Modelo de Estratégia e Coordenação Genérico para Sistemas Multi-Agente

João Pedro Bugalho Certo

Dissertação realizada no âmbito do
Mestrado Integrado em Engenharia Electrotécnica e de Computadores
sob orientação de
Professor Doutor Luís Paulo Gonçalves dos Reis
Professor Doutor José Nuno Panelas Nunes Lau

_____

(O Presidente do Júri, Professor Doutor João Carlos Pascoal de Faria)

Faculdade de Engenharia da Universidade do Porto
Departamento de Engenharia Electrotécnica e de Computadores
Rua Roberto Frias, s/n, 4200-465 Porto, Portugal

Março de 2008

Faculdade de Engenharia da Universidade do Porto

A Generic Coordination and Strategy Model for Multi-Agent Systems

João Pedro Bugalho Certo

joao.certo@fe.up.pt

Master's thesis of the Mestrado Integrado em Eng.ª Electrotécnica e de Computadores

Faculdade de Engenharia da Universidade do Porto

supervised by Luís Paulo Reis[1] and Nuno Lau[2].

Faculdade de Engenharia da Universidade do Porto

Departamento de Engenharia Electrotécnica e de Computadores

Rua Roberto Frias, s/n, 4200-465 Porto, Portugal

March 2008

---

1 Luís Paulo Reis – PhD, Assistant Professor at the University of Porto (FEUP) Researcher and Member of the Directive Board of LIACC (NIAD&R)

2 Nuno Lau – PhD, Assistant Professor at the University of Aveiro Researcher at IEETA

Abstract

Managing a team of heterogeneous robots performing a cooperative task in a dynamic environment poses a challenging job. In this thesis a model for a multi-purpose, real-time, adaptable, strategical coordination layer is presented.

Based on previous work developed for the RoboCup Soccer simulation, small-size, middle-size and legged leagues, a generic coordination model was built. Communication was an important factor to consider only introducing a minor communication overhead as both centralized and distributed architectures are handled by the layer. A multi-level hierarchical approach is proposed with hybrid methods used to switch between concepts.

Strategies may be designed with the help of a developed graphical tool. The tool facilitates the strategy design by allowing a graphical interconnection between strategical concepts and the quick reuse of parts of previously designed strategies.

The strategical coordination model was instantiated in the RoboCup Soccer and RoboCup Rescue Simulation domains. For the RoboCup Soccer a graphical debugging tool was also developed in order to better understand the layer effects on the domain.

For the RoboCup Rescue domain, a collection of domain related strategical objects was created as a basis for strategy design. In this domain a full model compliant strategy, based on the previous behaviour of FC Portugal's team was tested. The former strategy was compared to a new search and rescue designed strategy showing the usefulness of the approach.

Resumo

A gestão de uma equipa de robots heterogéneos que executam uma tarefa cooperativa num ambiente dinâmico é um desafio considerável. Nesta tese apresenta-se um modelo de uma camada de coordenação estratégica genérica para ambientes de tempo real.

O modelo de coordenação genérico foi construído com base em trabalho prévio de investigação nas ligas de futebol do RoboCup, nomeadamente as de simulação, robôs pequenos, médios e de quatro patas. A comunicação foi um factor a ter em consideração uma vez que a camada suporta quer ambientes centralizados quer descentralizados onde esta é limitada. Seguiu-se uma arquitectura hierárquica com diferentes níveis de abstracção conceptual usando-se métodos híbridos para interligar conceitos.

De modo a facilitar o desenvolvimento de diferentes estratégias, foi construída uma aplicação gráfica. Esta ferramenta facilita a concepção de estratégias ao permitir a interligação gráfica entre objectos de conceitos estratégicos. A ferramenta também facilita a reutilização de partes de estratégias criadas previamente.

O modelo de coordenação estratégica foi instanciado para os domínios de aplicação de futebol e de busca e salvamento do RoboCup. Para o domínio de futebol do RoboCup foi ainda desenvolvida uma ferramenta de depuração gráfica de modo a permitir uma melhor compreensão dos efeitos da camada estratégica no domínio.

Para o domínio da busca e salvamento foi projectada uma colecção de objectos estratégicos do domínio de modo a servirem de base ao projecto de estratégias. Foi efectuada uma adaptação dos comportamentos existentes da equipa FC Portugal neste domínio convertendo-os numa estratégia que segue o modelo proposto. A estratégia referida anteriormente foi comparada com uma nova estratégia de busca e salvamento criada de raiz para seguir o modelo proposto, demonstrado a utilidade da abordagem desenvolvida.

Acknowledgements

I would like to acknowledge the following institutions:

FCT - Fundação para a Ciência e a Tecnologia - who, under the grant POSC/EIA/63240/2004 financed part of the work presented in this thesis.

IEETA - Instituto de Engenharia Electrónica e Telemática de Aveiro – for the excellent working conditions and working environment in particular the "Transverse Activity on Intelligent Robotics" (ATRI) group in which I was inserted as part of the FC Portugal's team/project.

LIACC - Artificial Intelligence and Computer Science Laboratory – in the research group of Distributed Artificial Intelligence and Robotics (NIAD&R) I learned a lot about science in general, but mostly that is possible to do good science and divulging it without being serious all the time and wearing those white coats. My view on science changed form an eight headed monster into a work in which you can have some fun and sense of accomplishment while aiming to have a good impact on humanity (although somewhat long term).

FEUP – Faculdade de Engenharia da Universidade do Porto – I learned a lot during this past few years, had some of the best teachers (unfortunately a few bad as well) certainly this work would not be possible without the knowledge gathered in here.

The next acknowledgement are of a more personal nature so I apologize to English readers but I will write the rest in Portuguese as some of the people mentioned don't understand English.

É sempre difícil escrever esta parte, há sempre o medo de deixar, injustamente, alguém de fora pelo qual peço desculpa antecipadamente.

Um obrigado muito especial a toda a minha família por todo o apoio, em particular aos meus pais e irmã.

Agradeço e aprecio bastante a amizade daquele pequeno grupo da figueira, vocês sabem quem são. Quero reconhecer ainda a amizade do Nuno, Vanessa, Zombie, João A., HC, Cyber, Billy, Ritinha, Marta, Dani, Fabí, Martinha, mais alguns amigos do curso, grupo do euro, grupo do teatro, resto do grupo dos encalhados, grupo das enfermeiras (que me matam por lhes referir assim), … Por fim, mas não de modo algum em último, um agradecimento ao Nuno e Luís Paulo, em particular a este último pela amizade, oportunidades e ensinamentos.

Contents

List of Figures

List of Tables

Chapter 1

1. Introduction

The work described in this thesis is related to the study and development of generic coordination methods for multi-agent systems (MAS) with more emphasis on coordinating heterogeneous teams for performing complex tasks in dynamic, and inaccessible environments like the domains used in the RoboCup international initiative (Kitano, et al., 1997).

Being a high-level software abstraction, an agent provides a convenient and powerful way to describe a complex software entity, capable of acting autonomously in order to accomplish tasks. But unlike objects, which are defined in terms of *methods* and *attributes*, an agent is defined in terms of its behaviour (Charniak and McDermott, 1985; Russell and Norvig, 1995).

RoboCup was created as an international research and education initiative, aiming to foster Artificial Intelligence (AI) and Robotics research, by providing standard problems (Kitano, et al., 1997). RoboCup has two main league types: simulation and robotics. Simulation leagues enable research on AI and multi-agent coordination while waiting for the availability of hardware to enable the same type of research.

The main application domain for RoboCup is soccer. The choice is justified as the sport enjoys worldwide popularity and presents a great number of scientific challenges. In RoboCup Soccer leagues two opposing teams play a soccer match, with simplified rules and setup, creating a dynamic multi-agent environment. The scientific challenges range from an individual level (perception, moving, dribbling, shooting) to a collective level (strategy, collective play, formations, passing, etc.). There are a number of RoboCup Soccer leagues including robotic small-size, middle-size, legged, humanoid, simulation 2d and simulation 3d.

Although its main application is soccer, RoboCup is also concerned in applying the methodologies developed to more socially useful problems. Search and rescue of victims in large-scale disasters are serious and very difficult tasks presenting several challenges from a scientific point of view. Unprepared cities can suffer tremendous consequences in a natural catastrophe as was the case in Kobe's earthquake (Pollitz and Sacks, 1997; Reinaldo, et al., 2005) or, more recently, the south Asian tsunami of 2004. Every city needs an emergency

plan, to reduce the loss of human life in a natural disaster. In recent years, staggering technological breakthroughs brought some science fiction dreams closer to us. The innovations in robotics and artificial intelligence have opened doors, and allowed for a complete new use of rescue agents in emergency plans (Reinaldo, et al., 2005).

Proposed by Kitano (Kitano, et al., 1999), the RoboCup Rescue simulated environment consists of a virtual city, immediately after a big catastrophe, in which heterogeneous, intelligent agents, acting in a dynamic environment, coordinate efforts to save people and property. The agents are of six different types: Fire Brigades, Police Forces, Ambulance Teams and the three respective center agents. Fire Brigades are responsible for extinguishing fires, Police Forces open up blocked routes and Ambulance Teams unbury Civilians and transport them to safety. In order to obtain a good score, all these agents work together communicating through supervising center agents.

1.1. Motivation

The author, as a member of FC Portugal research team, sets his research focus on the development of new coordination methodologies. After successfully developing such methodologies for soccer simulation leagues, focus changed to the adaptation of these methodologies to the Rescue Simulation League, already with some success[3].

Members of the team are also involved in different robotic soccer teams (simulation 2D, simulation 3D, small-size, middle-size and legged) that, in order to collaborate between themselves, need a common strategic layer. Furthermore, a strategy developed for one soccer league, has many similarities with strategies in other soccer leagues. Also in some of the leagues our participation includes collaboration with other universities and thus the need of a common strategy enabling cooperation from robots developed by different universities. One of the expectations of RoboCup is to stimulate technology development in the hope that it can be applied to other areas.

---

3 FC Portugal won several World and European championships in different RoboCup soccer leagues in the past eight years, the rescue simulation team achieved very good results in RoboCup, including winning a rescue European championship.

1.2. Objectives

The objective of this thesis is to develop the specification and application of a multi-purpose, multi-domain, adaptable, strategical coordination layer for multi-agent systems. This layer should allow the management of homogeneous and heterogeneous agents and the centralized or decentralized management of a flexible strategy. The layer should be tested on the Search and Rescue Simulation League and be compatible with the Soccer Simulation Leagues. The model and tools developed should aim to simplify the portability of research between RoboCup leagues and expand its usefulness to areas outside RoboCup.

The thesis specific objectives are:

- Analysis of existing technology in multi-agent coordination;
- Study of the testing domains for the layer;
- Development and formalization of a generic coordination and strategy model;
- Validation of the generic coordination and strategy model.

1.3. Thesis Outline

After this introduction, the rest of this thesis is organized as follows. The next chapter presents a technical overview of the application domains discussed in the thesis namely the RoboCup Soccer and RoboCup Rescue domains. In chapter 3, related research in the area is presented and analysed including the analysis of existing coordination frameworks either generic or for the test domains. At chapter 4 the strategic layer and related concepts are described and formalized. Chapter 5 presents a graphical tool developed for building strategies compliant with the defined model. On chapter 6, a strategy for the soccer domain is demonstrated as well as a developed visual debugger for the better understanding of the layer effect in the domain. Chapter 7 presents an implementation on the rescue team, a modelling of existing behaviours, the creation of a new, domain related, strategy model objects and the comparison of newly developed strategies. Finally, chapter 8 presents the thesis conclusions and points out to future work.

Chapter 2

2. Application Domains

Although generic the work discussed in this thesis was developed with two main application domains in mind. In this chapter a presentation of the RoboCup Soccer 2D Simulation and RoboCup Search and Rescue Simulation Domains is made. The domains are presented in terms of simulated environments, architectures, simulator components and human to agent modelling including agent sensors and actuators.

2.1. RoboCup Soccer Simulation League

The RoboCup soccer simulator is a system that enables various teams to compete in a simulated game of soccer. A full sized soccer field is simulated in the competition. Since the match is carried out in a client-server style, each client is a separate process and a team can have up to 12 clients, i.e. 11 players (10 fielders + 1 goalie) and a coach. Players receive noisy perception information (visual, aural and physical) from the server. Using this information, players decide the actions to perform and send requests to the server regarding the actions they want to perform (e.g. kick the ball, turn, run, etc.). The server executes the actions in an imperfect way and updates the world moving the objects on the field and enforcing the game rules.

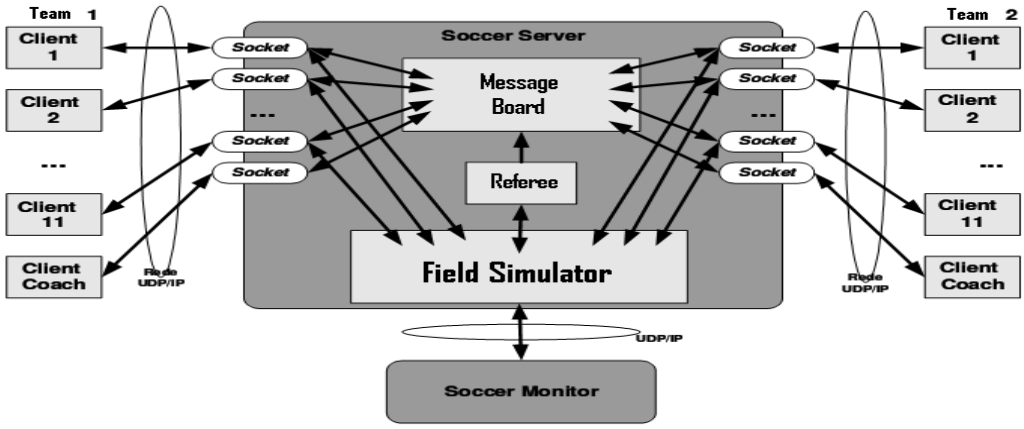

Figure 1: 2D Simulation Soccer Server Architecture (Reis, 2003).

The simulator architecture, on Figure 1, has the following components (Chen, et al., 2002):

- rcssbase is the base code used by the other modules.
- rcssserver performs the actual simulation.
- rcsslogplayer allows you to replay logs (*.rcg files) created by rcsserver.
- rcssmonitor and rcssmonitorclassic allow to a watch game in progress or a game being replayed by the log player.

2.1.1. The server

The server receives messages from the clients and monitor, handles the requests, and updates the environment accordingly. In addition, the server provides all players with sensory information (e.g. visual data regarding the position of objects on the field, or data about the player's resources like stamina or speed). It is important to mention that the server is a real-time system working with discrete time intervals (or cycles). Each cycle has a specified duration, and actions that need to be executed in a given cycle (default 100ms), must arrive at the server during the right interval. Therefore, slow performance of a player that results in missing action opportunities has a major impact on the performance of the team as a whole.

2.1.2. The Soccer Monitor

The Soccer Monitor is a visualization tool that allows people to see what is happening within the server during a game. Currently the monitor comes in two flavours, the rcssmonitor and the rcssmonitor classic.

Figure 2: A screenshot of the rcssmonitor.

As shown on Figure 2, the information shown on both monitors include the time and score, team names, and the positions of all the players and the ball. Monitors also provide simple interfaces to the server. For example, when both teams have connected, the ”Kick-Of” button on the monitor allows a human referee to start the game.

2.1.3. The Logplayer

The Logplayer can be thought of as a video player. It is a tool that is used to replay matches. When running the server, certain options can be used that will cause the server to store all the data for a given match on the hard drive. Then, the program rcsslogplayer combined with a monitor can be used to replay that game as many times as needed. This is quite useful for doing team analysis and discovering the strong or weak points of a team. Much like a video player, the logplayer is equipped with play, stop, fast forward and rewind buttons. Also the logplayer allows you to jump to a particular cycle in a game (for example if you only want to see the goals).

2.1.4. Human Player Modelling

The modelled player has three different kinds of sensors. The hearing sensor receives messages from the referee, coach, team players and opponents. The visual sensor models human vision and receives visual information including object’s (ball, flags) and player’s distance and direction if they are in the field of view. The physical sensor detects the state of the agent including its stamina, recovery rate, velocity and neck angle (Reis, 2003).

In simulated 2D soccer the players, like in real world, are heterogeneous and have different capabilities. Some players are faster but have smaller kickable areas around them, some have increased power for kicking the ball, others have increased effort recovery capabilities, some possess more stamina, etc... The coach agent has the power to select distinct players for his team and thus has to weight the tradeoffs between choosing one player or another as the capabilities are randomly generated for each game.

Sensory Information

In simulated soccer the hearing is very limited as a player can only hear one message each cycle. Messages are also limited in size (10 bytes) and so, choosing the right message to hear can be crucial. The soccer server also limits the hearing distance (default 50 m).

Visual information is critical in soccer matches. Modelled players can alter their peripheral view to 45, 90 or 180 degrees. Figure 3 shows how the vision sensor works. In a 3 m circle around the player other player’s presence is perceived even if not in the field of vision (a). Also, for those players outside the field of vision (f) is only possible to detect their presence (not the

team). As the players become closer it is possible to identify the team and the player's number and, with a probability inverse to the distance, the direction and distance ("d" more likely than "e"). For the closest players, it is possible to know with good precision their direction and distance along with the player's identification (c). All the visual information is subjected to noise that increases with distance.

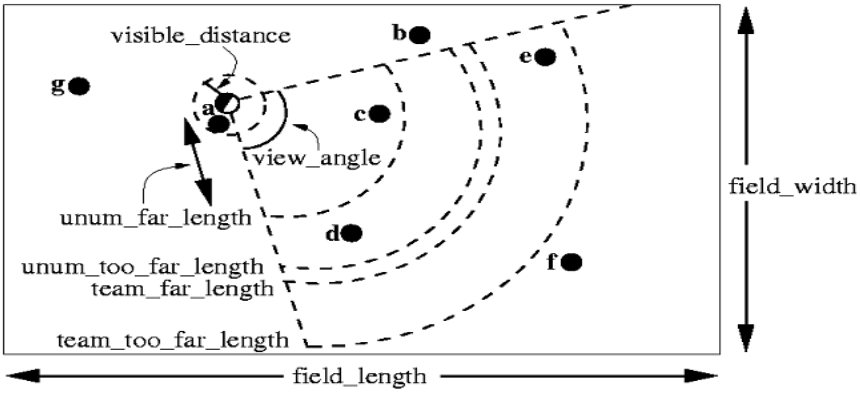

Figure 3: Agent's Visual Perception (Stone, 2000).

The quality and receiving frequency of the visual information is also influenced by the view mode as seen on Table 1. The narrower the choice of viewing angle bigger the frequency of the visual information updates. Likewise, decreasing the viewing quality of the sensor will increase the frequency of the information at the cost of losing some objects' and players' properties like distance and direction.

Table 1: Soccer View Modes (Reis, 2003).

Width / *Quality*	***Narrow***	***Normal***	***Wide***
High	*send_step/2* *75 ms*	*send_step* *150 ms*	*send_step*2* *300 ms*
Low	*send_step/4* *37.5 ms*	*send_step/2* *75 ms*	*send_step* *150 ms*

Actuators

There are four main types of actions in simulated soccer: movement (dash, turn, move), ball interaction (kick, tackle, catch), perception control (turn_neck, attentionto, change_view) and communication (say, pointto). Players are restricted to executing only one movement or ball interaction per cycle while being able of executing any number of distinct perception control and communication commands.

2.2. RoboCup Search and Rescue Agent Competition

In RoboCup search and rescue simulator the action takes place in a simulated city, where a natural disaster (earthquake) has just taken place (Certo and Cordeiro, 2005; Certo, et al., 2007a; Committee, 2000; Kitano, et al., 1999). This city is dynamically modelled by the following equations,

$$e(t) = f(x(t),u(t),t) \quad (1)$$

$$x(t + \Delta t) = g(x(t),e(t)) \quad (2)$$

where:

e(t) represents the effects that create change in the city, calculated by **f**.

t represents the current time instant.

f is the function describing how **x(t)** e **u(t)** affect the simulated world, changing its status.

x(t) is the status variable. It represents the disaster situation in instant t. Every variable such as the strength of fire or the speed of cars is saved in the form of a vector.

u(t) is the input vector in instant **t**, representing external effects like water sprayed by Fire Brigades and debris removed by Police Forces.

Δt is the time step used to forward the simulation discretely.

Finally, g is the function that describes the values of status **x(t)** at the instant immediately after **t**, i.e. instant **t + Δt** (Committee, 2000).

At **t=0**, **x(t)** represents the initial situation.

