A ROLE-BASED MULTI-AGENT MODEL

FOR CONCURRENT NAVIGATION SYSTEMS

S. Fournier, T. Devogele and C. Claramunt

Naval Academy Research Institute
BP 600, 29240, Brest Naval, France

Email: {fournier, devogele, claramunt}@ecole-navale.fr

1. INTRODUCTION

Since the eighties constant augmentation of the demand on worldwide maritime transportation together with acute economical competition have increased the pressure on ship owners and commercial fleets. Crews are often reduced to a minimum and sometimes to even one officer of the watch in the deck. In order to counterbalance these negative effects on safety navigation, and to facilitate the task of navigation officers, a new generation of automated-navigation systems has appeared over the past years. Those include the Automatic Radar Plotting Aids (ARPA) [1], Automatic Identification System (AIS) [2] and Electronic Charts Display Information System [3].

The ARPA is currently one of the most frequently implemented systems for automatic navigation. It is based on a radar system that makes tracking of ships possible, and derives important navigation information such as the Closest Point of Approach (i.e. minimal distance that separates two ships whose routes are crossing) and the Time to Closest Point of approach (TCPA). The AIS is a communication system that allows exchange of navigation data between ships and navigation authorities (one VHF transmitter, two VHF Time Division Multiple Access receivers and one VHF Digital Selective Calling receiver). It is a solution comparable to aeronautic transponders. The ECDIS is an electronic mapping tool that includes GPS information, navigation tools, route planning and warning functionalities. It can also integrate ARPA and AIS data.

These automated navigation systems are part of the resources used by an officer of the watch, contributing thus to safe navigation. Further work in the development of
navigation-aided systems are at least of two orders. First, there is still a need for some affordable, light and mobile solutions that can be used by small ships, particularly for coastal navigation. This is the objective of the Share-Loc prototype developed in a related work [4]. Share-Loc provides a mobile solution, based on either WML or HTML, together with a centralised application server that monitors the concurrent displacements of several ships. Secondly, current navigation-aided systems are mainly monitoring tools that evaluate the configuration of a given navigation area or the neighbourhood of a ship at a given time, with so far a lack of reasoning and prediction mechanisms although anticipation is a constant objective of officers of the watch to guaranty a safe navigation. Therefore, extensions of current navigation-aided systems towards decision-aided tools are still expected developments.

Navigation decisions are based on several parameters from physical data on the bathymetry, weather, swell and stream to ship movements. In a previous work, we have developed a rule-based prototype that monitors the concurrent displacement of several ships [5]. Navigation decisions are explicitly modelled by condition-action rules depending on navigation configurations. However rule-based systems imply a-priori modelling of navigation configurations, this being a difficult computing target when the number of ships is high in a given area. Also there is a lack of autonomy, flexibility and interaction between the different ships represented. This leads us to explore a different alternative, based on a multi-agent system, and potentially adapted to our modelling context, where ships are modelled as autonomous agents that behave and interact in their environment.

This paper objective is to explore the application of a multi-agents approach to model a concurrent navigation system. It integrates some of the modelling concepts recently introduced in multi-agent system such as roles and group behaviour, and extends those to the concepts of permanent roles and the integration of priorities in role definition. A case study illustrates these concepts by a maritime navigation case study and prototype.

2. MODELLING BACKGROUND

Multi-agent systems haven’t been applied so far, to the best of our knowledge, to model and simulate a maritime navigation system despite the fact that the modelling concepts underlying multi-agent systems are particularly and potentially adapted to those systems: ships can be considered as autonomous agents acting, behaving and interacting (exchange of messages) in a maritime environment. In some ultimate cases some ships might even change their behaviour and take the role of a different agent: this is the case of an emergency event where several ships (i.e. agents) change their state and behaviour to act as rescuers. This is somehow related to the concept of role in agent modelling [6], initially introduced in database modelling [7][8], and object-oriented modelling [9]. Different modelling properties have been associated to the concept of role: a role can be unique, organised per group where a group qualifies an agent’s community in relation (by interaction, division of an environment, a goal or a common ontology) [10][11]. The concept of agent extended to the notion of role allows us to identify the foundations of a meta-model, applied to the simulation of a maritime navigation system, upon which we will design and add several additional concepts particularly adapted to our application area, but also adapted to other application contexts.

