A Generic Multi-Robot Coordination Strategic Layer

João Certo***, Nuno Lau****, Luis P. Reis******
** IEETA – Institute of Electronics and Telematics Engineering of Aveiro, Portugal*
*** FEUP – Faculty of Engineering of the University of Porto, Portugal*
**** DETI –Informatics, Electronics and Telecommunications Dep., University of Aveiro, Portugal*
***** LIACC – Artificial Intelligence and Computer Science Lab., Univ. Porto, Portugal*
joao.certo@fe.up.pt, lau@det.ua.pt, lpreis@fe.up.pt

Abstract—**Managing a team of heterogeneous robots in a dynamic environment poses a challenging job. In this paper 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. As both centralized and distributed environment are handled by the layer, communication was an important factor to consider only introducing a minor overhead. A multi-level hierarchical approach was followed with hybrid methods used to switch between concepts. The model was tested with two strategy instances, RoboCup Rescue Simulation and RoboCup Soccer. Strategies are designed with the help of a graphical tool. Results achieved by the team in RoboCup Rescue and Soccer Simulation competitions demonstrate the usefulness of this approach.**

Index Terms—**Distributed coordination of mobile robots.**

I. Introduction

RoboCup was created as an international research and education initiative, aiming to foster Artificial Intelligence (AI) and Robotics research, by providing standard problems. 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.

Proposed by Kitano [1], 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. In order to obtain a good score, all these agents work together communicating through supervising center agents.

In RoboCup Soccer leagues two opposing teams play a soccer match, hence creating a dynamic environment. Developing robots that are able to play a soccer match provides important scientific challenges, both at an individual level (hardware, perception, moving, dribbling, shooting) and at a collective level (strategy, collective play, communication, formations, passing, etc.)

FC Portugal's research focus is on the development of new coordination methodologies. After successfully developing such methodologies for soccer simulation leagues[1], the team is working on adapting these methodologies to the Rescue Simulation League already with some success[2].

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. The model and tools developed aim to simplify the portability of research between RoboCup leagues and expand its usefulness to areas outside this domain.

This paper describes the specification and application of a multi-purpose, multi-domain, adaptable, strategical layer on multi-agent systems. This layer allows the management of homogeneous and heterogeneous agents, and the centralized or decentralized management of the strategy. A graphical strategy building tool, compliant with the layer is also presented.

Manuscript received September 4, 2007. This project is collaboration between the Universities of Porto and Aveiro. The research was supported by FCT under grant # POSI/EIA/63240/2004. The authors wish to thank the Portuguese FCT - National Science and Technology Foundation for the fellowship granted. The authors also wish to thank LIACC – Artificial Intelligence and Computer Science Lab of the University of Porto, IEETA – Institute of Electronics and Telematics Engineering of Aveiro, FEUP – Faculty of Engineering of the University of Porto, and the University of Aveiro for their support to the project.

[1] FC Portugal won several World and European championships in different RoboCup soccer leagues in the past seven years.
[2] FC Portugal rescue simulation team achieved very good results in RoboCup, including winning a rescue European champion using these coordination methodologies.

ROBOCOMM 2007, 14th – 16th Oct 2007, Athens, Greece.
Copyright © 2011 – 2012 ICST
ISBN 978-963-9799-08-0
DOI 10.4108/ICST.ROBOCOMM2007.2207

The rest of this paper is organized as follows. The next section presents related work together with some related previous work. In section III the strategic layer is described. Section IV presents an implementation on the rescue team. Section V presents the graphical tool, showing a strategy for the soccer team. Section VI concludes this paper and points out to future work.

II. Related Work

Authors Stone and Veloso previously defined periodic team synchronization (PST) domains as domains with the following characteristics: “There is a team of autonomous agents that collaborate towards the achievement of a joint long-term goal” [2, 3]. Then they decomposed the task at hand, into multiple rigid roles, assigning one agent to each role. Thus each component of the task was accomplished and there were no conflicts among agents in terms of how they should accomplish the team goal. As it was defined, a role consisted of a specification of an agent's internal and external behaviors. The conditions and arguments of any behavior could depend on the agent's current role, which was a function of its internal state.