From **x(0)** we can obtain **Sint** as the total vitality or health points (HP) of all agents at start, **Bint** as the total undamaged building area at start.

At any time step we can obtain **P** as the number of living agents, **S** as the remaining total HP of all agents, **B** as the total undamaged area of buildings. The simulation score V is calculated using the following equation:

$$V = \left(P + \frac{S}{Sint} \right) * \sqrt{\frac{B}{Bint}} \quad (3)$$

Given any simulation, the higher the V value, the better the rescue operation (Akin, et al., 2004).

Note that at the beginning of the simulation (**t=0**):

$$V = (P + 1) \tag{4}$$

As the simulation proceeds, more buildings are damaged and people hurt, causing the score to drop till its final value at **t=300**.

As such the initial value of **V** is the maximum possible score for a given simulation.

2.2.1. System Structure

A schematic representation of the simulation system can be seen on Figure 4.

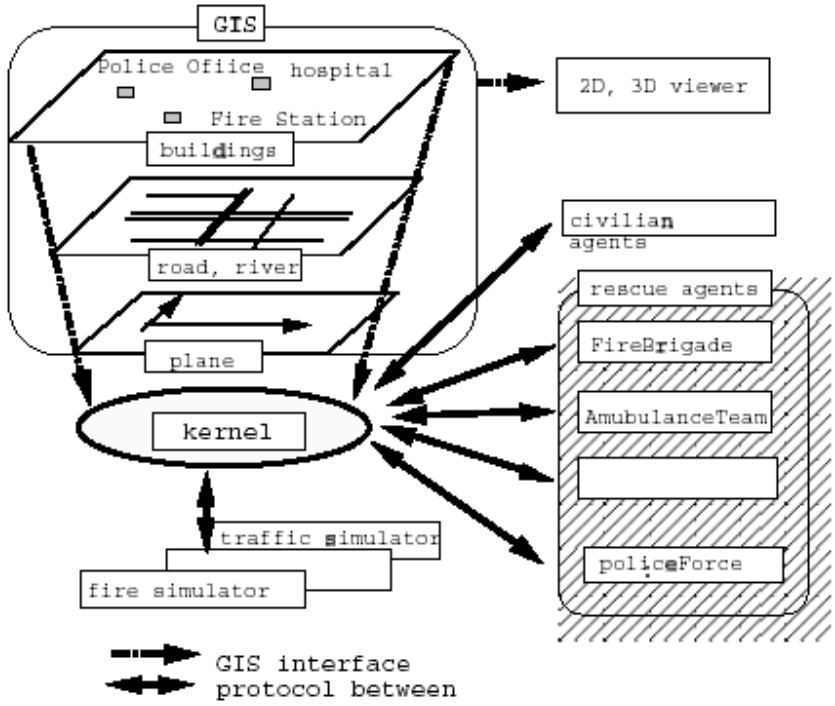

Figure 4: Simulation System functional outline (Committee, 2000).

This structure allows a relatively autonomous development of the different simulator modules, since once the communication protocol is defined, the modules are mostly independent.

The communication between modules takes place by message exchange. The organization depicted on Figure 4 is a flexible one and, due to recent evolutions, it may change into something slightly different since, for example, the new fire simulator is now able to communicate directly with the collapse simulator – although the collapse simulator module is not yet ready for this. A description of the different modules and their representation on the system is given below.

2.2.2. Kernel

The kernel is the central processing unit of the system, controlling the simulation process and facilitating information exchange between modules. It is responsible for establishing and maintaining communication with the Geographic Information System (GIS), the Simulators (Collapse, Fire, Traffic, etc.), the Viewer, and the Agents; as is depicted on Figure 5.

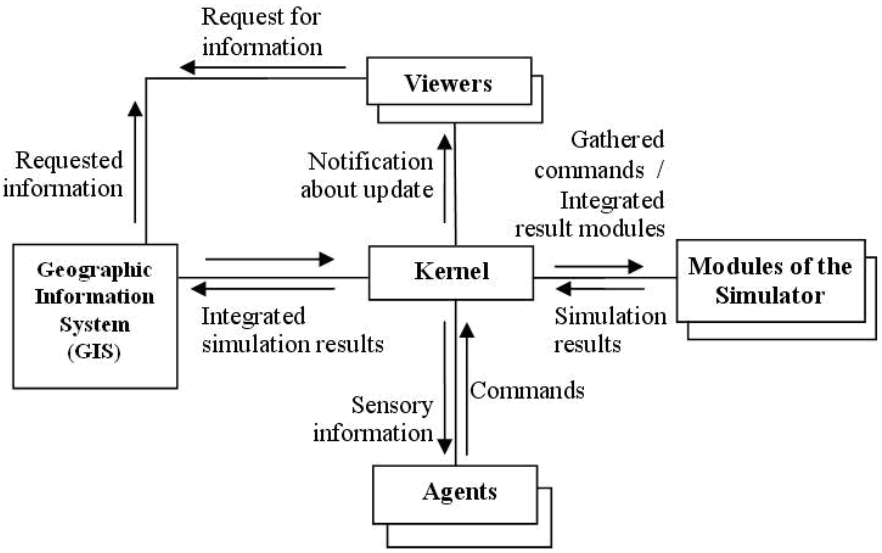

Figure 5: RoboCup Rescue Simulation System.

When the program starts, the Kernel receives from the GIS module the initial configuration of the simulated world. At every step of the simulation, the Kernel sends sensory information to all agents and receives their action commands. Information is sent and received from the modules as necessary and, for each data exchange, the command and information validity is verified (Takahashi, et al., 2000).

2.2.3. GIS (Geographical Information System)

The GIS module is responsible for the initial configuration of the simulated world. This is composed by the location and properties of buildings, roads, nodes, refuges, agent centers, Civilians, Ambulances, Fire Brigades, Police Forces and initial fires. It also records the simulation progress into a log file, enabling a detailed offline analysis. Additionally, this module is responsible for feeding data to the viewer.

2.2.4. Current Simulators Modules

As the project evolves, simulator modules are added or improved, deepening the complexity and adding realism to the simulation. The current simulator is composed by the collapse simulator responsible for modelling building integrity, the blockade simulator which models road obstructions caused by debris, the traffic simulator that models people's movements, the fire simulator modelling building consumption by fire and fire spread through the city and finally the miscellaneous simulator which models human physical state including the vitality and degree of buriedness.

Collapse Simulator

This module acts on the physical state of buildings after the earthquake. On a large scale disaster like the one RoboCupRescue aims to emulate, around 80% to 90% of households are at least partially collapsed, shortly after the calamity. Currently, this simulator is triggered only once, at the beginning of the simulation. In the images displayed by the default viewer[4], like the one displayed in Figure 6, the more damaged a building is, the darker it looks – as can be seen by the buildings numbered 1 to 4.

This is one of the modules schedule to be revised shortly, as new features – such as earthquake aftershocks – are expected.

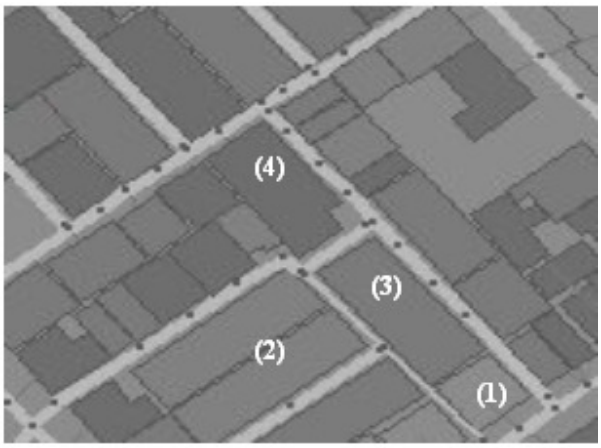

Figure 6: Buildings with growing collapse levels (from 1 to 4).

[4] The default viewer, developed by Morimoto can be found at http://ne.cs.uec.ac.jp/~morimoto/rescue/viewer/index.html

Blockade Simulator

This is the module responsible for defining the state of road obstructions. After the earthquake, a large part of the roads gets blocked, hindering traffic flow. These obstructions may have different causes such as crowds, debris from buildings and traffic accidents. Blocked roads can only be cleared by Police Force agents, this way allowing other agents to freely move through.

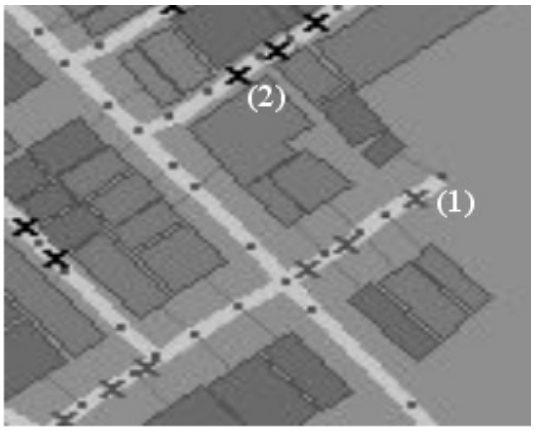

Figure 7: Blocked Roads.

As seen on Figure 7, the default viewer represents road blocks with crosses, which can be either gray or black. A gray cross (1) means that the road is only partially blocked, while a black one (2) marks a block which cannot be crossed. Although the viewer simplifies this feature it is a fairly complex one; blocks are defined and simulated in millimetres, even though some do not show up in the viewer. These affect the speed at which agents can travel through roads and the number of usable lanes, impacting on traffic jams.

Traffic Simulator

Every agent's movement in the world, including Civilians, is modelled by this component, which defines the pace allowed on every road section. The road width, number of agents present, and the level of "blockness", are some of the factors affecting maximum speed on a street. Usually, a road which is over 50% blocked is not traversable.

Fire Simulator

This module simulates the spread of fire in the city. It is currently one of the most evolved components of the simulator package. Right after the earthquake, some buildings ignite and start radiating heat to nearby structures. This component is responsible for the physical simulation of combustion and heat spread. This is done resorting to an intelligent model in which the temperature of a building is, on the one hand increased, either by the its own combustion or by the radiation waves from neighbouring buildings, and on the other hand decreased, due to the

evaporation of water, pumped by Fire Brigade agents. In the simplified combustion model used, the critical factors are temperature and fuel (buildings), with the supply of oxygen being disregarded. When a building's temperature rises above its material's flash point it bursts into flames, as seen on Figure 8. In contrast, when the temperature drops below this temperature, its fire is extinguished (Nüssle, et al., 2004).

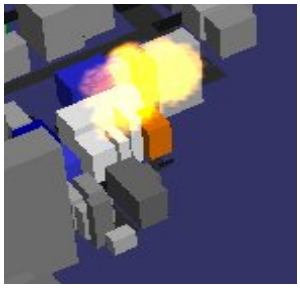

Figure 8: An image from Freiburg's 3D viewer shows a building on fire.

Miscellaneous Simulator

The agent's status is modelled by this simulator. When an agent is inside a burning building or trapped under debris, its health is affected and starts decreasing. This is the module responsible for controlling the agent's properties in these situations. As a simple example, a large value for the agent's property *buriedness* describes its state as trapped under debris. When Ambulances use their rescue ability, this value is progressively reduced until the agent is free.

In Figure 9, the status of the depicted Civilians has changed. In the default viewer, Civilians are represented by green circles that get darker as their health degrades. The Civilian in the bottom changed from healthy to deceased in just a few cycles, because the building he was trapped in burned to the ground.

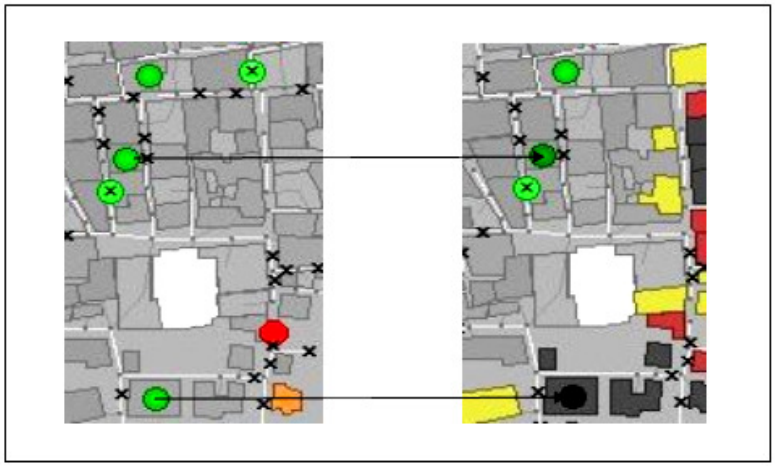

Figure 9: Civilians with evolving status.

2.2.5. Simulated World Objects

When trying to model a city, a number of simulated world objects were incorporated. These objects include:

- World - The parameters about the simulated world are stored in this object. These parameters include location, time, and wind (strength and direction);
- Roads and Crossings - The road network is represented by a graph, in which edges are roads and nodes are crossings;
- Rivers and River Nodes - Like with roads, the river network is also represented by a graph. Although possible, there are no current implementations of these objects;
- Buildings - Buildings have parameters like position, number of floors, primary construction material, fieriness, brokenness and total area, amongst many others;
- Refuge - The refuge (Figure 10) is a special kind of building where Civilians take shelter and Fire Brigades can fill their water tanks. It has mostly the same properties as any other building, but it is indestructible.

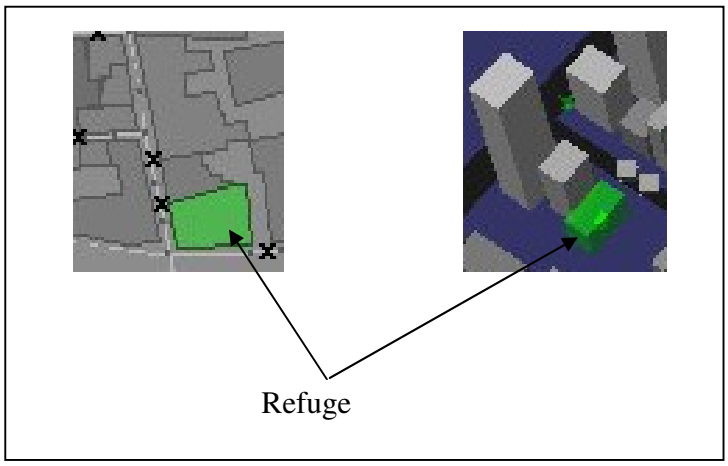

Figure 10: The refuge 2D and 3D views.

2.2.6. Agents

Whereas every team is responsible for the creation of their own rescue agents, Civilian agents are directly controlled by the simulator. Some new types of agents were recently proposed and their possible implementation is being discussed. The number of each type of agent is defined for every simulation.

Civilians / Cars

Civilians represent people in the system, but due to current computational limitations, every Civilian represents a family or similar aggregate. They possess parameters such as location, health and buriedness, amongst others (Committee, 2000). Figure 11 shows how the default viewer represents different health values for Civilians. Civilians buried under debris can be rescued by Ambulances.

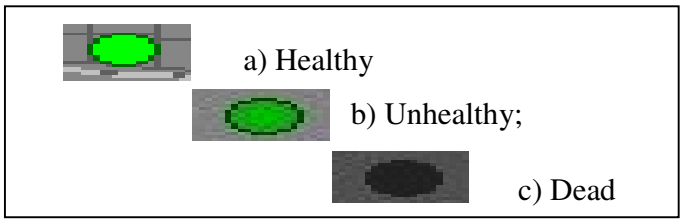

Figure 11: Civilian status.

The car is a purely conceptual agent, created because Civilian agents act like motorized vehicles when moving around the city. From every other point of view, cars and civilians are only one type of agent.

Field Agents

These types of agents are developed by rescue teams. They all have the same properties as a Civilian, plus some type specific ones. Field agent types are described below. In the default viewer, agents are represented by colored circles. Police and Ambulance activity is manifested by a surrounding halo, while Fire Brigades shoot blue water streams (Figure 17 in section 2.2.7).

Ambulance Team agent

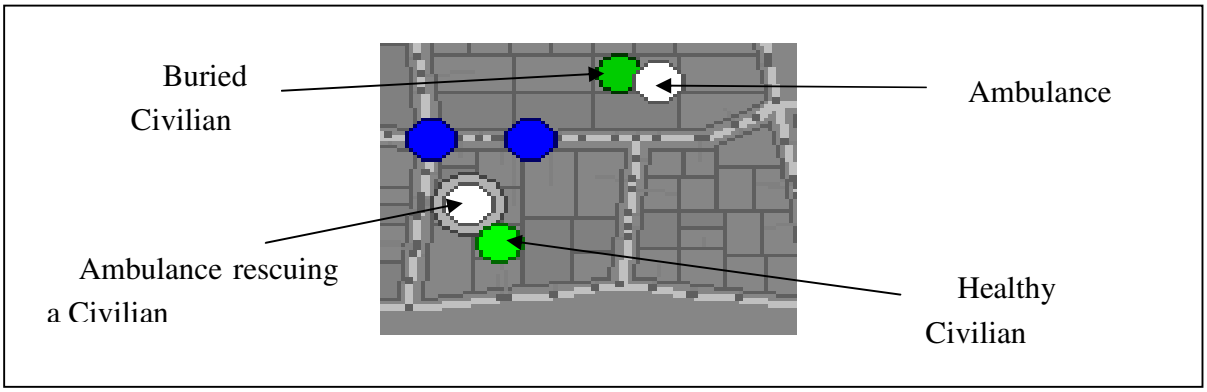

Figure 12: Ambulance and buried Civilian.

This is the agent responsible for rescuing Civilians from collapsed buildings. It bears all properties of a Civilian plus the ones specific to its function, namely: unbury Civilians, load

them to the Ambulance and unload them in refuges. Figure 12 depicts two Ambulances, with one of them rescuing a Civilian.

Fire Brigade agent

Fire Brigades are responsible for putting out fires all over the city. They share all the properties of a Civilian, plus the ones related to its ability to throw water at buildings - such as water quantity. Figure 13 shows two Fire Brigades. One is inside a burnt out building fighting the flames around him (as denoted by the water stream), and the other one is finding a suitable position to cooperate with his teammate.

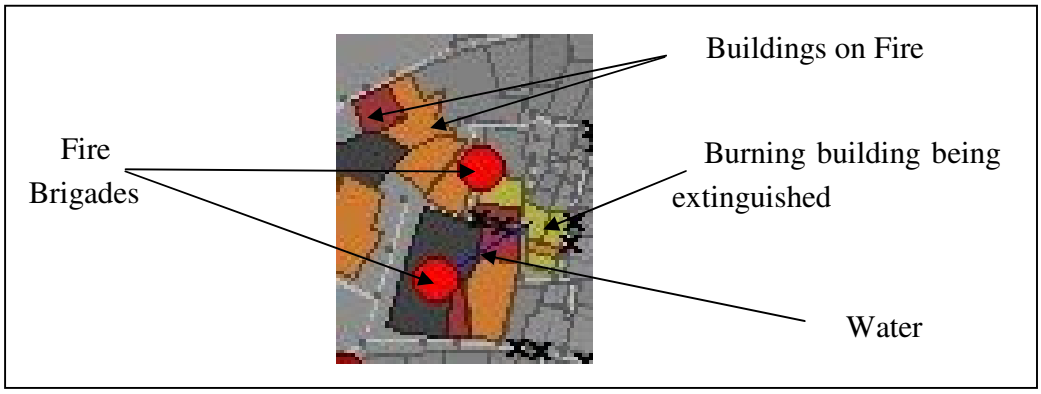

Figure 13: Fire Brigades extinguishing a fire.

Police Force agent

This agent is responsible for cleaning up road blocks such as debris, traffic accidents and other kind of obstacles. It possesses all properties of a Civilian, plus the ability to remove these obstructions. Figure 14 shows two Police Forces clearing roads. One of them is approaching an obstacle while the other is actively removing another.

