel"
delta.RecordSource = "delta_gsel"
delta.Refresh
datac.RecordSource = qdstring
datac.Refresh

With datac.Recordset

    If Not .EOF Then
        .MoveFirst
    Else
        End If
    While Not .EOF
        If tcmd = "Sug" Then

            If tcmd <> !cmd Then
                Set cqd = db.CreateQueryDef("cqd", "select d
from d_gsel_cmd where cmd1='" & tcmd & "'" and cmd2 = '" & !cmd &
'")

                temp.RecordSource = "cqd"
                temp.Refresh
                If Not temp.Recordset.EOF Then
                    temp1 = temp.Recordset!d
                Else
                    End If

                db.QueryDefs.Delete ("cqd")
                Set cqd = db.CreateQueryDef("cqd", "select
weight from w_cmd_gsel where cmd='" & tcmd & "'")
                temp.RecordSource = "cqd"
                temp.Refresh
                If Not temp.Recordset.EOF Then
                    temp1 = temp1 * temp.Recordset!Weight
                Else
                    End If
                lambda = lambda + temp1
                temp1 = 0
                db.QueryDefs.Delete ("cqd")
            Else
                End If
        Else
            End If
        lambda = lambda + lambda_for_cr("d_gsel_cr",
"w_cr_gsel")
        If (!sel = True And Not tsel) Or (!sel = False And tsel)
Then

            If tsel Then
                Set cqd = db.CreateQueryDef("cqd", "select d
from d_gsel_sel where sel1= Yes and sel2=No")
            Else
                Set cqd = db.CreateQueryDef("cqd", "select d
from d_gsel_sel where sel1= No and sel2=Yes")
            End If
            temp.RecordSource = "cqd"
            temp.Refresh
            If Not temp.Recordset.EOF Then
                temp1 = temp.Recordset!d
            End If
        End If
    End While
End With

```

```

        Else
        End If

        db.QueryDefs.Delete ("cqd")

        If tsel Then
            Set cqd = db.CreateQueryDef("cqd", "select
weight from w_sel_gsel where sel= Yes")
        Else
            Set cqd = db.CreateQueryDef("cqd", "select
weight from w_sel_gsel where sel= No")
        End If
        temp.RecordSource = "cqd"
        temp.Refresh
        If Not temp.Recordset.EOF Then
            temp1 = temp1 * temp.Recordset!Weight
        Else
        End If
        lambda = lambda + temp1
        temp1 = 0
        db.QueryDefs.Delete ("cqd")
    Else
    End If
    If tsel Then
        lambda = lambda + lambda_for_scr("d_gsel_scr",
"w_scr_gsel")
    Else
    End If

    BeginTrans
        'Ver aqui se o gsel fica bem na tabela
        delta.Recordset.AddNew
        delta.Recordset!user = !user
        delta.Recordset!Date = !Date
        delta.Recordset!Time = !Time
        delta.Recordset!number = !number
        delta.Recordset!gsel = !gsel
        delta.Recordset!delta = lambda
        delta.Recordset.Update
    CommitTrans
    .MoveNext
Wend
End With
db.Close

```

End Sub

Private Sub delta_gscr(qdstring As String)

```

    Dim db As Database
    Dim qd As QueryDef
    Dim cqd As QueryDef
    Dim lambda As Double
    Dim temp1 As Double

```

```

    lambda = 0
    temp1 = 0

```

```

Set db =
OpenDatabase("C:\projects\webagent\experience\experience.mdb")
db.Execute "delete * from delta_gscr"
delta.RecordSource = "delta_gscr"
delta.Refresh
datac.RecordSource = qdstring
datac.Refresh

With datac.Recordset
  If Not .EOF Then
    .MoveFirst
  Else
    End If
  While Not .EOF
    If !gsel Then

      If max_date = Date Then
        'testar primeiro se cmds sao iguais
        If tcmd <> !cmd Then
          Set cqd = db.CreateQueryDef("cqcd", "select d
from d_gscr_cmd where cmd1='" & tcmd & "'" and cmd2 = '" & !cmd &
'")

          temp.RecordSource = "cqcd"
          temp.Refresh
          If Not temp.Recordset.EOF Then
            temp1 = temp.Recordset!d
          Else
            End If

          db.QueryDefs.Delete ("cqcd")
          Set cqd = db.CreateQueryDef("cqcd", "select
weight from w_cmd_gscr where cmd='" & tcmd & "'")
          temp.RecordSource = "cqcd"
          temp.Refresh
          If Not temp.Recordset.EOF Then
            temp1 = temp1 * temp.Recordset!Weight
          Else
            End If
          lambda = lambda + temp1
          temp1 = 0
          db.QueryDefs.Delete ("cqcd")
        Else
          End If

      Else
        End If
        'testar primeiro se sao iguais
        lambda = lambda + lambda_for_cr("d_gscr_cr",
"w_cr_gscr")
        'testar primeiro se sao iguais
        If (!sel = True And Not tsel) Or (!sel = False And tsel)
Then

          If tsel Then
            Set cqd = db.CreateQueryDef("cqcd", "select d
from d_gscr_sel where sel1=Yes and sel2=No")
          Else
            Set cqd = db.CreateQueryDef("cqcd", "select d
from d_gscr_sel where sel1=No and sel2=Yes")
          End If
          temp.RecordSource = "cqcd"

```

```

temp.Refresh
If Not temp.Recordset.EOF Then
    temp1 = temp.Recordset!d
Else
End If

db.QueryDefs.Delete ("cqd")

If tsel Then
    Set cqd = db.CreateQueryDef("cqd", "select
weight from w_sel_gscr where sel= Yes")
Else
    Set cqd = db.CreateQueryDef("cqd", "select
weight from w_sel_gscr where sel= No")
End If
temp.RecordSource = "cqd"
temp.Refresh
If Not temp.Recordset.EOF Then
    temp1 = temp1 * temp.Recordset!Weight
Else
End If
lambda = lambda + temp1
temp1 = 0
db.QueryDefs.Delete ("cqd")
Else
End If
'testar primeiro se sao iguais
If tsel Then
    lambda = lambda + lambda_for_scr("d_gscr_scr",
"w_scr_gscr")
Else
End If

BeginTrans
delta.Recordset.AddNew
delta.Recordset!user = !user
delta.Recordset!Date = !Date
delta.Recordset!Time = !Time
delta.Recordset!number = !number
delta.Recordset!gsleft = !gsleft
delta.Recordset!gsright = !gsright
delta.Recordset!gsbottom = !gsbottom
delta.Recordset!gstop = !gstop
delta.Recordset!delta = lambda
delta.Recordset.Update
CommitTrans
.MoveNext
Else
.MoveNext
End If
Wend
End With
db.Close

End Sub

```

```

Private Function CalculateNextAction(ext As MapObjects.Rectangle,
ByVal values As Object, ByVal arguments As Object) As String
    Dim db As Database
    Dim qdstring As String

```

```
'get the previous action to consider, (previous goal fields, or
state in the current request)
GetPreviousAction ext, values, arguments
```

```
'for each value of the predictor fields search for the one with
the highest weight in the Memory
GetHighestPField
```

```
'restrict the memory to those records with the highest weight
value
```

```
qdsting = RestrictMemory(maxweight, pf)
```

```
If max_date = Date Then
```

```
    delta_gcmd (qdsting)
```

```
Else
```

```
End If
```

```
delta_gcr (qdsting)
```

```
If max_date = Date Then
```

```
    delta_gsel (qdsting)
```

```
    delta_gscr (qdsting)
```

```
Else
```

```
End If
```

```
'calcular deltas invertidos e decidir quais as accoes a propor
```

```
'criar o comando retrieve no java que nao faz nada no servidor
```

```
'datac.RecordSource = qdsting
```

```
'datac.Refresh
```

```
Set db =
```

```
OpenDatabase("C:\projects\webagent\experience\experience.mdb")
```

```
db.QueryDefs.Delete (qdsting)
```

```
If pf = "cr" Or pf = "scr" Then
```

```
    db.TableDefs.Delete ("CMBR")
```

```
Else
```

```
End If
```

```
db.Close
```

```
Private Function equal_extents(ext1 As MapObjects.Rectangle, ext2 As
MapObjects.Rectangle) As Boolean
```

```
Dim area1 As Double
```

```
Dim area2 As Double
```

```
Dim area3 As Double
```

```
Dim intersect As New MapObjects.Rectangle
```

```
area1 = area_extent(ext1)
```

```
area2 = area_extent(ext2)
```

```
If ((area1 < area2 * 0.5) Or (area2 < area1 * 0.5)) Then
```

```
    equal_extents = False
```

```
Else
```

```
    intersect.Left = ext1.Left
```

```
    intersect.Right = ext1.Right
```

```
    intersect.Top = ext1.Top
```

```
    intersect.Bottom = ext1.Bottom
```

```
    intersect.intersect ext2
```

```
    area3 = area_extent(intersect)
```

```
If (area3 < (area1 * 0.5)) And (area3 < (area2 * 0.5)) Then
```

```
    equal_extents = False
```



```
        Else
            equal_extents = True
        End If
    End If
End Function
```

Appendix C: SIFIA Memory Manager Code

```
Private Sub rec_frequence_cr()  
    Dim ext1 As New MapObjects.Rectangle  
    Dim rec As Recordset  
    Dim bb As Variant  
    Dim sair As Boolean  
    Dim ext2 As New MapObjects.Rectangle  
  
    compare2.RecordSource = "frequence_cr"  
    compare2.Refresh  
    compare1.RecordSource = "frequence_cr"  
    compare1.Refresh  
    compare1.Recordset.MoveFirst  
    sair = False  
    compare2.Recordset.MoveFirst  
    compare2.Recordset.MoveNext  
  
    While Not ((compare1.Recordset.EOF) Or sair)  
  
        With compare2.Recordset  
            If Not .EOF Then  
                ext1.Left = compare1.Recordset!Left  
                ext1.Right = compare1.Recordset!Right  
                ext1.Bottom = compare1.Recordset!Bottom  
                ext1.Top = compare1.Recordset!Top  
                ext2.Left = !Left  
                ext2.Right = !Right  
                ext2.Top = !Top  
                ext2.Bottom = !Bottom  
                Dim equal As Integer  
                equal = (equal_extents(ext1, ext2))  
                If (equal = True) Then  
                    BeginTrans  
                    compare1.Recordset.Edit  
                    compare1.Recordset!Ccr = compare1.Recordset!Ccr + !Ccr  
                    compare1.Recordset!Mbef = "X"  
                    compare1.Recordset.Update  
                    .Edit  
                    !rec_del = "X"  
                    .Update  
                    CommitTrans  
                    Do  
                        .MoveNext  
                        If .EOF Then  
                            Exit Do  
                        End If  
                    Loop Until IsNull(!rec_del) And IsNull(!Mbef)  
                Else  
                    Do  
                        .MoveNext  
                        If .EOF Then  
                            Exit Do  
                        End If  
                    Loop Until IsNull(!rec_del) And IsNull(!Mbef)  
                End If  
            Else  
                Else  
            End If  
        End With  
    End While  
End Sub
```

```

        Do
            compare1.Recordset.MoveNext
            If compare1.Recordset.EOF Then
                Exit Do
            End If
            Loop Until IsNull(compare1.Recordset!rec_del) And
IsNull(compare1.Recordset!Mbef)
            If (Not (compare1.Recordset.EOF)) Then
                BeginTrans
                    compare1.Recordset.Edit
                    compare1.Recordset!Mbef = "X"
                    compare1.Recordset.Update
                CommitTrans
                compare2.Recordset.MoveFirst
                If (Not (compare2.Recordset.EOF)) Then
                    If (compare2.Recordset!rec_del = "X") Or
(compare2.Recordset!Mbef = "X") Then

                        Do
                            compare2.Recordset.MoveNext
                            If (compare2.Recordset.EOF) Then
                                Exit Do
                            End If
                            Loop Until
IsNull(compare2.Recordset!rec_del) And
IsNull(compare2.Recordset!Mbef)
                                End If
                            Else
                                sair = True
                            End If
                        End If
                    End If
                End With
            Wend

            compare1.Recordset.MoveFirst
            Do
                BeginTrans
                    compare1.Recordset.Edit
                    If Not IsNull(compare1.Recordset!rec_del) Then
                        compare1.Recordset.Delete
                    Else
                        compare1.Recordset!Mbef = Null
                        compare1.Recordset.Update
                    End If
                CommitTrans
                compare1.Recordset.MoveNext
            Loop Until compare1.Recordset.EOF
        End Sub

```

```

Private Sub rec_frequence_scr()
    Dim ext1 As New MapObjects.Rectangle
    Dim rec As Recordset
    Dim bb As Variant
    Dim sair As Boolean
    Dim ext2 As New MapObjects.Rectangle

    compare2.RecordSource = "frequence_scr"
    compare2.Refresh

```

```

compare1.RecordSource = "frequence_scr"
compare1.Refresh
compare1.Recordset.MoveFirst
sair = False
compare2.Recordset.MoveFirst
compare2.Recordset.MoveNext

While Not ((compare1.Recordset.EOF) Or sair)

With compare2.Recordset
If Not .EOF Then
    ext1.Left = compare1.Recordset!sleft
    ext1.Right = compare1.Recordset!sright
    ext1.Bottom = compare1.Recordset!sbottom
    ext1.Top = compare1.Recordset!stop
    ext2.Left = !sleft
    ext2.Right = !sright
    ext2.Top = !stop
    ext2.Bottom = !sbottom
    Dim equal As Integer
    equal = (equal_extents(ext1, ext2))
    If (equal = True) Then
        BeginTrans
            compare1.Recordset.Edit
            compare1.Recordset!Cscr = compare1.Recordset!Cscr +
!Cscr

            compare1.Recordset!Mbef = "X"
            compare1.Recordset.Update
            .Edit
            !rec_del = "X"
            .Update
        CommitTrans
        Do
            .MoveNext
            If .EOF Then
                Exit Do
            End If
            Loop Until IsNull(!rec_del) And IsNull(!Mbef)
        Else
            Do
                .MoveNext
                If .EOF Then
                    Exit Do
                End If
                Loop Until IsNull(!rec_del) And IsNull(!Mbef)
            End If
        Else
            Do
                compare1.Recordset.MoveNext
                If compare1.Recordset.EOF Then
                    Exit Do
                End If
                Loop Until IsNull(compare1.Recordset!rec_del) And
IsNull(compare1.Recordset!Mbef)
                If (Not (compare1.Recordset.EOF)) Then
                    BeginTrans
                        compare1.Recordset.Edit
                        compare1.Recordset!Mbef = "X"
                        compare1.Recordset.Update
                    CommitTrans
                    compare2.Recordset.MoveFirst

```

```

                If (Not (compare2.Recordset.EOF)) Then
                    If (compare2.Recordset!rec_del = "X") Or
(compare2.Recordset!Mbef = "X") Then

                        Do
                            compare2.Recordset.MoveNext
                            If (compare2.Recordset.EOF) Then
                                Exit Do
                            End If
                            Loop Until
IsNull(compare2.Recordset!rec_del) And
IsNull(compare2.Recordset!Mbef)
                                End If
                            Else
                                sair = True
                            End If
                        End If
                    End If
                End With
            Wend

            compare1.Recordset.MoveFirst
            Do
                BeginTrans
                    compare1.Recordset.Edit
                    If Not IsNull(compare1.Recordset!rec_del) Then
                        compare1.Recordset.Delete
                    Else
                        compare1.Recordset!Mbef = Null
                        compare1.Recordset.Update
                    End If
                CommitTrans
                compare1.Recordset.MoveNext
            Loop Until compare1.Recordset.EOF
        End Sub

        Private Sub rec_frequence_cmd_gcr()
            Dim ext1 As New MapObjects.Rectangle
            Dim rec As Recordset
            Dim bb As Variant
            Dim sair As Boolean
            Dim ext2 As New MapObjects.Rectangle

            compare2.RecordSource = "frequence_cmd_gcr"
            compare2.Refresh
            compare1.RecordSource = "frequence_cmd_gcr"
            compare1.Refresh
            compare1.Recordset.MoveFirst
            sair = False
            compare2.Recordset.MoveFirst
            compare2.Recordset.MoveNext

            While Not ((compare1.Recordset.EOF) Or sair)

                With compare2.Recordset
                    If Not .EOF Then
                        ext1.Left = compare1.Recordset!gleft
                        ext1.Right = compare1.Recordset!gright
                        ext1.Bottom = compare1.Recordset!gbottom
                        ext1.Top = compare1.Recordset!gtop
                        ext2.Left = !gleft

```

```

ext2.Right = !gright
ext2.Top = !gtop
ext2.Bottom = !gbottom
Dim equal As Integer
equal = (equal_extents(ext1, ext2))
If (equal = True) And (!cmd = compare1.Recordset!cmd) Then
    BeginTrans
        compare1.Recordset.Edit
        compare1.Recordset!Cgcr = compare1.Recordset!Cgcr +
!Cgcr

        compare1.Recordset!Mbef = "X"
        compare1.Recordset.Update
        .Edit
        !rec_del = "X"
        .Update
    CommitTrans
    Do
        .MoveNext
        If .EOF Then
            Exit Do
        End If
        Loop Until IsNull(!rec_del) And IsNull(!Mbef)
    Else
        Do
            .MoveNext
            If .EOF Then
                Exit Do
            End If
            Loop Until IsNull(!rec_del) And IsNull(!Mbef)
        End If
    Else
        Do
            compare1.Recordset.MoveNext
            If compare1.Recordset.EOF Then
                Exit Do
            End If
            Loop Until IsNull(compare1.Recordset!rec_del) And
IsNull(compare1.Recordset!Mbef)
            If (Not (compare1.Recordset.EOF)) Then
                BeginTrans
                    compare1.Recordset.Edit
                    compare1.Recordset!Mbef = "X"
                    compare1.Recordset.Update
                CommitTrans
                compare2.Recordset.MoveFirst
                If (Not (compare2.Recordset.EOF)) Then
                    If (compare2.Recordset!rec_del = "X") Or
(compare2.Recordset!Mbef = "X") Then

                        Do
                            compare2.Recordset.MoveNext
                            If (compare2.Recordset.EOF) Then
                                Exit Do
                            End If
                            Loop Until
IsNull(compare2.Recordset!rec_del) And
IsNull(compare2.Recordset!Mbef)
                        End If
                    Else
                        sair = True
                    End If
                End If
            End If
        End If
    End If
End If

```

```

        End If
    End If
End With
Wend

compare1.Recordset.MoveFirst
Do
    BeginTrans
        compare1.Recordset.Edit
        If Not IsNull(compare1.Recordset!rec_del) Then
            compare1.Recordset.Delete
        Else
            compare1.Recordset!Mbef = Null
            compare1.Recordset.Update
        End If
    CommitTrans
    compare1.Recordset.MoveNext
Loop Until compare1.Recordset.EOF
End Sub

Private Sub rec_frequence_cmd_gscr()
    Dim ext1 As New MapObjects.Rectangle
    Dim rec As Recordset
    Dim bb As Variant
    Dim sair As Boolean
    Dim ext2 As New MapObjects.Rectangle

    compare2.RecordSource = "frequence_cmd_gscr"
    compare2.Refresh
    compare1.RecordSource = "frequence_cmd_gscr"
    compare1.Refresh
    compare1.Recordset.MoveFirst
    sair = False
    compare2.Recordset.MoveFirst
    compare2.Recordset.MoveNext

    While Not ((compare1.Recordset.EOF) Or sair)

        With compare2.Recordset
            If Not .EOF Then
                ext1.Left = compare1.Recordset!gsleft
                ext1.Right = compare1.Recordset!gsright
                ext1.Bottom = compare1.Recordset!gsbottom
                ext1.Top = compare1.Recordset!gstop
                ext2.Left = !gsleft
                ext2.Right = !gsright
                ext2.Top = !gstop
                ext2.Bottom = !gsbottom
                Dim equal As Integer
                equal = (equal_extents(ext1, ext2))
                If (equal = True) And (!cmd = compare1.Recordset!cmd) Then
                    BeginTrans
                        compare1.Recordset.Edit
                        compare1.Recordset!Cgscr = compare1.Recordset!Cgscr +
!Cgscr

                        compare1.Recordset!Mbef = "X"
                        compare1.Recordset.Update
                        .Edit
                        !rec_del = "X"
                        .Update
                    CommitTrans
                End If
            End If
        End With
    End While
End Sub

```

```

Do
    .MoveNext
    If .EOF Then
        Exit Do
    End If
    Loop Until IsNull(!rec_del) And IsNull(!Mbef)
Else
    Do
        .MoveNext
        If .EOF Then
            Exit Do
        End If
        Loop Until IsNull(!rec_del) And IsNull(!Mbef)
    End If
Else
    Do
        compare1.Recordset.MoveNext
        If compare1.Recordset.EOF Then
            Exit Do
        End If
        Loop Until IsNull(compare1.Recordset!rec_del) And
IsNull(compare1.Recordset!Mbef)
        If (Not (compare1.Recordset.EOF)) Then
            BeginTrans
                compare1.Recordset.Edit
                compare1.Recordset!Mbef = "X"
                compare1.Recordset.Update
            CommitTrans
            compare2.Recordset.MoveFirst
            If (Not (compare2.Recordset.EOF)) Then
                If (compare2.Recordset!rec_del = "X") Or
(compare2.Recordset!Mbef = "X") Then