Due to inflexibility to short-term changes (e.g. one robot is non-operational), inflexibility to long-term changes (e.g. a route is blocked), and a lack of facility for reassigning roles, a formation was introduced as a teamwork structure within the team member agent architecture. A formation decomposes the task space defining a set of roles with associated behaviors. In a general scenario with heterogeneous agents, subsets of homogeneous agents could flexibly switch roles within formations, and agents could change formations dynamically. Formations included as many roles as there were agents in the team, so that each role is filled by one agent.

Much of FC Portugal’s related research was done for soccer simulation leagues. The rest of this section explains the concepts and mechanisms developed for those 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.

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 (defense, attack, etc) [4]. Formations assign each player a positioning (that determines the strategic behavior) and each positioning a player type (that determines the active behavior).

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 behaviors 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 [5]. 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 is used for strategic situations (in which the agent believes that it is not going to enter in active behavior soon) [5, 6]. 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 behavior 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. The concept was extended and players may 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 [4].

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) [4]. 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 modeling to estimate these utilities.

III. 3. 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. The terms *scenario* and *agent* should be considered as broader terms. *Scenario* can be a simulation, a game, or any other kind of set where there is an environment, with *agents* who have one or more objectives. 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 do *teamwork* [7, 8]. In this model, homogeneous and heterogeneous *agents* can be used. In heterogeneous environments the term *agent type* is used for differentiation.

A. *Structure*

In order to better explain the model, a top-down approach will be followed. Both informal and formal definitions will be given for each concept.

Figure 1 represents the proposed model and depicts the interconnections between the concepts presented in this model. The figure only expands one branch for each concept.

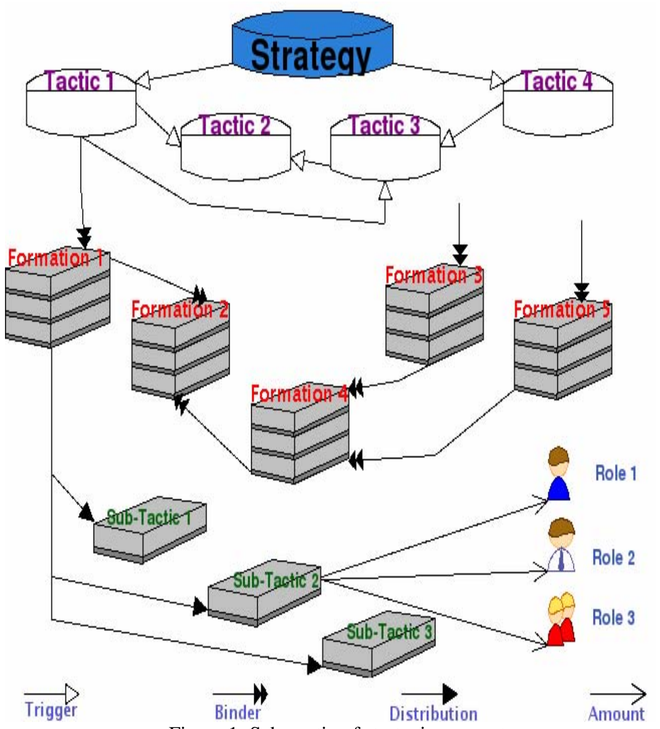

Figure 1. Schematic of strategic concepts.

1) Strategy: is the combining 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:

$$\text{Strategy} = \{\text{Tactics, Triggers}\} \quad (1)$$

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

$$\text{Tactics} = \{\text{Tactic 1, Tactic 2, ... , Tactic t}\}, \forall \text{ t} \in N \quad (2)$$

Triggers set the conditions to interchange tactics:

$$\text{Triggers} = \{\text{Trigger 1, Trigger 2, ... , Trigger tg}\}, \forall \text{ tg} \in N \quad (3)$$

2) *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 more particular problems and *binders* 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.

$$\text{Tactic} = \{\text{Formations, Situations, Binders, [Tactical Parameters]}\} \quad (4)$$

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

$$\text{Formations} = \{\text{Formation1, Formation2, ... , Formation f}\}, \forall \text{ f} \in N \quad (5)$$