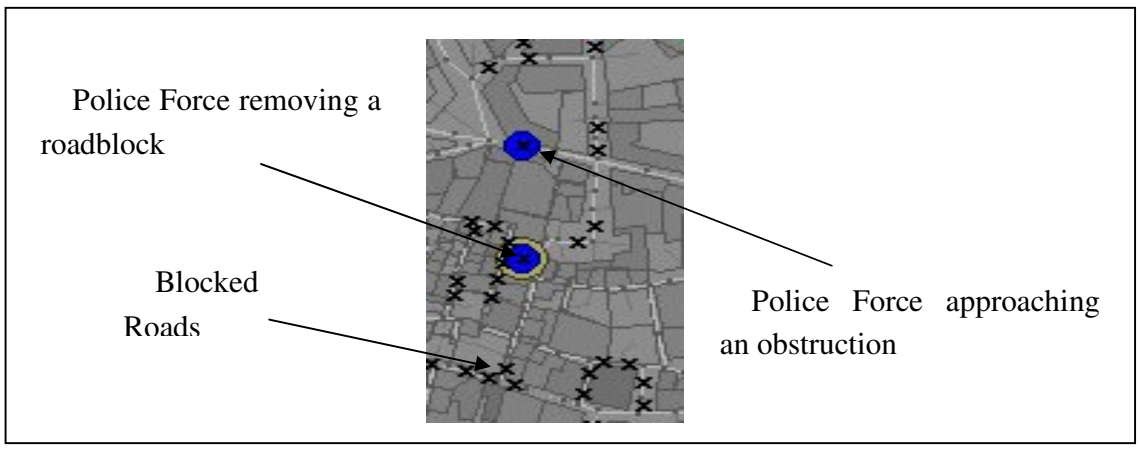

Figure 14: Police agents cleaning roads.

Center Agents

These agents are represented as ordinary buildings – they appear in a lighter colour on the default viewer – possessing all their properties, but having the particularity of being indestructible. Center agents include:

- Fire Station. Center agent for the Fire Brigades;
- Police Office. Center agent for the Police Forces;
- Ambulance Center. Center agent for the Ambulances Teams.

Although represented as buildings, Centers are, conceptually, very different. They are developed by rescue teams and, as will be seen in section 2.2.9 they have, relatively to field agents, enhanced communication skills. This makes them ideal for the coordination and tactical organization of field agents. Each Center can communicate with the field agents of the related type and with the other Center agents.

2.2.7. Viewer

A viewer is the graphical interface used to display the actions taking place in the simulated city. Several viewers exist, but the one included in the official package, and used in the competitions, is Morimoto Viewer, developed by Morimoto[5].

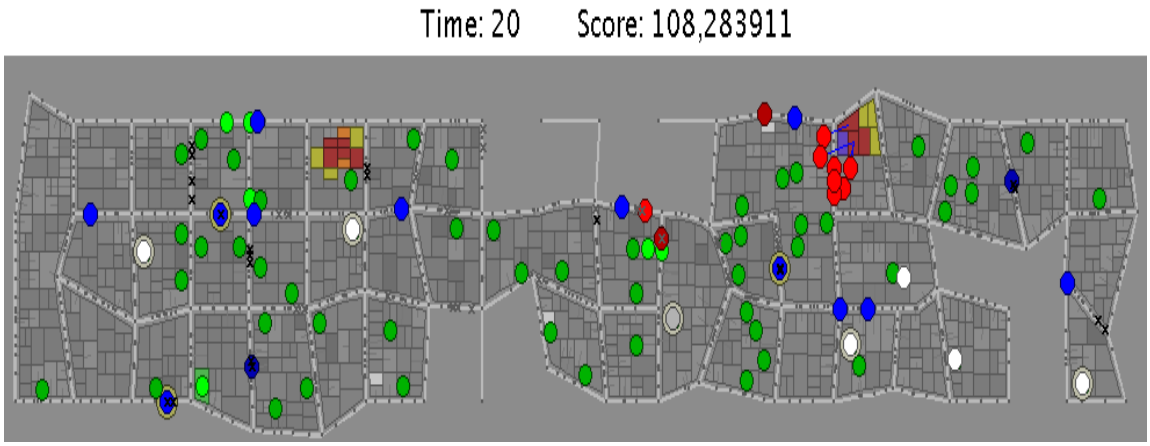

Figure 15: RandomMedium map from RoboCup Osaka 2005.

[5] Takeshi Morimoto is a PhD candidate at the Graduate School of Electro-Communications and is affiliated with the Takeuchi Laboratory.

Morimoto Viewer

Snapshots of the Morimoto Viewer can be found in Figure 15 and Figure 16. This viewer shows agents as colored circles. Ambulance Teams are white; Fire Brigades red; Police Forces blue; Civilians green and all of them get darker when hurt, turning completely black if they die (see Figure 17). Buildings also have different colours, according to their function or status. While refuges are green and Center (agent) buildings are white, those on fire evolve from yellow, to orange, to red. Flooded and extinguished buildings have different shades of blue, while those burnt down are dark grey (almost black). Roadblocks are marked with crosses, and current time and score are displayed on top of the map. The team name can also be displayed on the top bar, but it is optional.

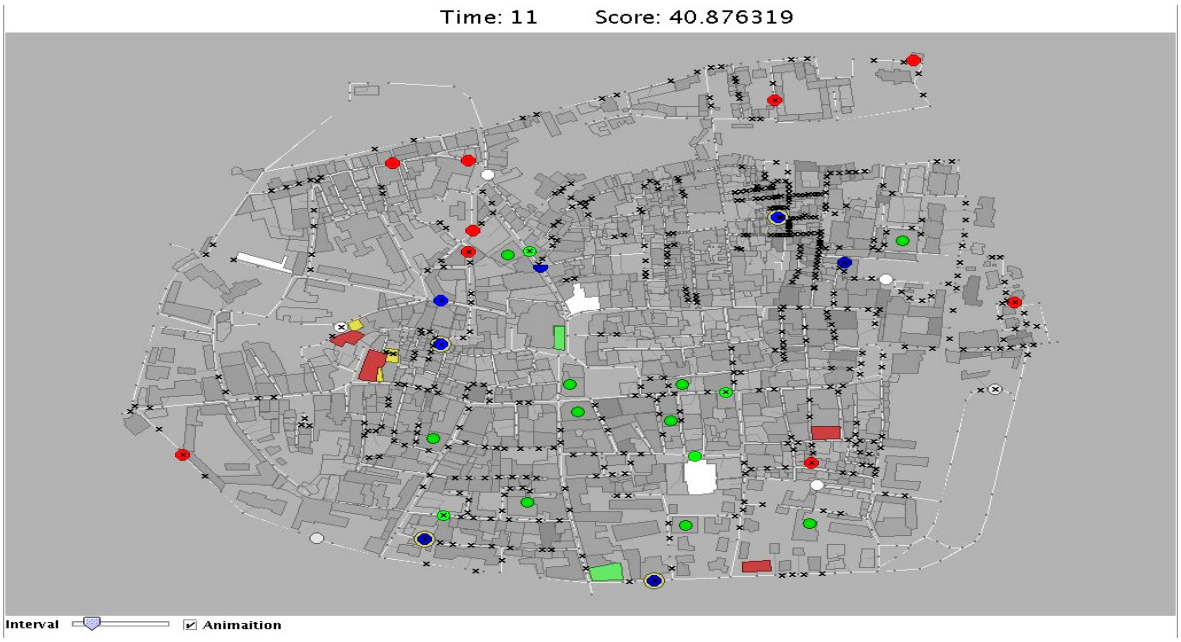

Figure 16: Foligno map.

As previously mentioned, Police and Ambulance activity is denoted with a hallo around the unit. On the other hand, Fire Brigade activity is represented by a blue stream of water shooting from the agent into a building. These agents are represented in Figure 17 along with a Civilian and a dead agent. Viewers can also be used to read log files, displaying an offline simulation.

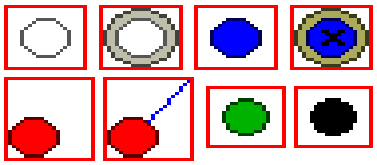

Figure 17: Agents and their respective states as represented in Morimoto Viewer.

Freiburg's 3D viewer

Also widely used is Freiburg's 3D viewer[6], mostly due to its appealing look to the public as seen on Figure 18.

This viewer uses familiar forms to represent agents. For example, Fire Brigades are represented by fire trucks, while on roads, and fire fighter helmets, when inside buildings. One further bonus presented is its ability to display statistical data. Although more attractive, Freiburg's 3D viewer is also resource intensive and conveys information in a more cluttered way, which makes it less suitable for developers, that tend to prefer simplicity.

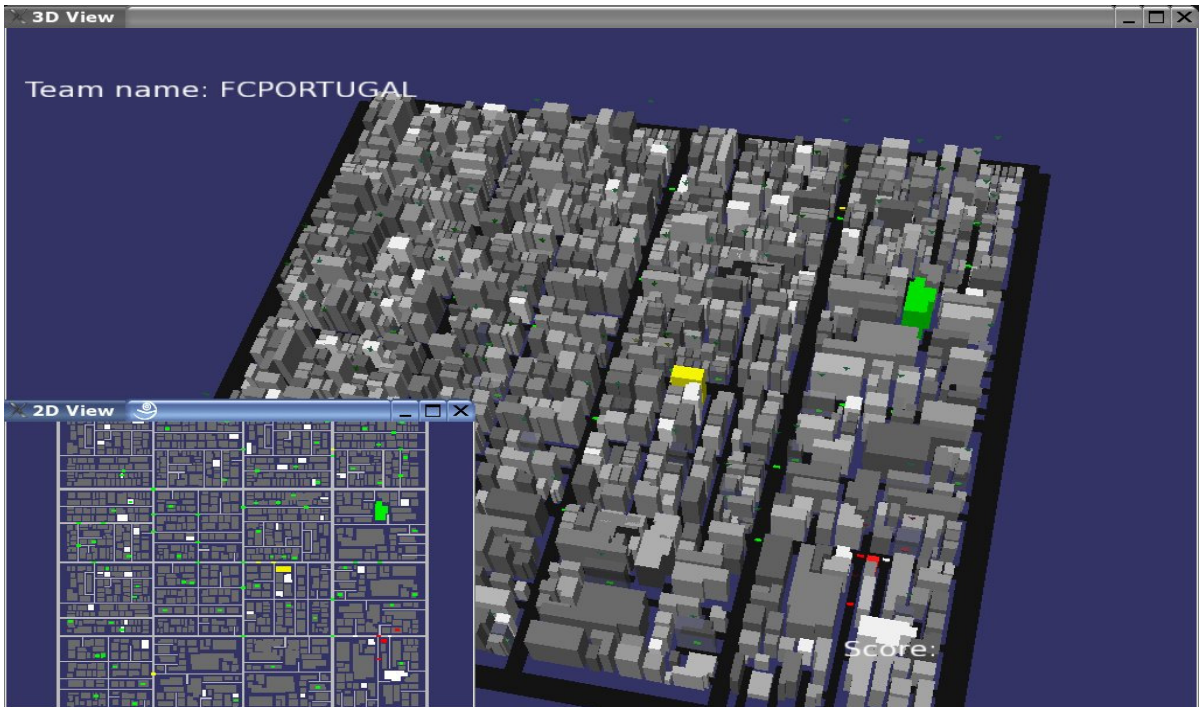

Figure 18: Freiburg's 3D viewer (also displaying its 2D view).

2.2.8. System Configuration

By default, the Simulator is configured with the rules for the RoboCupRescue World Championship (Committee, 2000; Skinner, et al., 2005). However, it is possible to change simulator parameters in order to test strategies, communication options or Civilian behaviour, amongst other possibilities.

[6] The 3D viewer, developed by Alexander Kleiner can be found at http://kaspar.informatik.uni-freiburg.de/~rescue3D/

To gain a general notion of the dimensions used in the simulation parameters, default value ranges for some of the most important parameters are displayed in Table 2 (Skinner, et al., 2005).

Table 2: Parameter ranges in RoboCup 2007 Atlanta.

Entity	Min.	Max.
Fire Brigade	0	15
Police Force	0	15
Ambulance	0	8
Civilian	70	90
Fire Brigade Center	0	1
Police Force Center	0	1
Ambulance Center	0	1
Refuges	0	5
Ignition points	2	8

Some other important parameters are:

- Simulation time: 300 steps (which corresponds to the 72 hours after the disaster);
- Range of agent eyesight: 10m;
- Range of agent voice: 30m.

The simulated map area is usually in the order of a few dozens of square kilometres.

2.2.9. Communication

As of simulator version 0.47 (2005), communication between modules is done in TCP, although UDP is still supported for legacy reasons. High-level communication between modules is described in the simulator manual (Committee, 2000); however, at this stage, we are only interested in the communications between the Kernel and Rescue agents.

At each time instant after the initial setup, Rescue agents receive sensory information from the Simulator, process it, and send their commands to the simulator.

Sensory information can be received in three forms:

- Visual;
- Field hearing;
- Radio hearing.

Visual information is only sent to field agents. Each agent receives all properties of all objects within the radius of his eyesight. The same is true for **field hearing**, also exclusive to field agents – each agent receives all voice messages sent by an agent within voice range, with sender identification and contents.

Radio hearing is very different. Although messages are still in the sender/contents format, all team agents (field and center) receive such a message, regardless of distance. The following restrictions apply: field agents can only hear messages from their Center or from agents of the same type; center agents can only hear messages from other Centers or from their field agents of the same kind. The agents' radio communication scheme can be seen in Figure 19.

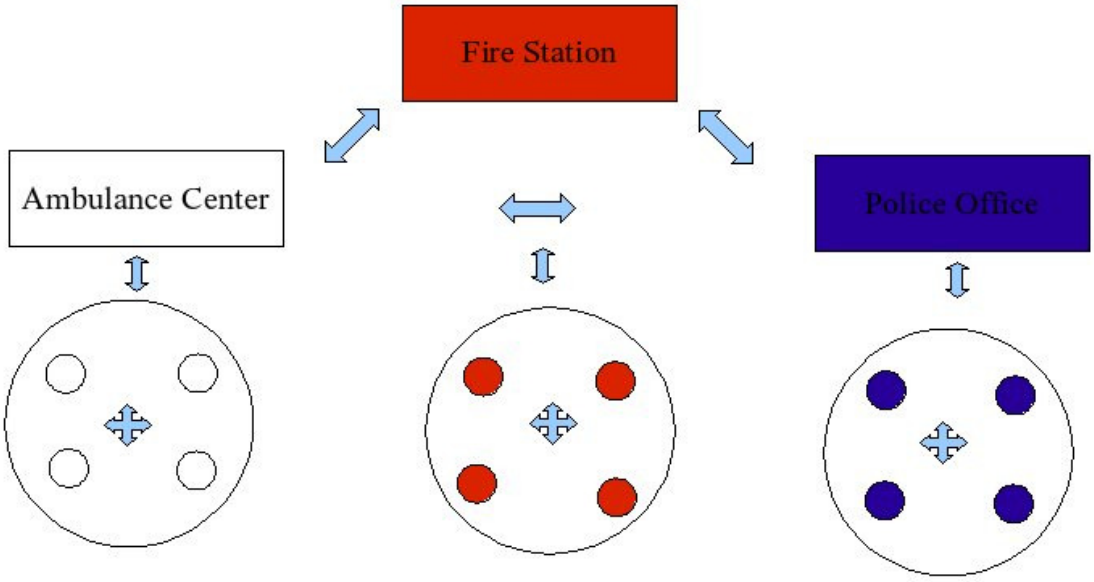

Figure 19: Radio-communications.

There are also other restriction limiting communications. With the changes introduced to the RoboCup 2004 rules (Akin, et al., 2004), all hearing information between team agents is also limited to the following:

- Field agents can only receive 4 messages per cycle;
- Center agents can only receive a maximum 2*n messages, where n is the number of field agents of the Center's type. For example, if there are 10 Fire Brigades, a Fire Station can receive 20 messages per cycle, at most. It should be noted that it doesn't matter if it is a radio or field (voice) message. The restriction applies to the total sum of messages heard.

Sending information works in the opposite way (to hearing). Each field agent has two forms of sending information. Those forms depend on the destination environment:

- Voice – if the destination is the field within voice range;
- Radio – if the destination is the radio channel.

Like hearing, sending information to team's agents has similar restrictions:

- Field agents can only send 4 messages;
- Center agents can only send a maximum 2*n messages, where n is the number of field agents of the Center's type. For example, if there are 10 Fire Brigades, a Fire Station can send 20 messages per cycle, at most.

The data part of any message sent is limited. The current limit is 256 bytes.

2.3. Final considerations

This section presented the RoboCup Soccer 2D Simulation and RoboCup Search and Rescue Simulation domains in terms of simulated environments, architectures, simulator components and human to agent modelling including agent sensors and actuators.

Both simulators systems are in constant evolution, as new modules are added to the package and the existing ones are updated. Particularly in rescue, the system architecture was designed with this in mind, allowing for independent simulator development and integration. Still, some challenges arise from the hierarchical organization, in which the kernel, being the central module, can become extremely complex

As new features and rules are introduced, teams are required to adapt their code. This leads to a tradeoff between advancing the simulator system and improving team strategies. As teams adapt to new features, the time left for the development and perfecting of high level strategies is significantly reduced. This is a typical problem in which equilibrium between depth and wideness is required, and the situation is actively discussed in the RoboCup mailing lists.

Challenges in both the domains presented range from controlling low level actions (like moving from point A to point B), to the of cause and effect implications (if goalkeeper moves to B to cover a opponent->goal left open->goal scored; if extinguish D -> E burns) to cooperative actions for reaching a goal (pass to number 4 for him to score; unblock road Y for Fire Brigade passage). Table 3 presents a challenge comparison between the RoboCup soccer simulation 2D and the RoboCup search and rescue agent simulation leagues.

Table 3: Comparison between soccer and rescue simulation domains.

Category	Soccer	Rescue
Environment	Dynamic	Dynamic
State Change	Real-Time	Real-Time
Control	Distributed	Distributed
Communication	Strongly limited	Limited
Sensors	Noisy, limited	Limited
Environment Reactivity	Aggressive	Low
Environment Pre-Knowledge	High	Low
Heterogeneity	Low	High
Number of Agents	Fixed	Variable

2.3.1. Soccer

The soccer architecture here described served as the basis for other RoboCup Soccer leagues, including the 3D simulation league and the Physical Visualization league that uses the concept of mixed reality.

One of the main challenges of this domain lies with its reactivity and dynamism. There is always another team trying to counter-act every positive action and thus prediction of opponent moves is very important. Another challenge lies on the partial cooperative/adversarial environment. In this domain it becomes as important to score a goal as to prevent the opposing team of scoring.

2.3.2. Search and Rescue

One of the shortcomings of the modular architecture in Rescue is the fact that, sometimes, specific modules are a lot more developed than others, leading to imbalances. The most current example lies in the fire simulator. This module was greatly improved by a Freiburg university team and some of the original concepts, in regards to communication with the kernel and, through it, to the agents, became outdated.

One example of the simulator package flexibility, and its ability to evolve beyond the predefined architecture, lies also in the fire simulation. Freiburg's fire simulator allows a direct communication to the collapse simulator enabling a new level of interaction and integration between the two modules (Nüssle, et al., 2004). When the earthquake hits the city, and the simulation begins, buildings crumble. These collapses can provide the fire simulator with useful information for realistic creation of ignition points. Furthermore, as fires spread, temperature data can be conveyed to the collapse simulator, as the fire weakens building structure and

facilitates collapses. Still, while the fire simulator allows this information exchange, the collapse simulator does not - it is currently outdated and requires a significant upgrade.

One of the main challenges of this domain lies in the agent's strong heterogeneity. Agents are dependent on each other to accomplish the goals. To be successful in this domain, there must be a good balance between what an agent wants to do and what other agents want him to do.

Chapter 3

3. Towards a Coordination Model

Several authors have proposed general models for flexible teamwork. However, most of the approaches either are not sufficiently reactive to perform efficiently in real time and dynamic domains or do not provide agents with sufficiently developed social behaviour to perform intelligently as a member of a team in continuous, multi-objective and complex multi-agent environments. Some notable exceptions may be recognized, like Stone's and Veloso's work (Stone and Veloso, 1999) that has been applied with success to RoboCup soccer and network routing, Tambe's STEAM (Tambe, 1997) successfully applied in virtual battlefield simulations and Jennings' GRATE* (Jennings, 1995) also applied in dynamic domains.

3.1. Coordination Frameworks

Peter Stone et al have proposed the use of "locker room agreements" (Stone, 2000; Stone and Veloso, 1999) as a mechanism for defining a pre-determined coordinated behaviour that may be used in environments with limited communication (Stone, 2000). The "agreement" defines a flexible teamwork structure including task decomposition and dynamic role assignment (Stone, 2000; Stone and Veloso, 1999) and it was implemented in the simulated robotic soccer team CMUnited (Stone, et al., 1999) that won several RoboCup world championships (Asada and Kitano, 1999; Kitano, 1997; Kitano, et al., 1995; Veloso, et al., 2000). Their team strategy is composed of formations, using simple protocols for switching between them. Each formation assigns each agent a role, but role exchange between the agents is theoretically possible. As it was defined, a role consisted of a specification of an agent's internal and external behaviours. The teamwork structure also includes set-plays, e.g. multi-step, multi-agent plans for execution in some situations. In this approach, roles may either be rigid or they may be somewhat flexible (Stone and Veloso, 1999).