                    Do
                        compare2.Recordset.MoveNext
                        If (compare2.Recordset.EOF) Then
                            Exit Do
                        End If
                    Loop Until
IsNull(compare2.Recordset!rec_del) And
IsNull(compare2.Recordset!Mbef)
                    End If
                Else
                    sair = True
                End If
            End If
        End If
    End With
Wend

compare1.Recordset.MoveFirst
Do
    BeginTrans
        compare1.Recordset.Edit
        If Not IsNull(compare1.Recordset!rec_del) Then
            compare1.Recordset.Delete
        Else
            compare1.Recordset!Mbef = Null
            compare1.Recordset.Update
        End If
    CommitTrans

```



```

        compare1.Recordset.MoveNext
    Loop Until compare1.Recordset.EOF
End Sub

Private Sub rec_frequence_scr_gcr()
    Dim ext1 As New MapObjects.Rectangle
    Dim rec As Recordset
    Dim bb As Variant
    Dim sair As Boolean
    Dim ext2 As New MapObjects.Rectangle
    Dim gext1 As New MapObjects.Rectangle
    Dim gext2 As New MapObjects.Rectangle

    compare2.RecordSource = "frequence_scr_gcr"
    compare2.Refresh
    compare1.RecordSource = "frequence_scr_gcr"
    compare1.Refresh
    compare1.Recordset.MoveFirst
    sair = False
    compare2.Recordset.MoveFirst
    compare2.Recordset.MoveNext

    While Not ((compare1.Recordset.EOF) Or sair)

        With compare2.Recordset
            If Not .EOF Then
                ' Master
                ext1.Left = compare1.Recordset!sleft
                ext1.Right = compare1.Recordset!sright
                ext1.Bottom = compare1.Recordset!sbottom
                ext1.Top = compare1.Recordset!stop
                gext1.Left = compare1.Recordset!gleft
                gext1.Right = compare1.Recordset!gright
                gext1.Top = compare1.Recordset!gtop
                gext1.Bottom = compare1.Recordset!gbottom

                'detail
                ext2.Left = !sleft
                ext2.Right = !sright
                ext2.Top = !stop
                ext2.Bottom = !sbottom
                gext2.Left = !gleft
                gext2.Right = !gright
                gext2.Top = !gtop
                gext2.Bottom = !gbottom

                Dim equal1 As Boolean
                Dim equal2 As Boolean
                equal1 = (equal_extents(ext1, ext2))
                equal2 = (equal_extents(gext1, gext2))
                If (equal1 And equal2) Then
                    BeginTrans
                        compare1.Recordset.Edit
                        compare1.Recordset!Cgcr = compare1.Recordset!Cgcr +
!Cgcr
                        compare1.Recordset!Mbef = "X"
                        compare1.Recordset.Update
                        .Edit
                        !rec_del = "X"
                        .Update
                    CommitTrans
                End If
            End If
        End With
    End While
End Sub

```

```

Do
    .MoveNext
    If .EOF Then
        Exit Do
    End If
    Loop Until IsNull(!rec_del) And IsNull(!Mbef)
Else
    Do
        .MoveNext
        If .EOF Then
            Exit Do
        End If
        Loop Until IsNull(!rec_del) And IsNull(!Mbef)
    End If
Else
    Do
        compare1.Recordset.MoveNext
        If compare1.Recordset.EOF Then
            Exit Do
        End If
        Loop Until IsNull(compare1.Recordset!rec_del) And
IsNull(compare1.Recordset!Mbef)
        If (Not (compare1.Recordset.EOF)) Then
            BeginTrans
                compare1.Recordset.Edit
                compare1.Recordset!Mbef = "X"
                compare1.Recordset.Update
            CommitTrans
            compare2.Recordset.MoveFirst
            If (Not (compare2.Recordset.EOF)) Then
                If (compare2.Recordset!rec_del = "X") Or
(compare2.Recordset!Mbef = "X") Then

                    Do
                        compare2.Recordset.MoveNext
                        If (compare2.Recordset.EOF) Then
                            Exit Do
                        End If
                        Loop Until
IsNull(compare2.Recordset!rec_del) And
IsNull(compare2.Recordset!Mbef)
                    End If
                Else
                    sair = True
                End If
            End If
        End With
    Wend

    compare1.Recordset.MoveFirst
    Do
        BeginTrans
            compare1.Recordset.Edit
            If Not IsNull(compare1.Recordset!rec_del) Then
                compare1.Recordset.Delete
            Else
                compare1.Recordset!Mbef = Null
                compare1.Recordset.Update
            End If
        CommitTrans

```

```

        compare1.Recordset.MoveNext
    Loop Until compare1.Recordset.EOF
End Sub

Private Sub rec_frequence_sel_gcr()
    Dim ext1 As New MapObjects.Rectangle
    Dim rec As Recordset
    Dim bb As Variant
    Dim sair As Boolean
    Dim ext2 As New MapObjects.Rectangle

    compare2.RecordSource = "frequence_sel_gcr"
    compare2.Refresh
    compare1.RecordSource = "frequence_sel_gcr"
    compare1.Refresh
    compare1.Recordset.MoveFirst
    sair = False
    compare2.Recordset.MoveFirst
    compare2.Recordset.MoveNext

    While Not ((compare1.Recordset.EOF) Or sair)

        With compare2.Recordset
            If Not .EOF Then
                ext1.Left = compare1.Recordset!gleft
                ext1.Right = compare1.Recordset!gright
                ext1.Bottom = compare1.Recordset!gbottom
                ext1.Top = compare1.Recordset!gtop
                ext2.Left = !gleft
                ext2.Right = !gright
                ext2.Top = !gtop
                ext2.Bottom = !gbottom
                Dim equal As Integer
                equal = (equal_extents(ext1, ext2))
                If (equal = True) And (!sel = compare1.Recordset!sel) Then
                    BeginTrans
                        compare1.Recordset.Edit
                        compare1.Recordset!Cgcr = compare1.Recordset!Cgcr +
!Cgcr
                        compare1.Recordset!Mbef = "X"
                        compare1.Recordset.Update
                        .Edit
                        !rec_del = "X"
                        .Update
                    CommitTrans
                    Do
                        .MoveNext
                        If .EOF Then
                            Exit Do
                        End If
                    Loop Until IsNull(!rec_del) And IsNull(!Mbef)
                Else
                    Do
                        .MoveNext
                        If .EOF Then
                            Exit Do
                        End If
                    Loop Until IsNull(!rec_del) And IsNull(!Mbef)
                End If
            Else
                Do

```

```

        compare1.Recordset.MoveNext
        If compare1.Recordset.EOF Then
            Exit Do
        End If
        Loop Until IsNull(compare1.Recordset!rec_del) And
IsNull(compare1.Recordset!Mbef)
        If (Not (compare1.Recordset.EOF)) Then
            BeginTrans
                compare1.Recordset.Edit
                compare1.Recordset!Mbef = "X"
                compare1.Recordset.Update
            CommitTrans
            compare2.Recordset.MoveFirst
            If (Not (compare2.Recordset.EOF)) Then
                If (compare2.Recordset!rec_del = "X") Or
(compare2.Recordset!Mbef = "X") Then

                    Do
                        compare2.Recordset.MoveNext
                        If (compare2.Recordset.EOF) Then
                            Exit Do
                        End If
                    Loop Until
IsNull(compare2.Recordset!rec_del) And
IsNull(compare2.Recordset!Mbef)
                    End If
                Else
                    sair = True
                End If
            End If
        End If
    End With
Wend

compare1.Recordset.MoveFirst
Do
    BeginTrans
        compare1.Recordset.Edit
        If Not IsNull(compare1.Recordset!rec_del) Then
            compare1.Recordset.Delete
        Else
            compare1.Recordset!Mbef = Null
            compare1.Recordset.Update
        End If
    CommitTrans
    compare1.Recordset.MoveNext
Loop Until compare1.Recordset.EOF
End Sub

Private Sub rec_frequence_sel_gscr()
    Dim ext1 As New MapObjects.Rectangle
    Dim rec As Recordset
    Dim bb As Variant
    Dim sair As Boolean
    Dim ext2 As New MapObjects.Rectangle

    compare2.RecordSource = "frequence_sel_gscr"
    compare2.Refresh
    compare1.RecordSource = "frequence_sel_gscr"
    compare1.Refresh
    compare1.Recordset.MoveFirst

```

```

sair = False
compare2.Recordset.MoveFirst
compare2.Recordset.MoveNext

While Not ((compare1.Recordset.EOF) Or sair)

With compare2.Recordset
If Not .EOF Then
    ext1.Left = compare1.Recordset!gsleft
    ext1.Right = compare1.Recordset!gsright
    ext1.Bottom = compare1.Recordset!gsbottom
    ext1.Top = compare1.Recordset!gstop
    ext2.Left = !gsleft
    ext2.Right = !gsright
    ext2.Top = !gstop
    ext2.Bottom = !gsbottom
    Dim equal As Integer
    equal = (equal_extents(ext1, ext2))
    If (equal = True) And (!sel = compare1.Recordset!sel) Then
        BeginTrans
            compare1.Recordset.Edit
            compare1.Recordset!Cgscr = compare1.Recordset!Cgscr +
!Cgscr
            compare1.Recordset!Mbef = "X"
            compare1.Recordset.Update
            .Edit
            !rec_del = "X"
            .Update
        CommitTrans
        Do
            .MoveNext
            If .EOF Then
                Exit Do
            End If
            Loop Until IsNull(!rec_del) And IsNull(!Mbef)
        Else
            Do
                .MoveNext
                If .EOF Then
                    Exit Do
                End If
                Loop Until IsNull(!rec_del) And IsNull(!Mbef)
            End If
        Else
            Do
                compare1.Recordset.MoveNext
                If compare1.Recordset.EOF Then
                    Exit Do
                End If
                Loop Until IsNull(compare1.Recordset!rec_del) And
IsNull(compare1.Recordset!Mbef)
                If (Not (compare1.Recordset.EOF)) Then
                    BeginTrans
                        compare1.Recordset.Edit
                        compare1.Recordset!Mbef = "X"
                        compare1.Recordset.Update
                    CommitTrans
                    compare2.Recordset.MoveFirst
                    If (Not (compare2.Recordset.EOF)) Then
                        If (compare2.Recordset!rec_del = "X") Or
(compare2.Recordset!Mbef = "X") Then

```

```

        Do
            compare2.Recordset.MoveNext
            If (compare2.Recordset.EOF) Then
                Exit Do
            End If
            Loop Until
IsNull(compare2.Recordset!rec_del) And
IsNull(compare2.Recordset!Mbef)
            End If
        Else
            sair = True
        End If
    End If
End With
Wend

compare1.Recordset.MoveFirst
Do
    BeginTrans
        compare1.Recordset.Edit
        If Not IsNull(compare1.Recordset!rec_del) Then
            compare1.Recordset.Delete
        Else
            compare1.Recordset!Mbef = Null
            compare1.Recordset.Update
        End If
    CommitTrans
    compare1.Recordset.MoveNext
Loop Until compare1.Recordset.EOF
End Sub

Private Function equal_extents(ext1 As MapObjects.Rectangle, ext2 As
MapObjects.Rectangle) As Boolean

    Dim area1 As Double
    Dim area2 As Double
    Dim area3 As Double
    Dim intersect As New MapObjects.Rectangle
    area1 = area_extent(ext1)
    area2 = area_extent(ext2)
    If ((area1 < area2 * 0.5) Or (area2 < area1 * 0.5)) Then
        equal_extents = False
    Else

        intersect.Left = ext1.Left
        intersect.Right = ext1.Right
        intersect.Top = ext1.Top
        intersect.Bottom = ext1.Bottom
        intersect.intersect ext2
        area3 = area_extent(intersect)
        If (area3 < (area1 * 0.5)) And (area3 < (area2 * 0.5)) Then
            equal_extents = False
        Else
            equal_extents = True
        End If
    End If
End Function

```

```

Private Function area_extent(ext As MapObjects.Rectangle) As Double

    area_extent = Abs(ext.Right - ext.Left) * Abs(ext.Top -
ext.Bottom)

End Function

Private Sub w_cmd_gcmd()

    Dim w As Double

    'delete all records from w_cmd_gcmd
    Dim dbs As Database
    Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
    dbs.Execute "DELETE * FROM " _
        & " w_cmd_gcmd;"
    dbs.Close

    'weight for cmd_gcmd
    compare1.RecordSource = "frequence_cmd"
    compare2.RecordSource = "frequence_cmd_gcmd"
    res.RecordSource = "w_cmd_gcmd"
    compare1.Refresh
    compare2.Refresh
    res.Refresh
    compare1.Recordset.MoveFirst
    compare2.Recordset.MoveFirst
    With compare2.Recordset

        While Not (compare1.Recordset.EOF Or compare2.Recordset.EOF)

            If (compare1.Recordset!cmd = !cmd) Then

                w = w + (!Cgcmd_cmd / compare1.Recordset!Ccmd) *
(!Cgcmd_cmd / compare1.Recordset!Ccmd)
                .MoveNext
            Else

                .MoveNext
            End If
            If .EOF Then
                w = Sqr(w)
                BeginTrans
                res.Recordset.AddNew
                res.Recordset!cmd = compare1.Recordset!cmd
                res.Recordset!Weight = w
                res.Recordset.Update
                w = 0
                CommitTrans
                .MoveFirst
                compare1.Recordset.MoveNext
            End If

        Wend

    End With
End Sub

Private Sub w_cmd_gcr()

```

```

Dim w As Double

'delete all records from w_cmd_gcmd
Dim dbs As Database
Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
dbs.Execute "DELETE * FROM " _
    & " w_cmd_gcr;"
dbs.Close

'weight for cmd_gcmd
compare1.RecordSource = "frequence_cmd"
compare2.RecordSource = "frequence_cmd_gcr"
res.RecordSource = "w_cmd_gcr"
compare1.Refresh
compare2.Refresh
res.Refresh
compare1.Recordset.MoveFirst
compare2.Recordset.MoveFirst
With compare2.Recordset

    While Not (compare1.Recordset.EOF Or compare2.Recordset.EOF)

        If (compare1.Recordset!cmd = !cmd) Then

            w = w + (!Cgcr / compare1.Recordset!Ccmd) * (!Cgcr /
compare1.Recordset!Ccmd)
            .MoveNext
        Else

            .MoveNext
        End If
        If .EOF Then
            w = Sqr(w)
            BeginTrans
            res.Recordset.AddNew
            res.Recordset!cmd = compare1.Recordset!cmd
            res.Recordset!Weight = w
            res.Recordset.Update
            w = 0
            CommitTrans
            .MoveFirst
            compare1.Recordset.MoveNext
        End If

    Wend

End With
End Sub

Private Sub w_cmd_gscr()

Dim w As Double

'delete all records from w_cmd_gcmd
Dim dbs As Database
Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
dbs.Execute "DELETE * FROM " _
    & " w_cmd_gscr;"
dbs.Close

```



```

'weight for cmd_gcmd
compare1.RecordSource = "frequence_cmd"
compare2.RecordSource = "frequence_cmd_gscr"
res.RecordSource = "w_cmd_gscr"
compare1.Refresh
compare2.Refresh
res.Refresh
compare1.Recordset.MoveFirst
compare2.Recordset.MoveFirst
With compare2.Recordset

    While Not (compare1.Recordset.EOF Or compare2.Recordset.EOF)

        If (compare1.Recordset!cmd = !cmd) Then

            w = w + (!Cgscr / compare1.Recordset!Ccmd) * (!Cgscr
/ compare1.Recordset!Ccmd)
            .MoveNext
        Else

            .MoveNext
        End If
        If .EOF Then
            w = Sqr(w)
            BeginTrans
            res.Recordset.AddNew
            res.Recordset!cmd = compare1.Recordset!cmd
            res.Recordset!Weight = w
            res.Recordset.Update
            w = 0
            CommitTrans
            .MoveFirst
            compare1.Recordset.MoveNext
        End If

    Wend

End With
End Sub

Private Sub w_cmd_gsel()

    Dim w As Double

    'delete all records from w_cmd_gcmd
    Dim dbs As Database
    Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
    dbs.Execute "DELETE * FROM " _
        & " w_cmd_gsel;"
    dbs.Close

    'weight for cmd_gcmd
    compare1.RecordSource = "frequence_cmd"
    compare2.RecordSource = "frequence_cmd_gsel"
    res.RecordSource = "w_cmd_gsel"
    compare1.Refresh
    compare2.Refresh
    res.Refresh
    compare1.Recordset.MoveFirst

```

```

compare2.Recordset.MoveFirst
With compare2.Recordset

    While Not (compare1.Recordset.EOF Or compare2.Recordset.EOF)

        If (compare1.Recordset!cmd = !cmd) Then

            w = w + (!Cgsel_cmd / compare1.Recordset!Ccmd) *
(!Cgsel_cmd / compare1.Recordset!Ccmd)
            .MoveNext
        Else

            .MoveNext
        End If
        If .EOF Then
            w = Sqr(w)
            BeginTrans
            res.Recordset.AddNew
            res.Recordset!cmd = compare1.Recordset!cmd
            res.Recordset!Weight = w
            res.Recordset.Update
            w = 0
            CommitTrans
            .MoveFirst
            compare1.Recordset.MoveNext
        End If

    Wend

End With
End Sub

Private Sub w_cr_gcmd()

    Dim w As Double
    Dim ext1 As New MapObjects.Rectangle
    Dim ext2 As New MapObjects.Rectangle
    Dim equal As Integer

    'delete all records from w_cr_gcmd
    Dim dbs As Database
    Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
    dbs.Execute "DELETE * FROM " _
        & " w_cr_gcmd;"
    dbs.Close

    'weight for cr_gcmd
    compare1.RecordSource = "frequence_cr"
    compare2.RecordSource = "frequence_cr_gcmd"
    res.RecordSource = "w_cr_gcmd"
    compare1.Refresh
    compare2.Refresh
    res.Refresh
    compare1.Recordset.MoveFirst
    compare2.Recordset.MoveFirst
    With compare2.Recordset

        While Not (compare1.Recordset.EOF Or compare2.Recordset.EOF)
            ext1.Left = compare1.Recordset!Left
            ext1.Right = compare1.Recordset!Right

```

```

ext1.Bottom = compare1.Recordset!Bottom
ext1.Top = compare1.Recordset!Top
ext2.Left = !Left
ext2.Right = !Right
ext2.Top = !Top
ext2.Bottom = !Bottom

equal = (ext1.Left = ext2.Left) And (ext1.Right =
ext2.Right) And _
        (ext1.Top = ext2.Top) And (ext1.Bottom =
ext2.Bottom)

If (equal = True) Then
    w = w + (!Ccr / compare1.Recordset!Ccr) * (!Ccr /
compare1.Recordset!Ccr)
    .MoveNext
Else
    .MoveNext
End If
If .EOF Then
    w = Sqr(w)
    BeginTrans
    res.Recordset.AddNew
    res.Recordset!Left = compare1.Recordset!Left
    res.Recordset!Right = compare1.Recordset!Right
    res.Recordset!Top = compare1.Recordset!Top
    res.Recordset!Bottom = compare1.Recordset!Bottom
    res.Recordset!Weight = w
    res.Recordset.Update
    w = 0
    CommitTrans
    .MoveFirst
    compare1.Recordset.MoveNext
End If

Wend

End With
End Sub

Private Sub w_cr_gcr()

    Dim w As Double
    Dim ext1 As New MapObjects.Rectangle
    Dim ext2 As New MapObjects.Rectangle
    Dim equal As Integer

    'delete all records from w_cr_gcr
    Dim dbs As Database
    Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
    dbs.Execute "DELETE * FROM " _
        & " w_cr_gcr;"
    dbs.Close

    'weight for cr_gcr
    compare1.RecordSource = "frequence_cr"
    compare2.RecordSource = "frequence_cr_gcr"
    res.RecordSource = "w_cr_gcr"
    compare1.Refresh

```

```

compare2.Refresh
res.Refresh
compare1.Recordset.MoveFirst
compare2.Recordset.MoveFirst
With compare2.Recordset

    While Not (compare1.Recordset.EOF Or compare2.Recordset.EOF)
        ext1.Left = compare1.Recordset!Left
        ext1.Right = compare1.Recordset!Right
        ext1.Bottom = compare1.Recordset!Bottom
        ext1.Top = compare1.Recordset!Top
        ext2.Left = !Left
        ext2.Right = !Right
        ext2.Top = !Top
        ext2.Bottom = !Bottom

        equal = (ext1.Left = ext2.Left) And (ext1.Right =
ext2.Right) And _
                (ext1.Top = ext2.Top) And (ext1.Bottom =
ext2.Bottom)

        If (equal = True) Then
            w = w + (!Cgcr / compare1.Recordset!Ccr) * (!Cgcr /
compare1.Recordset!Ccr)
            .MoveNext
        Else
            .MoveNext
        End If
        If .EOF Then
            w = Sqr(w)
            BeginTrans
            res.Recordset.AddNew
            res.Recordset!Left = compare1.Recordset!Left
            res.Recordset!Right = compare1.Recordset!Right
            res.Recordset!Top = compare1.Recordset!Top
            res.Recordset!Bottom = compare1.Recordset!Bottom
            res.Recordset!Weight = w
            res.Recordset.Update
            w = 0
            CommitTrans
            .MoveFirst
            compare1.Recordset.MoveNext
        End If

    Wend

End With
End Sub

Private Sub w_scr_gcmd()

    Dim w As Double
    Dim ext1 As New MapObjects.Rectangle
    Dim ext2 As New MapObjects.Rectangle
    Dim equal As Integer

    'delete all records from w_scr_gcmd
    Dim dbs As Database
    Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")

```