3. MODELLING APPROACH

3.1 UML Meta-Model

In order to set up the modelling support of our simulation environment, we define a UML agent-based metamodel that takes into account the concepts of role and group (Fig. 1). The main properties of this metamodel are as follows. An agent is part and acts in its environment, its definition includes properties and methods, and it has control on some
resources (e.g. a ship needs fuel to progress). An agent may have some objectives (e.g. a destination). It belongs to one-to-many groups. Within a group, an agent takes at least one role where its behaviour is defined according to the role(s) it takes. In fact the role paradigm allows an agent to act in its environment. In some cases, an agent may depend on some resources and functionalities owned by other agents to perform an action \cite{12,13} to use those resources and dependencies; an agent has to interact with those agents. Roles enlarge the degree of interactivity and increase the semantics and flexibility of the model.

![Fig. 1 Meta-model (UML)](image)

### 3.2 Meta-Model properties

In order to enrich the semantics of the meta-model we add additional properties to the concept of role. First, in order to avoid some conflicts, we introduce a priority stamping in the attribution of roles. The roles an agent can take at a given time are prioritised by a total order. Role priorities can change during the existence of an agent according to its objectives. Secondly we make the distinction between permanent and temporary roles. For a given agent, we define a role as permanent when an agent keeps this role during its lifetime, as temporary when an agent keeps it during a sub-interval of its lifetime. We define an exclusive role as a role that agent cannot play with any other temporary role, while an exclusive role can be played together with one-to-many permanent roles. On the one hand, if an agent takes a role defined as permanent and exclusive, then this agent cannot take any other role during its lifetime. On the other hand, if an agent has one-to-many non-exclusive
permanent roles, then it can take several temporary roles, but just one both exclusive and temporary. There is also the option to give user-defined constraints at the meta-model level. Those constraints monitor the compatibility - or not - of different agent roles. For example, a role is incompatible with another role, even in a same group, when both cannot be taken at a same time. An agent cannot play two identical roles identified by a same name (i.e. same functionalities) at the same time either within a same group or in two different groups. Let us give the formal notations and properties of these modelling concepts.

Let $A$ be the set of agents, $G$ the set of groups, $R$ the set of roles
Let $I$ be the set of time intervals $[t_1,t_2]$ with $t_1,t_2$ the set of real and $t_1 < t_2$.
An agent’s life $a \in A$ is given by a time interval $l_a$ denoted $[T(a)_{\text{start}}, T(a)_{\text{end}}]$.
A group’s life $g \in G$ is given by a time interval $l_g$ denoted $[T(g)_{\text{start}}, T(g)_{\text{end}}]$.
Let $GR$ denote a subset of the Cartesian product $G \times R$ where $(r,g) \in GR$ iff $r$ is a role of the group $g$.
A role may belong to one but only one group, that is, $\forall (r,g_1), (r,g_2) \in GR$ then $r_1 \neq r_2$.
Let $GA$ denote a subset of $G \times A$ where $(a,g) \in GA$ iff the agent $a$ belongs to the group $g$.
Let $AGRT$ denote a subset of $A \times R \times G \times I$ where $(a,r,g,i) \in AGRT$ iff the agent $a$ takes the role $r$ in the group $g$ during $i$ with $(r,g) \in GR$ and $(a,g) \in GA$ and the constraint that $i \subseteq l_a \cap l_g \neq \emptyset$.

**Property 1: Play**

The predicate $Play$ defines the role an agent takes in a group during a given time interval $i$. $Play$ is defined as follows:

$$ \text{Play}(a,r,g,i) = \begin{cases} 1 & \text{if } (a,r,g) \in GRAT \\ 0 & \text{otherwise} \end{cases} $$

When $\text{Play}(a,r,g,i) = 1$ we say that the agent $a$ plays the role $r$ in the group $g$ during the time interval $i$. Let us take the example of an agent $ship$ that plays the role $tanker$ in a group $monitoring$. Let $ship \in A$, $tanker \in R$, $monitoring \in G$. During the time interval $i$ the agent $ship$ takes the role $tanker$ in the group $monitoring$, that is, $(ship,tanker,monitoring,i) \in GRAT$. Then we have $\text{Play}(ship,tanker,monitoring,i) = 1$.
Property 2: Role permanence

A permanent role is a role taken by an agent during its lifetime. This property is defined as follows:

\[
\text{Permanent}(a,r) = \begin{cases} 
1 & \text{if } \exists g \in G/ \text{Play}(a, r, g, i) = 1 \forall i \subseteq I_a \\
0 & \text{otherwise} 
\end{cases} 
\] (2)

With the consequence that \( I_a \subseteq I_g \)

When \( \text{Permanent}(a, g, r) = 1 \) we say that the role \( r \) is permanent for the agent \( a \). A role is permanent when an agent plays it during its lifetime. For example, an agent ship plays the role tanker in the group monitoring during its lifetime \( I_{\text{ship}} \). Let \( \text{ship} \in A, \text{tanker} \in R, \text{monitoring} \in G \) and \( I_{\text{ship}} \in I \). As the role tanker is permanent for the agent ship we have

\[
\text{Permanent(\text{ship}, monitoring, tanker)} = 1
\]

With the consequence that \( I_{\text{ship}} \subseteq I_{\text{monitoring}} \) where \( I_{\text{monitoring}} \) denotes the lifetime of monitoring.