Most implementations of multi-agent coordination frameworks rely on domain specific coordination. However, some relevant exceptions may be identified. ARCHON project (Wittig, 1992), proposes a multi-agent cooperation system, in the domain of electricity transportation management, based on joint-intentions and on a teamwork's general model. Other example is Jennings' (Jennings, 1995) joint responsibility framework, which is based on a joint commitment to the team's joint goal which was implemented in the GRATE* system. GRATE* has a layered architecture in which the behaviour of an agent is guided by its mental attitudes, beliefs, desires, intentions and joint intentions. Agents are composed of two layers: a domain level system and a cooperation and control layer. In GRATE*, teamwork is executed when an organizer agent detects the need for joint action, becoming then the responsible for establishing the team and ensuring all member's commitments (Jennings, 1995).

One other general model for teamwork is STEAM (simply, a Shell for Teamwork) (Tambe, 1997). STEAM is based in the joint intentions theory (Levesque, et al., 1990) but also on the SharedPlan theory (Grosz, 1996; Grosz and Kraus, 1996). STEAM uses joint intentions as the basis for teamwork allowing team members to build a hierarchical structure of joint intentions, individual intentions and beliefs about the teammates intentions (Tambe, 1997). STEAM has been applied in several domains like the attack and transport domains and RoboCup soccer server (Chen, et al., 2002), in the context of the ISIS team. Although being far from a complete model of teamwork, STEAM attempt to bridge the gap from cooperation theory to its implementation is a remarkable one.

3.2. Hierarchical Task Networks

One other approach to automated planning is discussed in hierarchical task network (HTN) where the dependency among actions can be given in the form of networks. In Hierarchical task networks there are primitive, compound and goal tasks. Primitive tasks are simple actions, compound tasks are a composition of simpler tasks or sets of actions and goal tasks that exist in order to satisfy a condition (Lekavý and Návrat, 2007).

Both goal and compound tasks require a sequence of primitive tasks to be performed yet goal is specified not has a set of primitive or compound tasks but rather as a set of conditions that have to be satisfied.

In HTNs there are a number of constraints among tasks that take the form of network. Such task network can be used as a condition for another compound or goal task. As such, it is possible to express that a given task is feasible only if a set of other actions mentioned in the network are done in such a way that the network' specified constraints among them are satisfied.

One of the most used formalisms for representing hierarchical task networks TAEMS (Decker, 1995; Horling, et al., 1999).

It is possible for a task network to specify that a condition has to be verified for a primitive task to be executed. When this network is used as the precondition for a compound or goal task, those compound or goal task require the primitive action to be executed and that the condition must be true for its execution to successfully achieve the compound or goal task (Erol, et al., 2005).

3.3. Domain Specific Coordination

One interesting coordination mechanisms widely used in soccer is the UVA Trilearn team's coordination graphs (Kok, et al., 2003). A coordination graph (CG) represents the coordination requirements of a system. A node in the graph represents an agent, while an edge in the graph defines a dependency between two agents. The topology of the graph is dynamically updated based on the current context, only the inter-connected agents have to coordinate their actions at any particular instance to find the joint optimal action. Via a context-specific decomposition of the problem into smaller sub-problems, CGs thus offers scalable solutions to the problem of multi- agent decision making. In order to apply CGs to the continuous domain, roles are assigned to the agents who then coordinate the different roles. Such an assignment provides a way to parameterize a coordination structure over a continuous domain (Kok, et al., 2003).

FC Portugal's team strategy definition extends the concepts introduced by Stone and is based on a set of player types (roles), and a set of tactics that include several formations for different game situations (defence, attack, etc) (Reis and Lau, 2001). Formations assign each player a positioning (that determines the strategic behaviour) and each positioning a player type (that determines the active behaviour).

When Stone defined a situation, the concept was bound to set-plays. A situation was a set of world state conditions that triggered a series of predefined behaviours within the roles. FC Portugal's members have expanded on this concept and defined situations as a group of easily identifiable logic conditions set for high-level, world state, parameters. These situations were defined so that they would not suffer a considerable, temporal, variation. The situations were then associated with formations, however not every situation had to have its own formation using, in this case, a set of replacement situations.

Situation Based Strategic Positioning (SBSP) mechanism , developed within the FC Portugal team, is used for strategic situations (in which the agent believes that it is not going to enter in

active behaviour soon) (Lau and Reis, 2002; Reis, et al., 2001). For active situations, the agent position on the field is calculated using ball possession and recovery or playoff decision mechanisms. To calculate its strategic positioning, the agent analyzes which is the game situation. Then the agent calculates its base strategic position in the field in that formation, adjusting it according to the ball position and velocity, situation and player type strategic information. This behaviour enables the team to move similarly to a real soccer team, covering the ball while the team remains distributed along the field.

The Dynamic Positioning and Role Exchange (DPRE), and Dynamic Covering, was based on previous work from Peter Stone which suggested the use of flexible agent roles with protocols for switching among them (although not implementing it in practice). The concept was extended and implemented in FC Portugal team enabling players to exchange their positionings and player types in the current formation if the utility of that exchange is positive for the team. Positioning exchange utilities are calculated using the distances from the player's present positions to their strategic positions and the importance of their positionings in the formation on that situation (Reis, 2003; Reis and Lau, 2001).

In the case of communication in single channel, low bandwidth, and unreliable domains the challenge is deciding what and when to communicate. In ADVCOM (Intelligent Communication Mechanism), agents use communication in order to maintain world states updated by sharing individual world states, and to increase team coordination by communicating useful events (e.g. a positioning swap) (Reis and Lau, 2001). The main innovation of this communication strategy is that agents communicate when they believe that the utility of their communication is higher than those of their teammates, using mutual modelling to estimate these utilities.

For the rescue domain most coordination methodologies use the concept of task allocation. One of the most interesting where the tasks arrive dynamically and can change in intensity (Nair, et al., 2001). Two approaches are described, one centralized based on a combinatorial auction mechanism and a distributed approach that relies on the agent choosing for himself which task to perform. In the combinatorial auction, the center agents assume the part of auctioneers. Each center is then responsible for controlling the action of tasks relating to their kind (Police Office for Police Forces' tasks, etc). At the beginning of each cycle, each agent makes a number of bids consisting in a series of tasks with cost estimation. The auctioneer collects all the bids and assigns the winning bids to those tack cover a larger number of tasks with the minimum cost. In the distributed approach, in used with some variations by the majority of teams, each agent reasons about the seriousness of the task before committing to it.

3.4. Final considerations

After searching and reviewing technologies that would fulfil the objectives defined in section 1.2 none was found fitted. Of the studied frameworks although being partially domain independent, they all lack a global strategic coordination view. Some rely on environment conditions to adapt agent's coordination; others are dependent on a coordination agent or on the identification of joint intentions.

The Hierarchical Task Networks was found to be much goal-oriented, although the hierarchy between action, compound tasks and goals is interesting and useful, the implementation of alternative approaches to goals would be arduous due to the relative low level of this approach.

Domain approaches, as the coordination frameworks don't have a global strategy view. They rely on the immediate needs to coordinate agents either trough coordination graphs of system requirements or through the use of tasks.

FC Portugal's members' research would be adequate for the objectives if the domain dependence were not an issue. Although it provides a high-level strategical overview, because of the domain specificity it does not deal with a variable number of agents and strong heterogeneous agents (with different action possibilities). Also the strategical view is always seen and set as a whole, without smaller organizational units (e.g. Defence, Midfield, Attack). As much of FC Portugal's related research was done for soccer simulation leagues, the work here presented either serves as a basis for the construction of the strategical layer or is directly usable in conjunction with this layer for the specific case of RoboCup Soccer. In conclusion, a generic strategy coordination model would have to take advantage of goal and task definition present in HTNs. Layered and hierarchical approaches like GRATE would allow a separation of in-field, local, coordination form a global objective view. Finally strategy, tactic, formation and situation concepts if properly expanded could provide a good basis for a generic coordination and strategy model.

Chapter 4

4. Model for the Strategic Layer

The model here depicted provides a structured method of representing, building and managing a strategy in a scenario where a team of agents is used to perform a given cooperative task. The terms scenario and agent should be considered as broader terms. Scenario can be a simulation, a game, a simplified view of the real world or any other kind of setup where there is an environment, where a team of agents have one or more objectives to fulfil. Likewise agents, besides being software computational entities, can be any kind of independent units like robots or even persons.

This model handles static, dynamic, reactive or nonreactive environments and is designed to manage team strategy and cooperation. A team is an aggregation of agents with common goals. When agents in a team work together cooperatively they may be viewed as team members performing teamwork (Cohen and Levesque, 1991; Tambe, 1997).

In this model, homogeneous and heterogeneous agents can be used. In heterogeneous environments the term “agent type” is used for the differentiation of agents with different capabilities.

4.1. Structure

In order to better explain the proposed model, a top-down approach will be followed. Figure 20 represents the proposed model and depicts the interconnections between the concepts presented in this model. The figure only expands one branch for each concept. A strategy is thus, in the proposed model, composed of tactics, formations, sub-tactics and roles. Triggers, binders, distributions and amounts are also used to configure the strategy.

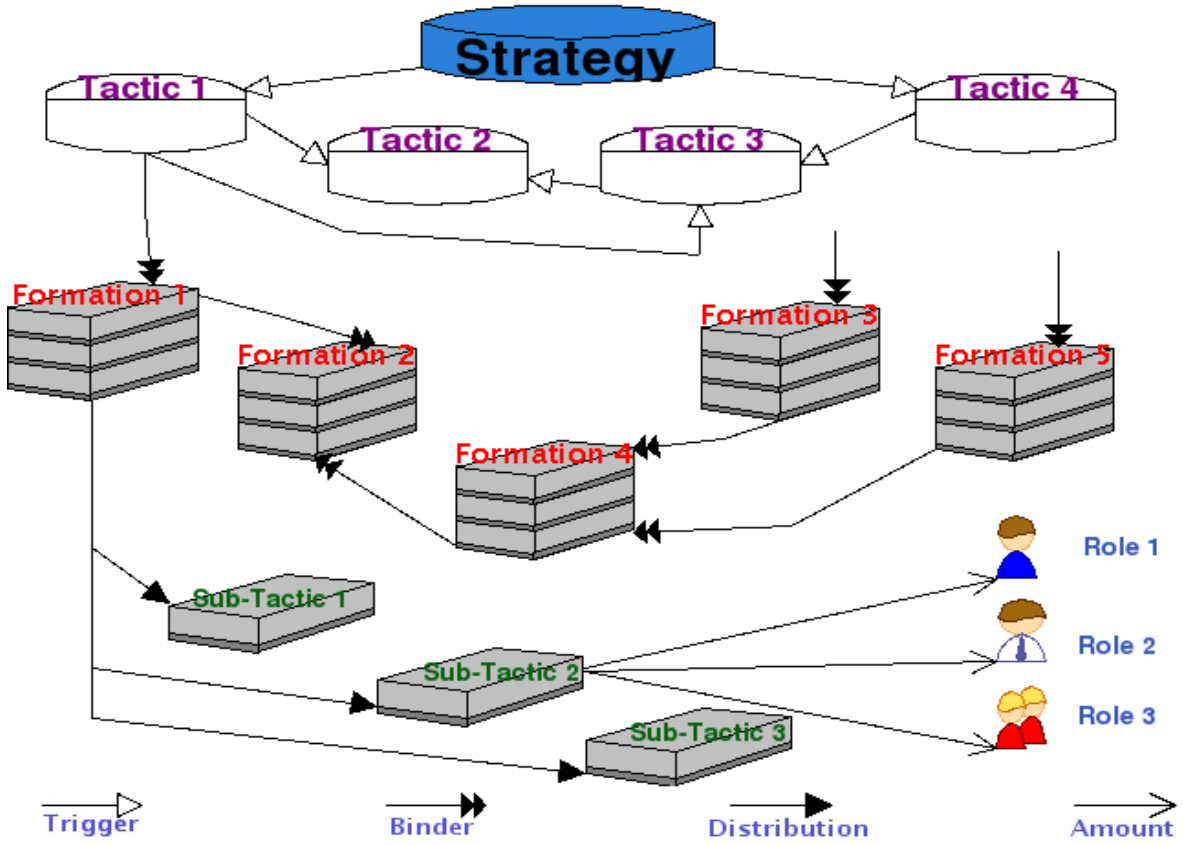

Figure 20: Schematic of strategic concepts

4.1.1. Strategy

Informally, a strategy is the combination and employment of means in large-scale, long-range planning and the act of directing operations for obtaining a specific goal or result.

Formally, a *strategy* is a combination of *tactics* used to face the scenario and the *triggers* to change between tactics:

$$\text{Strategy} = \{\text{Tactics}, \text{Triggers}\} \tag{5}$$

A *strategy* can have several available *tactics*:

$$\text{Tactics} = \{\text{Tactic 1}, \text{Tactic 2}, \ldots, \text{Tactic t}\}, \forall\, t \in N \tag{6}$$

Triggers set the conditions to interchange tactics:

$$\text{Triggers} = \{\text{Trigger 1}, \text{Trigger 2}, \ldots, \text{Trigger tg}\}, \forall\, tg \in N \tag{7}$$

4.1.2. Tactic

A tactic is an approach to face the scenario in order to achieve a goal. Tactics deal with the identification of different situations and the correspondent use and deployment of agents in the scenario for those situations.

Formally, a *tactic* defines *agents*' *formations* as the arrangement of *agents, situations* as the combination of scenario conditions that can be seen as a more particular problem. Also defined in *tactics* is a *binder* as the association between a *formation* and a *situation* or between several *situations* and a *formation*. *Tactics* can optionally also set *tactical parameters,* the default thresholds on which *agents* base their decisions.

A *tactic* should be self-sufficient, i.e., it does not need other tactics to function through all the simulation. There can be only one *tactic* active at one given time.

$$\text{Tactic} = \{\text{Formations, Situations, Binders, [Tactical Parameters]}\} \quad (8)$$

A *tactic* has several formations that can be used:

$$\text{Formations} = \{\text{Formation1, Formation2, ... , Formation f }\},\ \forall\, f \in \mathbb{N} \quad (9)$$

A *tactic* also defines different, useful, *situations*:

$$\text{Situations} = \{\text{Situation 1, Situation 2, ... , Situation s}\},\ \forall\, s \in \mathbb{N} \quad (10)$$

Tactics have *binders* in order to associate *formations* with *situations*:

$$\text{Binders} = \{\text{Binder 1, Binder 2, ... , Binder b}\},\ \forall\, b \in \mathbb{N} \quad (11)$$

Tactics can optionally have tactical parameters:

$$\text{Tactical Parameters} = \{\text{Tactical Parameter1,.., Tactical Parameter tp}\},\ \forall tp \in \mathbb{N} \quad (12)$$

In a *situation*, the conditions that make it unique are defined:

$$\text{Situation} = \{\text{Condition 1, Condition 2, ... , Condition cd }\},\ \forall\, cd \in \mathbb{N} \quad (13)$$

A *binder* sets the *situations* that lead to a *formation*. Optionally, a *binder* can set the connection between several origin *formations* and a terminus *formation* through *situations*:

$$\text{Binder} = \{[\text{Origin Formations}], \text{Situations, Terminus Formation}\}\ , \quad (14)$$
$$[\text{Origin Formations}]\ ,\ \text{Terminus Formation} \in \text{Formations}$$

4.1.3. Formation

A formation is a high-level structure that aggregates all the agents with the intent of assigning them to specific sub-tactics. The aggregation is either wrought by using agents that belong to the same type, have the same more immediate goals, or both.

Formally, a *formation* is a specific association of *sub-tactics* with a defined *distribution* that may specify an *agent type*. Only one *formation* can be active at any given time. As such, the *formation* must include *sub-tactics* for all *agents*.

$$\text{Formation} = \{ \text{Distribution, Sub-Tactics, [Agent Types]} \} \quad (15)$$

The same *sub-tactic* can be used more than once in a *formation*. This allows an implicit definition of *Group*. Let *sub-tactics* be a multiset (Stanley, 1997) where m(Sub-Tactic st) defines the multiplicity of a *sub-tactic*:

$$\text{Sub-Tactics} = \{(\text{Sub-Tactic1, m(Sub-Tactic1)}), (\text{Sub-Tactic2, m(Sub-Tactic2)}), \ldots, (\text{Sub-Tactic st, m(Sub-Tactic st)})\}, \forall\, \text{st} \in \mathbb{N} \quad (16)$$

For each element in *sub-tactics* there is correspondent value in a *distribution*:

$$\text{Distribution} = \{\text{Value1, Value2, } \ldots \text{, Value v}\}, \text{v} = \sum m(\textit{Sub-Tactics st}) \quad (17)$$

A distribution specifies either absolute or percentage distribution values for each sub-tactic in the formation. Distribution *values* always refer to *agent types* when applicable. In this manner, the total of *values* can surpass 100%, but not for a specific *agent type.*

The association with *agent type* is implicit when a *sub-tactic* can only be applied to one *agent type*. Otherwise, when more than one *agent type* can be used (see section 3.1.5), an *agent type* must be specified for that *sub-tactic*:

$$\text{[Agent Types]} = \{\text{Type1, Type2, ..., Type ty}\}, \forall\, \text{ty} \in \mathbb{N} \quad (18)$$

4.1.4. Sub-Tactic

A sub-tactic reflects the approach to face the scenario of a limited set of agents either partially for a number of situations or during the whole scenario.

Formally, a *sub-tactic* is an association of *roles* with one default *amount* of *agents* assigned to those *roles*. Additionally a *sub-tactic* may also have *sub-tactical parameters* to reflect specific

thresholds, *agent* parameters, coordination options or other values that are needed to configure the *roles* used on the *sub-tactic*.

$$\text{Sub-Tactic} = \{\text{Amounts, Roles, [Sub-Tactical Parameters]}\} \quad (19)$$

A *sub-tactic* can have one or more *roles*:

$$\text{Roles} = \{\text{Role 1, Role 2, } \ldots \text{, Role r }\},\ \forall\ r \in N \quad (20)$$

For each *role* in *sub-tactic* there is an *amount* in *amounts*:

$$\text{Amounts} = \{\text{ Amount 1, Amount 2, } \ldots \text{, Amount a}\}\ ,\ a = \sum role\ r \quad (21)$$

Sub-Tactics can optionally have tactical parameters:

$$\text{Sub-Tactical Parameters} = \{\text{Sub-Tactical Parameter1,.., Sub-Tactical Parameter stp}\}\ ,\ \forall\ stp \in N \quad (22)$$

Like in a distribution, an *amount* specifies either absolute or percentage values for each *role* in the sub-*tactic*. Percentage *amounts* in a given *sub-tactic* must total 100%.

Sub-tactics can be divided into Typed Sub-Tactics and Generic Sub-Tactics. In a typed sub-tactic at least one of the roles is associated with an agent type, which becomes the sub-tactic's type.

In order to ease the handling of different *agent types*, it is not possible to use *roles* of different *agent types* in the same *sub-tactic.* As such, *typed sub-tactic* can only use *roles* for one *agent type* together with *generic roles*. As a consequence, to build a *formation* with different *agent types,* there should be at least one *sub-tactic* for each *agent type*.

A *generic sub-tactic* is a particular kind of *sub-tactic* without any association with an *agent type*. Thus, in a generic *sub-tactic,* only *generic roles* can be used. As it was previously stated, if a *generic sub-tactic* is used in a *formation* that contains *sub-tactics* for more than one *agent type*, an *agent type* must be specified. This type is specified together with a *distribution value* when *agents* are assigned to a *generic sub-tactic*.

In the event that there are no *agent types,* or there is only one type of agent in the *tactic*, all *sub-tactic* kinds are generic and can be refereed simply as *sub-tactic*.

4.1.5. Role

A role is a normal or customary activity of an agent in a particular environment.

Formally, a *role* is a set of *algorithms* in a defined sequence that describes an *agent*'s behaviour. Behaviour is the aggregation of the responses or movements made by an agent in any situation as the manner of acting or controlling himself. This behaviour description is expected to include, when relevant, the specification on how the *agent* should coordinate with *agents* in the same *role* or in other *roles*.

The *agent* coordination can be of three different kinds as all *agents* with the same *role* form one *group, all agents* with the same *role* form several smaller *groups* (with a rule specified inside the *role*)or all *agents* with the same *role* act individually.