```

dbs.Execute "DELETE * FROM " _
    & " w_scr_gcmd;"
dbs.Close

'weight for cr_gcmd
compare1.RecordSource = "frequence_scr"
compare2.RecordSource = "frequence_scr_gcmd"
res.RecordSource = "w_scr_gcmd"
compare1.Refresh
compare2.Refresh
res.Refresh
compare1.Recordset.MoveFirst
compare2.Recordset.MoveFirst
With compare2.Recordset

    While Not (compare1.Recordset.EOF Or compare2.Recordset.EOF)
        ext1.Left = compare1.Recordset!sleft
        ext1.Right = compare1.Recordset!sright
        ext1.Bottom = compare1.Recordset!sbottom
        ext1.Top = compare1.Recordset!stop
        ext2.Left = !sleft
        ext2.Right = !sright
        ext2.Top = !stop
        ext2.Bottom = !sbottom

        equal = (ext1.Left = ext2.Left) And (ext1.Right =
ext2.Right) And _
            (ext1.Top = ext2.Top) And (ext1.Bottom =
ext2.Bottom)

        If (equal = True) Then
            w = w + (!Cgcmd / compare1.Recordset!Cscr) * (!Cgcmd
/ compare1.Recordset!Cscr)
            .MoveNext
        Else
            .MoveNext
        End If
        If .EOF Then
            w = Sqr(w)
            BeginTrans
            res.Recordset.AddNew
            res.Recordset!sleft = compare1.Recordset!sleft
            res.Recordset!sright = compare1.Recordset!sright
            res.Recordset!stop = compare1.Recordset!stop
            res.Recordset!sbottom = compare1.Recordset!sbottom
            res.Recordset!Weight = w
            res.Recordset.Update
            w = 0
            CommitTrans
            .MoveFirst
            compare1.Recordset.MoveNext
        End If

    Wend

End With
End Sub

Private Sub w_scr_gcr()

```

```

Dim w As Double
Dim ext1 As New MapObjects.Rectangle
Dim ext2 As New MapObjects.Rectangle
Dim equal As Integer

'delete all records from w_scr_gcr
Dim dbs As Database
Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
dbs.Execute "DELETE * FROM " _
    & " w_scr_gcr;"
dbs.Close

'weight for cr_gcr
compare1.RecordSource = "frequence_scr"
compare2.RecordSource = "frequence_scr_gcr"
res.RecordSource = "w_scr_gcr"
compare1.Refresh
compare2.Refresh
res.Refresh
compare1.Recordset.MoveFirst
compare2.Recordset.MoveFirst
With compare2.Recordset

    While Not (compare1.Recordset.EOF Or compare2.Recordset.EOF)
        ext1.Left = compare1.Recordset!sleft
        ext1.Right = compare1.Recordset!sright
        ext1.Bottom = compare1.Recordset!sbottom
        ext1.Top = compare1.Recordset!stop
        ext2.Left = !sleft
        ext2.Right = !sright
        ext2.Top = !stop
        ext2.Bottom = !sbottom

        equal = (ext1.Left = ext2.Left) And (ext1.Right =
ext2.Right) And _
            (ext1.Top = ext2.Top) And (ext1.Bottom =
ext2.Bottom)

        If (equal = True) Then
            w = w + (!Cgcr / compare1.Recordset!Cscr) * (!Cgcr /
compare1.Recordset!Cscr)
            .MoveNext
        Else
            .MoveNext
        End If
    If .EOF Then
        w = Sqr(w)
        BeginTrans
        res.Recordset.AddNew
        res.Recordset!sleft = compare1.Recordset!sleft
        res.Recordset!sright = compare1.Recordset!sright
        res.Recordset!stop = compare1.Recordset!stop
        res.Recordset!sbottom = compare1.Recordset!sbottom
        res.Recordset!Weight = w
        res.Recordset.Update
        w = 0
        CommitTrans
        .MoveFirst
        compare1.Recordset.MoveNext
    End If
End With

```

```

        End If

    Wend

End With
End Sub

Private Sub w_scr_gscr()

    Dim w As Double
    Dim ext1 As New MapObjects.Rectangle
    Dim ext2 As New MapObjects.Rectangle
    Dim equal As Integer

    'delete all records from w_scr_gscr
    Dim dbs As Database
    Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
    dbs.Execute "DELETE * FROM " _
        & " w_scr_gscr;"
    dbs.Close

    'weight for scr_gscr
    compare1.RecordSource = "frequence_scr"
    compare2.RecordSource = "frequence_scr_gscr"
    res.RecordSource = "w_scr_gscr"
    compare1.Refresh
    compare2.Refresh
    res.Refresh
    compare1.Recordset.MoveFirst
    compare2.Recordset.MoveFirst
    With compare2.Recordset

        While Not (compare1.Recordset.EOF Or compare2.Recordset.EOF)
            ext1.Left = compare1.Recordset!sleft
            ext1.Right = compare1.Recordset!sright
            ext1.Bottom = compare1.Recordset!sbottom
            ext1.Top = compare1.Recordset!stop
            ext2.Left = !sleft
            ext2.Right = !sright
            ext2.Top = !stop
            ext2.Bottom = !sbottom

            equal = (ext1.Left = ext2.Left) And (ext1.Right =
ext2.Right) And _
                (ext1.Top = ext2.Top) And (ext1.Bottom =
ext2.Bottom)

            If (equal = True) Then
                w = w + (!Cgscr / compare1.Recordset!Cscr) * (!Cgscr
/ compare1.Recordset!Cscr)
                .MoveNext
            Else
                .MoveNext
            End If
            If .EOF Then
                w = Sqr(w)
                BeginTrans
                res.Recordset.AddNew
                res.Recordset!sleft = compare1.Recordset!sleft

```

```

        res.Recordset!sright = compare1.Recordset!sright
        res.Recordset!stop = compare1.Recordset!stop
        res.Recordset!sbottom = compare1.Recordset!sbottom
        res.Recordset!Weight = w
        res.Recordset.Update
        w = 0
        CommitTrans
        .MoveFirst
        compare1.Recordset.MoveNext
    End If

Wend

End With
End Sub

Private Sub w_scr_gsel()

    Dim w As Double
    Dim ext1 As New MapObjects.Rectangle
    Dim ext2 As New MapObjects.Rectangle
    Dim equal As Integer

    'delete all records from w_scr_gsel
    Dim dbs As Database
    Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
    dbs.Execute "DELETE * FROM " _
        & " w_scr_gsel;"
    dbs.Close

    'weight for cr_gcr
    compare1.RecordSource = "frequence_scr"
    compare2.RecordSource = "frequence_scr_gsel"
    res.RecordSource = "w_scr_gsel"
    compare1.Refresh
    compare2.Refresh
    res.Refresh
    compare1.Recordset.MoveFirst
    compare2.Recordset.MoveFirst
    With compare2.Recordset

        While Not (compare1.Recordset.EOF Or compare2.Recordset.EOF)
            ext1.Left = compare1.Recordset!sleft
            ext1.Right = compare1.Recordset!sright
            ext1.Bottom = compare1.Recordset!sbottom
            ext1.Top = compare1.Recordset!stop
            ext2.Left = !sleft
            ext2.Right = !sright
            ext2.Top = !stop
            ext2.Bottom = !sbottom

            equal = (ext1.Left = ext2.Left) And (ext1.Right =
ext2.Right) And _
                (ext1.Top = ext2.Top) And (ext1.Bottom =
ext2.Bottom)

            If (equal = True) Then
                w = w + (!Cgsel / compare1.Recordset!Cscr) * (!Cgsel
/ compare1.Recordset!Cscr)
                .MoveNext
            End If
        End While
    End With
End Sub

```



```

Else

    .MoveNext
End If
If .EOF Then
    w = Sqr(w)
    BeginTrans
    res.Recordset.AddNew
    res.Recordset!sleft = compare1.Recordset!sleft
    res.Recordset!sright = compare1.Recordset!sright
    res.Recordset!stop = compare1.Recordset!stop
    res.Recordset!sbottom = compare1.Recordset!sbottom
    res.Recordset!Weight = w
    res.Recordset.Update
    w = 0
    CommitTrans
    .MoveFirst
    compare1.Recordset.MoveNext
End If

Wend

End With
End Sub

Private Sub w_sel_gcr()

    Dim w As Double

    'delete all records from w_sel_gcr
    Dim dbs As Database
    Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
    dbs.Execute "DELETE * FROM " _
        & " w_sel_gcr;"
    dbs.Close

    'weight for sel_gcr
    compare1.RecordSource = "frequence_sel"
    compare2.RecordSource = "frequence_sel_gcr"
    res.RecordSource = "w_sel_gcr"
    compare1.Refresh
    compare2.Refresh
    res.Refresh
    compare1.Recordset.MoveFirst
    compare2.Recordset.MoveFirst
    With compare2.Recordset

        While Not (compare1.Recordset.EOF Or compare2.Recordset.EOF)

            If (compare1.Recordset!sel = !sel) Then

                w = w + (!Cgcr / compare1.Recordset!Csel) * (!Cgcr /
compare1.Recordset!Csel)
                .MoveNext
            Else

                .MoveNext
            End If
            If .EOF Then
                w = Sqr(w)

```

```

        BeginTrans
        res.Recordset.AddNew
        res.Recordset!sel = compare1.Recordset!sel
        res.Recordset!Weight = w
        res.Recordset.Update
        w = 0
        CommitTrans
        .MoveFirst
        compare1.Recordset.MoveNext
    End If

Wend

    End With
End Sub

Private Sub w_sel_gscr()

    Dim w As Double

    'delete all records from w_sel_gscr
    Dim dbs As Database
    Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
    dbs.Execute "DELETE * FROM " _
        & " w_sel_gscr;"
    dbs.Close

    'weight for sel_gscr
    compare1.RecordSource = "frequence_sel"
    compare2.RecordSource = "frequence_sel_gscr"
    res.RecordSource = "w_sel_gscr"
    compare1.Refresh
    compare2.Refresh
    res.Refresh
    compare1.Recordset.MoveFirst
    compare2.Recordset.MoveFirst
    With compare2.Recordset

        While Not (compare1.Recordset.EOF Or compare2.Recordset.EOF)

            If (compare1.Recordset!sel = !sel) Then

                w = w + (!Cgscr / compare1.Recordset!Csel) * (!Cgscr
/ compare1.Recordset!Csel)
                .MoveNext
            Else

                .MoveNext
            End If
            If .EOF Then
                w = Sqr(w)
                BeginTrans
                res.Recordset.AddNew
                res.Recordset!sel = compare1.Recordset!sel
                res.Recordset!Weight = w
                res.Recordset.Update
                w = 0
                CommitTrans
                .MoveFirst
                compare1.Recordset.MoveNext
            End If
        End With
    End Sub

```

```

        End If

    Wend

    End With
End Sub

Private Sub w_sel_gsel()

    Dim w As Double

    'delete all records from w_sel_gsel
    Dim dbs As Database
    Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
    dbs.Execute "DELETE * FROM " _
        & " w_sel_gsel;"
    dbs.Close

    'weight for sel_gsel
    compare1.RecordSource = "frequence_sel"
    compare2.RecordSource = "frequence_sel_gsel"
    res.RecordSource = "w_sel_gsel"
    compare1.Refresh
    compare2.Refresh
    res.Refresh
    compare1.Recordset.MoveFirst
    compare2.Recordset.MoveFirst
    With compare2.Recordset

        While Not (compare1.Recordset.EOF Or compare2.Recordset.EOF)

            If (compare1.Recordset!sel = !sel) Then

                w = w + (!Cgsel_sel / compare1.Recordset!Csel) *
(!Cgsel_sel / compare1.Recordset!Csel)
                .MoveNext
            Else

                .MoveNext
            End If
            If .EOF Then
                w = Sqr(w)
                BeginTrans
                res.Recordset.AddNew
                res.Recordset!sel = compare1.Recordset!sel
                res.Recordset!Weight = w
                res.Recordset.Update
                w = 0
                CommitTrans
                .MoveFirst
                compare1.Recordset.MoveNext
            End If

        Wend

    End With
End Sub

Private Sub d_gcnd_cmd()

```

```

Dim d As Double
Dim fp1 As Integer
Dim fp2 As Integer
Dim fgp1 As Integer
Dim fgp2 As Integer
Dim cg As String
Dim cp1 As String
Dim cp2 As String
Dim sair As Boolean

'delete all records from w_cmd_gcmd
Dim dbs As Database
Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
dbs.Execute "DELETE * FROM " _
& " d_gcmd_cmd;"
dbs.Close

d = 0
f1 = 0
f2 = 0
fgp1 = 0
fgp2 = 0
sair = False

'weight for sel_gsel
compare1.RecordSource = "frequence_cmd"
compare3.RecordSource = "frequence_cmd"
compare2.RecordSource = "select * from frequence_cmd_gcmd order
by gcmd"
res.RecordSource = "d_gcmd_cmd"
compare1.Refresh
compare3.Refresh
compare2.Refresh
res.Refresh
compare1.Recordset.MoveFirst
compare2.Recordset.MoveFirst
compare3.Recordset.MoveFirst
If Not compare3.Recordset.EOF Then
    compare3.Recordset.MoveNext
End If
If Not compare3.Recordset.EOF Then
    With compare2.Recordset
        cg = !gcmd
        cp1 = compare1.Recordset!cmd
        cp2 = compare3.Recordset!cmd
        fp1 = compare1.Recordset!Ccmd
        fp2 = compare3.Recordset!Ccmd
        Do While (Not compare1.Recordset.EOF) And (Not sair)

            Do While Not .EOF
                If !cmd = cp1 Then
                    fgp1 = fgp1 + !Cgcmd_cmd
                Else
                    End If
                If !cmd = cp2 Then
                    fgp2 = fgp2 + !Cgcmd_cmd
                Else
                    End If
                .MoveNext
            End While
        End With
    End If
End If

```

```

        If Not .EOF Then
            If !gcmd <> cg Then
                d = d + (fgp1 / fp1 - fgp2 / fp2) *
(fgp1 / fp1 - fgp2 / fp2)
                fgp1 = 0
                fgp2 = 0
                cg = !gcmd
            Else
                End If
            Else
                Dim v As Double
                v = (fgp1 / fp1 - fgp2 / fp2) * (fgp1 / fp1
- fgp2 / fp2)
                d = d + v
                fgp1 = 0
                fgp2 = 0
            End If
        Loop
    If .EOF Then
        BeginTrans
            res.Recordset.AddNew
            res.Recordset!cmd1 = cp1
            res.Recordset!cmd2 = cp2
            res.Recordset!d = d
            res.Recordset.Update
            res.Recordset.AddNew
            res.Recordset!cmd1 = cp2
            res.Recordset!cmd2 = cp1
            res.Recordset!d = d
            res.Recordset.Update
        CommitTrans
        d = 0
        fgp1 = 0
        fgp2 = 0
        compare3.Recordset.MoveNext
        If compare3.Recordset.EOF Then
            compare1.Recordset.MoveNext
            compare3.Recordset.AbsolutePosition =
compare1.Recordset.AbsolutePosition
            compare3.Recordset.MoveNext
            If compare3.Recordset.EOF Then
                sair = True
            Else
                End If
        End If
        If Not sair = True Then
            .MoveFirst
            cg = !gcmd
            cp1 = compare1.Recordset!cmd
            cp2 = compare3.Recordset!cmd
            fp1 = compare1.Recordset!Ccmd
            fp2 = compare3.Recordset!Ccmd
        Else
            End If
        End If
    End If

    Loop
End With
End If

```

```

End Sub

Private Sub d_gcmd_scr()

    Dim d As Double
    Dim fp1 As Integer
    Dim fp2 As Integer
    Dim fgp1 As Integer
    Dim fgp2 As Integer
    Dim cg As String
    Dim cp1 As New MapObjects.Rectangle
    Dim cp2 As New MapObjects.Rectangle
    Dim cgp As New MapObjects.Rectangle
    Dim sair As Boolean

    'delete all records from d_gcmd_scr
    Dim dbs As Database
    Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
    dbs.Execute "DELETE * FROM " _
        & " d_gcmd_scr;"
    dbs.Close

    d = 0
    f1 = 0
    f2 = 0
    fgp1 = 0
    fgp2 = 0
    sair = False

    compare1.RecordSource = "frequence_scr"
    compare3.RecordSource = "frequence_scr"
    compare2.RecordSource = "select * from frequence_scr_gcmd order
by gcmd"
    res.RecordSource = "d_gcmd_scr"
    compare1.Refresh
    compare3.Refresh
    compare2.Refresh
    res.Refresh
    compare1.Recordset.MoveFirst
    compare2.Recordset.MoveFirst
    compare3.Recordset.MoveFirst
    If Not compare3.Recordset.EOF Then
        compare3.Recordset.MoveNext
    End If
    If Not compare3.Recordset.EOF Then
        With compare2.Recordset
            cg = !gcmd
            cp1.Left = compare1.Recordset!sleft
            cp1.Right = compare1.Recordset!sright
            cp1.Bottom = compare1.Recordset!sbottom
            cp1.Top = compare1.Recordset!stop
            cp2.Left = compare3.Recordset!sleft
            cp2.Right = compare3.Recordset!sright
            cp2.Bottom = compare3.Recordset!sbottom
            cp2.Top = compare3.Recordset!stop
            cgp.Left = !sleft
            cgp.Right = !sright
            cgp.Bottom = !sbottom
            cgp.Top = !stop

```

```

fp1 = compare1.Recordset!Cscr
fp2 = compare3.Recordset!Cscr
Do While (Not compare1.Recordset.EOF) And (Not sair)

Do While Not .EOF
    If equal_extents(cgp, cp1) Then
        fgp1 = fgp1 + !Cgcmd
    Else
    End If
    If equal_extents(cgp, cp2) Then
        fgp2 = fgp2 + !Cgcmd
    Else
    End If
    .MoveNext

    If Not .EOF Then
        cgp.Left = !sleft
        cgp.Right = !sright
        cgp.Bottom = !sbottom
        cgp.Top = !stop

        If !gcmd <> cg Then
            d = d + (fgp1 / fp1 - fgp2 / fp2) *
(fgp1 / fp1 - fgp2 / fp2)
            fgp1 = 0
            fgp2 = 0
            cg = !gcmd
        Else
        End If
    Else
        Dim v As Double
        v = (fgp1 / fp1 - fgp2 / fp2) * (fgp1 / fp1
- fgp2 / fp2)

        d = d + v
        fgp1 = 0
        fgp2 = 0

    End If
Loop
If .EOF Then
    BeginTrans
    res.Recordset.AddNew

    res.Recordset!sleft1 = cp1.Left
    res.Recordset!sright1 = cp1.Right
    res.Recordset!sbottom1 = cp1.Bottom
    res.Recordset!stop1 = cp1.Top
    res.Recordset!sleft2 = cp2.Left
    res.Recordset!sright2 = cp2.Right
    res.Recordset!sbottom2 = cp2.Bottom
    res.Recordset!stop2 = cp2.Top

    res.Recordset!d = d
    res.Recordset.Update
    res.Recordset.AddNew
    res.Recordset!sleft1 = cp2.Left
    res.Recordset!sright1 = cp2.Right
    res.Recordset!sbottom1 = cp2.Bottom
    res.Recordset!stop1 = cp2.Top
    res.Recordset!sleft2 = cp1.Left
    res.Recordset!sright2 = cp1.Right

```

```

        res.Recordset!sbottom2 = cp1.Bottom
        res.Recordset!stop2 = cp1.Top

        res.Recordset!d = d
        res.Recordset.Update
    CommitTrans
    d = 0
    fgp1 = 0
    fgp2 = 0
    compare3.Recordset.MoveNext
    If compare3.Recordset.EOF Then
        compare1.Recordset.MoveNext
        compare3.Recordset.AbsolutePosition =
compare1.Recordset.AbsolutePosition
        compare3.Recordset.MoveNext
        If compare3.Recordset.EOF Then
            sair = True
        Else
            End If
    End If
    If Not sair = True Then
        .MoveFirst
        cgp.Left = !sleft
        cgp.Right = !sright
        cgp.Bottom = !sbottom
        cgp.Top = !stop

        cg = !gcmd
        cp1.Left = compare1.Recordset!sleft
        cp1.Right = compare1.Recordset!sright
        cp1.Bottom = compare1.Recordset!sbottom
        cp1.Top = compare1.Recordset!stop
        cp2.Left = compare3.Recordset!sleft
        cp2.Right = compare3.Recordset!sright
        cp2.Bottom = compare3.Recordset!sbottom
        cp2.Top = compare3.Recordset!stop
        cgp.Left = !sleft
        cgp.Right = !sright
        cgp.Bottom = !sbottom
        cgp.Top = !stop

        fp1 = compare1.Recordset!Cscr
        fp2 = compare3.Recordset!Cscr
    Else
        End If
    End If

        Loop
    End With
End If
End Sub

```

```
Private Sub d_gsel_cmd()
```

```

    Dim d As Double
    Dim fp1 As Integer
    Dim fp2 As Integer
    Dim fgp1 As Integer
    Dim fgp2 As Integer
    Dim cg As Boolean
    Dim cp1 As String

```



```

Dim cp2 As String
Dim sair As Boolean

'delete all records from d_gsel_cmd
Dim dbs As Database
Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
dbs.Execute "DELETE * FROM " _
    & " d_gsel_cmd;"
dbs.Close

d = 0
f1 = 0
f2 = 0
fgp1 = 0
fgp2 = 0
sair = False

compare1.RecordSource = "frequence_cmd"
compare3.RecordSource = "frequence_cmd"
compare2.RecordSource = "select * from frequence_cmd_gsel order
by gsel"
res.RecordSource = "d_gsel_cmd"
compare1.Refresh
compare3.Refresh
compare2.Refresh
res.Refresh
compare1.Recordset.MoveFirst
compare2.Recordset.MoveFirst
compare3.Recordset.MoveFirst
If Not compare3.Recordset.EOF Then
    compare3.Recordset.MoveNext
End If
If Not compare3.Recordset.EOF Then
    With compare2.Recordset
        cg = !gsel
        cp1 = compare1.Recordset!cmd
        cp2 = compare3.Recordset!cmd
        fp1 = compare1.Recordset!Ccmd
        fp2 = compare3.Recordset!Ccmd
        Do While (Not compare1.Recordset.EOF) And (Not sair)