Property 3: Role incompatibility

The incompatibility of two roles reflects that fact that two given roles cannot be played at the same time by a same agent. This property is defined for all the agents that take those roles. The incompatibility of two roles is defined as follows:

\[
\text{Incompatibility}(r_1, r_2) = \begin{cases} 
1 & \text{if } \forall (a, g_1, g_2) \in GA (g_1 = g_2 \text{ or } g_1 \neq g_2), i \subseteq I_a \\
0 & \text{otherwise} 
\end{cases} 
\] (3)

When \( \text{Incompatibility}(r_1, r_2) = 1 \) we say that the roles \( r_1 \) and \( r_2 \) are incompatible. Let us consider the example of an agent ship that can play the roles tanker and frigate in the group monitoring. Let us also assume that these roles are incompatible. Let \( \text{ship} \in A, \text{tanker} \in R, \text{frigate} \in R, \text{monitoring} \in G \). We consider that the agent ship plays the role tanker in the group monitoring during the time interval \( i_1 \), then we have

\[
\text{Play(\text{ship}, \text{tanker}, \text{monitoring}, i)} = 1 \Rightarrow \text{Play(\text{ship}, \text{frigate}, \text{monitoring}, i)} = 0 \quad \forall i \in I_{\text{ship}}
\]

Property 4: Role exclusivity

An exclusive role is a role that can be only played concurrently with permanent roles. This property applies to all the groups and roles an agent belongs during the time it plays such an exclusive role. The role exclusivity is defined as follows:

\[
\text{Exclusive}(r) = \begin{cases} 
1 & \text{if } \forall (a, g_1, g_2) \in GA, r_i \in R, i \subseteq I_a \\
0 & \text{otherwise} 
\end{cases} 
\] (4)

When \( \text{Exclusive}(r) = 1 \) we say that the role \( r \) is exclusive. Note that the groups \( g \) and \( g_1 \) can be the same. Let us take the example of an exclusive role rescuer in a group monitoring where \( \text{Exclusive(rescuer)} = 1 \). Let \( \text{ship} \in A, \text{rescuer} \in R, \text{monitoring} \in G \). We consider that the agent ship plays the role rescuer in the group monitoring during the time interval \( i_1 \), then
we the role monitoring can (resp. cannot) be taken at the same time than the role rescuer if \( \text{Exclusive(monitoring)} = 1 \) (resp. = 0).

Agents need to share information and cooperate. Within multi-agent systems, agents communicate by messages. Messages are made off two parts: the message header and the corpse of the message. The header contains basic data on the message type and corpse the contain of the message. Agents take messages into account according to their internal state. An agent can only send a message to other agents in groups to which it belongs (i.e. the ones it interacts with). Messages can be either broadcasted to all agents in a group or to one-to-many agents with some particular roles. An agent can also send a direct message to one other agent in same group. Roles attached to agent that receives messages deal with them according to their time of arrival and some priority ordering. When an agent through one of its roles does accept to deal with a message, it takes action accordingly.

4. CASE STUDY

In order to illustrate the potential of our approach we briefly introduce a case study. In a given navigation area, two trawlers are fishing, near them three frigates with a leader are exercising. Tugboat is alongside. Some vessel traffic stations are monitoring the area. A tanker comes in the area and goes straight towards a dangerous area. The authorities inform the tanker of a risk of running aground (Fig. 2). However, the tanker doesn’t send any response and then run aground and finally sends a S.O.S message. Authority detects that tanker is in difficulty and send a message. This message asks the ships in the zone to rescue tanker. Authority sends also a tugboat. A trawler, a frigate and the tugboat attempt to rescue the tanker (Fig. 3).