The *role* also defines partial objectives accordingly to the coordination method used. Although *roles* can describe the behaviour for an entire scenario, they can also describe the behaviour for only a given time frame or *situation*. *Teams* construct their *roles* by combining different motion and action mechanisms with partial objectives.

The *role* level is the lowest in the proposed model. For teams who use sequenced task/objective/state based *agents*, a conversion to *role* based *agent* is discussed in section 7.2.

Similarly to the *sub-tactics*, *roles* can be divided into *Typed Role* or *Generic Role*. A *typed role* is a particular kind of *role* that can only be assumed by one *agent type*. Using heterogeneous *agents* does not necessarily means that *typed roles* or *agent types* will be used in the *strategy*. *Typed roles* are used when, in heterogeneous *agents*, there is a need to use the different *agent*'s properties or capabilities.

A *generic role* is a kind of *role* that can be assumed by any of the *agent types* used in a *tactic*. Analogously to a *generic sub-tactic*, in the event that there are no *agent types,* or there is only one type of *agent* in the *tactic*, all *role* kinds are generic.

4.2. Decision, Supervising and Communication

The decision making depends on the agents' organization and types set by the scenario. In teams where there is only a supervisor and all the agents are “dummy”, the strategical layer will obviously only be applied to the supervisor.

In multi-agent systems, the first rule is that all *agents* have full knowledge of the strategical layer being used. Then if all *agents* have a good, shared, world state knowledge using the layer

can be done with no extra communication. This is accomplished because all the *agents* switch their *tactics*, *situations* and *formations* based on the same conditions and at almost the same time. When a team uses a mechanisms like ADVCOM (section 3.3), the no-communication version of the strategical layer can be applied to scenarios where the normal communications are limited and unreliable.

If *agents* have more limited computational resources but still have good world state knowledge synchronization, the layer can be computed only by a supervising *agent*. This supervising *agent* would only have to communicate a new *formation* whenever declared by the strategical layer.

The supervising *agent* is chosen taking into account the *agent* who normally has more computational resources. Some scenarios specifically have supervising *agents*. In environments where the world state sharing is unreliable, the layer must be computed by a supervising *agent* choosing typically, the best informed *agent*.

4.3. Agent Assignment

The strategical layer defines both absolute and percentage forms for *distribution values* and *role amounts.* This possibility is given so that strategies can be built independently from the *agent* number used in the scenario.

Another possibility of the model is to use both absolute and percentage forms simultaneous. In this model, for both *distribution values* and *role amounts*, absolute forms for values take priority over percentage value forms. This means that *agents* are assigned first to *roles* in a *sub-tactic* specified with absolute *distribution values* and with absolute *role amounts* in the referred *role.* Next *agents* are assigned to *sub-tactics* with only absolute forms of *distribution values.* The succeeding priority is assigning *agents* to *roles* specified by absolute *role amounts*, in a sub-tactic with a percentage *distribution values.*

Finally, for the remainder *agents* that use percentage forms in the mixed method, or when the percentage form is the only assignment method used, the assignment priorities follow. When converting to absolute numbers, the values are truncated and assigned. If there are any *agents* left, one *agent* is assigned to each *sub-tactics* and *roles* that did not received any *agents* in the decreasing order of their respective percentages. If there are any available *agents* left they are assigned sequentially to the *sub-tactics* and *roles* with the highest remainder values.

If *agent types* are in use, the previously defined assignment method is applied separately to each *agent type.* As it is easily concluded the mixed method allows the definition of priority *roles* in environment where the total *agent* number is unknown.

The *agent* assignment methods defined what *roles* needed to be used, particularly for environments where the total *agent* number is unknown. Next, the assignment of a specific *agent* to a specific *role* is discussed.

In order to assign *roles*, each *agent* must be capable of differentiate himself from others. Generally, this differentiation consists of attributing a different number or an id to each *agent*. There are a number of methods used to get an unique id namely it can be hard coded, attributed by a simulator or a referee, or even defined based on a relative position rule.

In its simpler form, the *role* assignment can be done by sequentially assigning one id to a *role*. Optimal *role* assignment depends on scenario conditions like proximity to objectives, relative *agents*' positions, etc… Based on this fact, the model does not specify a method. In fact, a method like DPRE (section 3.3) that uses dynamic *role* exchanges is strongly advisable. To be noted that the strategical layer is still compatible with dynamic, situation based positioning like SBSP (section 3.3). This is accomplished because the positioning systems are specified inside the *role*.

4.4. Final Considerations

The new concepts of sub-tactics simplify the management of heterogeneous agents taking advantage of the different capabilities. The concept of generic-role maintains the capability with homogeneous teams and allows an efficient use of common capabilities in heterogeneous teams. Binders allow the use of formations in a delimited timeframe or situation and at the same time simplify the use of substitute formations. Absolute and Percentage value forms allow a dynamic, real time, adaptation of the layer to changes in the number of available agents while assuring the enforcement of priority roles

Chapter 5

5. Graphical Tool for Building Strategies

Having specified and formalized a strategical coordination model that would fit the enunciated objectives, the next step was to implement it. The customary way for describing the available strategies, formations, etc, was based on the use of textual strategical files containing all the information about the strategy and its parameters. A more natural way would be to design and specify a strategy in a way more close to the human manner of thinking the strategy – as interconnecting, visual, objects. Thus, a graphical tool was projected and implemented in order to enable high-level strategy definition based on the graphical definition and connection of visual objects.

5.1. Requirements

The requirements for a visual tool that would allow the graphical construction of strategies would be visual object management, object interconnection, object editing, copy-paste functionalities and format export features to allow integration with agent's code. Instead of building such a tool from scratch it seemed more time effective to use an existing flowcharting or diagram application as the basis of the graphical tool. In order to maximize the model integration with the tool and facilitate the broad use of the tool it was decided to consider only open-source applications. A preference for Linux based applications was also given as the agents for both the domains in this thesis use the Linux operative system.

5.2. Base Application

The considered available tools that satisfied the requirements were Dia (Breit, et al., 2000), Kivio (Lamb, 2003), OpenOffice.org Draw (Belzunce, et al., 2006), Umbrello UML Modeller (Hensgen and Authors, 2003) and Xfig (Sato and Smith, 2002). Although allowing a

diagramming construction, Xfig was more oriented for vector drawing and as such had too many features not related to the requirements. OpenOffice.org Draw, like Xfig, was also more of a general purpose vector graphics editor and was disregarded despite having a wide user spread and a versatile "connectors" between shapes for diagramming. The Umbrello UML Modeller featured a good shape interconnection system and a clean and consistent user interface however creating new shapes and objects and describing their relations proved difficult. For the remaining tools, Dia and Kivio, both had a decent user interface, easy addition of new objects and the ability to export the diagrams to xml. The choice leaned on Kivio since it is less developed than Dia, which in this case is a good feature as it would be simpler to understand and customize a simpler tool. Also Kivio features a plug-in system and a multi-sheet feature that would prove useful in the layer implementation (Lamb, 2003). Kivio is also the flowcharting and diagramming application for the KOffice[7] application suite (Team, 2008) which makes its availability high and already has a relatively high support community.

5.3. Features and Usage

After choosing Kivio for the graphical tool's GUI, a customized, installable, stencil set with the layer objects was built as seen in Figure 21.

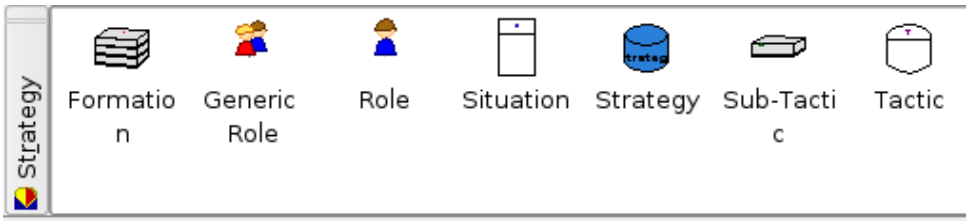

Figure 21: Strategy stencil set.

Installation of the stencil set is fairly simple and is accomplish by using the in-built Kivio's plug-in system. Usage is intuitive as by using graphical representations of the strategic layer's components it is possible to interconnect them. For example one drags a situation from the toolbar, uses the situation object zone of title to assign a situation number and the conditions zone to define the conditions that make that situation unique. The triggers, binders, distributions and amounts are set trough the use of the connector feature. The values for these objects are

---

7 KOffice is an office suite for the K Desktop Environment released under free software/open source licenses. Available at http://koffice.org/.

simply set by double-clicking the connector. Strategical objects and interconnections can also be easily selected and copy-pasted into other distinct strategies.

In a simple strategy it is enough to use one sheet however, for more complex strategies a multi-sheet is recommended separating *strategy*, each *tactic*, *formations*, *situations* and *sub-tactics* (Figure 22).

Figure 22: Multi-sheet feature.

In fact, when defining several *tactics* which use at least one *binder* with no precedence (*origin formation)* a separate sheet for each tactic is mandatory. Figure 23 illustrates the use of formations with no precedences (Formation 3 and Formation 5). In this tactic if the binder condition of Formation 3 becomes true, no matter what the current formation is, it automatically should change to Formation 3.

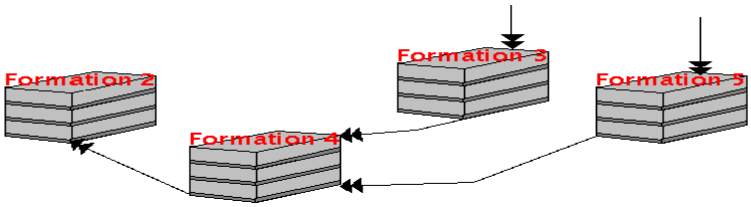

Figure 23: Formation bindings.

The tool exports the edited strategy to a XML (Extensible Markup Language) (W3C, 2006) file which can be used to implement the layer in *agents*.

```
        <KivioLineStyle width="1" capStyle="0" pattern="1" joinStyle="128" color="#000000"/>
      + <KivioTextStyle vTextAlign="64" isHtml="0" hTextAlign="4" text="T 2"></KivioTextStyle>
        <KivioFillStyle gradientColor="#ffffff" colorStyle="1" gradientType="0" color="#ffffff"/>
      </KivioShapeData>
    </KivioShape>
  - <KivioShape shapeType="12" name="TacticDescription">
    - <KivioShapeData>
        <Position x="2" y="40"/>
        <Dimension w="78" h="40"/>
        <KivioLineStyle width="1" capStyle="0" pattern="1" joinStyle="128" color="#000000"/>
      + <KivioTextStyle vTextAlign="16" isHtml="0" hTextAlign="1" text="focus on fire"></KivioTextStyle>
        <KivioFillStyle gradientColor="#ffffff" colorStyle="1" gradientType="0" color="#ffffff"/>
      </KivioShapeData>
    </KivioShape>
  </KivioSMLStencil>
- <KivioPluginStencil setId="Kivio - Internal - Do Not Touch" id="Dave Marotti - Straight Connector">
  - <KivioStencilProperties>
      <Kivio1DProperties connectorWidth="36" needsWidth="0"/>
      <KivioLineStyle width="1" capStyle="0" pattern="1" joinStyle="128" color="#000000"/>
      <KivioFillStyle gradientColor="#ffffff" colorStyle="1" gradientType="0" color="#ffffff"/>
    + <KivioTextStyle vTextAlign="64" isHtml="0" hTextAlign="4" text="explored buildings > 80% && rescued civilians > 60%"></KivioTextStyle>
    - <KivioConnectorList>
        <KivioConnectorPoint x="347.244" y="105.591" connectable="1" targetId="23"/>
        <KivioConnectorPoint x="456.379" y="105.88" connectable="1" targetId="25"/>
        <KivioConnectorPoint x="401.764" y="123.735" connectable="0"/>
        <KivioConnectorPoint x="401.859" y="87.7356" connectable="0"/>
        <KivioConnectorPoint x="337.323" y="115.73" connectable="0"/>
      </KivioConnectorList>
```

Figure 24: Partial xml of a Strategy.

As seen on Figure 24 the XML file contains both graphical and strategical information (marked). As the strategical organization for a domain is thought in a hierarchical, relational form the combination of graphical and strategical information allows the preservation of the relations between strategical concepts. Additionally, the reuse of parts of a strategy is also facilitated by storing the graphical information.

The construction tool also features a simple C++ code generator, still in its early stages. The code generator parses the compressed XML and generates the code necessary to execute the constructed layer as well as a strategical file which includes configuration parameters such as tactical parameters and parameter for the quick disabling of tactic present in the strategy.

5.4. Final Considerations

While very useful in the design of strategies, the developed tool is not without limitations. For instance, when using multi-sheets (the usual) situations are defined in a separate sheet. This means that when using the situations in the tactics sheet the identification is only done by the situation number (S1, S2, etc...). If the user is not familiar with the correspondence between situation numbers and the actual situations (and conditions that defines them) the legibility of the tactic and its formation's commutations are diminished.

The graphical tool should also feature a validator capability. As the strategies are designed by humans, they are prone to mistakes. If, for example, the user makes a mistake by surpassing the percentages in agent's assignment the results would be unforeseeable. A domain plug-in for the validator would also assure that for instance in previously shown Figure 23 the binders for formations 3 and 5 would be mutually exclusive.

The code generator is not yet usable in practical applications. The multi-sheet feature, the semantics on defining conditions and parameter as well as implicit assignments in heterogeneous agents make the generator development a time consuming project. Due to these facts, unfortunately, it was not possible to finish the generator in the context of this thesis.

One of the most powerful features of using the graphical tool is the reuse capability. For example, formations and situations, designed for one tactic, can be simply used in another; sub-tactics used in one strategy can be copy-pasted into another. Furthermore, when the domain is similar, like in simulated soccer and robotic soccer, whole strategies can be reused with minor adaptations.

Chapter 6

6. Coordination Model on Soccer

Using the proposed coordination layer on the soccer domain results in a relatively, simple instantiation. This fact comes from the fixed number of agents, limited heterogeneity (except for the goalkeeper all other players can do the same actions although with distinct capabilities).

As the coordination layer was based on previous FC Portugal's research, the adaptation of the graphical tool to operate soccer strategies was straightforward as demonstrated on the next section.

6.1. A Soccer Strategy

Using soccer as an example, for a simple strategy, the same sheet is used to represent the entire layer as seen in Figure 25.

In this example the soccer team has a different offensive or defensive *tactic* depending on the opposing team. Aggressive or defensive formations are used depending on the current score.

As shown, a trigger can originate in a *strategy* thus defining the initial *tactic*. Likewise a *binder* can originate in a tactic thus defining the initial *formation*. Note the absolute values assignment mode and the implicit value assignment to certain roles.

The novel use of *sub-tactics* allows the partial strategic management of agents, for example the defence. Creating new soccer sub-tactics based on existing is straightforward. In this example, if Sbt2 is copied and the *sub-tactical parameter* that defines hard tackling is changed to normal tackling, an entire new sub-tactic is created and ready to use on other formations.

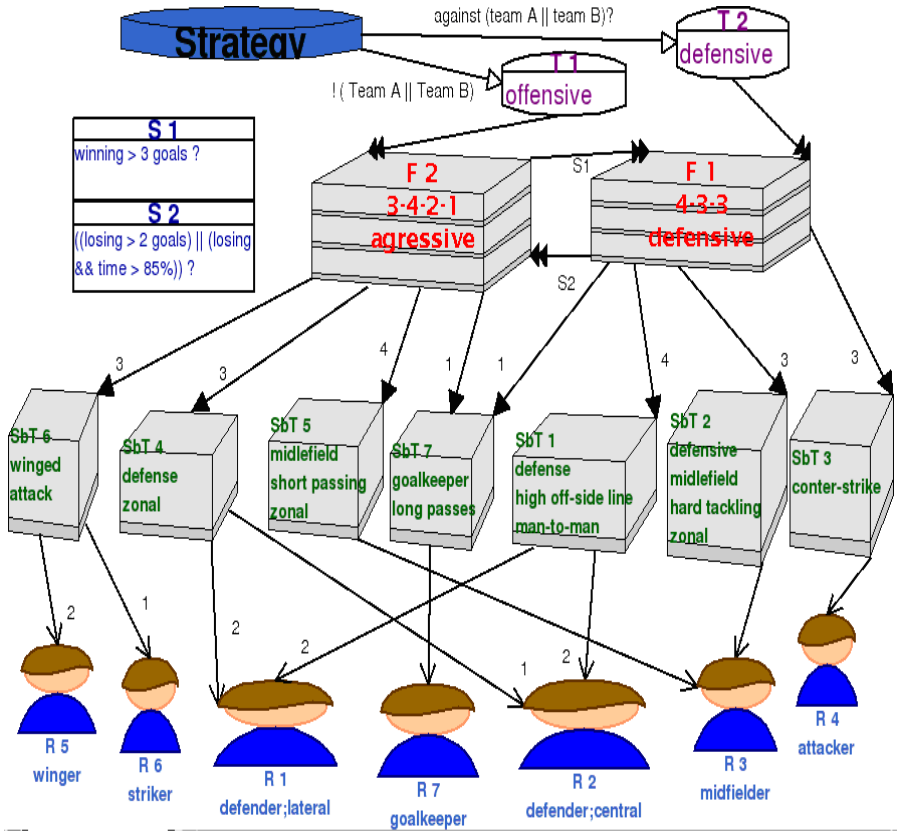

Figure 25: A soccer strategy.

6.2. Soccer Visual Debugger

During the preparation of FC Portugal's participation in RoboCup 2007- Atlanta a need for a debugging tool arouse. As previously demonstrated in chapter 2.1, the soccer domain is very reactive to agent's actions due to the opposing team's counteraction efforts. As such, the effects of switching tactics were hard to perceive and understand in this context, and although not initially planned in the thesis, a visual debugging tool was developed.

The main debugging tool of FC Portugal is called Visual Debugger (Figure 26). Its implementation is based on CMUnited99 layered disclosure tool (Stone, 2000; Stone, et al., 2000) and the soccerserver's logplayer application. CMUnited layered disclosure tool included the possibility of synchronous visualization of the game (using soccermonitor) and of one of the players reasoning (at several levels of abstraction) saved in action log files. The two applications (logplayer with layered disclosure and soccermonitor) have been integrated in a powerful team debugging tool and added the visual debugging capabilities and real and believed world-states superposition (Lau, et al., 2007b).

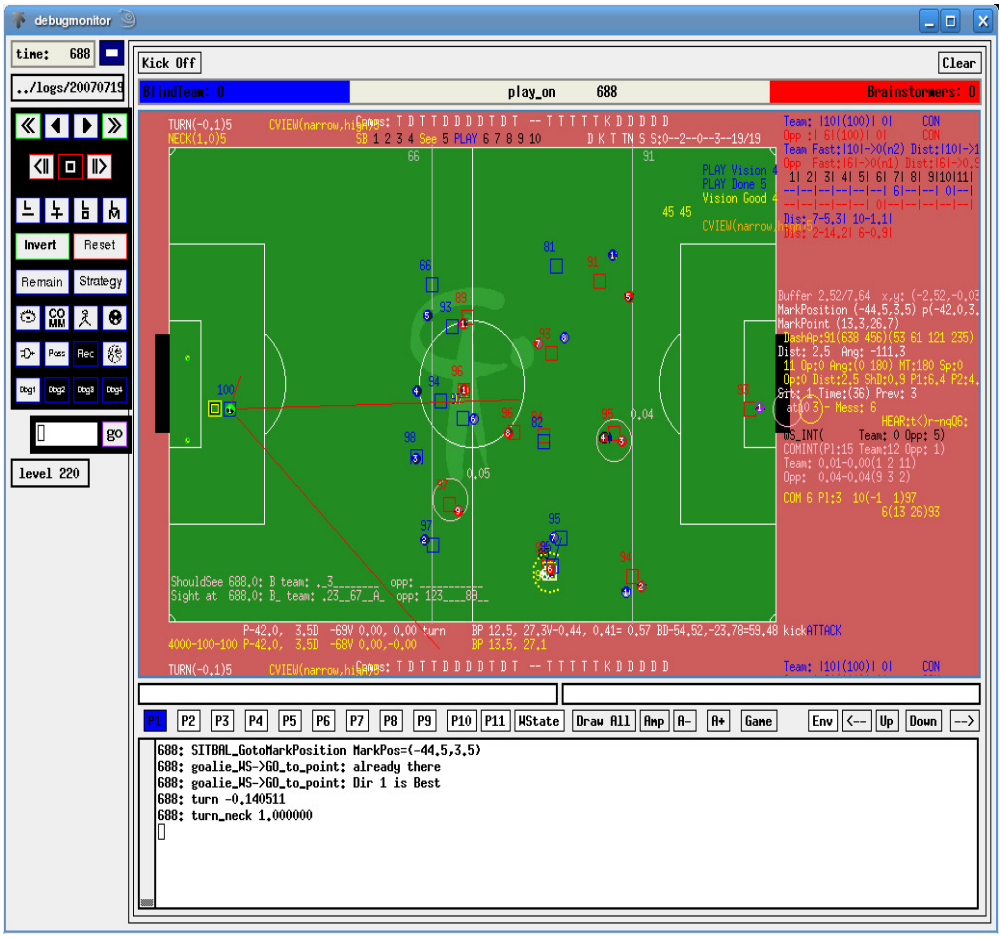

Figure 26: The Visual Debugger.