            Do While Not .EOF
                If !cmd = cp1 Then
                    fgp1 = fgp1 + !Cgsel_cmd
                Else
                    End If
                If !cmd = cp2 Then
                    fgp2 = fgp2 + !Cgsel_cmd
                Else
                    End If
                .MoveNext
                If Not .EOF Then

                    If !gsel <> cg Then
                        d = d + (fgp1 / fp1 - fgp2 / fp2) *
(fgp1 / fp1 - fgp2 / fp2)
                        fgp1 = 0
                        fgp2 = 0
                        cg = !gsel
                    
```

```

        Else
        End If
    Else
        Dim v As Double
        v = (fgp1 / fp1 - fgp2 / fp2) * (fgp1 / fp1
- fgp2 / fp2)

        d = d + v
        fgp1 = 0
        fgp2 = 0

    End If

Loop
If .EOF Then
    BeginTrans
        res.Recordset.AddNew
        res.Recordset!cmd1 = cp1
        res.Recordset!cmd2 = cp2
        res.Recordset!d = d
        res.Recordset.Update
        res.Recordset.AddNew
        res.Recordset!cmd1 = cp2
        res.Recordset!cmd2 = cp1
        res.Recordset!d = d
        res.Recordset.Update
    CommitTrans
    d = 0
    fgp1 = 0
    fgp2 = 0
    compare3.Recordset.MoveNext
    If compare3.Recordset.EOF Then
        compare1.Recordset.MoveNext
        compare3.Recordset.AbsolutePosition =
compare1.Recordset.AbsolutePosition
        compare3.Recordset.MoveNext
        If compare3.Recordset.EOF Then
            sair = True
        Else
        End If
    End If
    If Not sair = True Then
        .MoveFirst
        cg = !gsel
        cp1 = compare1.Recordset!cmd
        cp2 = compare3.Recordset!cmd
        fp1 = compare1.Recordset!Ccmd
        fp2 = compare3.Recordset!Ccmd
    Else
    End If
End If

    Loop
End With
End If
End Sub

Private Sub d_gscr_cmd()

    Dim d As Double
    Dim fp1 As Integer
    Dim fp2 As Integer
    Dim fgp1 As Integer

```

```

Dim fgp2 As Integer
Dim cg As New MapObjects.Rectangle
Dim cgp As New MapObjects.Rectangle
Dim cp1 As String
Dim cp2 As String
Dim sair As Boolean

'delete all records from d_gscr_cmd
Dim dbs As Database
Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
dbs.Execute "DELETE * FROM " _
    & " d_gscr_cmd;"
dbs.Close

d = 0
f1 = 0
f2 = 0
fgp1 = 0
fgp2 = 0
sair = False

compare1.RecordSource = "frequence_cmd"
compare3.RecordSource = "frequence_cmd"
compare2.RecordSource = "select * from frequence_cmd_gscr order
by gsleft,gsright,gsbottom,gstop"
res.RecordSource = "d_gscr_cmd"
compare1.Refresh
compare3.Refresh
compare2.Refresh
res.Refresh
compare1.Recordset.MoveFirst
compare2.Recordset.MoveFirst
compare3.Recordset.MoveFirst
If Not compare3.Recordset.EOF Then
    compare3.Recordset.MoveNext
End If
If Not compare3.Recordset.EOF Then
    With compare2.Recordset
        cg.Left = !gsleft
        cg.Right = !gsright
        cg.Bottom = !gsbottom
        cg.Top = !gstop
        cp1 = compare1.Recordset!cmd
        cp2 = compare3.Recordset!cmd
        fp1 = compare1.Recordset!Ccmd
        fp2 = compare3.Recordset!Ccmd
        Do While (Not compare1.Recordset.EOF) And (Not sair)

        Do While Not .EOF
            If !cmd = cp1 Then
                fgp1 = fgp1 + !Cgscr
            Else
                End If
            If !cmd = cp2 Then
                fgp2 = fgp2 + !Cgscr
            Else
                End If
            .MoveNext
            If Not .EOF Then

```

```

        cgp.Left = !gsleft
        cgp.Right = !gsright
        cgp.Bottom = !gsbottom
        cgp.Top = !gstop
        If Not equal_extents(cgp, cg) Then
            d = d + (fgp1 / fp1 - fgp2 / fp2) *
(fgp1 / fp1 - fgp2 / fp2)
            fgp1 = 0
            fgp2 = 0
            cg.Left = !gsleft
            cg.Right = !gsright
            cg.Bottom = !gsbottom
            cg.Top = !gstop
        Else
            End If
        Else
            Dim v As Double
            v = (fgp1 / fp1 - fgp2 / fp2) * (fgp1 / fp1
- fgp2 / fp2)
            d = d + v
            fgp1 = 0
            fgp2 = 0

            End If
    Loop
    If .EOF Then
        BeginTrans
            If (d > 1) Then
                d = 1
            Else
                End If
            res.Recordset.AddNew
            res.Recordset!cmd1 = cp1
            res.Recordset!cmd2 = cp2
            res.Recordset!d = d
            res.Recordset.Update
            res.Recordset.AddNew
            res.Recordset!cmd1 = cp2
            res.Recordset!cmd2 = cp1
            res.Recordset!d = d
            res.Recordset.Update
        CommitTrans
        d = 0
        fgp1 = 0
        fgp2 = 0
        compare3.Recordset.MoveNext
        If compare3.Recordset.EOF Then
            compare1.Recordset.MoveNext
            compare3.Recordset.AbsolutePosition =
compare1.Recordset.AbsolutePosition
            compare3.Recordset.MoveNext
            If compare3.Recordset.EOF Then
                sair = True
            Else
                End If
        End If
    End If
    If Not sair = True Then
        .MoveFirst
        cg.Left = !gsleft
        cg.Right = !gsright
        cg.Bottom = !gsbottom

```

```

        cg.Top = !gstop
        cp1 = compare1.Recordset!cmd
        cp2 = compare3.Recordset!cmd
        fp1 = compare1.Recordset!Ccmd
        fp2 = compare3.Recordset!Ccmd
    Else
    End If
End If

    Loop
End With
End If
End Sub

Private Sub d_gscr_sel()

    Dim d As Double
    Dim fp1 As Integer
    Dim fp2 As Integer
    Dim fgp1 As Integer
    Dim fgp2 As Integer
    Dim cg As New MapObjects.Rectangle
    Dim cgp As New MapObjects.Rectangle
    Dim cp1 As Boolean
    Dim cp2 As Boolean
    Dim sair As Boolean

    'delete all records from d_gscr_sel
    Dim dbs As Database
    Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
    dbs.Execute "DELETE * FROM " _
        & " d_gscr_sel;"
    dbs.Close

    d = 0
    f1 = 0
    f2 = 0
    fgp1 = 0
    fgp2 = 0
    sair = False

    compare1.RecordSource = "frequence_sel"
    compare3.RecordSource = "frequence_sel"
    compare2.RecordSource = "select * from frequence_sel_gscr order
by gsleft,gsright,gsbottom,gstop"
    res.RecordSource = "d_gscr_sel"
    compare1.Refresh
    compare3.Refresh
    compare2.Refresh
    res.Refresh
    compare1.Recordset.MoveFirst
    compare2.Recordset.MoveFirst
    compare3.Recordset.MoveFirst
    If Not compare3.Recordset.EOF Then
        compare3.Recordset.MoveNext
    End If
    If Not compare3.Recordset.EOF Then
        With compare2.Recordset
            cg.Left = !gsleft
            cg.Right = !gsright

```

```

cg.Bottom = !gsbottom
cg.Top = !gstop
cp1 = compare1.Recordset!sel
cp2 = compare3.Recordset!sel
fp1 = compare1.Recordset!Csel
fp2 = compare3.Recordset!Csel
Do While (Not compare1.Recordset.EOF) And (Not sair)

Do While Not .EOF
    If !sel = cp1 Then
        fgp1 = fgp1 + !Cgscr
    Else
    End If
    If !sel = cp2 Then
        fgp2 = fgp2 + !Cgscr
    Else
    End If
    .MoveNext
    If Not .EOF Then
        cgp.Left = !gsleft
        cgp.Right = !gsright
        cgp.Bottom = !gsbottom
        cgp.Top = !gstop
        If Not equal_extents(cgp, cg) Then
            d = d + (fgp1 / fp1 - fgp2 / fp2) *
(fgp1 / fp1 - fgp2 / fp2)
            fgp1 = 0
            fgp2 = 0
            cg.Left = !gsleft
            cg.Right = !gsright
            cg.Bottom = !gsbottom
            cg.Top = !gstop
        Else
        End If
    Else
        Dim v As Double
        v = (fgp1 / fp1 - fgp2 / fp2) * (fgp1 / fp1
- fgp2 / fp2)
        d = d + v
        fgp1 = 0
        fgp2 = 0
    End If
Loop
If .EOF Then
    BeginTrans
    If d > 1 Then
        d = 1
    Else
    End If
    res.Recordset.AddNew
    res.Recordset!sel1 = cp1
    res.Recordset!sel2 = cp2
    res.Recordset!d = d
    res.Recordset.Update
    res.Recordset.AddNew
    res.Recordset!sel1 = cp2
    res.Recordset!sel2 = cp1
    res.Recordset!d = d
    res.Recordset.Update
    CommitTrans

```

```

        d = 0
        fgp1 = 0
        fgp2 = 0
        compare3.Recordset.MoveNext
        If compare3.Recordset.EOF Then
            compare1.Recordset.MoveNext
            compare3.Recordset.AbsolutePosition =
compare1.Recordset.AbsolutePosition
            compare3.Recordset.MoveNext
            If compare3.Recordset.EOF Then
                sair = True
            Else
                End If
        End If
        If Not sair = True Then
            .MoveFirst
            cg.Left = !gsleft
            cg.Right = !gsright
            cg.Bottom = !gsbottom
            cg.Top = !gstop
            cp1 = compare1.Recordset!sel
            cp2 = compare3.Recordset!sel
            fp1 = compare1.Recordset!Csel
            fp2 = compare3.Recordset!Csel
        Else
            End If
        End If
    Loop
End With
End If
End Sub

```

```
Private Sub d_gscr_cr()
```

```

    Dim d As Double
    Dim fp1 As Integer
    Dim fp2 As Integer
    Dim fgp1 As Integer
    Dim fgp2 As Integer
    Dim cg As New MapObjects.Rectangle
    Dim cgp1 As New MapObjects.Rectangle
    Dim cp1 As New MapObjects.Rectangle
    Dim cp2 As New MapObjects.Rectangle
    Dim cgp As New MapObjects.Rectangle
    Dim sair As Boolean

    'delete all records from d_gscr_cr
    Dim dbs As Database
    Set dbs =
OpenDatabase("c:\projects\webagent\experience\experience.mdb")
    dbs.Execute "DELETE * FROM " _
        & " d_gscr_cr;"
    dbs.Close

    d = 0
    f1 = 0
    f2 = 0
    fgp1 = 0
    fgp2 = 0
    sair = False

```

```

compare1.RecordSource = "frequence_cr"
compare3.RecordSource = "frequence_cr"
compare2.RecordSource = "select * from frequence_cr_gscr order
by gsleft,gsright,gsbottom,gstop"
res.RecordSource = "d_gscr_cr"
compare1.Refresh
compare3.Refresh
compare2.Refresh
res.Refresh
compare1.Recordset.MoveFirst
compare2.Recordset.MoveFirst
compare3.Recordset.MoveFirst
If Not compare3.Recordset.EOF Then
    compare3.Recordset.MoveNext
End If
If Not compare3.Recordset.EOF Then
    With compare2.Recordset
        cg.Left = !gsleft
        cg.Right = !gsright
        cg.Bottom = !gsbottom
        cg.Top = !gstop
        cp1.Left = compare1.Recordset!Left
        cp1.Right = compare1.Recordset!Right
        cp1.Bottom = compare1.Recordset!Bottom
        cp1.Top = compare1.Recordset!Top
        cp2.Left = compare3.Recordset!Left
        cp2.Right = compare3.Recordset!Right
        cp2.Bottom = compare3.Recordset!Bottom
        cp2.Top = compare3.Recordset!Top
        cgp.Left = !Left
        cgp.Right = !Right
        cgp.Bottom = !Bottom
        cgp.Top = !Top

        fp1 = compare1.Recordset!Ccr
        fp2 = compare3.Recordset!Ccr
        Do While (Not compare1.Recordset.EOF) And (Not sair)

        Do While Not .EOF
            If equal_extents(cgp, cp1) Then
                fgpl = fgpl + !Cgscr
            Else
                End If
            If equal_extents(cgp, cp2) Then
                fgp2 = fgp2 + !Cgscr
            Else
                End If
            .MoveNext

            If Not .EOF Then
                cgpl.Left = !gsleft
                cgpl.Right = !gsright
                cgpl.Bottom = !gsbottom
                cgpl.Top = !gstop

                cgp.Left = !Left
                cgp.Right = !Right
                cgp.Bottom = !Bottom
                cgp.Top = !Top

```



```

                If Not equal_extents(cgp1, cg) Then
                    d = d + (fgp1 / fp1 - fgp2 / fp2) *
(fgp1 / fp1 - fgp2 / fp2)
                    fgp1 = 0
                    fgp2 = 0
                    cg.Left = !gsleft
                    cg.Right = !gsright
                    cg.Bottom = !gsbottom
                    cg.Top = !gstop
                Else
                End If
            Else
                Dim v As Double
                v = (fgp1 / fp1 - fgp2 / fp2) * (fgp1 / fp1
- fgp2 / fp2)

                d = d + v
                fgp1 = 0
                fgp2 = 0

            End If
        Loop
    If .EOF Then
        BeginTrans
            If (d > 1) Then
                d = 1
            Else
            End If

            res.Recordset.AddNew

            res.Recordset!left1 = cp1.Left
            res.Recordset!right1 = cp1.Right
            res.Recordset!bottom1 = cp1.Bottom
            res.Recordset!top1 = cp1.Top
            res.Recordset!left2 = cp2.Left
            res.Recordset!right2 = cp2.Right
            res.Recordset!bottom2 = cp2.Bottom
            res.Recordset!top2 = cp2.Top

            res.Recordset!d = d
            res.Recordset.Update
            res.Recordset.AddNew
            res.Recordset!left1 = cp2.Left
            res.Recordset!right1 = cp2.Right
            res.Recordset!bottom1 = cp2.Bottom
            res.Recordset!top1 = cp2.Top
            res.Recordset!left2 = cp1.Left
            res.Recordset!right2 = cp1.Right
            res.Recordset!bottom2 = cp1.Bottom
            res.Recordset!top2 = cp1.Top

            res.Recordset!d = d
            res.Recordset.Update
        CommitTrans
        d = 0
        fgp1 = 0
        fgp2 = 0
        compare3.Recordset.MoveNext
        If compare3.Recordset.EOF Then
            compare1.Recordset.MoveNext

```

```

        compare3.Recordset.AbsolutePosition =
compare1.Recordset.AbsolutePosition
        compare3.Recordset.MoveNext
        If compare3.Recordset.EOF Then
            sair = True
        Else
            End If
    End If
    If Not sair = True Then
        .MoveFirst
        cgpl.Left = !gsleft
        cgpl.Right = !gsright
        cgpl.Bottom = !gsbottom
        cgpl.Top = !gstop

        cg.Left = !gsleft
        cg.Right = !gsright
        cg.Bottom = !gsbottom
        cg.Top = !gstop

        cp1.Left = compare1.Recordset!Left
        cp1.Right = compare1.Recordset!Right
        cp1.Bottom = compare1.Recordset!Bottom
        cp1.Top = compare1.Recordset!Top

        cp2.Left = compare3.Recordset!Left
        cp2.Right = compare3.Recordset!Right
        cp2.Bottom = compare3.Recordset!Bottom
        cp2.Top = compare3.Recordset!Top

        cgp.Left = !Left
        cgp.Right = !Right
        cgp.Bottom = !Bottom
        cgp.Top = !Top

        fp1 = compare1.Recordset!Ccr
        fp2 = compare3.Recordset!Ccr
    Else
        End If
    End If

        Loop
    End With
    End If
End Sub

```

Appendix D: Car Park code

Car Class

```
import java.awt.Point;
import java.awt.Color;
import java.util.Random;

/* import bitpix.agent.*;
import bitpix.think.*;

*/

public class Car /*extends RbAgent*/ {

    /*attributes of the car */
    public long creation_time;
    public Point creation_location;
    public Point current_location;
    public int speed;
    public double priority_distance_to_exit;
    public double priority_time_to_park;
    public long parking_limit;
    public boolean in_car_park;
    public boolean parked;
    public boolean exiting;
    private CarThread car_t;
    public boolean forward;
    public String name;

    /* constructor methods according to RbAgent and initializing
    car's attributes */
    public Car(Point p, int s, double pde, double ptp, boolean pa,
boolean e, String n)
    {
/*      super(); */
        this.initializeCar(p,s,pde,ptp,pa,e,n);
    }

    private void initializeCar(Point p, int s, double pde, double
ptp, boolean pa, boolean e, String n)
    {
        System.out.println("Car "+n+" Creating car ..."+n);

        this.creation_time=randomMillis()+TimeThread.current_time; /*
seconds in milliseconds */
        System.out.println("Car "+n+" Creation_time
"+this.creation_time);
        this.creation_location=p;
        System.out.println("Car "+n+" Creation location
"+p.x+", "+p.y);
        this.current_location=new Point(p.x, p.y);
        System.out.println("Car "+n+" Current location
"+p.x+", "+p.y);
        this.speed=s;
        System.out.println("Car "+n+" Speed "+s);
        this.priority_distance_to_exit=pde;
```

```

        this.priority_time_to_park=ptp;
        /* generating how long car will be in Car park - less
that 60 seconds*/
        this.parking_limit=Math.abs(randomMillis());
        System.out.println("Car "+n+" parking time
"+this.parking_limit);
        if (this.creation_time < TimeThread.current_time)
            {
                this.in_car_park=true;
            }
        else {
            this.in_car_park=false;
        }
        System.out.println("Car "+n+" Currently in car park ?
"+this.in_car_park);
        this.parked=pa;
        System.out.println("Car "+n+" Currently Parked ?
"+this.parked);
        this.exiting=e;
        System.out.println("Car "+n+" Currently exiting ?
"+this.exiting);
        this.forward=true;
        System.out.println("Car "+n+" Currently moving forward ?
"+this.forward);
        this.name=n;
    }

    private long randomMillis()
    /* gives a number of seconds in milliseconds */
    {
        Random ra=new Random();
        long pl1;
        long t=ra.nextLong();
        if (t < 0)
            {
                t=-1;
            }
        if (t >= 60)
            {
                pl1=t;
                t= t/60;
                t=(pl1-t*60);
            }
        return (long)t*1000*4;
    }

    public Point nextPosition(Point p)
    {
        double b=0;;
        double a;
        double c;
        double sin_alfa;
        double cos_alfa;
        double alfa;
        int next_x;
        int next_y;
        Point loc=this.current_location;
        b= Math.abs (Math.sqrt(((Math.abs(p.x-
loc.x))^2)+((Math.abs(-p.y+loc.y))^2)));
        if (Math.abs(b-28) < 28)

```

```

        {
            return p;
        }
    else
    {

        a=-p.y+loc.y;
        c=p.x-loc.x;
        sin_alfa=a/b;
        cos_alfa=c/b;
        alfa=Math.asin(sin_alfa);
        next_x=(int) (- (28*sin_alfa)-loc.x);
        next_y=(int) (- (28*cos_alfa)+loc.y);
        return new Point(next_x,next_y);
    }
}

/* Method that moves a car towards a point */
public synchronized boolean moveTowards(Point p)
{
    double b=0;
    double a;
    double c;
    double sin_alfa;
    double cos_alfa;
    double alfa;
    int next_x;
    int next_y;
    if ((p.x==0) && (p.y==0))
    {
        System.out.println(this.name+" barraca !!!");
    }
    Point loc=this.current_location;
    b= Math.abs (Math.sqrt(((Math.abs(p.x-loc.x))*(Math.abs(p.x-loc.x)))+(Math.abs(-p.y+loc.y))*(Math.abs(-p.y+loc.y)))));
    if (Math.abs(b-28) < 28)
    {
        this.current_location=p;
        return true;
    }
    else
    {
        /* p - objetivo
           p.y - e -p.y em trigonometria
           loc.y e -loc.y em trigonometria */
        a=-p.y+loc.y;
        c=p.x-loc.x;
        sin_alfa=a/b;
        cos_alfa=c/b;
        alfa=Math.asin(sin_alfa);
        next_x=(int) (28*cos_alfa+loc.x);
        next_y=(int) (- (28*sin_alfa)+loc.y);
        this.current_location.x=next_x;
        this.current_location.y=next_y;
        System.out.println("Car "+this.name+" is going
to"+p.x+", "+p.y);
        System.out.println("Car "+this.name+" moved
to"+this.current_location.x+", "+this.current_location.y);
        return false;
    }
}

```