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

$$\text{Situations} = \{\text{Situation 1, Situation 2, ... , Situation s}\}, \forall \text{ s} \in N \quad (6)$$

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

$$\text{Binders} = \{\text{Binder 1, Binder 2, ..., Binder b}\}, \forall \text{ b} \in N \quad (7)$$

Tactics can optional have tactical parameters:

$$\text{Tactical Parameters} = \{\text{Tactical Parameter1, ... , Tactical Parameter tp}\}, \text{tp} \in N \quad (8)$$

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

$$\text{Situation} = \{\text{Condition 1, Condition 2, ... , Condition cd}\}, \forall \text{ cd} \in N \quad (9)$$

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*:

$$\text{Binder} = \{\text{[Origin Formations], Situations, Terminus Formation}\}, \text{[Origin Formations], Terminus Formation} \in \text{Formations} \quad (10)$$

3) 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 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*.

$$\text{Formation} = \{\text{Distribution, Sub-Tactics, [Agent Types]}\} \quad (11)$$

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 [9]. Here, m(Sub-Tactic st) defines the multiplicity of a *sub-tactic*:

$$\text{Sub-Tactics} = \{(\text{Sub-Tactic1}, m(\text{Sub-Tactic1})), (\text{Sub-Tactic2}, m(\text{Sub-Tactic2}), ..., (\text{Sub-Tactic st}, (\text{Sub-Tactic st}))\}, \forall \text{ st} \in N \quad (12)$$

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

$$\text{Distribution} = \{\text{Value1, Value2, ... , Value v}\}, \quad v = \sum m(\text{Sub-Tactic st}) \quad (13)$$

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 A5), an *agent type* must be specified for that *sub-tactic*:

$$[\text{Agent Types}] = \{\text{Type1, Type2, ..., Type ty}\}, \quad \forall \text{ ty} \in N \quad (14)$$

4) 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*.

$$\text{Sub-Tactic} = \{\text{Amounts, Roles, [Sub-Tactical Parameters]}\} \quad (15)$$

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

$$\text{Roles} = \{\text{Role 1, Role 2, ... , Role r}\}, \forall \text{ r} \in N \quad (16)$$

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

$$\text{Amounts} = \{\text{Amount 1, Amount 2, ... , Amount a}\}, \quad a = \sum \text{role r} \quad (17)$$

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*.

5) *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 behavior. The behavior 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:

- 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*);
- 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 behavior for an entire scenario, they can also describe the behavior for only a given time frame or *situation*. *Teams* form 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 IV.

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 use 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.

B. Decision, Supervising and Communication

The decision maker 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 II), the strategical layer can be applied to scenarios were 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*.

C. 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 differentiating himself from others. Generally, this differentiation consists of attributing a different number or a 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 II) 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 II). This is accomplished because the positioning systems are specified inside the *role*.

IV. Implementation on Search and Rescue

In order to adopt the strategic layer, our rescue team needed to use role concept. The previous code was based on a sequential selection of algorithms based on world state conditions. 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.

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 2 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.

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.

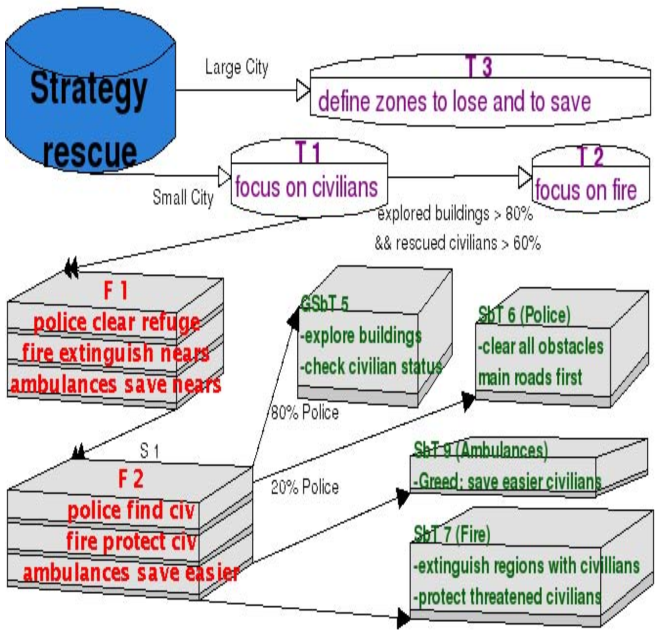

Figure 2. Partial rescue strategy

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 3 the *situation* (S1) to switch from *formation* F1 to *formation* F2 is defined.