This case study is modelled according to different groups, roles and role priorities as follows:

- **Group task force.** An agent ship plays the role leader while the others play the roles follower. The roles leader and follower are incompatible.
- **Group fishing boat.** All agents ship take the role fishing. This group is created to reflect some specific interactions (e.g. as fishing ships want to not diffuse any information on their fishing area, they avoid interactions).
- **Group monitoring.** This group contains the agents ship, one agent authority and one agent buoy. The agents ship takes incompatible and permanent roles (frigate, trawler, tanker, tugboat) while the agent authority plays a role vessel traffic station and buoy plays role danger. The role danger is exclusive. The tugboat plays role alongside. When the agent with the role tanker is damaged, the role difficulties is taken, then this agent sends a S.O.S. Finally three of these ships take an additional temporary role rescuer. The roles difficulties and rescuer are exclusive roles, thus the roles follower and fishing are temporary placed in stand-by.
- **Priorities.** The role frigate has priority over the roles follower or leader. The role trawler has priority over the role fishing. The roles difficulties, alongside and rescuers have priority over the other roles. The role rescuer has priority over role alongside.
The prototype implementation is developed under Java 1.4.1. Abstract classes support the definition of agents, roles and groups. Properties are specified at the abstract class level. The definition of the application classes (e.g. ship, tanker, monitoring) are inherited from abstract classes.

A simulation is triggered by a Java virtual machine. When a simulation scenario begins (Fig. 2), an agent ship takes the role tanker and goes straight towards a dangerous zone symbolised by a cross in a circle and modelled by an agent buoy with the role danger. The agent buoy demarcates the dangerous zone. A circle estimates when the agent ship with the role tanker cannot avoid the dangerous zone. The Agent ship with the role tugboat plays the role alongside. Two agents ship have the roles fishing and trawler. Due to their role fishing and the higher priority of this role, these two agents make a circuit around their fishing area. Another agent ship with roles frigate and leader and two agents ship with roles frigate and follower make an exercise, they move in formation. The agent ship with role leader gives orders to the both agents ship with role follower. These orders are broadcasted to all agents in the group task force that play the role follower. The agent authority with the role vessel traffic station sends a cautious message to the tanker several times to report the proximity of a dangerous area. However for whatever reason this message is ignored.
When the agent ship with the role tanker encounters a dangerous zone, it runs aground (Fig. 3). Therefore this agent broadcasts a S.O.S message to all agents in the group monitoring and takes the role difficulties. This message contains its coordinates. The role difficulties takes priority over the role tanker.

When the agent ship with the role fishing receives the S.O.S message, one of them decides to rescue the tanker. It takes the exclusive role rescuer. This role forces this agent ship to go to the coordinates of the S.O.S message emitter. The second agent ship with the role fishing continues to play its role. In fact, the role fishing deals with the message and it decides that it is not close enough to rescue the tanker.

When an agent ship with the role frigate receives the S.O.S message, the agents ship with the role follower ignores the message (i.e. the role follower deletes the S.O.S message) and waits for the agent ship leader orders. The frigate leader commands (by sending a direct message) to one frigate follower - the one closest to the tanker - to rescue it. The message contains the coordinates given by the S.O.S message. The other one stays in formation.

As soon as an agent authority with the role vessel traffic station detects that an agent ship is in difficulty (i.e. has role difficulties), such an agent broadcasts a rescue S.O.S message to all agents in the group monitoring. This message asks ships in the zone to rescue the tanker and communicates the coordinates of that ship in difficulty. Thus, if a ship in difficulty cannot send a S.O.S message or has not a sufficient range, ships around receive a message on the accident from the authorities. The agents ship in that zone decide for whatever reason to ignore this message. In fact, all of them have received a S.O.S message and act according to their roles. The authority sends a direct message to the agent...
ship with the role tugboat that commands it to rescue the agent ship at given coordinates. Finally the agent ship with the role tugboat takes the role rescuer.

5. CONCLUSION

Although many navigation-based systems have been developed over the past years there is still a need for additional reasoning and prediction mechanisms that improve the monitoring and planning of navigation decisions. This paper develops a multi-agent meta-model that supports micro-simulation capabilities where ships are modelled as autonomous agents acting in their environment. We introduce several new modelling concepts that enhance the semantics of this meta-model. Those allow an agent to be part of a group and to act according to the roles defined at the group level. Role priorities and constraints give additional flexibility to the meta-model.

Further work includes addition of continuous phenomena like tides, streams and wind that interact with ships. These interactions allow ships to move more realistically. Physical ship behaviours (rate of turn, weight, height, speed) must also condition ship movements. Integration of navigation rules to avoid collisions and running aground are parts of our further work. In order to make the simulation more plausible we also plan to integrate additional entities such as vessel traffic stations, buoys and lighthouses and maritime polices. Further application of the meta-model concerns other terrestrial or aerial navigation systems.

6. REFERENCES