```

}

/* Method that attempts to move a car forward */
public synchronized boolean moveForward(int sp) {
    if (!this.forward)
    {
        this.forward=true;
    }
    if (this.current_location.y==320)
    {
        this.current_location.y=292;
        System.out.println("Car "+this.name+" Current
location "+this.current_location.x+", "+this.current_location.y);
        return true;
    }
    else
    {
        if (this.current_location.y==0)
        {
            System.out.println("Car "+this.name+"
Current location
"+this.current_location.x+", "+this.current_location.y);
            return false;
        }
        else
        {
            if (this.current_location.y-32 < 0)
            {
                System.out.println("Car "+this.name+"
Current location
"+this.current_location.x+", "+this.current_location.y);
                return false;
            }
            else
            {
                this.current_location.y=this.current_location.y-32;
                System.out.println("Car "+this.name+"
Current location
"+this.current_location.x+", "+this.current_location.y);
                return true;
            }
        }
    }
}

public Point forwardPosition(int sp) {
    Point next=new Point(this.current_location.x,
this.current_location.y);
    if (next.y==320)
    {
        next.y=292;
        System.out.println("Car "+this.name+" Next
location "+next.x+", "+next.y);
        return next;
    }
    else
    {

```

```

        if (next.y-32 >=0)
            {
                next.y=next.y-32;
                System.out.println("Car "+this.name+"
Next location "+next.x+", "+next.y);
                return next;
            }
        else
        {
            System.out.println("Car "+this.name+" Next
location -1, -1");
            return new Point(-1, -1);
        }
    }
}

/* Method that attempts to move a car back */
public synchronized boolean moveBack(int sp) {

    if (this.forward)
    {
        this.forward=false;
    }
    if (this.current_location.y==292)
    {
        this.current_location.y=320;
        System.out.println("Car "+this.name+" Current
location "+this.current_location.x+", "+this.current_location.y);
        return true;
    }
    else
    {
        if (this.current_location.y==320)
        {
            System.out.println("Car "+this.name+"
Current location
"+this.current_location.x+", "+this.current_location.y);
            return false;
        }
        else
        {

            this.current_location.y=this.current_location.y+32;
            System.out.println("Car "+this.name+"
Current location
"+this.current_location.x+", "+this.current_location.y);
            return true;
        }
    }
}

public synchronized Point backPosition(int sp) {

    Point next=new Point(this.current_location.x,
this.current_location.y);
    if (next.y==292)
    {
        next.y=320;
        System.out.println("Car "+this.name+" Next
location "+next.x+", "+next.y);
    }
}

```

```

        return next;
    }
    else
    {
        if (next.y==320)
        {
            System.out.println("Car "+this.name+" Next
location "+next.x+", "+next.y);
            return new Point(-1,-1);
        }
        else
        {
            next.y=next.y+32;
            System.out.println("Car "+this.name+" Next
location "+next.x+", "+next.y);
            return next;
        }
    }
}

public synchronized boolean park(Space spa)

{

    if (spa.park(this))
    {
        System.out.println("Car "+this.name+" has
succceccfully parked at "+spa.x+", "+spa.y);
        this.parked=true;
        return true;
    }
    else
    {
        System.out.println("Car "+this.name+" has
unsuccessfully tried to park at "+spa.x+", "+spa.y);
        return false;
    }
}

public boolean carInFront(CarPark cp)
{
    int i;
    for (i=0; i < cp.car_threads.size(); i=i+1)
    {
        CarThread
ct=(CarThread)cp.car_threads.elementAt(1);
        Car c=ct.owner;
        if (c.current_location.y ==
this.current_location.y - 32)
        {
            System.out.println("Car "+this.name+" There
is a car in front of car "+this.name);
            return true;
        }
    }
    System.out.println("Car "+this.name+" No car in front of
car "+this.name);
    return false;
}
}

```



```

public Space nearestSpaceInCarPark(CarPark cp)
{
    Point p=this.current_location;
    System.out.println("Car park: "+"Looking for nearest
space in car park for location: "+p.x+","+p.y);
    int i;
    Space sdp=new Space(0,0,false,10,10);
    int n_ele=cp.spaces.size();
    for (i=0;i < n_ele;i=i+1)
    {
        Space x= (Space) cp.spaces.elementAt(i);
        if (i == 0)
        {
            sdp=x;
        }
        else
        {
            long d1=x.distanceTo(p);
            long d2=sdp.distanceTo(p);
            if (d1<d2)
            {
                sdp=x;
            }
        }
    }
    System.out.println("Car park: "+"Nearest space is
"+sdp.x+","+sdp.y);
    return sdp;
}

Space nearestSpaceToExit(Space s1, Space s2, Point
exit_location)
{
    long dist1=s1.distanceTo(exit_location);
    long dist2=s2.distanceTo(exit_location);

    if (dist1 < dist2)
    {
        return s1;
    }
    else
    {
        return s2;
    }
}

public boolean birth(CarThread ct)
{
    while (this.creation_time > TimeThread.current_time)
    {
        try {
            System.out.println("Car "+this.name+": Going
to sleep ...waiting for birth");
            ct.sleep(10000);
        }
        catch (InterruptedException i)
        {
            return false;
        }
    }
}

```

```

        }
        return true;
    }

    public boolean unpark(CarPark cp, CarThread ct, long
parking_time, long park_limit, Space s)
    {
        if (TimeThread.current_time <
parking_time+parking_limit)
        {
            try {
                System.out.println("Car "+this.name+": Going
to sleep ...waiting unparking");
                ct.sleep(10000);
            }
            catch (InterruptedException i)
            {
                return false;
            }
            return false;
        }
        else
        {
            this.parked=false;
            this.exiting=true;
            this.forward=true;
            cp.removeParkedCar();
            System.out.println("Unparked ... leaving car
park");

            s.exit();
            return true;
        }
    }

    public void exit()
    {
        this.current_location=new Point(0,320);
        this.in_car_park=false;
        this.parked=false;
        this.exiting=false;
        this.forward=true;
        System.out.println(this.name+" leaving car park");
    }
}

```

Car Park Class

```
import java.awt.Point;
import java.awt.Polygon;
import java.util.*;

public class CarPark {

    public Polygon shape;
    public String name=new String ();
    public int maxncars;
    public Vector spaces;
    public int full_spaces;
    public boolean full;
    public boolean coll_detect;
    public int n_entrances;
    public Vector entrances;
    public int n_exits;
    public Vector exits;
    public Vector car_threads=new Vector(15);

    /* constructor methods */
    public CarPark(Polygon p, String n, int max, int nen, Vector
en, int nex, Vector ex, Vector car_ts)
    {

        this.shape=p;
        this.name=n;
        System.out.println("Car park: "+"Car Park name:
"+this.name);
        this.maxncars=max;
        this.spaces=new Vector(10,10);
        this.full_spaces=0;
        System.out.println("Car park: "+"Maximum number of cars:
"+this.maxncars);
        System.out.println("Car park: "+"Full Spaces
"+this.full_spaces);
        this.full=false;
        System.out.println("Car park: "+"Car park is full
"+this.full);
        this.coll_detect = false;
        this.defineEntrances(nen,en);
        this.defineExits(nex,ex);
        this.car_threads=car_ts;
        System.out.println("Car park: "+"Car Park has been
created");
    }

    public void addSpace(Space s)
    {
        this.spaces.addElement(s);
        System.out.println("Car park: "+"Space has been added to
car park: "+s.x+", "+s.y);
    }

    public void defineEntrances(int ne, Vector e)
    {
        this.n_entrances=ne;
    }
}
```

```

        this.entrances=e;
    }

    public void defineExits(int nex, Vector ex)
    {
        this.n_exits=nex;
        this.exits=ex;
    }

    public void addThreadToPark(CarThread ct)
    {
        this.car_threads.addElement(ct);
        System.out.println("Car park: "+"Car Thread has been
added to Car Park");
    }

    public Space firstEmptySpace()
    {
        Space f;
        int i=0;
        while (((Space)this.spaces.elementAt(i)).full) &&
(i<this.spaces.size()))
        {
            if (i < this.spaces.size())
            {
                i=i+1;
            }
        }
        return (Space)this.spaces.elementAt(i);
    }

    public Space nearestEmptySpaceNearestToExit(Car c)
    {
        int i=0;
        Space min=this.firstEmptySpace();

        for (i=0; i < this.spaces.size(); i=i+1)
        {
            Space s=(Space)this.spaces.elementAt(i);
            if (!s.full)
            {
                Point saida=(Point)this.exits.elementAt(0);
                long d1=min.distanceTo(saida);
                long d2=s.distanceTo(saida);
                if (d2<d1)
                {
                    min=s;
                }
            }
        }
        return min;
    }

    public Space nearestSpaceInCarPark(Point p)
    {
        System.out.println("Car park: "+"Looking for nearest
space in car park for location: "+p.x+", "+p.y);
        int i;
        Space sdp=new Space(0,0,false,10,10);
    }

```

```

int n_ele=this.spaces.size();
for (i=0;i < n_ele;i=i+1)
{
    Space x= (Space) this.spaces.elementAt(i);
    if (i == 0)
    {
        sdp=x;
    }
    else
    {
        long d1=x.distanceTo(p);
        long d2=sdp.distanceTo(p);
        if (d1<d2)
        {
            sdp=x;
        }
    }
}
System.out.println("Car park: "+"Nearest space is
"+sdp.x+", "+sdp.y);
return sdp;
}

public boolean addParkedCar()
{
    this.full_spaces=this.full_spaces+1;
    if (this.full_spaces == this.maxncars)
    {
        this.full=true;
    }
    return this.full;
}

public long distanceBetween(Point p1, Point p2)
{
    /* This method should be redone and added to an
interface
car
so that it can be changed as the configuration of the
park changes */
return (long) Math.abs (Math.sqrt(((Math.abs(p1.x-
p2.x))*(Math.abs(p1.x-p2.x)))+(Math.abs(p1.y-p2.y))*(Math.abs(p1.y-
p2.y)))));
}

public boolean removeParkedCar()
{
    this.full_spaces=this.full_spaces -1;
    if (this.full)
    {
        this.full=false;
    }
    return this.full;
}

public boolean full()
{
    return this.full;
}

```

```

    }

    public Point emptyCentre()
    {
        Vector sps=this.spaces;
        int i;
        int ax=0;
        int ay=0;
        int count=0;

        for (i=0;i<sps.size(); i=i+1)
        {
            Space sp=(Space)sps.elementAt(i);

            if (!sp.full)
            {
                ax=ax+sp.x;
                ay=ay+sp.y;
                count=count+1;
            }
        }
        ax=ax/count;
        ay=ay/count;
        return new Point(ax,ay);
    }

    public int spaceIndex(Space sp)
    {
        int i;
        Space s;
        int r=-1;

        for (i=0; i < this.spaces.size(); i=i+1)
        {
            s=(Space)this.spaces.elementAt(i);
            if (s.equals(sp))
            {
                r=i;
            }
        }
        return r;
    }

    public double carsInCarPark()
    {
        int carsin=0;
        int i;
        Car c;
        for (i=0;i<this.car_threads.size();i=i+1)
        {
            c=(Car) (((CarThread)this.car_threads.elementAt(i)).owner);
            if (c.in_car_park)
            {
                carsin=carsin+1;
            }
        }
    }

```

```

        if (this.car_threads.size() >0)
        {
            return
            ((double)carsin)/((double)(this.car_threads.size()));
        }
        return 0;
    }

    public double carsExiting()
    {
        int carsin=0;
        int carsexit=0;
        int i;
        Car c;
        for (i=0;i<this.car_threads.size();i=i+1)
        {

            c=(Car)(((CarThread)this.car_threads.elementAt(i)).owner);
            if (c.exiting)
            {
                carsexit=carsexit+1;
            }
            if (c.in_car_park)
            {
                carsin=carsin+1;
            }
        }
        if (carsin >0)
        {
            return ((double)carsexit)/((double)carsin);
        }
        return 0;
    }

    public double carsParked()
    {
        int carsin=0;
        int carsparked=0;
        int i;
        Car c;
        for (i=0;i<this.car_threads.size();i=i+1)
        {

            c=(Car)(((CarThread)this.car_threads.elementAt(i)).owner);
            if (c.parked)
            {
                carsparked=carsparked+1;
            }
            if (c.in_car_park)
            {
                carsin=carsin+1;
            }
        }
        if (carsin >0)
        {
            return ((double)carsparked)/((double)maxncars);
        }
        return 0;
    }

    public double carsEntering()

```

```

{
    int carsin=0;
    int carsenter=0;
    long location_entrance;
    int i;
    Point entrance=(Point)this.entrances.elementAt(0);
    Car c;
    for (i=0;i<this.car_threads.size();i=i+1)
    {

        c=(Car)(((CarThread)this.car_threads.elementAt(i)).owner);

        location_entrance=this.distanceBetween((Point)c.current_location, (Point)entrance)/32;
        if (location_entrance < 6)
        {
            carsenter=carsenter+1;
        }
        if (c.in_car_park)
        {
            carsin=carsin+1;
        }
    }
    if (carsin >0)
    {
        return ((double)carsenter)/((double)carsin);
    }
    return 0;
}
}

```


Car Thread Class

```
import java.awt.Point;
import java.io.IOException;

public class CarThread extends Thread {

    Car owner;
    CarPark car_park;
    Learning learn;
    Point empty_centre;

    public CarThread(Car c, CarPark cp) throws IOException
    {
        owner=c;
        car_park=cp;
        learn=new Learning(this);
        System.out.println("CarThread for "+c.name+": Created
CarThread of car "+c.name);
    }

    public void run()
    {
        Space near;
        Point centre;
        long parking_time=0;
        long current_time=0;
        Car c=this.owner;
        Space s=new Space(0,0,false,32,24);
        int next;
        boolean
teste=((Space) car_park.spaces.elementAt(10)).full;
        int t1;

        if (c.birth(this))
        {
            c.in_car_park=true;
            while (c.in_car_park)
            {
                if (!c.parked)
                {
                    if (c.exiting)
                    {
                        Point
exit=(Point) car_park.exits.elementAt(0);
                        if
(car_park.distanceBetween(exit,c.current_location) <= 32)
                        {
                            try
                            {

                                learn.writeLearning(c);

                            }
                            catch (IOException e)
                            {
                                c.exit();
                            }
                        }
                    }
                }
            }
        }
    }
}
```

```

        if (c.exiting)
        {
            c.moveTowards(exit);
        }
        System.out.println(c.name+" vai para a saida");
    }
    else
    {
    }
}
else
{

    teste=((Space) car_park.spaces.elementAt(10)).full;

    current_time=TimeThread.current_time;
    /* Space
near=car_park.nearestEmptySpaceNearestToExit(c); */

    next=learn.q_learning(car_park.emptyCentre(),c,car_park);
    if (next!=learn.space)
    {

        centre=car_park.emptyCentre();
        if
        (learn.distanceBetween(centre, c.current_location) >= 32)
        {

            c.moveTowards(centre);

        }
        else
        {

        }
    }
    else
    {

        near=learn.target_space;
        System.out.println("
Distance from car "+c.name+" To space "+near.x+", "+near.y+"
:"+near.distanceTo(c.current_location));
        if
        (near.distanceTo(c.current_location) == 61)
        {

            System.out.println("E agora !!!");
        }
        else
        {
        }
    }
    if
    (near.distanceTo(c.current_location) <= 32)
    {
        if (c.park(near))
        {

            s=near;

            parking_time=TimeThread.current_time;

```


Learning Class

```
import java.awt.Point;
import java.io.IOException;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.PrintStream;
import java.io.FileNotFoundException;
import java.io.StreamTokenizer;

public class Learning
{
    /* states */
    public static final int centre=0;
    public static final int space=1;
    public static final int exit=2;
    public static final int out=3;
    public static final int beginner=4;
    /* */

    public Point initial_direction; /* first direction given */

    public int steps_initial_direction; /* number steps to
initial direction */

    public Point j_location; /* current location */
    public Point j_direction; /* target space of previous step */

    public Point i_location; /* initial location at previous step
*/
    public Point i_direction; /* initial direction at previous
step - before running q_learning */

    public Space target_space; /* target space defined by learning
algorithm - after running q_lerning */
    public int n_previous_targets; /* number of targets previously
assigned */
    public int n_previous_steps; /* number of steps taken until
now */
    public int n_steps_target; /* number of steps to take to reach
current target */
    public double current_reward; /* reward of current location
and direction depending on environment */
    public int action; /* action taken (centre, space or beginner)
*/

    /* Q_learning function */
    public int q_maxx=19; /* number of possible actions to take */
    public int q_maxy=19; /* number of possible states */
    public double q[][]=new double[q_maxx][q_maxy];

    /* number of times an action is taken at each state - a
counter array*/
    public int n[][]=new int[19][19];

    public Learning(CarThread ct) throws IOException
    {
        Car c=ct.owner;
```

```

        boolean f=true;
        j_location=c.current_location;
        /* j_direction=c.current_location; TESTE */
        n_previous_steps=-1;
        n_steps_target=-1;
        n_previous_targets=0;
int i,j,u=0,t=0;
        FileInputStream inputfile1, inputfile2;
        action=beginner;

        if (f)
        {
            try
            {
                inputfile1=new
FileInputStream("c:/projects/ParkingApplet/"+c.name+".q");
                inputfile2=new
FileInputStream("c:/projects/ParkingApplet/"+c.name+".n");
                StreamTokenizer token1;
                StreamTokenizer token2;

                token1=new StreamTokenizer(inputfile1);
                token2=new StreamTokenizer(inputfile2);

                for (i=0; i<q_maxx; i=i+1)
                {
                    for (j=0;j<q_maxy;j=j+1)
                    {
                        token1.nextToken();
                        token2.nextToken();
                        q[i][j]=(int)token1.nval;
                        n[i][j]=(int)token2.nval;
                    }
                }

            }
            catch (FileNotFoundException e)
            {
                System.out.println("Cannot find file");
            }
            catch (IOException E)
            {
                System.out.println("Cannot Process File");
            }
        }
        else
        {
            for (i=0; i<q_maxx; i=i+1)
            {
                for (j=0;j<q_maxy;j=j+1)
                {
                    q[i][j]=-1;
                    n[i][j]=0;
                }
            }
        }
    }
}

```

```

public long distanceBetween(Point p1, Point p2)
{
    return (long) Math.abs(Math.sqrt(((Math.abs(p1.x-
p2.x))*(Math.abs(p1.x-p2.x)))+(Math.abs(p1.y-p2.y))*(Math.abs(p1.y-
p2.y)))));
}

public double maxQ(CarPark cp, int state)
{
    int i;
    double maxq=0, r=0;
    for (i=0;i<q_maxx; i=i+1)
    {
        if (q[i][state] != -1)
        {
            if (q[i][state] > maxq)
            {
                maxq = q[i][state];
            }
        }
        else
        {
            try {
                r=this.reward(cp);
                if (r > maxq)
                {
                    maxq=r;
                }
            }
            catch (IOException e)
            {
            }
        }
    }
    return maxq;
}

public double reward(CarPark cp) throws IOException
{
    Point entrance;
    double pos=0, icp, pa, cen, cex;
    double r=0, location_direction=0,
direction_exit=0, dist=0;
    Point exit=(Point) (cp.exits.elementAt(0));
    if (action==space)
    {

        location_direction=distanceBetween((Point)j_location,
(Point)j_direction)/32;
        direction_exit=distanceBetween((Point)j_direction,
exit)/32;

        if (location_direction!=0)
        {
            if (direction_exit != 0)
            {
                r=(1/location_direction)*(1/direction_exit);
            }
            else
            {

```

```

        r=1/location_direction;
    }
}
else
{
    if (direction_exit != 0)
    {
        r=1/direction_exit;
    }
    else
    {
        r=1;
    }
}

j_location))
    if (((Space) j_direction).betterCar(cp,
    {
        r=r*0.3;
    }
    else
    {
        r=r*0.7;
    }
}
if (action==centre)
{
    entrance=(Point) (cp.entrances.elementAt(0));
    dist=distanceBetween(j_location,entrance) /32 ;
    if (dist !=0)
    {
        pos=1/dist;
    }
    else
    {
        dist=1;
    }
    icp=cp.carsInCarPark();
    pa=cp.carsParked();
    cen=cp.carsEntering();
    cex=cp.carsExiting();
    if (icp < 0.7)
    {
        r=0.002;
    }
    else
    {
        r=icp*pa*cen*(1-cex)*pos;
        if ((icp > 1) || (pa > 1) || (cen > 1) ||
(cex > 1) || (pos > 1))
        {
            System.out.println("Mais problemas no
reward !!");
        }
    }

    //r=cp.carsInCarPark()*cp.carsParked()*cp.carsEntering()*(1-
cp.carsExiting())*pos;
}
}

```

```

        /*if (r > 1)
        {
            r=1;
            System.out.println("Porcaria com o reward");
        } */
        return r;
    }
    public double reward1(CarPark cp) throws IOException
    {
        double r1=1,r2=1,r3=1,r4=1;

        boolean b;

        if (action==space)
        {
            if (distanceBetween(j_location, j_direction) < 32)
            {
                if (!((Space) j_direction).full)
                {
                    return 1;
                }
                else
                {
                    return 0;
                }
            }
            else
            {
                if (!((Space) j_direction).full)
                {
                    r1=0.8;
                }
                else
                {
                    r1=0;
                }
            }
        }
        else
        {
            if (action==centre)
            {
                b=((Space)
i_direction).betterCar(cp,j_location);
                if (b)
                {
                    r1=0.2;
                }
                else
                {
                    r1=0.8;
                }
            }
            else
            {
                return 0;
            }
        }
    }
}

```