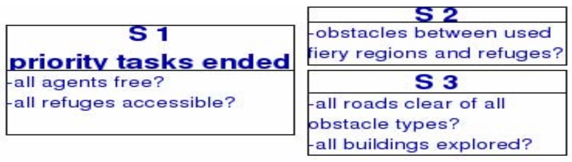

Figure 3. Some rescue situations.

In Figure 4 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.

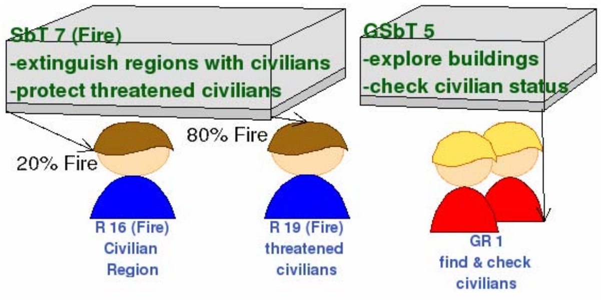

Figure 4. Two rescue sub-tactics.

The *generic sub-tactic* GSbT 5 has 80% of the Police Force assigned to it (Figure 2). In Figure 4 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 checkup 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.

V. GRAPHICAL TOOL FOR BUILDING STRATEGIES

The graphical tool provides a visual interface for building strategies. By using graphical reorientations of the strategic layer components, it is possible to interconnect them. The tool exports the edited strategy to an XML file which can be used to implement the layer in *agents*.

The graphical tool's Graphical User Interface (GUI) is provided by Kivio, a flowcharting and diagramming application for the KOffice[3] application suite. A customized, installable, stencil set with the layer objects was built as seen in Figure 5.

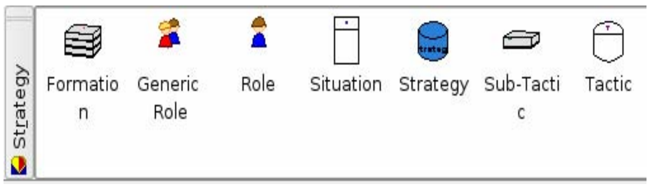

Figure 5. Strategy stencil set.

In order to implement the tool on agents, a C++ code generator was created. Although still in its early stages, this application already generates code for strategies using homogeneous and heterogeneous agents with real-time dynamic agent number. This application also generates a configuration file that contains the tactical parameters but also has the possibility for quick disabling a particular formation or tactic of the given strategy.

Using soccer as an example, for a simple strategy, the same sheet can be used to represent the entire layer as seen in Figure 6.

In this example the soccer team as 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* (Strategy to T1 and to T2). Likewise a *binder* can originate in a tactic thus defining the initial *formation* (T1 to F2 and T2 to F1). Note the absolute values assignment mode and the implicit value assignment to certain roles (eg. In Sbt3 all 3 robots are assigned to role R4). The fact that soccer is a domain with a fixed number of players in the team enables this strict assignment mode.