Visual information is much more easily handled than text information. Soccermonitor (Chen, et al., 2002) includes the possibility of drawing points, lines and circles over the field, but this functionality is not completely being used by other teams. The possibility of drawing over the field provided by soccermonitor was exploited, modified and extended (possibility of drawing text over the field).

Soccermonitor capabilities were limited relating the analysis of a particular situation so an amplification mode was developed (Figure 27). In this mode, the area around the ball is magnified with a desired zoom level and offset to the ball. When in amplification mode the ball is drawn with a vector indicating its direction and velocity. Additionally all players are drawn with orientation lines (showing were the player is facing), a moving vector (showing the direction and velocity of the player), a view zone (that indicates where the player is looking and what he can see) and with a kickable area circle adapted to each player's physical characteristics. Whenever a player is able to kick the ball he is automatically highlighted.

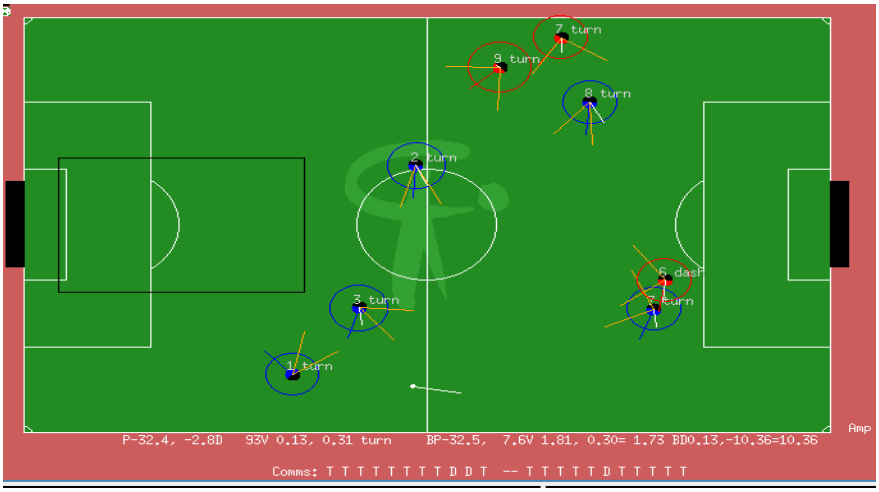

Figure 27: Amplification mode of the Visual Debugger.

On the other hand, on the team side, the log action files syntax was extended so that they could include draw commands (Stone, et al., 1999). Draw commands can be inserted at different reasoning levels of abstraction. A programming interface allows the player to save visual logs by combining simple primitives as points, lines, circles, crosses, squares and text over the field.

Pure textual action debug messages are shown in scrollable debug text windows. There is one of these windows for each of the players and one window with the real situation without errors derived from the simulator log file. Finally two extra textual windows with only log file information were created: an event board and a world state board. The event board displays the initial team selection of heterogeneous agents, and time stamped major environment events like goals, free kicks off-sides and substitutions. The world state board keeps track of the ball and all players' information (stamina, recovery rate, neck angle…).

The debugger determines the real world state based on the visualization information saved in server record log files. The real positions and velocities of all objects are calculated and last player commands determined either by direct inspection of log (version 3) or are deduced from a set of simple rules that reason on world state changes (version 2 logs). This internal view of the real world state is shown in a text window. Some features (ball position/velocity, selected player position/velocity and ball distance) are shown directly over the field.

The possibility of drawing text over the field was already implemented in FC Portugal's code and used quite extensively to show player action mode, synchronization, basic action, lost cycles, evaluation metrics, position confidences and to compare players beliefs on its (and ball) position/velocities with real values. Drawing circles and lines over the field was used to show player beliefs on object positions, player view area, best pass, shoot or forward, communication events, evaluation parameters of each type of action, etc (Figure 28).

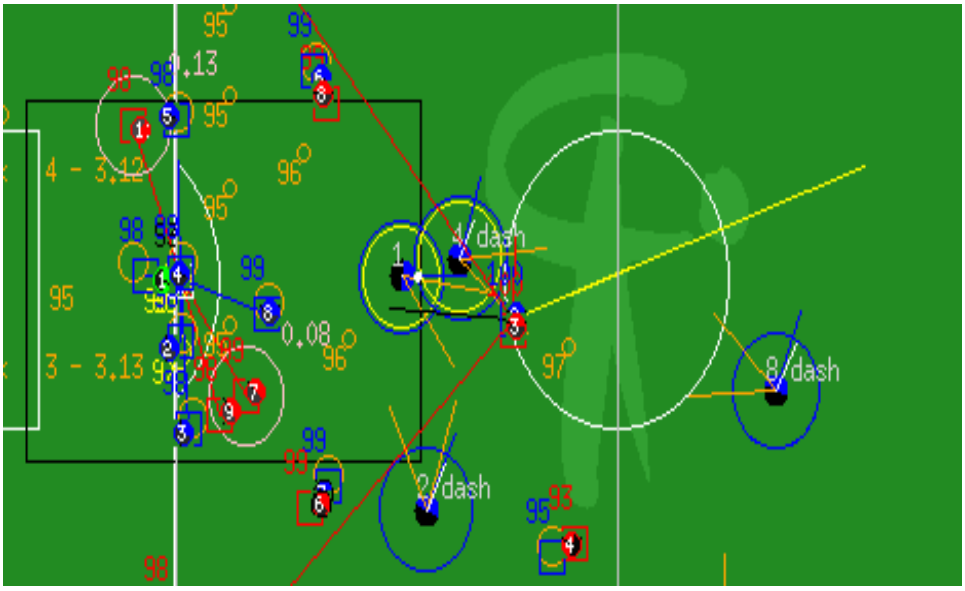

Figure 28: Debugging for Player 9.

Using Figure 28 as example, player 9's point of view is shown and one can see both normal views and amplified views overlapped. This allows the user to directly follow the immediate action (amplified on the ball) and at the same time accompany the global player's view. For instance small colored balls are other players' positions and the near squares represent the position where player 9 thinks they are with a certain degree of confidence (attached number). At the same time with amplification, two large yellow circles are shown, highlighting players that can kick the ball.

This tool can be seen as a 4 dimensional debugging tool where the developer can rapidly position himself and receive the required visual and textual information:

- For each cycle time - dimension 1 -;
- In a given field space - dimension 2 -;
- With a given focus area (eg. Communication) -dimension 3 -;
- With a specified information depth – dimension 4 -.

The tool has been used to tune strategic positions, test new behaviours, tune the importance and precision of each of the evaluation metrics used in the high-level decision module, test world update, communication and looking strategies empirically, analyze previous games of other teams, etc.

6.3. Final Considerations

The layer's introduction of small organizational units into the soccer domain allowed the independent planning of sub-tactics for defence, midfield and attack. New formations can now be rapidly formed from those available sub-tactics. If in use, the percentage values form in assigning roles to a sub-tactics, allows the usage of the same sub-tactic with a different number of agents assigned in different formations. Once more, when domains have similar nature like in soccer simulation and soccer robotic leagues, the strategies defined in one, can easily be adapted to the others. This is achieved by only modifying the roles in the existing sub-tactics.

The visual debugger tool has been used to tune strategic positions, test new behaviours, tune the importance and precision of each of the evaluation metrics used in the high-level decision module, test world update, communication and looking strategies empirically, analyze previous games of other teams, etc. The developed debugger proved to be quite valuable in identifying situations, making tactical changes and predicting their effects.

Chapter 7

7. Coordination Model on Search and Rescue

In order to adopt the strategic layer, our rescue team needed to use the role concept. The previous code was based on a sequential selection of algorithms dependant on world state conditions (Certo and Cordeiro, 2005; Certo, et al., 2007a). To reach the role level the following classifications were used:

- Action: a simple deed performed by an agent. E.g.: Action: Refill; Description: filling a Fire Brigade tank in a refuge.
- Task: set of actions performed by an agent that leads to a goal. E.g.: Task: Rescue civilian; Actions: Move to civilian; Unbury Civilian; Load Civilian; Move to refuge; Unload Civilian.
- Algorithm: set of tasks performed by one or more agents used to solve a particular field problem in a specific manner. E.g.: Algorithm: Clear main roads by prioritizing the main roads; Tasks: Each chosen road or set of roads is assigned to a specific Police Force, and then Police force agents clear the roads.

After identifying the algorithms, they were associated into *roles*. If there were two relevant algorithms with the same function but with different manners of solving the problem, they would be associated with two different *roles*. Some partial, *generic roles* like finding civilians were also created. Although these *roles* only included algorithms related to search and dislocation, and do not have algorithms to act after all civilians are found, they are extremely useful.

7.1. A Search and Rescue Strategy

The following figures depict a simplified rescue strategy. Some additional knowledge of the rescue simulation league is advisable to fully perceive the strategy. In Figure 29 the strategy is

only expanded in one tactic and one formation. As shown, there is a different initial tactic depending on city size, T1 for small cities as T3 for large.

For a large city (T3) the losses will be unavoidable so a tactic that marks city zones as lost from the start would be more effective.

For a small city (T1) an option to focus on human life was made so, at start, agents will be more focused on finding and rescuing civilians. When more than 60% of known civilians are rescued and 80% of the buildings are explored, the tactic changes to T2 giving priority to fire fighting.

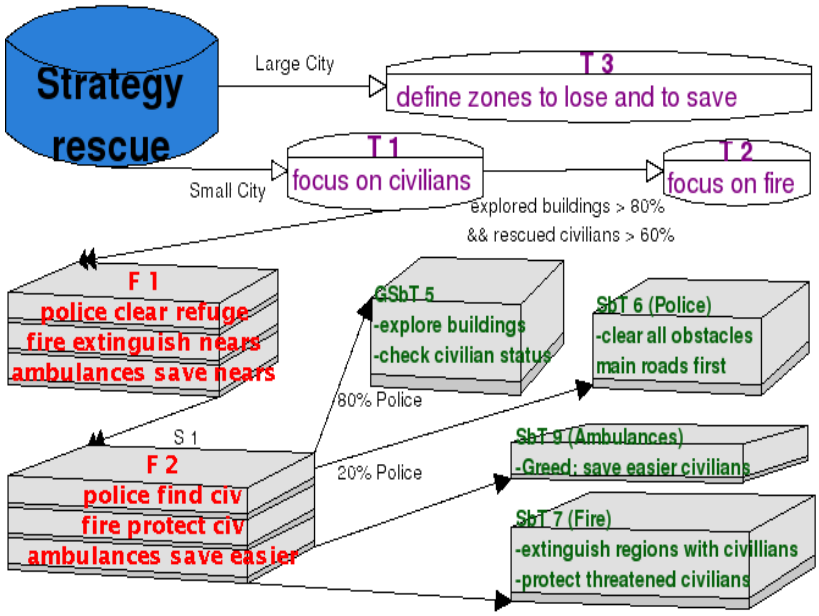

Figure 29: Partial rescue strategy.

Tactic 1 has two *formations*: the initial F1 and F2. F1 is used to ensure that rescue agents are saved as soon as possible and that the refuges are reachable. Note that refuges are essential buildings as Fire Brigades use them to refill their tanks and Ambulances Teams to unload civilians.

Formation 2 is used to find, protect and rescue civilians. Instead of focusing on unblocking roads, Police Forces explore buildings trying to find civilians. Likewise, Fire Brigades opt for extinguish buildings near trapped civilians instead of minimizing fire spread. In Figure 30 the *situation* (S1) to switch from *formation* F1 to *formation* F2 is defined.

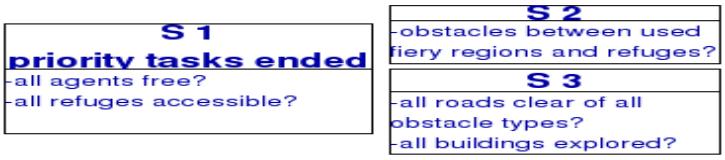

Figure 30: Some rescue situations.

In Figure 31 the *sub-tactic* SbT 7 is expanded. In this *sub-tactic* 80% of the Fire Brigades assume the *role* of protecting civilians that are directly threatened by fire. The remaining Fire Brigades chose to put out fires in city regions with civilians.

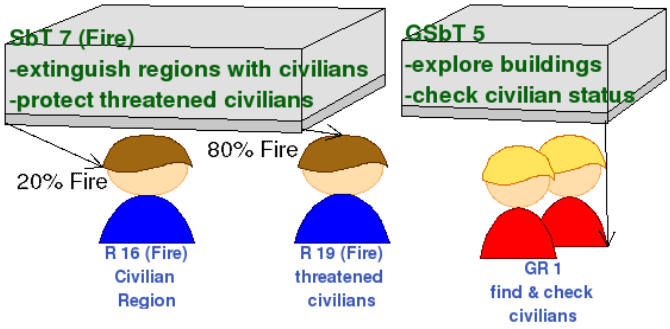

Figure 31: Two rescue sub-tactics.

The *generic sub-tactic* GSbT 5 has 80% of the Police Force assigned to it (Figure 29). In Figure 31 is seen that all of those agents are assigned to the *generic-role* Gr1 used for exploring the city in search of civilians and to check-up on their status. As Gr1 is a *generic role* it could also be assumed by Ambulance Teams or Fire Brigades yet, Police Forces can unblock roads in their path thus reaching any building which was found to be more useful in this case.

7.2. Modelling Behaviours into Strategy

In order to validate the generic coordination layer, two different strategies were built. The first replicates the author's previous developed strategy (Certo and Cordeiro, 2005; Certo, et al., 2007a). This strategy was balanced in order to deal with the most possible number of different situations. A behaviour based description for the three different kinds of rescue agents follows.

7.2.1. Ambulance Team

Based on the known Civilian properties, mainly buriedness and health points, they estimate the time of death and schedule the order in which Civilians should be saved. The time taken in travelling, and whether a path to the Civilian exists, are considered.

The next steps for an Ambulance are: go to the position of the first reachable Civilian on list; unbury the civilian; load him into the Ambulance; travel to the closest refuge and unload the Civilian to safety, moving on to the next one. However, this behaviour can change due to several events, such as receiving updated or new information about a Civilian, which may change rescue order and priorities. Behaviour can also change to a self-preservation mode if the ambulance is on a burning building. Bear in mind that buried field agents will always have higher priority.

In FC Portugal's implementation, all Ambulance Teams form and act as a single group. This happens because the action of digging an agent is cumulative, and directly proportional to the number of Ambulance Teams. Therefore, the more Ambulance Teams, the faster the agent will be unburied. There are some exceptions to this tactic, since the moving cost is considered when estimating the Civilian time of death, and sometimes it is more efficient for Ambulances to act individually. Another exception is at the beginning of the simulation given that, due to road blocks, Ambulance Teams don't usually have possible paths to the same agent, they also act individually.

One final remark on Ambulance behaviour is that, when there are no Civilians to rescue, Ambulance Teams scatter, dividing the map in cells, and search for new Civilians.

7.2.2. Fire Brigade

Fire Brigade agents are the most complex field agents and their function is to extinguish fires. The strategy for Fire Brigades is the following:

Depending on map size, Fire Brigades are organized into groups - usually two or three. Based on relative positions, size, and proximity to refuges, fiery regions are prioritized, and the ones with the highest priority are assigned, sequentially, to the available groups.

At the beginning of the simulation, road blocks prevent Fire Brigades from reaching the assigned regions and targets. As such, in order for the Fire Brigade to be useful, only reachable targets are considered. As soon as the assigned region is reachable, Fire Brigades leave this initial state.

Each group then considers its assigned fiery region, and prioritizes the burning buildings (from now on they are called targets) to extinguish, based on relative position in the fiery region, percentage of building area unburned, proximity with building with buried Civilian, amongst other factors.

The next step is to choose a suitable neighbour building that is in the water range of the target. The conditions for this are: the building cannot be on fire; it must be reachable; and as close to the target as possible. If no suitable building is found, then a road near the target is used. The

disadvantage to this solution is that the Brigade occupies a lane, which in turn increases the risk of a traffic jam.

After moving to the selected building, the Fire Brigade starts watering the target. Note that if for some of the above stated reasons the target changes, and if the new target is in range of the water cannon, the Fire Brigade simply starts watering the new target without requiring a move action.

After some cycles, water in tanks is depleted and, therefore, Fire Brigades go to the nearest refuge to refill their tanks. As this action takes some cycles, when refilling finishes, Fire Brigades reprioritize between the previous assigned region and the currently unassigned ones - the Fire Brigade proceedings are then repeated. Logically, if at some point there are more groups than fiery regions, the available group will be assigned to a fiery region using the remaining criteria.

Akin to Ambulance Teams, when Fire Brigades run out of fires to extinguish, the groups are scattered and the map is divided into cells, so that new fiery buildings may be found. If none is discovered, Fire Brigades start searching for new civilians.

Additionally, if all buildings have been explored, Fire Brigades keep on visiting all known living, buried, Civilians in order to update the information on their properties, allowing Ambulances to better estimate the Civilian time of death.

7.2.3. Police Force

The first strategic decision made is to only clear road blocks at not passable roads. This means that partially obstructed roads will not be cleared, and the reason for this is that they only affect the speed of an agent by halving it. The speed reduction isn't significant, when compared to the time it would take a Police Force to move to that road and remove the partial block.

The main problem for Police Forces has to do with the order in which road blocks are removed. On FC Portugal's implementation there are several ways to do this:

- Clearing blocks until a refuge is reached;
- Clearing blocks from a specific point to a refuge;
- Clear blocks around a refuge;
- Clear a specific path;
- Clear a specific cell.

These possible options are called tasks. Each of these tasks is given a weight, which is a parameter specified in a Police Force configuration file, and can be set for specific maps.

All tasks are requested by other agents, with the exception of clearing a specific cell, which is used when no other tasks are requested. When a Police Force receives a task request, it calculates how long each task is going to take, and multiplies it with the weight factor, executing the cheapest one. If the cost of doing a certain task is too high, the task is not considered.

Police force agents are always performing tasks, unlike the other agents that must move themselves to a certain position, in order to perform a certain action. This means that a Police Force agent, when moving, is always clearing impassable blocks in its way, and no detour of blocks is made.

7.2.4. Resulting Roles and Sub-tactics

In order to represent the previously defined behaviours in the strategic layer the following roles and sub-tactics were defined. As this is an adaptation of existing behaviours into the strategy coordination model, for each sub-tactic there is only one role.

Ambulance Teams

Sbt etd (Ambulance) – In this sub-tactic ambulances estimate de time of death (etd) and try to avoid the death of the first rescuable civilian in the list, as described in 7.2.1. The corresponding role is **R etd (Ambulance).**

Fire Brigades

Sbt initial usefulness (**Fire**) – In this sub-tactic, Fire Brigades extinguish the best, reachable, burning building. This sub-tactic is particularly useful at the beginning of the simulation when the desired region isn't reachable. The corresponding role is **R initial usefulness (Fire).**

Sbt balanced (Fire) – Fire control is made taking into account both the civilians at risk and possible fire spread, as described in section 7.2.2. The corresponding role is **R balanced (Fire)**.

Police Forces

Sbt Optimized (Police) – Police forces clear paths around refuges, respond to requests from other agents and equally divide the map for clearing, as described in section 7.2.3. The corresponding role is **R Optimized (Police)**.

Generic

Usually used in the previous, behaviourally, strategy when the field agents could not perform the tasks specific to their kind, or ran out of tasks.

GSbt fire search – Assigned agents split the map equally amongst them and search the city for fires. The corresponding role is **GR fire search.**

GSbt civilian search – Assigned agents split the map equally amongst them and search the city for civilians, updating civilian status upon completion of the search. The corresponding role is **GR civilian search.**

7.2.5. Resulting Strategy

The next figure (Figure 32) shows the equivalent formations to the original strategy built using the previously defined sub-tactics and roles.