```

        if
((n_previous_steps+((long)(distanceBetween(j_location,j_direction)/3
2))) != 0)
    {
        r3=(double)steps_initial_direction/(double)
(n_previous_steps+distanceBetween(j_location,j_direction));
    }

    if ((double) n_previous_targets != 0)
    {
        r4=(double) 1/(double) n_previous_targets;
    }
    if ((r1!=0) && (r2 != 0) && (r3 != 0) && (r4 != 0))
    {
        return r1*r2*r3*r4;
    }
    else
    {
        return 0;
    }
}

```

```

public int q_learning(Point empty_centre,Car c, CarPark cp)
{

```

```

    Space sp;
    int i;
    int k;
    int temp;
    double qi,maxq;
    double qi_old=0;
    int action_old;

```

```

    /* getting present state */
    j_location=c.current_location;
    n_previous_steps=n_previous_steps+1;
    if (n_previous_steps != 0)
    {
        if (action == space)
        {
            j_direction=(Point)target_space;
        }
        else
        {
            j_direction=empty_centre;
        }
        if (j_direction != i_direction)
        {
            n_previous_targets=n_previous_targets+1;
        }
    }
}

```

```

    /* updating tables of use and utility */
    if (n_previous_steps != 0)
    {
        if (action == space)
        {
            i=cp.spaceIndex(target_space);

```

```

        if ((i_direction == null)
|| (i_direction.getClass().getName() != "Space"))
        {
            k=18;
        }
        else
        {
            k=cp.spaceIndex((Space)i_direction);
        }
    }
    else
    {
        i=18;
        if (target_space != null)
        {
            k=cp.spaceIndex((Space)i_direction);
        }
        else
        {
            k=18;
        }
    }

    n[i][k]=n[i][k]+1;
    maxq=maxQ(cp,i);
    if (q[i][k] != -1)
    {
        qi_old=q[i][k];
    }
    else
    {
        try
        {
            qi_old=this.reward(cp);
        }
        catch (IOException e)
        {
        }
    }
    qi=(qi_old+(0.7*(current_reward+Math.abs(maxq-
qi_old)))));
    q[i][k]=qi;

}
else
{

}

if ((action==centre) || (action==space) || (action==beginner))
{
    action_old=action;
    i_location=j_location;
    i_direction=j_direction;
    try
    {
        current_reward=this.reward(cp);
    }
}

```

```

        catch (IOException e)
        {
        }
        action=this.max_action(cp,c);
        if (action_old != action)
        {
            if (action==space )
            {

                n_steps_target=(int) (this.distanceBetween(j_location, (Point)target
                rget_space)/32);
            }
            else
            {
                n_steps_target=(int)
                (this.distanceBetween(j_location, empty_centre)/32);
            }
            System.out.println("Car "+c.name+"'s action is
            "+action+" (0-centre, 1-space)");
            return action;
        }
        return 0;
    }

    public int less_used_space(CarPark cp)
    {
        int i,j,i_less_used=0;
        int n_i, n_less_used=0;

        for (j=0;j<q_maxx; j=j+1)
        {
            n_less_used=n_less_used+n_less_used+n[0][j];
        }
        for (i=0;i<q_maxx; i=i+1)
        {
            n_i=0;
            for (j=1;j<q_maxy; j=j+1)
            {
                n_i=n_i+n[i][j];
            }
            if ((n_less_used>n_i) && (n_i < 10) )
            {
                if (((i < 18) && (!((Space)
                (cp.spaces.elementAt(i))).full))) || (i==18))
                {
                    n_less_used=n_i;
                    i_less_used=i;
                }
            }
        }
        return i_less_used;
    }

    public boolean littleExperience(CarPark cp)
    {
        int i,j=0;
        int line;
        int defeito=10;
        for (i=0; i< q_maxx; i=i+1)

```

```

    {
        defeito=10;
        line=0;
        for(j=0; j<q_maxy; j=j+1)
        {
            line=line+n[i][j];
        }
        if (line < defeito)
        {
            if ((i!=18) &&
(!(((Space)cp.spaces.elementAt(i)).full)))
            {
                defeito=line;
            }
        }
        if (defeito > 10)
        {
            return false;
        }
        return true;
    }

public boolean littleExperience(CarPark cp, int action)
{
    int j,line=0;
    for(j=0;j<q_maxy;j=j+1)
    {
        line=line+n[action][j];
    }
    if (line > 10)
    {
        return false;
    }
    else
    {
        if ((action!=18) &&
(!(((Space)cp.spaces.elementAt(action)).full)))
        {
            return true;
        }
    }
    return false;
}

public double actionQ(int action)
{
    int j=0;
    double action_q=0;
    int n_nonzero=q_maxy;
    for (j=0;j<q_maxy;j=j+1)
    {
        if (q[action][j]!=-1)
        {
            action_q=action_q+q[action][j];
        }
        else
        {

```

```

        n_nonzero=n_nonzero-1;
    }

}
if (n_nonzero!=0)
{
    return action_q/n_nonzero;
}
return -1;
}

public int bestQInd(CarPark cp)
{
    int i,j=0;
    double line,q_line;
    double best=0;
    int ind=-1;
    int n_nonzero=q_maxy;
    for (i=0; i< q_maxx; i=i+1)
    {
        line=0;
        for(j=0; j<q_maxy; j=j+1)
        {
            if (q[i][j]!=-1)
            {
                line=line+q[i][j];
            }
            else
            {
                n_nonzero=n_nonzero-1;
            }
        }
        if (n_nonzero !=0)
        {
            q_line=line/n_nonzero;
        }
        else
        {
            q_line=-1;
        }
        if (q_line > best)
        {
            if ((i!=18) &&
(!(((Space) cp.spaces.elementAt(i)).full)))
            {
                best=q_line;
                ind=i;
            }
        }
    }
    return ind;
}

public double bestQ(CarPark cp)
{
    int i,j=0;
    double line,q_line;
    double best=0;

```

```

int ind=-1;
int n_nonzero=q_maxy;
for (i=0; i< q_maxx; i=i+1)
{
    line=0;
    for(j=0; j<q_maxy; j=j+1)
    {
        if (q[i][j]!=-1)
        {
            line=line+q[i][j];
        }
        else
        {
            n_nonzero=n_nonzero-1;
        }
    }
    if (n_nonzero !=0)
    {
        q_line=line/n_nonzero;
    }
    else
    {
        q_line=-1;
    }
    if (q_line > best)
    {
        if ((i!=18) &&
(!(((Space)cp.spaces.elementAt(i)).full)))
        {
            best=q_line;
            ind=i;
        }
    }
}
return best;
}

public int max_action(CarPark cp, Car c)
{
    double qi=0, qi_old=0;
    int k1=0,index, i,v,j, k,ns=0;
    Space near;
    int i_less_used=0;
    int ind;

    if (this.littleExperience(cp))
    {
        if ((action==beginner) || ((action==space) &&
(target_space.full)))
        {
            i_less_used=less_used_space(cp);

            if (i_less_used != 18)
            {
                target_space=(Space)cp.spaces.elementAt(i_less_used);
                action=space;
                return action;
            }
        }
    }
}

```

```

        else
        {
            action=centre;
            target_space=null;
            return action;
        }
    }
    else
    {
        if (action == centre)
        {
            if ((cp.carsInCarPark() < 0.7) ||
(cp.carsParked() < 0.7))
            {
                i_less_used=less_used_space(cp);

                if (i_less_used != 18)
                {
                    target_space=(Space)cp.spaces.elementAt(i_less_used);
                    action=space;
                    return action;
                }
                else
                {
                    /* arriscado */

                    target_space=(Space)cp.spaces.elementAt(17);
                    action=space;
                    return action;
                }
            }
        }
    }
}
else
{
    if ((action==beginner) || ((action == space) &&
target_space.full))
    {
        ind=this.bestQInd(cp);
        if (ind !=18)
        {
            target_space=(Space)cp.spaces.elementAt(ind);
            action=space;
            return action;
        }
        else
        {
            action=centre;
            target_space=null;
            return action;
        }
    }
}
else
{

```

```

        if (action==centre)
        {
            i=18;
        }
        else
        {
            i=cp.spaceIndex((Space)j_direction);
        }
        System.out.println("The number of possible aims is
"+q_maxy);
        for (k=0;k<q_maxy;k=k+1)
        {
            if (q[i][k] != -1)
            {
                qi_old=q[i][k];
            }
            else
            {
                try
                {
                    qi_old=this.reward(cp);
                }
                catch (IOException e)
                {
                }
            }
            if (qi_old > qi)
            {
                qi=qi_old;
                if (k<18)
                {

                    target_space=(Space)cp.spaces.elementAt(k);
                    System.out.println("Car
"+c.name+" is going to space with index "+k);
                    k1=k;
                    action=space;
                    return action;
                }
            }
            else
            {
                action=centre;
                k1=18;
                return action;
            }
        }
        else
        {
            k1=k;
        }
    }
}
return action;
}

public int max_action_second(CarPark cp, Car c)
{
    double qi=0, qi_old=0;
    int k1=0,index, i,v,j, k,ns=0;

```



```

    Space near;
    int i_less_used=less_used_space(cp);

    if ((action==beginner) || ((action==space) &&
(target_space.full)))
    {
        if (i_less_used != -1)
        {
            if (i_less_used != 18)
            {

target_space=(Space)cp.spaces.elementAt(i_less_used);
                action=space;
                return action;
            }
            else
            {
                action=centre;
                return action;
            }
        }
    }

    if (action==beginner)
    {

        near=cp.nearestEmptySpaceNearestToExit(c);
        v=cp.spaceIndex(near);
        action=space;
        target_space=near;
    }
    else
    {
        if (action==centre)
        {
            i=18;
        }
        else
        {
            i=cp.spaceIndex((Space)j_direction);
        }
        System.out.println("The number of possible aims is
"+q_maxy);
        for (k=0;k<q_maxy;k=k+1)
        {
            if (q[i][k] != -1)
            {
                qi_old=q[i][k];
            }
            else
            {
                try
                {
                    qi_old=this.reward(cp);
                }
                catch (IOException e)
                {
                }
            }
        }
        if (qi_old > qi)

```

```

        {
            qi=qi_old;
            if (k<18)
            {
                target_space=(Space) cp.spaces.elementAt(k);
                System.out.println("Car "+c.name+" is
going to space with index "+k);
                k1=k;
                action=space;
            }
            else
            {
                action=centre;
                k1=18;
            }
        }
        else
        {
            k1=k;
        }
    }

    }
    if ((action != centre) &&(target_space.full))
    {
        near=cp.nearestEmptySpaceNearestToExit(c);
        v=cp.spaceIndex(near);
        ns=0;
        action=space;
        target_space=near;
    }

    return action;
}

public int max_action_initial(CarPark cp, Car c)
{
    double qi=0, qi_old=0;
    int k1=0,index, i,v,j, k,ns=0;
    Space near;

    if (action==beginner)
    {
        near=cp.nearestEmptySpaceNearestToExit(c);
        v=cp.spaceIndex(near);
        for (j=0; j< q_maxx; j++)
        {
            ns=ns+n[j][v];
        }
        if (ns > 10)
        {
            action=centre;
        }
        else
        {
            action=space;
            target_space=near;
        }
    }
}

```

```

    }
}
else
{
    if (action==centre)
    {
        i=18;
    }
    else
    {
        i=cp.spaceIndex((Space)j_direction);
    }
    System.out.println("The number of possible aims is
"+q_maxy);
    for (k=0;k<q_maxy;k=k+1)
    {
        if (q[i][k] != -1)
        {
            qi_old=q[i][k];
        }
        else
        {
            try
            {
                qi_old=this.reward(cp);
            }
            catch (IOException e)
            {
            }
        }
        if (qi_old > qi)
        {
            qi=qi_old;
            if (k<18)
            {

                target_space=(Space)cp.spaces.elementAt(k);
                System.out.println("Car "+c.name+" is
going to space with index "+k);
                k1=k;
                action=space;
            }
            else
            {
                action=centre;
                k1=18;
            }
        }
        else
        {
            k1=k;
        }
    }
}
if ((action != centre) &&(target_space.full))
{
    near=cp.nearestEmptySpaceNearestToExit(c);
    v=cp.spaceIndex(near);
    ns=0;
    for (j=0; j< q_maxx; j++)

```

```

        {
            ns=ns+n[j][v];
        }
    if (ns > 10)
    {
        action=centre;
    }
    else
    {
        action=space;
        target_space=near;
    }
}

return action;
}

public void writeLearning (Car c) throws IOException
{
    int i,j;
    FileOutputStream outputfile1=new
FileOutputStream("c:/projects/ParkingApplet/"+c.name+".q");
    FileOutputStream outputfile2=new
FileOutputStream("c:/projects/ParkingApplet/"+c.name+".n");
    PrintStream output1=new PrintStream(outputfile1);
    PrintStream output2=new PrintStream(outputfile2);

    for (i=0; i< q_maxx; i=i+1)
    {
        for (j=0;j<q_maxy; j=j+1)
        {
            output1.print(q[i][j]);
            output2.print(n[i][j]);
            output1.print(" ");
            output2.print(" ");
        }
        output1.println();
        output2.println();
    }
    output1.close();
    output2.close();
    outputfile1.close();
    outputfile2.close();
    System.out.println(c.name+"is closing...");
}
}

```

Space Class

```
import java.awt.Point;
import java.awt.Polygon;
import java.awt.Rectangle;
import java.lang.Math;
import java.util.Vector;

public class Space extends Point{

    public boolean full=false;
    public int width;
    public int height;
    public Rectangle place;
    public Car parked_car;

    /* constructor methods */
    public Space(int x, int y, boolean f, int w, int h)
    {
        super(x,y);

        this.full=f;
        this.width=w;
        this.height=h;
        this.place=new Rectangle(x,y,w,h);
    }

    public long distanceTo(Point p)
    {
        /* This method should be redone and added to an
interface
car
        so that it can be changed as the configuration of the
        car
        park changes */
        long d =(long) Math.abs (Math.sqrt(((Math.abs(this.x-
p.x))*(Math.abs(this.x-p.x)))+(Math.abs(this.y-
p.y))*(Math.abs(this.y-p.y)))));
        return d;
    }

    public synchronized boolean park(Car c)
    {
        if (!this.full)
        {
            this.full=true;
            this.parked_car=c;
            return true;
        }
        return false;
    }

    public boolean exit()
    {
        this.full= false;
        return true;
    }

    public boolean betterCar(CarPark cp, Point location)
    {
        int i;
```

```

Vector ct;
Car c;
long d1,d2=0;