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

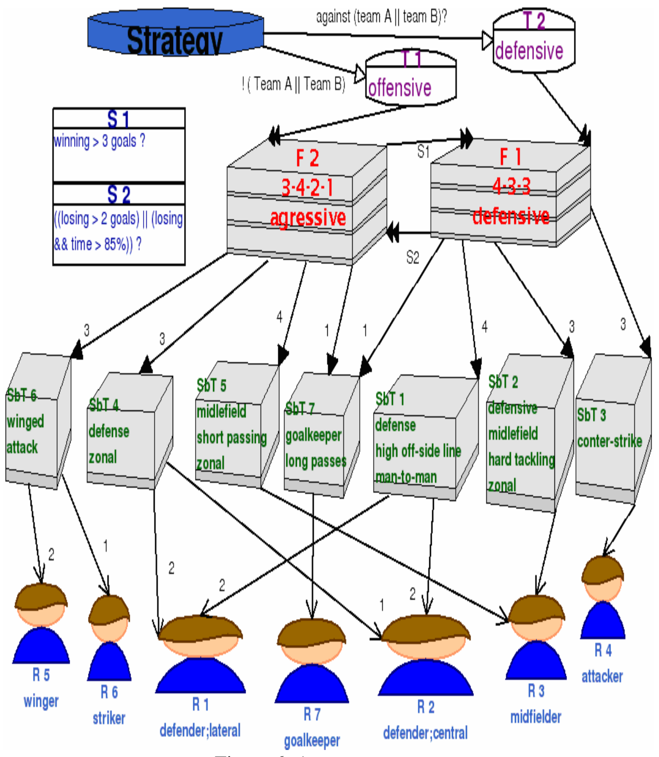

Figure 6. A soccer strategy.

For a more complex *strategy* a multi-sheet is recommended separating *strategy*, each *tactic*, *formations*, *situations* and *sub-tactics* (Figure 7).

Figure 7. 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 1 although only expands on one strategy branch (not possible under the layer), it uses formation F3 and formation F5 with no precedences.

VI. Conclusion and Future Work

The proposed strategical layer is now fully integrated with our soccer and rescue teams and is successfully being used in our rescue team. The layer also maintains full compatibility with all our RoboCup soccer teams, as the soccer model is a particular case of the specified generic layer. Results in international competitions for both the domains tested, proved the success of the layer.

The model flexibility enables using it in an environment where a single program manages all homogeneous "dummy" robots, to its 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.

Strategies are easily developed through the use of a very user-friendly graphical tool. By using a frequently improved, open source, editor as its base, the developed graphical tool can take advantages of its innovations.

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. These developments will enable a more generalized use of the strategic layer in the context of RoboCup and in other cooperative domains. Thus, we plan to use the strategical layer, with different instantiations, built using the graphical tool, in all our teams (simulation 2D, simulation 3D, small-size, middle-size, legged, simulation rescue and physical visualization) participating in European and world RoboCup competitions in 2007.

References

[1] H. Kitano, S. Tadokoro, I. Noda, H. Matsubara, T. Takahashi, A. Shinjou, and S. Shimada, "RoboCup Rescue: search and rescue in large-scale disasters as a domain for autonomous agents research," *IEEE International Conference on Systems, Man, and Cybernetics*, pp. 739-743.

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

[3] P. Stone, *Layered Learning in Multiagent Systems: A Winning Approach to Robotic Soccer* MIT Press, 2000.

[4] L.P. Reis, and N. Lau, "FC Portugal Team Description: RoboCup 2000 Simulation League Champion," *RoboCup-2000: Robot Soccer World Cup IV*2019, P. Stone, T. Balch, and G. Kraetzschmar eds., LNAI Springer-Verlag, 2001, pp. 29-40.

[5] L.P. Reis, Lau, Nuno, and E.C. Oliveira, "Situation Based Strategic Positioning for Coordinating a Team of Homogeneous Agents," *Balancing Reactivity and Social Deliberation in Multiagent Sytems: From RoboCup to Real Word Applications*, M. Hannenbauer, J. Wendler, and E. Pagello eds., Springer-Verlag LNAI 2103, 2001.

[6] N. Lau, and L.P. Reis, "FC Portugal 2001 Team Description: Configurable Strategy and Flexible Teamwork," *RoboCup 2001: Robot Soccer World Cup V*, Lecture Notes in Computer Science 2377, A. Birk, S. Coradeschi, and S. Tadokoro eds., Springer Berlin / Heidelberg, 2002, pp. 1-10.

[7] P.R. Cohen, and H.J. Levesque, "Teamwork," *Noûs, Special Issue on Cognitive Science and Artificial Intelligence*, vol. 25, no. 4, 1991, pp. 487-512.

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

[9] R.P. Stanley, *Enumerative Combinatorics,* Cambridge University Press., 1997.