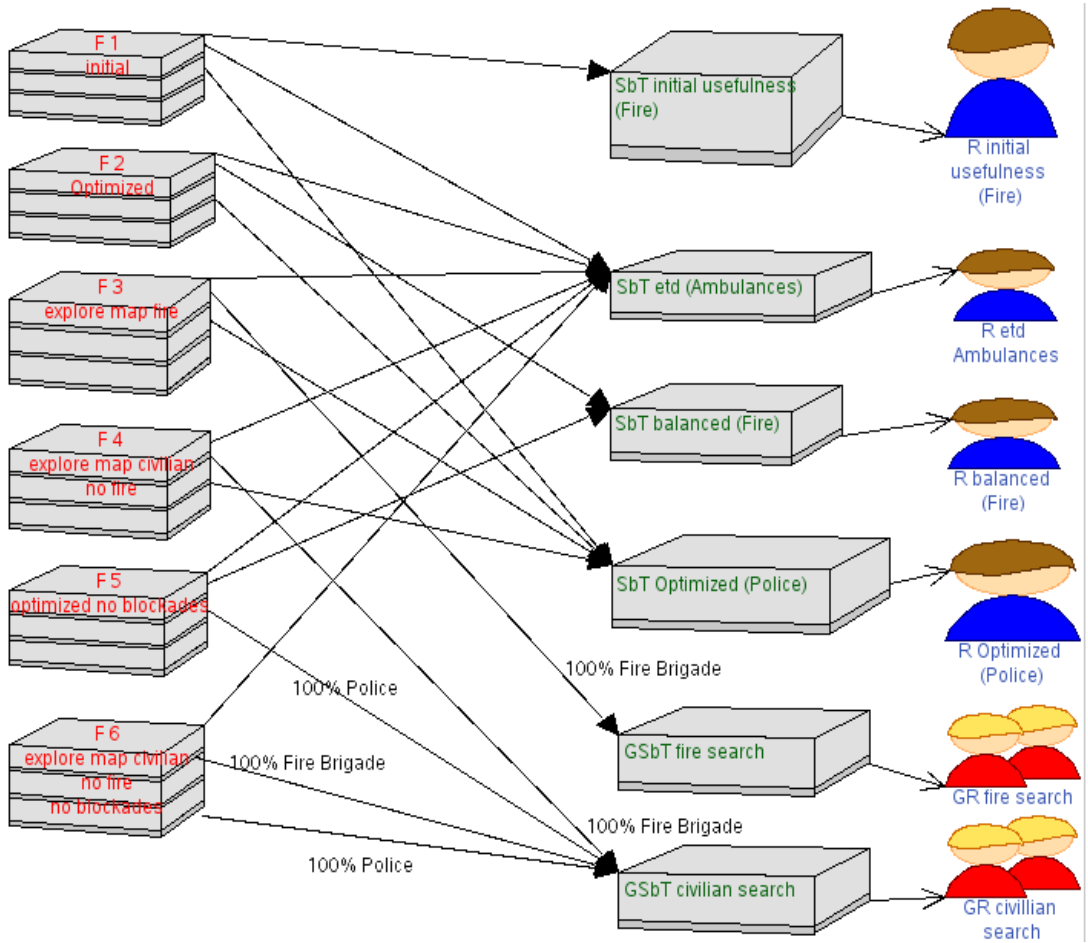

Figure 32: Equivalent original formations.

An overview of the representation of the original strategy in the strategic layer can be found in Figure 33. Note that this is a capture of the tactic sheet as the only information in the Strategy sheet is an interconnection between the strategy object and T1 (original strategy only had one tactic).

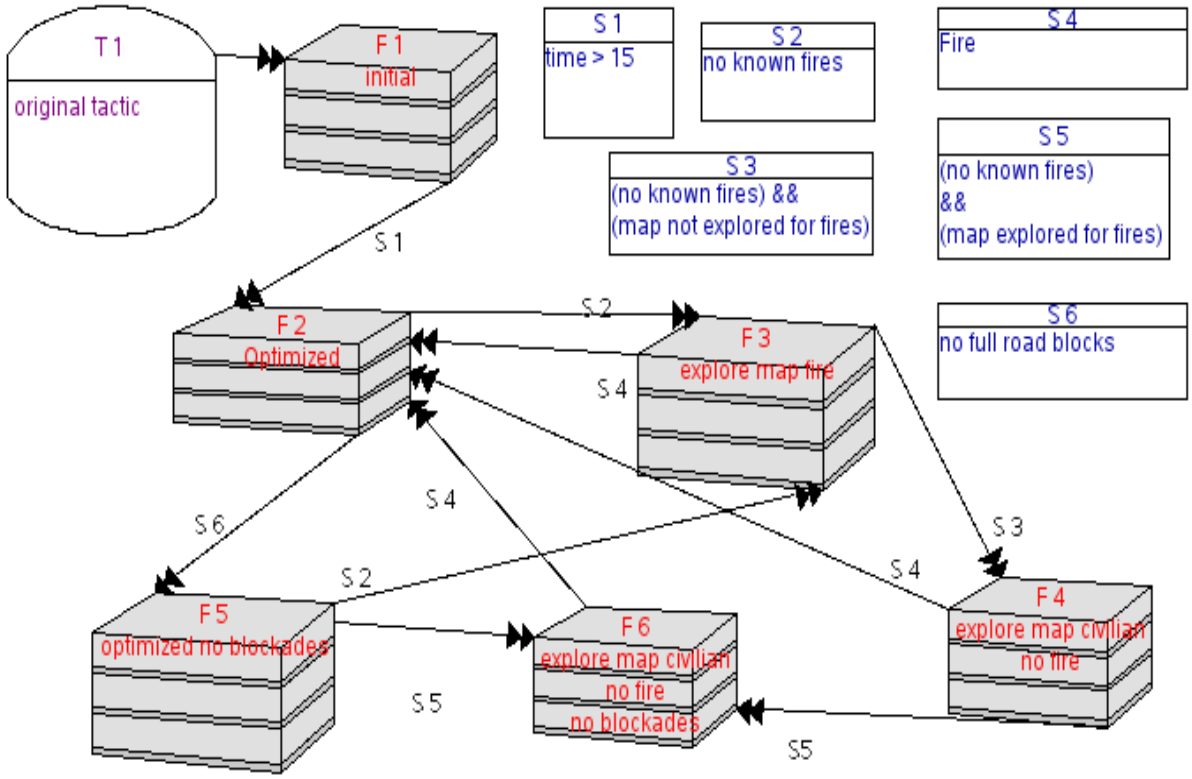

Figure 33: Equivalent tactic for original strategy.

7.3. Definition of Rescue Sub-tactics and Roles

For the RoboCup Rescue domain, a collection of domain related strategical objects was created as a basis for strategy design. In this context, the following sub-tactics and roles were defined.

7.3.1. Ambulance Teams

Sbt easy first (Ambulance) – In this sub-tactic ambulances try to save in the largest number of civilians in the short term choosing the least buried civilians first. The corresponding role is **R easy save (Ambulance).**

Sbt fire threat (Ambulance) – In this sub-tactic ambulances try to save the civilians threatened by close fires. The corresponding role is **R fire threat (Ambulance).**

7.3.2. Fire Brigades

Sbt civilian protection (**Fire**) – In this sub-tactic, Fire Brigades extinguish preferably fiery regions, buildings on fire with civilians or buildings on fire close to civilians. Two corresponding roles were created for this sub-tactic. In the role **R regional civilian protection (Fire)** Fire Brigades focus on protecting civilians from fire only within their assigned regions**.** In the role **R individual civilian protection (Fire)** Fire Brigades don't have an assigned region and distribute themselves to protecting civilians from fire in the entire city.

Sbt min area (Fire) – Fire control is made taking only into account the effect of area damage in fire spreading. As seen on equation (3), in section 2.2, the score is a function of the burned area and civilian casualties. As a result, in some situations, an extra burned building could harm the evaluation function more than an extra casualty. The corresponding role is **R min area (Fire)**.

7.3.3. Police Forces

Sbt help ambulances (Police) – Police forces clear paths between ambulances and refuges. The corresponding role is **R help ambulances (Police)**. Note that this is a partial sub-tactic, in the general strategy after the paths between ambulances and refuges have been cleared another sub-tactic should be used. For example, in the new formation, the general sub-tactic defined in section 7.2.4 **GSbt civilian search** could be used for the police.

Sbt help fire (Police) – Police forces clear paths between fire brigades and refuges as well as around fiery regions. The corresponding role is **R help fire (Police)**. Note that this is also partial sub-tactic, in the general strategy after the sub-tactic related tasks have been accomplished, another sub-tactic should be used.

Sbt main roads (Police) – Police forces clear paths on the roads that are predicted to be the most used. The corresponding role is **R main roads (Police)**. This sub-tactic also uses the role **Sbt Optimized (Police)** defined in section 7.2.4 if the number of Police Force agents is bigger than a certain threshold.

7.4. Strategy Comparison

Until the work described in this thesis FC Portugal's search and rescue team relied on a single approach that was activated for all rescue scenarios. Although some behaviours were adapted by the agents, depending on the city or on real-time scenario conditions, there was no real option to approach scenarios in a totally different way. The strategy layer provided the option to test new approaches and, in this section, a comparison of the old strategy/approach with a new one will be made. It is not expected that the new strategy will be better than the old for all scenarios, but if the new strategy proves more effective for certain scenarios, a better, newer mixed strategy could be made that uses tactics from both the compared strategies.

7.4.1. Competing strategy

The following strategy was created to compare with the original one. For comparison purposes this strategy will be referred as new. Besides the comparison purpose, another reason to build this strategy was the demonstration of the quick reusing capabilities of the coordination and strategy model. As such, formation 2 in Figure 33 was replaced by formation 7 presented on Figure 34.

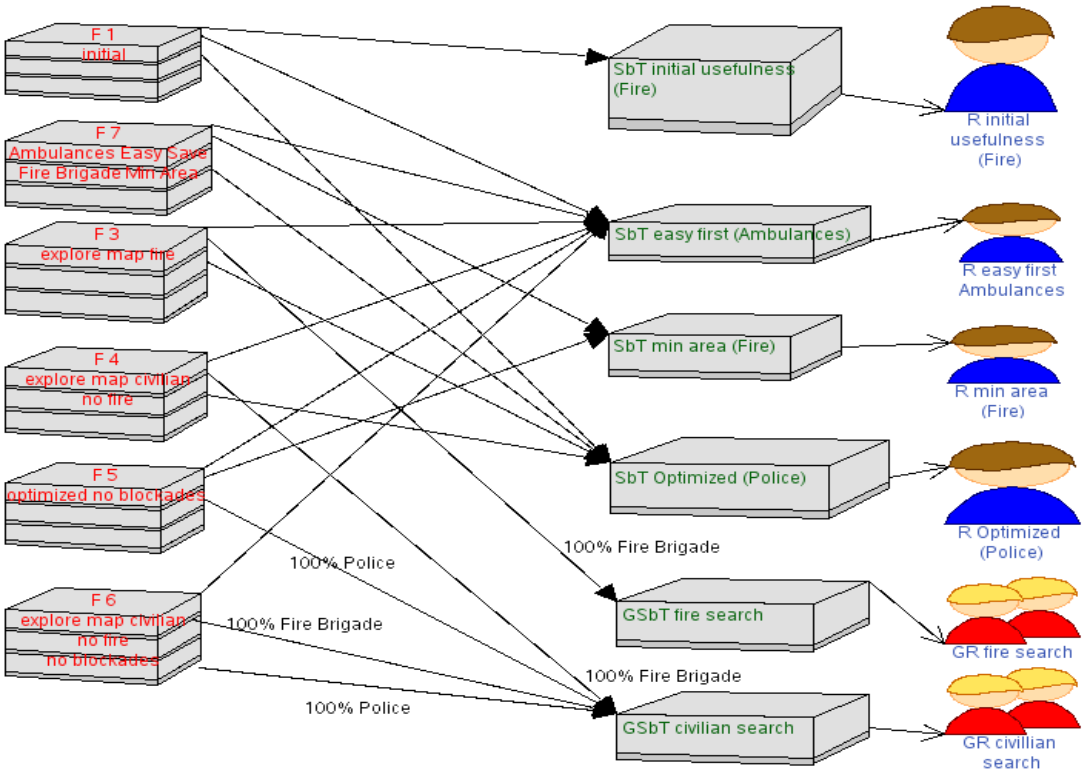

Figure 34: The new formations.

All formations other than 7, changed the ambulance's sub-tactics from **Sbt etd (Ambulance)** defined in 7.2.4 to **Sbt easy first (Ambulance)** defined 7.3.1 adjusting the correspondent roles. Likewise, all fire brigade formations that formerly used the sub-tactic **Sbt balanced (Fire)** defined in 7.2.4 now use **Sbt min area (Fire)** defined in 7.3.2, with the correspondent role adjusted.

The purpose of this new tactic is allowing the fire brigades to try to contain the fire as much as possible. In this tactic, fire brigades do not take into account civilian locations while deciding the best regions and buildings to extinguish. As civilians that formerly would be protected by fire brigades can now die by fire, it was decided that maybe a greedy approach by ambulances could counter-weight this effect. As such, ambulances now try to quickly move the less buried civilians to safety instead of working on the rescuable list. The expected result is that, in some situations, the fire would not spread as much and that the reduction in burned area compensates and exceeds the score loss due to civilian casualties (although this is not a very politically-correct tactic).

7.4.2. Test Setup

In order to test the competing strategies and ensure a smooth simulation run, two computers were used. One was used to run the simulator, another to run the agents. A configuration of the computers used can be found in Table 4.

Table 4: Computer Configuration.

	Server	Agent clients
Processor	Intel Pentium IV	Intel Pentium IV
Memory	1536 MB	512MB
Operative System	Linux - Ubuntu	Linux - Ubuntu

The latest, stable, simulator version is 0.49plus however there are no analysis tools for this version and, as such, the only automated result that is possible to obtain in this version is the final score. For this reason, it was decided to use the previous stable version 0.48. For this version an analysis tool called FCPx was previously developed (Certo, et al., 2006).

It is important to test strategies several times as the simulator is of a stochastic nature. One other important factor to consider is the diversity of scenarios as some tactics may be better suited for some scenarios than others. Considering the requirements, both strategies were simulated 3 times for the following 3 test scenarios.

The Virtual City (VC) scenario is composed of uniformly distributed building blocks. The configurations for this city are found in Table 5 and both 2D and 3D views of this scenario can be found in Figure 18 at section 2.2.7, used in the tests.

Table 5: Virtual City configuration.

	Value		Value
Initial Fires	15	Refuges	2
Civilians	89	Ambulance Centers	1
Ambulance Teams	7	Fire Stations	1
Fire Brigades	14	Police Offices	1
Police Forces	13	Initial Score	124

The Kobe scenario is a model of part the Japanese city of Kobe (scale 1/10). This city was hit by a major earthquake (7.2 on the Richter magnitude scale) and as a consequence, approximately 6,434 people died. Table 6 shows the configurations for this scenario, used in the tests.

Table 6: Kobe configuration.

	Value		Value
Initial Fires	6	Refuges	1
Civilians	70	Ambulance Centers	1
Ambulance Teams	6	Fire Stations	1
Fire Brigades	13	Police Offices	1
Police Forces	10	Initial Score	100

The final test scenario is the Foligno map. Foligno is an ancient town of Italy, stage to an earthquake (5.5 on the Richter magnitude scale) set in 1997 that caused two deaths and serious damage to historical buildings. A 2D view of the scenario can be found in Figure 16 at section 2.2.7 and

Table 7 shows the configurations for this scenario, used in the tests.

Table 7: Foligno configuration.

	Value		Value
Initial Fires	6	Refuges	2
Civilians	16	Ambulance Centers	1
Ambulance Teams	6	Fire Stations	1
Fire Brigades	11	Police Offices	1
Police Forces	7	Initial Score	41

7.4.3. Results

The data presented in this section was gathered from the log files through the "FCPx" tool. Although FCPx permits the analysis and comparison of simulations over an extensive number of parameters, the strategy comparison will be based only on a selection of those considered as the most relevant. These parameters are presented in the tool's summary table called "Simulation Analysis – Tables" (Certo, et al., 2006).

Table 8 presents the results for each simulation run in the Virtual City scenario. From the analysis of the table is possible to perceive that overall, based on the final score, the "new" strategy was slightly better. However, when comparing the burned area at table's line 3, the difference between the two strategies becomes more evident. One of the factors that may have contributed for this difference is the great number (15) of initial fires (Table 5). As the "new" strategy disregards civilians in favour of fire control the destroyed area values seem to be justified. To notice is that the improvement in the fire control came at the cost of "Points Lost Due to Casualties".

Table 8: Summary of VC simulation results.

Score Data\ Strategy	original	original	original	new	new	new
Total Number of Agents	123	123	123	123	123	123
Number of Agents not Killed	89	83	92	92	84	84
Percentage of Building Area Destroyed	25.59%	25.26%	25.91%	12.70%	12.37%	20.62%
Initial Score	124.00	124.00	124.00	124.00	124.00	124.00
Points Lost Due to Casualties	31.30	36.70	28.63	29.86	37.37	36.26
Points Lost Due to Burned Buildings	15.46	15.18	15.70	7.72	7.50	12.42
Final Score	77.23	72.12	79.67	86.42	79.13	75.32
Casualties	original	original	original	new	new	new
Total Number of Civilians	89	89	89	89	89	89
Civilians Killed by Fire	15	14	9	17	11	14
Civilians Killed by Debris	19	26	22	14	28	25
Civilians not Killed	55	49	58	58	50	50
Rescue Agents Killed	0	0	0	0	0	0

The following Table 9 presents the results for each simulation run in the Kobe scenario. From the analysis of the table it is possible to perceive that overall, based on the final score, the original strategy was quite better than the "new" one. The reason for this result is that in both strategies fires were controlled in a very early stage were both strategies use the same role for fire brigades (**R initial usefulness (Fire)**). It becomes evident, from the civilian's casualties'

numbers, that the poorer performance of the "new" strategy comes from the greedy approach of the ambulances.

Table 9: Summary of Kobe simulation results.

Score Data\ Srategy	original	original	original	new	new	new
Total Number of Agents	99	99	99	99	99	99
Number of Agents not Killed	88	82	82	72	72	71
Percentage of Building Area Destroyed	0.39%	1.03%	1.08%	1.16%	0.88%	1.83%
Initial Score	100.00	100.00	100.00	100	100	100
Points Lost Due to Casualties	11.29	17.31	17.31	27.27	27.25	28.18
Points Lost Due to Burned Buildings	0.19	0.51	0.54	0.58	0.44	0.91
Final Score	88.51	82.17	82.15	72.15	72.31	70.91
Casualties	original	original	original	new	new	new
Total Number of Civilians	70	70	70	70	70	70
Civilians Killed by Fire	0	0	0	0	1	0
Civilians Killed by Debris	11	17	17	27	26	28
Civilians not Killed	59	53	53	43	43	42
Rescue Agents Killed	0	0	0	0	0	0

Finally, Table 10 presents the simulation results for the map of the Foligno city.

Table 10: Summary of Foligno simulation results.

Score Data\ Srategy	original	original	original	new	new	new
Total Number of Agents	40	40	40	40	40	40
Number of Agents not Killed	36	33	37	37	36	37
Percentage of Building Area Destroyed	40.19%	39.51%	42.60%	38.01%	39.38%	34.99%
Initial Score	41.00	41.00	41.00	41.00	41.00	41.00
Points Lost Due to Casualties	3.97	6.54	3.10	3.05	3.97	3.05
Points Lost Due to Burned Buildings	8.56	8.17	9.25	8.14	8.36	7.43
Final Score	28.47	26.29	28.65	29.81	28.66	30.52
Casualties	original	original	original	new	new	new
Total Number of Civilians	16	16	16	16	16	16
Civilians Killed by Fire	1	0	0	1	1	1
Civilians Killed by Debris	3	7	2	2	2	2
Civilians not Killed	12	9	14	13	13	13
Rescue Agents Killed	0	0	1	0	1	0

As seen on Table 10, there is no clear best strategy for Foligno although the “new“ strategy has a minor advantage on overall score. The “new” strategy was better at controlling the fires yet it came at the cost of fire casualties.

7.5. Final Considerations

The task of adapting existing behaviours into roles in search and rescue so that they would be compliant with the strategy coordination model can be arduous. However, the identifying, singling out and combining of existing algorithms into separate roles led to the powerful possibility of combining different tactics as shown on the conceptual strategy presented in section 7.1. By providing and additional set of roles, new tactics like the one in the “Competing Strategy” can be built with specific formations to different situations. Trough the combination of several and different tactics, different approaches can be set in a strategy and tested in search and rescue scenarios.