ct=cp.car_threads;
for (i=0;i<ct.size();i=i+1)
{
    c= ((CarThread)ct.elementAt(i)).owner;
    if ((!c.exiting) && (c.in_car_park))
    {
        d1=this.distanceTo(location);
        d2=this.distanceTo(c.current_location);
        if (d2 < d1)
        {
            return true;
        }
    }
}
return false;
}
}

```

Appendix E: Car Park Simulation Report

Parking: Initializing ...
Parking: Car Park Polygon 0,0232,00,232232,232
Parking: 1 Entrance : 232,72
Parking: 1 Exit : 40,232
Car park: Car Park name: Armanda
Car park: Maximum number of cars: 17
Car park: Full Spaces 0
Car park: Car park is full false
Car park: Car Park has been created
Time Thread: Initial Time 933972181770
Time Thread: Time Step 2000
Parking: Adding Car spaces ...
Car park: Space has been added to car park: 168,4
Parking: Car space at168 4 size 24 32
Parking: Intersects car park ?true
Car park: Space has been added to car park: 136,4
Parking: Car space at136 4 size 24 32
Parking: Intersects car park ?true
Car park: Space has been added to car park: 104,4
Parking: Car space at104 4 size 24 32
Parking: Intersects car park ?true
Car park: Space has been added to car park: 72,4
Parking: Car space at72 4 size 24 32
Parking: Intersects car park ?true
Car park: Space has been added to car park: 40,4
Parking: Car space at40 4 size 24 32
Parking: Intersects car park ?true
Car park: Space has been added to car park: 4,36
Parking: Car space at4 36 size 32 24
Parking: Intersects car park ?true
Car park: Space has been added to car park: 4,68
Parking: Car space at4 68 size 32 24
Parking: Intersects car park ?true
Car park: Space has been added to car park: 4,100
Parking: Car space at4 100 size 32 24
Parking: Intersects car park ?true
Car park: Space has been added to car park: 4,132
Parking: Car space at4 132 size 32 24
Parking: Intersects car park ?true
Car park: Space has been added to car park: 4,164
Parking: Car space at4 164 size 32 24
Parking: Intersects car park ?true
Car park: Space has been added to car park: 4,196
Parking: Car space at4 196 size 32 24
Parking: Intersects car park ?true
Car park: Space has been added to car park: 76,196
Parking: Car space at76 196 size 24 32
Parking: Intersects car park ?true
Car park: Space has been added to car park: 108,196
Parking: Car space at108 196 size 24 32
Parking: Intersects car park ?true
Car park: Space has been added to car park: 140,196
Parking: Car space at140 196 size 24 32
Parking: Intersects car park ?true
Car park: Space has been added to car park: 196,172
Parking: Car space at196 172 size 32 24

Parking: Intersects car park ?true
Car park: Space has been added to car park: 196,140
Parking: Car space at196 140 size 32 24
Parking: Intersects car park ?true
Car park: Space has been added to car park: 196,108
Parking: Car space at196 108 size 32 24
Parking: Intersects car park ?true
Car park: Space has been added to car park: 196,76
Parking: Car space at196 76 size 32 24
Parking: Intersects car park ?true
Parking: Creating cars
Parking: Creating car number 1
Car Cr1 Creating car ...Cr1
Car Cr1 Creation_time 933972177770
Car Cr1 Creation_location 232,72
Car Cr1 Current location 232,72
Car Cr1 Speed 1
Car Cr1 parking time 232000
Car Cr1 Currently in car park ? true
Car Cr1 Currently Parked ? false
Car Cr1 Currently exiting ? false
Car Cr1 Currently moving forward ? true
CarThread for Cr1: Created CarThread of car Cr1
Car park: Car Thread has been added to Car Park
Parking: Creating car number 2
Car Cr2 Creating car ...Cr2
Car Cr2 Creation_time 933972393770
Car Cr2 Creation_location 232,72
Car Cr2 Current location 232,72
Car Cr2 Speed 1
Car Cr2 parking time 212000
Car Cr2 Currently in car park ? false
Car Cr2 Currently Parked ? false
Car Cr2 Currently exiting ? false
Car Cr2 Currently moving forward ? true
CarThread for Cr2: Created CarThread of car Cr2
Car park: Car Thread has been added to Car Park
Parking: Creating car number 3
Car Cr3 Creating car ...Cr3
Car Cr3 Creation_time 933972405770
Car Cr3 Creation_location 232,72
Car Cr3 Current location 232,72
Car Cr3 Speed 1
Car Cr3 parking time 224000
Car Cr3 Currently in car park ? false
Car Cr3 Currently Parked ? false
Car Cr3 Currently exiting ? false
Car Cr3 Currently moving forward ? true
Car Cr2: Going to sleep ...waiting for birth
CarThread for Cr3: Created CarThread of car Cr3
Car park: Car Thread has been added to Car Park
Parking: Creating car number 4
Car Cr4 Creating car ...Cr4
Car Cr4 Creation_time 933972177770
Car Cr4 Creation_location 232,72
Car Cr4 Current location 232,72
Car Cr4 Speed 1
Car Cr4 parking time 4000
Car Cr4 Currently in car park ? true
Car Cr4 Currently Parked ? false

Car Cr4 Currently exiting ? false
Car Cr1's action is 1 (0-centre, 1-space)
Car Cr3: Going to sleep ...waiting for birth
Car Cr4 Currently moving forward ? true
Distance from car Cr1 To space 196,76 :36
CarThread for Cr4: Created CarThread of car Cr4
Car park: Car Thread has been added to Car Park
Parking: Creating car number 5
Car Cr4's action is 0 (0-centre, 1-space)
Car Cr5 Creating car ...Cr5
Car Cr5 Creation_time 933972177770
Car Cr5 Creation_location 232,72
Car Cr5 Current location 232,72
Car Cr5 Speed 1
Car Cr5 parking time 4000
Car Cr5 Currently in car park ? true
Car Cr5 Currently Parked ? false
Car Cr5 Currently exiting ? false
Car Cr5 Currently moving forward ? true
Car Cr4 is going to91,100
Car Cr4 moved to204,77
CarThread for Cr5: Created CarThread of car Cr5
Car park: Car Thread has been added to Car Park
Car Cr5's action is 0 (0-centre, 1-space)
Car Cr5 is going to91,100
Car Cr5 moved to204,77
Parking: Creating car number 6
Car Cr6 Creating car ...Cr6
Car Cr6 Creation_time 933972177770
Car Cr6 Creation_location 232,72
Car Cr6 Current location 232,72
Car Cr6 Speed 1
Car Cr6 parking time 4000
Car Cr6 Currently in car park ? true
Car Cr6 Currently Parked ? false
Car Cr6 Currently exiting ? false
Car Cr6 Currently moving forward ? true
CarThread for Cr6: Created CarThread of car Cr6
Car park: Car Thread has been added to Car Park
Parking: Creating car number 7
Car Cr6's action is 1 (0-centre, 1-space)
Distance from car Cr6 To space 4,68 :228
Car Cr6 is going to4,68
Car Cr6 moved to204,71
Car Cr7 Creating car ...Cr7
Car Cr7 Creation_time 933972177770
Car Cr7 Creation_location 232,72
Car Cr7 Current location 232,72
Car Cr7 Speed 1
Car Cr7 parking time 4000
Car Cr7 Currently in car park ? true
Car Cr7 Currently Parked ? false
Car Cr7 Currently exiting ? false
Car Cr7 Currently moving forward ? true
Time Thread: New Time 933972183770
CarThread for Cr7: Created CarThread of car Cr7
Car park: Car Thread has been added to Car Park
Parking: Creating car number 8
Car Cr7's action is 0 (0-centre, 1-space)
Car Cr7 is going to91,100

Car Cr7 moved to204,77
Car Cr8 Creating car ...Cr8
Car Cr8 Creation_time 933972179770
Car Cr8 Creation location 232,72
Car Cr8 Current location 232,72
Car Cr8 Speed 1
Car Cr8 parking time 4000
Car Cr8 Currently in car park ? true
Car Cr8 Currently Parked ? false
Car Cr8 Currently exiting ? false
Car Cr8 Currently moving forward ? true
CarThread for Cr8: Created CarThread of car Cr8
Car park: Car Thread has been added to Car Park
Parking: Creating car number 9
Car Cr8's action is 1 (0-centre, 1-space)
Distance from car Cr8 To space 140,196 :154
Car Cr8 is going to140,196
Car Cr8 moved to215,94
Car Cr9 Creating car ...Cr9
Car Cr9 Creation_time 933972179770
Car Cr9 Creation location 232,72
Car Cr9 Current location 232,72
Car Cr9 Speed 1
Car Cr9 parking time 4000
Car Cr9 Currently in car park ? true
Car Cr9 Currently Parked ? false
Car Cr9 Currently exiting ? false
Car Cr9 Currently moving forward ? true
CarThread for Cr9: Created CarThread of car Cr9
Car park: Car Thread has been added to Car Park
Parking: Creating car number 10
Car Cr9's action is 1 (0-centre, 1-space)
Distance from car Cr9 To space 196,140 :76
Car Cr9 is going to196,140
Car Cr9 moved to218,96
Car Cr10 Creating car ...Cr10
Car Cr10 Creation_time 933972179770
Car Cr10 Creation location 232,72
Car Cr10 Current location 232,72
Car Cr10 Speed 1
Car Cr10 parking time 4000
Car Cr10 Currently in car park ? true
Car Cr10 Currently Parked ? false
Car Cr10 Currently exiting ? false
Car Cr10 Currently moving forward ? true
CarThread for Cr10: Created CarThread of car Cr10
Car park: Car Thread has been added to Car Park
Parking: Creating car number 11
Car Cr11 Creating car ...Cr11
Car Cr11 Creation_time 933972179770
Car Cr11 Creation location 232,72
Car Cr11 Current location 232,72
Car Cr11 Speed 1
Car Cr11 parking time 4000
Car Cr11 Currently in car park ? true
Car Cr11 Currently Parked ? false
Car Cr11 Currently exiting ? false
Car Cr11 Currently moving forward ? true
Car Cr10's action is 1 (0-centre, 1-space)
Distance from car Cr10 To space 4,100 :229

Car Cr10 is going to4,100
Car Cr10 moved to204,75
CarThread for Cr11: Created CarThread of car Cr11
Car park: Car Thread has been added to Car Park
Parking: Creating car number 12
Car Cr12 Creating car ...Cr12
Car Cr12 Creation_time 933972179770
Car Cr12 Creation_location 232,72
Car Cr11's action is 1 (0-centre, 1-space)
Car Cr12 Current location 232,72
Distance from car Cr11 To space 104,4 :144
Car Cr12 Speed 1
Car Cr11 is going to104,4
Car Cr12 parking time 4000
Car Cr11 moved to207,58
Car Cr12 Currently in car park ? true
Car Cr12 Currently exiting ? false
Car Cr12 Currently moving forward ? true
CarThread for Cr12: Created CarThread of car Cr12
Car park: Car Thread has been added to Car Park
Parking: Creating car number 13
Car Cr13 Creating car ...Cr13
Car Cr13 Creation_time 933972179770
Car Cr13 Creation_location 232,72
Car Cr13 Current location 232,72
Car Cr13 Speed 1
Car Cr13 parking time 4000
Car Cr13 Currently in car park ? true
Car Cr13 Currently Parked ? false
Car Cr13 Currently exiting ? false
Car Cr13 Currently moving forward ? true
Car Cr12's action is 1 (0-centre, 1-space)
Distance from car Cr12 To space 196,172 :106
Car Cr12 is going to196,172
Car Cr12 moved to222,98
CarThread for Cr13: Created CarThread of car Cr13
Car park: Car Thread has been added to Car Park
Car Cr13's action is 1 (0-centre, 1-space)
Distance from car Cr13 To space 108,196 :175
Car Cr13 is going to108,196
Car Cr13 moved to212,91
Parking: Creating car number 14
Car Cr14 Creating car ...Cr14
Car Cr14 Creation_time 933972179770
Car Cr14 Creation_location 232,72
Car Cr14 Current location 232,72
Car Cr14 Speed 1
Car Cr14 parking time 4000
Car Cr14 Currently in car park ? true
Car Cr14 Currently Parked ? false
Car Cr14 Currently exiting ? false
Car Cr14 Currently moving forward ? true
CarThread for Cr14: Created CarThread of car Cr14
Car park: Car Thread has been added to Car Park
Parking: Creating car number 15
Car Cr14's action is 1 (0-centre, 1-space)
Distance from car Cr14 To space 140,196 :154
Car Cr14 is going to140,196
Car Cr14 moved to215,94
Car Cr15 Creating car ...Cr15

Car Cr15 Creation_time 933972179770
Car Cr15 Creation location 232,72
Car Cr15 Current location 232,72
Car Cr15 Speed 1
Car Cr15 parking time 4000
Car Cr15 Currently in car park ? true
Car Cr15 Currently Parked ? false
Car Cr15 Currently exiting ? false
Car Cr15 Currently moving forward ? true
CarThread for Cr15: Created CarThread of car Cr15
Car park: Car Thread has been added to Car Park
Parking: Creating car number 16
Car Cr15's action is 1 (0-centre, 1-space)
Distance from car Cr15 To space 168,4 :93
Car Cr15 is going to168,4
Car Cr15 moved to212,51
Car Cr16 Creating car ...Cr16
Car Cr16 Creation_time 933972179770
Car Cr16 Creation location 232,72
Car Cr16 Current location 232,72
Car Cr16 Speed 1
Car Cr16 parking time 4000
Car Cr16 Currently in car park ? true
Car Cr16 Currently Parked ? false
Car Cr16 Currently exiting ? false
Car Cr16 Currently moving forward ? true
CarThread for Cr16: Created CarThread of car Cr16
Car park: Car Thread has been added to Car Park
Parking: Creating car number 17
Car Cr16's action is 1 (0-centre, 1-space)
Distance from car Cr16 To space 196,108 :50
Car Cr17 Creating car ...Cr17
Car Cr17 Creation_time 933972179770
Car Cr17 Creation location 232,72
Car Cr17 Current location 232,72
Car Cr17 Speed 1
Car Cr17 parking time 4000
Car Cr17 Currently in car park ? true
Car Cr17 Currently Parked ? false
Car Cr17 Currently exiting ? false
Car Cr17 Currently moving forward ? true
CarThread for Cr17: Created CarThread of car Cr17
Car park: Car Thread has been added to Car Park
Parking: Creating car number 18
Car Cr17's action is 1 (0-centre, 1-space)
Car Cr18 Creating car ...Cr18
Distance from car Cr17 To space 4,68 :228
Car Cr18 Creation_time 933972179770
Car Cr17 is going to4,68
Car Cr18 Creation location 232,72
Car Cr17 moved to204,71
Car Cr18 Current location 232,72
Car Cr18 Speed 1
Car Cr18 parking time 4000
Car Cr18 Currently in car park ? true
Car Cr18 Currently Parked ? false
Car Cr18 Currently exiting ? false
Car Cr18 Currently moving forward ? true
CarThread for Cr18: Created CarThread of car Cr18
Car park: Car Thread has been added to Car Park

Parking: Creating car number 19
Car Cr18's action is 1 (0-centre, 1-space)
Distance from car Cr18 To space 196,76 :36
Car Cr19 Creating car ...Cr19
Car Cr19 Creation_time 933972179770
Car Cr19 Creation location 232,72
Car Cr19 Current location 232,72
Car Cr19 Speed 1
Car Cr19 parking time 4000
Car Cr19 Currently in car park ? true
Car Cr19 Currently Parked ? false
Car Cr19 Currently exiting ? false
Car Cr19 Currently moving forward ? true
Time Thread: New Time 933972185770
CarThread for Cr19: Created CarThread of car Cr19
Car park: Car Thread has been added to Car Park
Parking: Creating car number 20
Car Cr20 Creating car ...Cr20
Car Cr20 Creation_time 933972181770
Car Cr20 Creation location 232,72
Car Cr20 Current location 232,72
Car Cr20 Speed 1
Car Cr20 parking time 4000
Car Cr20 Currently in car park ? true
Car Cr20 Currently Parked ? false
Car Cr20 Currently exiting ? false
Car Cr20 Currently moving forward ? true
Car Cr19's action is 1 (0-centre, 1-space)
Distance from car Cr19 To space 4,132 :235
Car Cr19 is going to4,132
Car Cr19 moved to204,79
CarThread for Cr20: Created CarThread of car Cr20
Car park: Car Thread has been added to Car Park
Parking: Creating car number 21
Car Cr21 Creating car ...Cr21
Car Cr21 Creation_time 933972181770
Car Cr21 Creation location 232,72
Car Cr21 Current location 232,72
Car Cr21 Speed 1
Car Cr21 parking time 4000
Car Cr21 Currently in car park ? true
Car Cr21 Currently Parked ? false
Car Cr21 Currently exiting ? false
Car Cr21 Currently moving forward ? true
Car Cr20's action is 1 (0-centre, 1-space)
Distance from car Cr20 To space 4,36 :230
Car Cr20 is going to4,36
Car Cr20 moved to204,67
CarThread for Cr21: Created CarThread of car Cr21
Car park: Car Thread has been added to Car Park
Parking: Creating car number 22
Car Cr22 Creating car ...Cr22
Car Cr22 Creation_time 933972181770
Car Cr22 Creation location 232,72
Car Cr22 Current location 232,72
Car Cr22 Speed 1
Car Cr22 parking time 4000
Car Cr22 Currently in car park ? true
Car Cr22 Currently Parked ? false
Car Cr22 Currently exiting ? false

Car Cr22 Currently moving forward ? true
Car Cr21's action is 1 (0-centre, 1-space)
Distance from car Cr21 To space 196,108 :50
CarThread for Cr22: Created CarThread of car Cr22
Car park: Car Thread has been added to Car Park
Parking: Creating car number 23
Car Cr22's action is 1 (0-centre, 1-space)
Distance from car Cr22 To space 108,196 :175
Car Cr22 is going to108,196
Car Cr22 moved to212,91
Car Cr23 Creating car ...Cr23
Car Cr23 Creation_time 933972181770
Car Cr23 Creation_location 232,72
Car Cr23 Current location 232,72
Car Cr23 Speed 1
Car Cr23 parking time 4000
Car Cr23 Currently in car park ? true
Car Cr23 Currently Parked ? false
Car Cr23 Currently exiting ? false
Car Cr23 Currently moving forward ? true
CarThread for Cr23: Created CarThread of car Cr23
Car park: Car Thread has been added to Car Park
Parking: Creating car number 24
Car Cr24 Creating car ...Cr24
Car Cr24 Creation_time 933972333770
Car Cr24 Creation_location 232,72
Car Cr24 Current location 232,72
Car Cr24 Speed 1
Car Cr24 parking time 148000
Car Cr24 Currently in car park ? false
Car Cr24 Currently Parked ? false
Car Cr24 Currently exiting ? false
Car Cr24 Currently moving forward ? true
Car Cr23's action is 1 (0-centre, 1-space)
Distance from car Cr23 To space 104,4 :144
Car Cr23 is going to104,4
Car Cr23 moved to207,58
CarThread for Cr24: Created CarThread of car Cr24
Car park: Car Thread has been added to Car Park
Parking: Creating car number 25
Car Cr25 Creating car ...Cr25
Car Cr25 Creation_time 933972209770
Car Cr25 Creation_location 232,72
Car Cr25 Current location 232,72
Car Cr25 Speed 1
Car Cr25 parking time 24000
Car Cr25 Currently in car park ? false
Car Cr25 Currently Parked ? false
Car Cr25 Currently exiting ? false
Car Cr25 Currently moving forward ? true
CarThread for Cr25: Created CarThread of car Cr25
Car park: Car Thread has been added to Car Park
Car Cr24: Going to sleep ...waiting for birth
Parking has created all cars
Time Thread: New Time 933972187770
Car Cr25: Going to sleep ...waiting for birth
Car Cr1's action is 1 (0-centre, 1-space)
Distance from car Cr1 To space 196,76 :0
Car Cr1 has successfully parked at 196,76
Car Cr4's action is 1 (0-centre, 1-space)

Distance from car Cr4 To space 168,4 :81
Car Cr4 is going to168,4
Car Cr4 moved to191,51
Car Cr5's action is 1 (0-centre, 1-space)
Distance from car Cr5 To space 136,4 :99
Car Cr5 is going to136,4
Car Cr5 moved to184,56
Car Cr6's action is 1 (0-centre, 1-space)
Distance from car Cr6 To space 4,68 :200
Car Cr6 is going to4,68
Car Cr6 moved to176,70
Car Cr7's action is 1 (0-centre, 1-space)
Distance from car Cr7 To space 136,4 :99
Car Cr7 is going to136,4
Car Cr7 moved to184,56
Car Cr8's action is 1 (0-centre, 1-space)
Distance from car Cr8 To space 140,196 :126
Car Cr8 is going to140,196
Car Cr8 moved to198,116
Car Cr9's action is 1 (0-centre, 1-space)
Distance from car Cr9 To space 196,140 :49
Car Cr10's action is 1 (0-centre, 1-space)
Distance from car Cr10 To space 4,100 :201
Car Cr10 is going to4,100
Car Cr10 moved to176,78
Car Cr11's action is 1 (0-centre, 1-space)
Distance from car Cr11 To space 104,4 :116
Car Cr11 is going to104,4
Car Cr11 moved to182,44
Car Cr12's action is 1 (0-centre, 1-space)
Distance from car Cr12 To space 196,172 :78
Car Cr12 is going to196,172
Car Cr12 moved to212,124
Car Cr13's action is 1 (0-centre, 1-space)
Distance from car Cr13 To space 108,196 :147
Car Cr13 is going to108,196
Car Cr13 moved to192,110
Time Thread: New Time 933972189770
Car Cr14's action is 1 (0-centre, 1-space)
Distance from car Cr14 To space 140,196 :126
Car Cr14 is going to140,196
Car Cr14 moved to198,116
Car Cr15's action is 1 (0-centre, 1-space)
Distance from car Cr15 To space 168,4 :64
Car Cr15 is going to168,4
Car Cr15 moved to192,30
Car Cr16's action is 1 (0-centre, 1-space)
Distance from car Cr16 To space 196,108 :0
Car Cr16 has successfully parked at 196,108
Car Cr17's action is 1 (0-centre, 1-space)
Distance from car Cr17 To space 4,68 :200
Car Cr17 is going to4,68
Car Cr17 moved to176,70
Car Cr18's action is 0 (0-centre, 1-space)
Car Cr18 is going to78,101
Car Cr18 moved to168,81
Car Cr19's action is 1 (0-centre, 1-space)
Distance from car Cr19 To space 4,132 :206
Car Cr19 is going to4,132
Car Cr19 moved to176,86

Car Cr20's action is 1 (0-centre, 1-space)
Distance from car Cr20 To space 4,36 :202
Car Cr20 is going to4,36
Car Cr20 moved to176,62
Car Cr21's action is 0 (0-centre, 1-space)
Car Cr21 is going to78,101
Car Cr21 moved to168,106
Time Thread: New Time 933972191770
Car Cr22's action is 1 (0-centre, 1-space)
Distance from car Cr22 To space 108,196 :147
Car Cr22 is going to108,196
Car Cr22 moved to192,110
Car Cr23's action is 1 (0-centre, 1-space)
Distance from car Cr23 To space 104,4 :116
Car Cr23 is going to104,4
Car Cr23 moved to182,44
Car Cr1: Going to sleep ...waiting unparking
Car Cr4's action is 1 (0-centre, 1-space)
Distance from car Cr4 To space 168,4 :52
Car Cr5's action is 1 (0-centre, 1-space)
Distance from car Cr5 To space 136,4 :70
Car Cr5 is going to136,4
Car Cr5 moved to165,35
Car Cr6's action is 1 (0-centre, 1-space)
Distance from car Cr6 To space 4,68 :172
Car Cr6 is going to4,68
Car Cr6 moved to148,69
Car Cr7's action is 1 (0-centre, 1-space)
Distance from car Cr7 To space 136,4 :70
Car Cr7 is going to136,4
Car Cr7 moved to165,35
Time Thread: New Time 933972193770
Car Cr8's action is 1 (0-centre, 1-space)
Distance from car Cr8 To space 140,196 :98
Car Cr8 is going to140,196
Car Cr8 moved to181,138
Car Cr9's action is 1 (0-centre, 1-space)
Distance from car Cr9 To space 196,140 :0
Car Cr9 has successfully parked at 196,140
Car Cr10's action is 1 (0-centre, 1-space)
Distance from car Cr10 To space 4,100 :173
Car Cr10 is going to4,100
Car Cr10 moved to148,81
Car Cr11's action is 1 (0-centre, 1-space)
Distance from car Cr11 To space 104,4 :87
Car Cr11 is going to104,4
Car Cr11 moved to157,31
Car Cr12's action is 1 (0-centre, 1-space)
Distance from car Cr12 To space 196,172 :50
Car Cr13's action is 1 (0-centre, 1-space)
Distance from car Cr13 To space 108,196 :120
Car Cr13 is going to108,196
Car Cr13 moved to172,130
Car Cr14's action is 1 (0-centre, 1-space)
Distance from car Cr14 To space 140,196 :98
Car Cr14 is going to140,196
Car Cr14 moved to181,138
Car Cr15's action is 1 (0-centre, 1-space)
Distance from car Cr15 To space 168,4 :35
Unparked ... leaving car park

Car Cr17's action is 1 (0-centre, 1-space)
Distance from car Cr17 To space 4,68 :172
Car Cr17 is going to4,68
Car Cr17 moved to148,69
Car Cr18's action is 1 (0-centre, 1-space)
Distance from car Cr18 To space 104,4 :100
Car Cr18 is going to104,4
Car Cr18 moved to150,59
Time Thread: New Time 933972195770
Car Cr19's action is 1 (0-centre, 1-space)
Distance from car Cr19 To space 4,132 :178
Car Cr19 is going to4,132
Car Cr19 moved to148,93
Car Cr20's action is 1 (0-centre, 1-space)
Distance from car Cr20 To space 4,36 :173
Car Cr20 is going to4,36
Car Cr20 moved to148,57
Car Cr21's action is 1 (0-centre, 1-space)
Distance from car Cr21 To space 168,4 :102
Car Cr21 is going to168,4
Car Cr21 moved to168,78
Car Cr22's action is 1 (0-centre, 1-space)
Distance from car Cr22 To space 108,196 :120
Car Cr22 is going to108,196
Car Cr22 moved to172,130
Car Cr23's action is 1 (0-centre, 1-space)
Distance from car Cr23 To space 104,4 :87
Car Cr23 is going to104,4
Car Cr23 moved to157,31
Car Cr24: Going to sleep ...waiting for birth
Time Thread: New Time 933972197770
Car Cr25: Going to sleep ...waiting for birth
Car Cr4's action is 1 (0-centre, 1-space)
Distance from car Cr4 To space 168,4 :0
Car Cr4 has successfully parked at 168,4
Car Cr5's action is 1 (0-centre, 1-space)
Distance from car Cr5 To space 136,4 :42
Car Cr6's action is 1 (0-centre, 1-space)
Distance from car Cr6 To space 4,68 :144
Car Cr6 is going to4,68
Car Cr6 moved to120,68
Car Cr7's action is 1 (0-centre, 1-space)
Distance from car Cr7 To space 136,4 :42
Car Cr8's action is 1 (0-centre, 1-space)
Distance from car Cr8 To space 140,196 :71
Car Cr8 is going to140,196
Car Cr8 moved to164,160
Unparked ... leaving car park
Car Cr10's action is 1 (0-centre, 1-space)
Distance from car Cr10 To space 4,100 :145
Car Cr10 is going to4,100
Car Cr10 moved to120,84
Car Cr11's action is 1 (0-centre, 1-space)
Distance from car Cr11 To space 104,4 :59
Car Cr11 is going to104,4
Car Cr11 moved to132,18
Car Cr12's action is 1 (0-centre, 1-space)
Distance from car Cr12 To space 196,172 :0
Car Cr12 has successfully parked at 196,172
Car Cr13's action is 1 (0-centre, 1-space)

Distance from car Cr13 To space 108,196 :91
Car Cr13 is going to108,196
Car Cr13 moved to152,150
Time Thread: New Time 933972199770
Car Cr14's action is 1 (0-centre, 1-space)
Distance from car Cr14 To space 140,196 :71
Car Cr14 is going to140,196
Car Cr14 moved to164,160
Car Cr15's action is 1 (0-centre, 1-space)
Distance from car Cr15 To space 168,4 :0
Car Cr15 has unsuccessfully tried to park at 168,4
Car Cr16 is going to40,232
Car Cr16 moved to150,99
Car Cr17's action is 1 (0-centre, 1-space)
Distance from car Cr17 To space 4,68 :144
Car Cr17 is going to4,68
Car Cr17 moved to120,68
Car Cr18's action is 1 (0-centre, 1-space)
Distance from car Cr18 To space 104,4 :71
Car Cr18 is going to104,4
Car Cr18 moved to132,37
Car Cr19's action is 1 (0-centre, 1-space)
Distance from car Cr19 To space 4,132 :149
Car Cr19 is going to4,132
Car Cr19 moved to120,100
Car Cr20's action is 1 (0-centre, 1-space)
Distance from car Cr20 To space 4,36 :145
Car Cr20 is going to4,36
Car Cr20 moved to120,52
Car Cr21's action is 1 (0-centre, 1-space)
Distance from car Cr21 To space 4,68 :149
Car Cr21 is going to4,68
Car Cr21 moved to122,93
Car Cr22's action is 1 (0-centre, 1-space)
Distance from car Cr22 To space 108,196 :91
Car Cr22 is going to108,196
Car Cr22 moved to152,150
Time Thread: New Time 933972201770
Car Cr23's action is 1 (0-centre, 1-space)
Distance from car Cr23 To space 104,4 :59
Car Cr23 is going to104,4
Car Cr23 moved to132,18
Car Cr2: Going to sleep ...waiting for birth
Car Cr3: Going to sleep ...waiting for birth
Unparked ... leaving car park
Car Cr5's action is 1 (0-centre, 1-space)
Distance from car Cr5 To space 136,4 :0
Car Cr5 has successfully parked at 136,4
Car Cr6's action is 1 (0-centre, 1-space)
Distance from car Cr6 To space 4,68 :116
Car Cr6 is going to4,68
Car Cr6 moved to92,68
Car Cr7's action is 1 (0-centre, 1-space)
Distance from car Cr7 To space 4,100 :163
Car Cr7 is going to4,100
Car Cr7 moved to113,20
Time Thread: New Time 933972203770
Car Cr8's action is 1 (0-centre, 1-space)
Distance from car Cr8 To space 140,196 :43
Car Cr9 is going to40,232

Car Cr9 moved to171,154
Car Cr10's action is 1 (0-centre, 1-space)
Distance from car Cr10 To space 4,100 :117
Car Cr10 is going to4,100
Car Cr10 moved to92,87
Car Cr11's action is 1 (0-centre, 1-space)
Distance from car Cr11 To space 104,4 :31
Car Cr11 has successfully parked at 104,4
Unparked ... leaving car park
Car Cr13's action is 1 (0-centre, 1-space)
Distance from car Cr13 To space 108,196 :63
Car Cr13 is going to108,196
Car Cr13 moved to132,170
Car Cr14's action is 1 (0-centre, 1-space)
Distance from car Cr14 To space 140,196 :43
Car Cr15's action is 1 (0-centre, 1-space)
Distance from car Cr15 To space 168,4 :0
Car Cr15 has successfully parked at 168,4
Car Cr16 is going to40,232
Car Cr16 moved to107,117
Car Cr17's action is 1 (0-centre, 1-space)
Distance from car Cr17 To space 4,68 :116
Car Cr17 is going to4,68
Car Cr17 moved to92,68
Car Cr18's action is 1 (0-centre, 1-space)
Distance from car Cr18 To space 140,196 :159
Car Cr18 is going to140,196
Car Cr18 moved to133,64
Time Thread: New Time 933972205770
Car Cr19's action is 1 (0-centre, 1-space)
Distance from car Cr19 To space 4,132 :120
Car Cr19 is going to4,132
Car Cr19 moved to93,107
Car Cr20's action is 1 (0-centre, 1-space)
Distance from car Cr20 To space 4,36 :117
Car Cr20 is going to4,36
Car Cr20 moved to92,48
Car Cr21's action is 1 (0-centre, 1-space)
Distance from car Cr21 To space 4,68 :114
Car Cr21 is going to4,68
Car Cr21 moved to81,104
Car Cr22's action is 1 (0-centre, 1-space)
Distance from car Cr22 To space 108,196 :63
Car Cr22 is going to108,196
Car Cr22 moved to132,170
Car Cr23's action is 1 (0-centre, 1-space)
Distance from car Cr23 To space 76,196 :186
Car Cr23 is going to76,196
Car Cr23 moved to123,44
Car Cr24: Going to sleep ...waiting for birth
Car Cr25: Going to sleep ...waiting for birth
Time Thread: New Time 933972207770
Car Cr1: Going to sleep ...waiting unparking
Car Cr4 is going to40,232
Car Cr4 moved to154,28
Unparked ... leaving car park
Car Cr6's action is 1 (0-centre, 1-space)
Distance from car Cr6 To space 4,68 :88
Car Cr6 is going to4,68
Car Cr6 moved to64,68

Car Cr7's action is 1 (0-centre, 1-space)
Distance from car Cr7 To space 4,100 :135
Car Cr7 is going to4,100
Car Cr7 moved to90,36
Car Cr8's action is 1 (0-centre, 1-space)
Distance from car Cr8 To space 140,196 :0
Car Cr8 has successfully parked at 140,196
Car Cr9 is going to40,232
Car Cr9 moved to146,168
Car Cr10's action is 1 (0-centre, 1-space)
Distance from car Cr10 To space 4,100 :88
Car Cr10 is going to4,100
Car Cr10 moved to64,91
Unparked ... leaving car park
Car Cr12 is going to40,232
Car Cr12 moved to169,182
Car Cr13's action is 1 (0-centre, 1-space)
Distance from car Cr13 To space 108,196 :35
Time Thread: New Time 933972209770
Car Cr14's action is 1 (0-centre, 1-space)
Distance from car Cr14 To space 81,104 :109
Car Cr14 is going to81,104
Car Cr14 moved to124,172
Unparked ... leaving car park
Car Cr16 is going to40,232
Car Cr16 moved to72,130
Car Cr17's action is 1 (0-centre, 1-space)
Distance from car Cr17 To space 4,68 :88
Car Cr17 is going to4,68
Car Cr17 moved to64,68
Car Cr18's action is 1 (0-centre, 1-space)
Distance from car Cr18 To space 169,182 :123
Car Cr18 is going to169,182
Car Cr18 moved to141,90
Car Cr19's action is 1 (0-centre, 1-space)
Distance from car Cr19 To space 4,132 :92
Car Cr19 is going to4,132
Car Cr19 moved to66,114
Car Cr20's action is 1 (0-centre, 1-space)
Distance from car Cr20 To space 4,36 :88
Car Cr20 is going to4,36
Car Cr20 moved to64,44
Car Cr21's action is 1 (0-centre, 1-space)
Distance from car Cr21 To space 4,68 :92
Car Cr21 is going to4,68
Car Cr21 moved to51,111
Car Cr22's action is 1 (0-centre, 1-space)
Distance from car Cr22 To space 108,196 :35
Time Thread: New Time 933972211770
Car Cr23's action is 1 (0-centre, 1-space)
Distance from car Cr23 To space 76,196 :159
Car Cr23 is going to76,196
Car Cr23 moved to114,70
Car Cr2: Going to sleep ...waiting for birth
Car Cr3: Going to sleep ...waiting for birth
Car Cr4 is going to40,232
Car Cr4 moved to140,52
Car Cr5 is going to40,232
Car Cr5 moved to83,63
Car Cr6's action is 1 (0-centre, 1-space)

Distance from car Cr6 To space 4,68 :60
Car Cr6 is going to4,68
Car Cr6 moved to36,68
Car Cr7's action is 1 (0-centre, 1-space)
Distance from car Cr7 To space 4,100 :87
Car Cr7 is going to4,100
Car Cr7 moved to57,74
Time Thread: New Time 933972213770
Unparked ... leaving car park
Car Cr9 is going to40,232
Car Cr9 moved to122,182
Car Cr10's action is 1 (0-centre, 1-space)
Distance from car Cr10 To space 4,100 :60
Car Cr10 is going to4,100
Car Cr10 moved to36,95
Car Cr11 is going to40,232
Car Cr11 moved to120,43
Car Cr12 is going to40,232
Car Cr12 moved to142,192
Car Cr13's action is 1 (0-centre, 1-space)
Distance from car Cr13 To space 108,196 :0
Car Cr13 has successfully parked at 108,196
Car Cr14's action is 1 (0-centre, 1-space)
Distance from car Cr14 To space 51,111 :95
Car Cr14 is going to51,111
Car Cr14 moved to102,154
Car Cr15 is going to40,232
Car Cr15 moved to126,76
Car Cr16 is going to40,232
Car Cr16 moved to48,138
Car Cr17's action is 1 (0-centre, 1-space)
Distance from car Cr17 To space 4,68 :60
Car Cr17 is going to4,68
Car Cr17 moved to36,68
Car Cr18's action is 1 (0-centre, 1-space)
Distance from car Cr18 To space 142,192 :102
Car Cr18 is going to142,192
Car Cr18 moved to141,117
Time Thread: New Time 933972215770
Car Cr19's action is 1 (0-centre, 1-space)
Distance from car Cr19 To space 4,132 :64
Car Cr19 is going to4,132
Car Cr19 moved to39,121
Car Cr20's action is 1 (0-centre, 1-space)
Distance from car Cr20 To space 4,36 :60
Car Cr20 is going to4,36
Car Cr20 moved to36,40
Car Cr21's action is 1 (0-centre, 1-space)
Distance from car Cr21 To space 4,68 :82
Car Cr21 is going to4,68
Car Cr21 moved to33,114
Car Cr22's action is 0 (0-centre, 1-space)
Car Cr22 is going to56,106
Car Cr22 moved to93,171
Car Cr23's action is 1 (0-centre, 1-space)
Distance from car Cr23 To space 76,196 :131
Car Cr23 is going to76,196
Car Cr23 moved to105,96
Car Cr24: Going to sleep ...waiting for birth
Car Cr25's action is 1 (0-centre, 1-space)

Distance from car Cr25 To space 102,154 :153
 Car Cr25 is going to102,154
 Car Cr25 moved to208,86
 Time Thread: New Time 933972217770
 Car Cr4 is going to40,232
 Car Cr4 moved to112,100
 Car Cr5 is going to40,232
 Car Cr5 moved to54,101
 Car Cr6's action is 1 (0-centre, 1-space)
 Distance from car Cr6 To space 4,68 :32
 Car Cr6 has successfully parked at 4,68
 Car Cr7's action is 1 (0-centre, 1-space)
 Distance from car Cr7 To space 4,100 :50
 Car Cr8 is going to40,232
 Car Cr8 moved to84,175
 Car Cr9 is going to40,232
 Car Cr9 moved to98,196
 Car Cr10's action is 1 (0-centre, 1-space)
 Distance from car Cr10 To space 4,100 :32
 Car Cr10 has successfully parked at 4,100
 Car Cr11 is going to40,232
 Car Cr11 moved to109,68
 Car Cr12 is going to40,232
 Car Cr12 moved to115,202
 Unparked ... leaving car park
 Car Cr14's action is 1 (0-centre, 1-space)
 Distance from car Cr14 To space 33,114 :79
 Car Cr14 is going to33,114
 Car Cr14 moved to66,153
 Time Thread: New Time 933972219770
 Car Cr15 is going to40,232
 Car Cr15 moved to98,124
 Car Cr16 is going to40,232
 Car Cr16 moved to34,141
 Car Cr17's action is 1 (0-centre, 1-space)
 Distance from car Cr17 To space 4,132 :71
 Car Cr17 is going to4,132
 Car Cr17 moved to23,93
 Car Cr18's action is 1 (0-centre, 1-space)
 Distance from car Cr18 To space 115,202 :88
 Car Cr18 is going to115,202
 Car Cr18 moved to132,143
 Car Cr19's action is 1 (0-centre, 1-space)
 Distance from car Cr19 To space 4,132 :36
 Car Cr20's action is 1 (0-centre, 1-space)
 Distance from car Cr20 To space 4,36 :32
 Car Cr20 has successfully parked at 4,36
 Car Cr21's action is 1 (0-centre, 1-space)
 Distance from car Cr21 To space 98,196 :84
 Car Cr21 is going to98,196
 Car Cr21 moved to55,159
 Car Cr22's action is 1 (0-centre, 1-space)
 Distance from car Cr22 To space 54,101 :80
 Car Cr22 is going to54,101
 Car Cr22 moved to79,146
 Time Thread: New Time 933972221770
 Car Cr23's action is 1 (0-centre, 1-space)
 Distance from car Cr23 To space 76,196 :104
 Car Cr23 is going to76,196
 Car Cr23 moved to97,122

Car Cr25's action is 1 (0-centre, 1-space)
Distance from car Cr25 To space 66,153 :157
Car Cr25 is going to66,153
Car Cr25 moved to182,97
Car Cr2: Going to sleep ...waiting for birth
Car Cr3: Going to sleep ...waiting for birth
Car Cr1: Going to sleep ...waiting unparking
Car Cr4 is going to40,232
Car Cr4 moved to84,148
Car Cr5 is going to40,232
Car Cr5 moved to51,128
Unparked ... leaving car park
Car Cr7's action is 1 (0-centre, 1-space)
Distance from car Cr7 To space 4,164 :64
Car Cr7 is going to4,164
Car Cr7 moved to4,128
Car Cr8 is going to40,232
Car Cr8 moved to57,179
Time Thread: New Time 933972223770
Car Cr9 is going to40,232
Car Cr9 moved to74,210
Unparked ... leaving car park
Car Cr11 is going to40,232
Car Cr11 moved to98,93
Car Cr12 is going to40,232
Car Cr12 moved to89,212
Car Cr13 is going to40,232
Car Cr13 moved to67,171
Car Cr14's action is 1 (0-centre, 1-space)
Distance from car Cr14 To space 55,159 :20
Car Cr14 has successfully parked at 55,159
Car Cr15 is going to40,232
Car Cr15 moved to71,172
Car Cr16 is going to40,232
Car Cr16 moved to49,186
Car Cr17's action is 1 (0-centre, 1-space)
Distance from car Cr17 To space 4,132 :43
Car Cr18's action is 1 (0-centre, 1-space)
Distance from car Cr18 To space 89,212 :81
Car Cr18 is going to89,212
Car Cr18 moved to117,166
Time Thread: New Time 933972225770
Car Cr19's action is 1 (0-centre, 1-space)
Distance from car Cr19 To space 4,132 :0
Car Cr19 has successfully parked at 4,132
Unparked ... leaving car park
Car Cr21's action is 1 (0-centre, 1-space)
Distance from car Cr21 To space 74,210 :34
Car Cr22's action is 1 (0-centre, 1-space)
Distance from car Cr22 To space 51,128 :45
Car Cr23's action is 1 (0-centre, 1-space)
Distance from car Cr23 To space 76,196 :76
Car Cr23 is going to76,196
Car Cr23 moved to89,148
Car Cr24: Going to sleep ...waiting for birth
Car Cr25's action is 1 (0-centre, 1-space)
Distance from car Cr25 To space 57,179 :149
Car Cr25 is going to57,179
Car Cr25 moved to158,112
Time Thread: New Time 933972227770

Car Cr4 is going to40,232
Car Cr4 moved to58,196
Car Cr5 is going to40,232
Car Cr5 moved to48,155
Car Cr6 is going to40,232
Car Cr6 moved to36,95
Car Cr7's action is 1 (0-centre, 1-space)
Distance from car Cr7 To space 4,164 :36
Car Cr10 is going to40,232
Car Cr10 moved to36,122
Car Cr11 is going to40,232
Car Cr11 moved to87,118
Car Cr13 is going to40,232
Car Cr13 moved to55,196
Unparked ... leaving car park
Time Thread: New Time 933972229770
Car Cr17's action is 1 (0-centre, 1-space)
Distance from car Cr17 To space 4,164 :32
Car Cr17 has successfully parked at 4,164
Car Cr18's action is 1 (0-centre, 1-space)
Distance from car Cr18 To space 89,212 :53
Unparked ... leaving car park
Car Cr20 is going to40,232
Car Cr20 moved to36,67
Car Cr21's action is 1 (0-centre, 1-space)
Distance from car Cr21 To space 74,210 :0
Car Cr21 has successfully parked at 74,210
Car Cr22's action is 1 (0-centre, 1-space)
Distance from car Cr22 To space 48,155 :0
Car Cr22 has successfully parked at 48,155
Time Thread: New Time 933972231770
Car Cr23's action is 1 (0-centre, 1-space)
Distance from car Cr23 To space 76,196 :49
Car Cr25's action is 1 (0-centre, 1-space)
Distance from car Cr25 To space 57,179 :121
Car Cr25 is going to57,179
Car Cr25 moved to134,127
Car Cr2: Going to sleep ...waiting for birth
Car Cr3: Going to sleep ...waiting for birth
Car Cr5 is going to40,232
Car Cr5 moved to45,182
Car Cr6 is going to40,232
Car Cr6 moved to36,122
Car Cr7's action is 1 (0-centre, 1-space)
Distance from car Cr7 To space 55,196 :60
Car Cr7 is going to55,196
Car Cr7 moved to27,178
Time Thread: New Time 933972233770
Car Cr10 is going to40,232
Car Cr10 moved to37,149
Car Cr11 is going to40,232
Car Cr11 moved to76,143
Unparked ... leaving car park
Car Cr18's action is 1 (0-centre, 1-space)
Distance from car Cr18 To space 89,212 :0
Car Cr18 has successfully parked at 89,212
Cr9is closing...
Cr9 leaving car park
Time Thread: New Time 933972235770
Car Cr19 is going to40,232

Car Cr19 moved to13,158
Car Cr20 is going to40,232
Car Cr20 moved to36,94
Cr12is closing...
Cr12 leaving car park
Unparked ... leaving car park
Cr8is closing...
Cr8 leaving car park
Unparked ... leaving car park
Cr15is closing...
Cr15 leaving car park
Cr16is closing...
Cr16 leaving car park
Car Cr23's action is 1 (0-centre, 1-space)
Distance from car Cr23 To space 76,196 :0
Car Cr23 has successfully parked at 76,196
Car Cr24: Going to sleep ...waiting for birth
Car Cr25's action is 1 (0-centre, 1-space)
Distance from car Cr25 To space 57,179 :92
Car Cr25 is going to57,179
Car Cr25 moved to110,142
Time Thread: New Time 933972237770
Car Cr6 is going to40,232
Car Cr6 moved to37,149
Car Cr7's action is 1 (0-centre, 1-space)
Distance from car Cr7 To space 55,196 :33
Car Cr10 is going to40,232
Car Cr10 moved to38,176
Cr4is closing...
Cr4 leaving car park
Car Cr11 is going to40,232
Car Cr11 moved to65,168
Time Thread: New Time 933972239770
Car Cr17 is going to40,232
Car Cr17 moved to22,184
Unparked ... leaving car park
Cr13is closing...
Cr13 leaving car park
Cr14is closing...
Cr14 leaving car park
Car Cr20 is going to40,232
Car Cr20 moved to36,121
Time Thread: New Time 933972241770
Unparked ... leaving car park
Car Cr25's action is 1 (0-centre, 1-space)
Distance from car Cr25 To space 57,179 :64
Car Cr25 is going to57,179
Car Cr25 moved to87,158
Car Cr2: Going to sleep ...waiting for birth
Car Cr3: Going to sleep ...waiting for birth
Car Cr6 is going to40,232
Car Cr6 moved to38,176
Car Cr7's action is 1 (0-centre, 1-space)
Distance from car Cr7 To space 55,196 :0
Car Cr7 has successfully parked at 55,196
Time Thread: New Time 933972243770
Cr5is closing...
Cr5 leaving car park
Car Cr10 is going to40,232
Car Cr10 moved to38,203

Car Cr11 is going to40,232
Car Cr11 moved to54,194
Time Thread: New Time 933972245770
Car Cr20 is going to40,232
Car Cr20 moved to37,148
Cr19is closing...
Cr19 leaving car park
Car Cr24: Going to sleep ...waiting for birth
Cr21is closing...
Cr21 leaving car park
Car Cr25's action is 1 (0-centre, 1-space)
Distance from car Cr25 To space 57,179 :36
Cr22is closing...
Cr22 leaving car park
Time Thread: New Time 933972247770
Car Cr6 is going to40,232
Car Cr6 moved to38,203
Unparked ... leaving car park
Time Thread: New Time 933972249770
Cr10is closing...
Cr10 leaving car park
Car Cr20 is going to40,232
Car Cr20 moved to37,175
Cr17is closing...
Cr17 leaving car park
Cr18is closing...
Cr18 leaving car park
Time Thread: New Time 933972251770
Car Cr25's action is 1 (0-centre, 1-space)
Distance from car Cr25 To space 57,179 :0
Car Cr25 has successfully parked at 57,179
Car Cr3: Going to sleep ...waiting for birth
Car Cr2: Going to sleep ...waiting for birth
Cr23is closing...
Cr23 leaving car park
Car Cr1: Going to sleep ...waiting unparking
Time Thread: New Time 933972253770
Cr6is closing...
Cr6 leaving car park
Cr11is closing...
Cr11 leaving car park
Time Thread: New Time 933972255770
Car Cr20 is going to40,232
Car Cr20 moved to38,202
Car Cr24: Going to sleep ...waiting for birth
Car Cr25: Going to sleep ...waiting unparking
Time Thread: New Time 933972257770
Cr7is closing...
Cr7 leaving car park
Time Thread: New Time 933972259770
Cr20is closing...
Cr20 leaving car park
Time Thread: New Time 933972261770
Car Cr3: Going to sleep ...waiting for birth
Car Cr2: Going to sleep ...waiting for birth
Time Thread: New Time 933972263770
Time Thread: New Time 933972265770
Car Cr24: Going to sleep ...waiting for birth
Car Cr1: Going to sleep ...waiting unparking
Time Thread: New Time 933972267770