In conclusion there were some scenarios were the new tactic obtained better results. As expected, no single tactic was better for all situations. However, it was proven that for some scenarios the different approach option provided by the tool could have a positive impact on the overall team performance. A further combination of the tested tactics, new tactics and sub-tactics with the definition of new situations should improve the overall performance of the team. The new model allowed a rapid change and adaptation of different simulation approaches.

Chapter 8

8. Conclusion and Future Work

This thesis presented a generic coordination and strategy model for multi-agent systems. The technical aspects of the application domains were studied as well as related research in the area of multi-agent coordination. A graphical tool was developed for building strategies compliant with the defined model and, as a proof of concept, some strategies were developed for the application domains. The tests showed that there is a room for different approaches to face scenarios and that the mechanisms provided by the layer are fit for this purpose.

From the study of the application domains it was concluded that the lowest level in the coordination and strategy model would be the role. From there on, the domains specificity would interfere in the generic aspect of the model. Using the role as basis for the model opens the possibility of devising global strategies without worrying about the low level actions and tasks.

The study of other research in the area led to the conclusion that no single technology was fitted for both the objectives of domain independence and global, high-level management of agent's coordination.

8.1. Main Contributions

The model's flexibility enables its use in environments where a single program manages all homogeneous "dummy" robots, and the collective use in heterogeneous, multi-agent systems. In fact, when domains have similar nature like in soccer simulation and soccer robotic leagues, the strategies defined in one, can easily be adapted to the others. This is achieved by only modifying the roles in the existing sub-tactics.

The new concepts of sub-tactics simplify the management of heterogeneous agents taking advantage of the different capabilities. The concept of generic-role maintains the capability with homogeneous teams and allows an efficient use of common capabilities in heterogeneous teams.

Binders allow the use of formations in a delimited timeframe or situation and at the same time simplify the use of substitute formations.

Absolute and Percentage value forms allow a dynamic, real time adaptation of the layer to changes in the number of available agents while assuring the enforcement of priority roles. In fact, the possibility of escalating the agent number without even changing the strategy is one of the strongest points of the layer.

Strategies are easily developed through the use of a very user-friendly graphical tool. The graphical design in addition to facilitating the understanding of other users' strategies allows a rapid reuse of parts of a strategy into a new strategy. By using a frequently improved, open source, editor as its base, the developed graphical tool can take advantages of its innovations. All the strategy figures used in this thesis come from screenshots of the graphical tool for building strategies.

8.2. Integration and Implementation

The proposed strategical layer is integrated with our soccer and rescue teams and is successfully being used in our rescue team. The layer also maintains full compatibility with all of our RoboCup soccer teams, as the soccer model is a particular case of the specified generic layer.

The layer's introduction of small organizational units into the soccer domain allowed the independent planning of sub-tactics for defence, midfield and attack. New formations can now be rapidly formed from those available sub-tactics. Reusing strategies in different soccer domains is now made easier.

The visual debugger tool, besides its valuable use as a debugging tool has been used to tune strategic positions and proved quite valuable in identifying situations, making tactical changes and predicting their effects.

On the RoboCup rescue domain they layer introduced the possibility of using different global strategic approaches to scenarios instead of just adapting behaviours to the environment conditions. By providing and additional set of roles, new tactics can be built with specific formations to different situations. The new model allowed a rapid design, change and adaptation of a previous tactic into another that resulted in a new way of facing the scenario. The testing of different tactics showed that for some scenarios and situations there is room for a different approach. The strategic coordination layer provides the means for further combination these tactics resulting in the improvement of the overall performance of the team.

8.3. Scientific Contributions

During the development of the strategy coordination model, the work was promoted and disseminated through demonstrations, participation in international RoboCup competitions and publication in international conferences and journals. In these publications, the paper "*A Generic Multi-Robot Coordination Strategic Layer*" describes the strategical layer which is the main subject of this thesis (Certo, et al., 2007b).

The work related to the visual debugger for soccer was integrated into two, wider scope papers, one about agent's reasoning entitled "*Understanding Dynamic Agent's Reasoning"* and another related to debugging methodologies in agents entitled "*Multi-Level, Functional, Spatial and Temporal Agent's Reasoning Debugging* " (Lau, et al., 2007a; Lau, et al., 2007b).

The paper "*Simulation Meets Reality: A Cooperative Approach to RoboCup's Physical Visualization Soccer League"* introduces FC Portugal's approach to a new RoboCup league that combines a virtual scenario with real robots and loosely includes some simple implementations of a few layer concepts (Gimenes, et al., 2007).

The papers (Certo, et al., 2006) and (Certo, et al., 2007a) describe the specific work developed to implement FC Portugal RoboCup Rescue simulated team with emphasis on the code development and FCPx tool developed to enable team performance analysis.

Finally, the paper "*A Generic Strategic Layer for Collaborative Networks"* discusses the cooperative strategy model implementation on the Virtual Enterprises domain (Certo, et al., 2007c). In this domain association between agent and virtual enterprises is made. In this way the strategic model is used taking into account that a strategy can be seen as a set of different configurations of collaborative networks or virtual organizations resulting in different approaches to reach the same ultimate goal. Tactics specify which resources of the collaborative network should be used to reach the different, partial, goals. As such, a formation can be seen as a collaborative network's organization for a pressing goal. A sub-tactic is seen as a limited association of resources, a virtual enterprise or an isolated network. Lastly, the role can be seen as the activity pattern of each participant in a virtual enterprise.

8.4. Limitations and Future Work

As the strategical coordination model uses roles as its base this implies an adaptation of existing agent programs to use a role based model. Furthermore, in order to maintain the layer generic some concessions had to be made. If the application domain has a spatial environment, the

spatial coordination is left out of the layer and built into roles. Due to domain specificity the agent assignment methods are also left out of the coordination model. It is up to the agent to find which strategical specified role is more fitted to be assumed.

The implementation of the strategic coordination layer in the RoboCupRescue domain left the possibility of using different tactics to face the scenarios. The correct choice of tactics for the different scenarios is not a simple task. For the future, it is possible to use the strategical layer, combined with a machine-learning algorithm to optimize the use of different approaches for the different scenarios.

In the future, further development of the graphical tool is expected, mostly on the source code generator. The graphical tool should also be able to generate efficient language independent code for the built strategy. Validating the strategies through the use of a domain plug-in should also be an interesting feature that would avoid some strategy building errors and unintentional effects. These developments will enable a more generalized use of the strategic layer in the context of RoboCup and in other cooperative domains. Thus, the plan is to use the strategical layer, with different instantiations, built using the graphical tool, in all the University teams (simulation 2D, simulation 3D, small-size, middle-size, legged, simulation rescue and physical visualization) participating in European and world RoboCup competitions in 2009.

References

(Akin, et al., 2004). Akin, H. L., Skinner, C., Habibi, J., Koto, T., and Casio, S. L., "Robocup 2004 Rescue Simulation League Rules V1.01", Robocup 2004 Rescue Simulation League Technical Commitee, http://robot.cmpe.boun.edu.tr/rescue2004/, 2004.

(Asada and Kitano, 1999). Asada, M. and Kitano, H., "RoboCup-98: Robot Soccer World Cup II", in *Lecture Notes in Artificial Intelligence*: Springer, 1999.

(Belzunce, et al., 2006). Belzunce, A., Carrera, D., Hall, L. M., Kupfer, P., Laurenson, I., Matins, A., Miller, P., Rentz, D., Roberts, C., Toberts, I., Uhlig, W., Weber, J. H., and Worthingtom, L., *OpenOffice.org 2.0 Draw Guide*, Friends of OpenDocument Inc, 2006, 978-1-4116-9691-4.

(Breit, et al., 2000). Breit, K., House, H., and Samson, J., "The Dia manual, version 0.2", Date Accessed: March 2008, http://www.gnome.org/projects/dia/doc/dia-manual.pdf, 2000.

(Certo and Cordeiro, 2005). Certo, J. and Cordeiro, N., "Search and Rescue in Urban Catastrophes", in *Projecto de Fim de Curso*: Faculdade de Engenharia da Universidade do Porto, 2005, pp. 93.

(Certo, et al., 2006). Certo, J., Cordeiro, N., Reinaldo, F., Reis, L. P., and Lau, N., "FCPx: A Tool for Evaluating Teams' Performance in RoboCup Rescue Simulation League", *Journal of Research in Computer Science*, vol. 26, 2006, pp. 137-148.

(Certo, et al., 2007a). Certo, J., Cordeiro, N., Reis, L. P., and Lau, N., "FC Portugal: Search and Rescue in Urban Catastrophes", in Proceedings of CISTI 2007 - 2ª Conferência Ibérica de Sistemas e Tecnologias de Informação, Novas Perspectivas em Sistemas e Tecnologias de Informação, Porto, Portugal, Edições Universidade Fernando Pessoa, 21-23 June 2007a, pp. 193-204.

(Certo, et al., 2007b). Certo, J., Lau, N., and Reis, L. P., "A Generic Multi-Robot Coordination Strategic Layer", in Proceedings of RoboComm 2007 - First International Conference on Robot Communication and Coordination, Athens, Greece, October 15-17 2007b, pp. 8.

(Certo, et al., 2007c). Certo, J., Lau, N., and Reis, L. P., "A Generic Strategic Layer for Collaborative Networks", in *Establishing the Foundations for Collaborative Networks, In IFIP International Federation for Information Processing*, Camarinha-Matos, L. A., H., Novais, P., Analide, C., Ed. Boston Springer, 2007c, pp. 273-282.

(Charniak and McDermott, 1985). Charniak, E. and McDermott, D., *Introduction to Artificial Intelligence*, Addison-Wesley, 1985.

(Chen, et al., 2002). Chen, M., Foroughi, E., Heintz, F., Huang, Z., Kapetanakis, S., Kostiadis, K., Kummeneje, J., Noda, I., Obst, O., Riley, P., Steffens, T., Wang, Y., and Yin, X., "RoboCup Soccer Server Manual", 2002, pp. 130.

(Cohen and Levesque, 1991). Cohen, P. R. and Levesque, H. J., "Teamwork", *Noûs, Vol. 25, No. 4, Special Issue on Cognitive Science and Artificial Intelligence* September 1991, pp. 487-512.

(Committee, 2000). Committee, R., "RoboCup Rescue simulator manual v0.4", The RoboCup Rescue Technical Committee 2000/07/01 2000.

(Decker, 1995). Decker, K., *Environment Centered Analysis and Design of Coordination Mechanisms*, University of Massachusetts, Phd Thesis, 1995.

(Erol, et al., 2005). Erol, K., Hendler, J., and Nau, D. S., "Complexity results for HTN planning", *Annals of Mathematics and Artificial Intelligence*, vol. 18, no. 1, March, 1996 2005, pp. Pages 69-93.

(Gimenes, et al., 2007). Gimenes, R., Mota, L., Lau, N., Reis, L. P., and Certo, J., "Simulation Meets Reality: A Cooperative Approach to RoboCup's Physical Visualization Soccer League", in Proceedings of 13th Portuguese Conference on Artificial Intelligence, EPIA 2007, Guimarães, Portugal, December 3-6 2007, pp. 12.

(Grosz, 1996). Grosz, B., "Collaborating systems", *AI magazine*, vol. 17, no. 2, 1996.

(Grosz and Kraus, 1996). Grosz, B. and Kraus, S., "Collaborative plans for complex group actions", *Artificial Intelligence*, vol. 86, 1996, pp. 269 - 358.

(Hensgen and Authors, 2003). Hensgen, P. and Authors, U. U. M., "Umbrello UML Modeller Handbook, v1.2", Date Accessed: March 2008, http://docs.kde.org/stable/en_GB/kdesdk/umbrello/index.html, 2003.

(Horling, et al., 1999).Horling, B., Lesser, V., Vincent, R., Wagner, T., Raja, A., Zhang, S., Decker, K., and Garvey, A., "The TAEMS White Paper", 1999.

(Jennings, 1995). Jennings, N. R., "Controlling cooperative problem solving in industrial multiagent systems using joint intentions", *Artificial Intelligence*, vol. 75, no. 2, 1995, pp. 195-240.

(Kitano, 1997). Kitano, H., "Robocup: The Robot World Cup Initiative", in Proceedings of 1st International Conference on Autonomous Agent (Agents97), Marina del Ray, The ACM Press, 1997.

(Kitano, et al., 1995). Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., and Osawa, E., "Robocup: The Robot World Cup Initiative", in Proceedings of IJCAI95 Workshop on Entertainment and AI/Alife, 1995.

(Kitano, et al., 1997). Kitano, H., Asada, M., Kuniyoshi, Y., Noda, I., Osawa, E., and Matsubara, H., "RoboCup: A Challenge Problem for AI and Robotics", *AI Magazine*, vol. 18, no. 1, 1997, pp. 73-85.

(Kitano, et al., 1999). Kitano, H., Tadokoro, S., Noda, I., Matsubara, H., Takahashi, T., Shinjou, A., and Shimada, S., "RoboCup Rescue: search and rescue in large-scale disasters as a domain for autonomous agents research", in Proceedings of IEEE International Conference on Systems, Man, and Cybernetics, 1999, pp. 739-743.

(Kok, et al., 2003). Kok, J. R., Vlassis, N., and Groen, F., "UvA Trilearn 2003 Team Description", in Proceedings of RoboCup 2003 Symposium (CD), RoboCup Federation, 2003, pp. 4.

(Lamb, 2003). Lamb, B., "The Kivio Handbook, rev 1.5", Date Accessed: March 2008, http://docs.kde.org/development/en/koffice/kivio/, 2003.

(Lau and Reis, 2002). Lau, N. and Reis, L. P., "FC Portugal 2001 Team Description: Configurable Strategy and Flexible Teamwork", in *RoboCup 2001: Robot Soccer World Cup V*, Birk, A., Coradeschi, S., and Tadokoro, S., Eds., Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2002, pp. 1-10.

(Lau, et al., 2007a). Lau, N., Reis, L. P., and Certo, J., "Multi-Level, Functional, Spatial and Temporal Agent's Reasoning Debugging", in Proceedings of 13th Portuguese Conference on Artificial Intelligence, EPIA 2007, Guimarães, Portugal, December 3-6 2007a, pp. 12.

(Lau, et al., 2007b). Lau, N., Reis, L. P., and Certo, J., "Understanding Dynamic Agent's Reasoning", in *Progress in Artificial Intelligence, 13th Portuguese Conference on Artificial Intelligence - EPIA 2007*, Neves, J. M., Santos, M. F., and Machado, J. M., Eds., Guimarães, Portugal, Lecture Notes in Computer Science, 4874 Springer Berlin / Heidelberg, 2007b, pp. 542-551.

(Lekavý and Návrat, 2007). Lekavý, M. and Návrat, P., "Expressivity of STRIPS-Like and HTN-Like Planning", in *Agent and Multi-Agent Systems: Technologies and Applications*, Lecture Notes in Artificial Intelligence, Springer Berlin / Heidelberg, 2007, pp. 121-130.

(Levesque, et al., 1990). Levesque, H. J., Cohen, P. R., and Nunes., J., "On acting together", in Proceedings of National Conference on Artificial Intelligence, Menlo Park, California, AAAI press, 1990.

(Nair, et al., 2001). Nair, R., Ito, T., Tambe, M., and Marsella, S., "Task Allocation in the RoboCup Rescue Simulation Domain", in *RoboCup-2001: The fifth Robot World Cup Games and Conferences*, Birk, S. C. a. S. T. a. A., Ed. Springer-Verlag, Heidelberg, Germany, 2001, pp. 4.

(Nüssle, et al., 2004). Nüssle, T., Kleiner, A., and Brenner, M., "Approaching Urban Disaster Reality: The ResQ Firesimulator", Universität Freiburg, Institut für Informatik 200, April 2004.

(Pollitz and Sacks, 1997). Pollitz, F. F. and Sacks, I. S., "The 1995 Kobe, Japan, earthquake: A long-delayed aftershock of the offshore 1944 Tonankai and 1946 Nankaido earthquakes", *Bulletin of the Seismological Society of America* no. 87, February 1997, pp. 1-10.

(Reinaldo, et al., 2005). Reinaldo, F., Certo, J., Cordeiro, N., Reis, L. P., Camacho, R., and Lau, N., "Applying Biological Paradigms to Emerge Behaviour in RoboCup Rescue Team", in *Progress in Artificial Intelligence: 12th Portuguese Conference on Artificial Intelligence, EPIA*, Bento, C., Cardoso, A., and Dias, G., Eds., Covilha, Portugal, LNCS, Springer-Verlag, 2005, pp. 422 - 434.

(Reis, 2003). Reis, L. P., *Coordination in Multi-Agent Systems: Applications in University Management and Robotic Soccer*, University of Porto, Porto, PhD Thesis, 2003.

(Reis, et al., 2001). Reis, L. P., Lau, Nuno, and Oliveira, E. C., "Situation Based Strategic Positioning for Coordinating a Team of Homogeneous Agents", in *Balancing Reactivity and Social Deliberation in Multiagent Sytems: From RoboCup to Real Word Applications*, Hannenbauer, M., Wendler, J., and Pagello, E., Eds., Berlin, Springer's Lecture Notes in Artificial Intelligence, 2103, Springer-Verlag, 2001, pp. 175-197.

(Reis and Lau, 2001). Reis, L. P. and Lau, N., "FC Portugal Team Description: RoboCup 2000 Simulation League Champion", in *RoboCup-2000: Robot Soccer World Cup IV*, Stone, P., Balch, T., and Kraetzschmar, G., Eds., Berlin, Springer's Lecture Notes in Artificial Intelligence, 2019, Springer-Verlag, 2001, pp. 29-40.

(Russell and Norvig, 1995). Russell, S. and Norvig, P., *Artificial Intelligence: A Modern Approach*, Prentice Hall, 1995.

(Sato and Smith, 2002). Sato, T. and Smith, B. V., "XFig Version 3.2.5-alpha User Manual", Date Accessed: March 2008, http://xfig.org/userman/, 2002.

(Skinner, et al., 2005).Skinner, C., Brenner, M., Paquet, S., Ito, N., and Rahimi, A., "Robocup 2005 Rescue Simulation League Rules": Robocup 2005 Rescue Simulation League Technical Commitee, 2005.

(Stanley, 1997). Stanley, R. P., *Enumerative Combinatorics* Cambridge University Press., 1997, ISBN: 0-521-55309-1.

(Stone, 2000). Stone, P., *Layered Learning in Multiagent Systems: A Winning Approach to Robotic Soccer* MIT Press, 2000, ISBN: 0262194384.

(Stone, et al., 1999). Stone, P., Riley, P., and Veloso, M., "CMUnited-99 source code, 1999", Date Accessed: March 2008, Accessible from http://www.cs.cmu.edu/~pstone/RoboCup/CMUnited99-sim.html, 1999.

(Stone, et al., 2000). Stone, P., Riley, P., and Veloso, M., "Layered Disclosure: Why is the agent doing what it's doing?" in *Agents 2000, Fourth Int. Conf. on Autonomous Agents*. Barcelona, 2000.

(Stone and Veloso, 1999). Stone, P. and Veloso, M., "Task Decomposition, Dynamic Role Assignment, and Low-Bandwidth Communication for Real-Time Strategic Teamwork", *Artificial Intelligence*, vol. 110, no. 2, June 1999, pp. 241–273.

(Takahashi, et al., 2000). Takahashi, T., Takeuchi, I., Tetsuhiko, K., Tadokoro, S., and Noda, I., "RoboCup-Rescue Disaster Simulator Architecture", in Proceedings of Robocup 2000: Robot Soccer World Cup IV, Springer-Verlag GmbH, 2000, pp. 379-284.

(Tambe, 1997). Tambe, M., "Towards Flexible Teamwork", *Journal of Artificial Intelligence Research*, vol. 7, 1997, pp. 83-124.

(Team, 2008). Team, T. K. P., "The KOffice Project", Date Accessed: March 2008, http://www.koffice.org/, 2008.

(Veloso, et al., 2000). Veloso, M., Pagello, E., and Kitano, H., "RoboCup-99: Robot Soccer World Cup III", in *Lecture Notes in Artificial Intelligence*: Springer, 2000.

(W3C, 2006). W3C, W. W. W. C.-. "Extensible Markup Language (XML) 1.1 (Second Edition)", World Wide Web Consortium - W3C, Date Accessed: March 2008, http://www.w3.org/TR/xml11/, 2006.

(Wittig, 1992).Wittig, T., *ARCHON: an architecture for multi-agent systems*, Ellis Horwood Limited, 1992, 0-13-044462-6.