A Collaborative Decision Support Model for Marine Safety and Security Operations

Uwe Glässer, Piper Jackson, Ali Khalili Araghi, Hans Wehn, Hamed Yaghoubi Shahir

A Collaborative Decision Support Model for Marine Safety and Security Operations*

Uwe Glässer[1], Piper Jackson[1], Ali Khalili Araghi[1],
Hans Wehn[2], and Hamed Yaghoubi Shahir[1]

[1] Software Technology Lab,
School of Computing Science, Simon Fraser University, Burnaby, BC, Canada
{glaesser, pjj, aka52, syaghoub}@cs.sfu.ca

[2] MacDonald, Dettwiler and Associates Ltd., Richmond, BC, Canada
hw@mdacorporation.com

Abstract. Collaboration and self-organization are hallmarks of many biological systems. We present the design for an intelligent decision support system that employs these characteristics: it works through a collaborative, self-organizing network of intelligent agents. Developed for the realm of Marine Safety and Security, the goal of the system is to assist in the management of a complex array of resources in both a routine and emergency role. Notably, this system must be able to handle a dynamic environment and the existence of uncertainty. The decentralized control structure of a collaborative self-organizing system reinforces its adaptiveness, robustness and scalability in critical situations.

Key words: Coastal Surveillance, Self-Organizing System, Automated Planning, Configuration Management, Abstract State Machines

1 Introduction

Canada and its allies have identified the vulnerability of sea lanes, their ports and harbors to a variety of threats and illegal activities. With a total length of over 243,000 kilometers (151,000 miles), Canada has the longest coastline of any country in the world [1]. Scarce surveillance and tracking capabilities make it difficult to perform large volume surveillance, keeping track of all marine traffic [2]. A collaborative research initiative by Defence R&D Canada, MDA and three academic partners addresses the design of intelligent decision support systems [3] for large volume coastal surveillance [4]. The NADIF research project [5] expands on the CanCoastWatch (CCW) project [4] by building on realistic marine safety and security scenarios studied in CCW. The aim is to facilitate complex command and control tasks of Marine Security Operation Centres (MSOC) [6] by improving situational awareness and automating routine coordination tasks. The proposed

* The work presented here is funded by MacDonald, Dettwiler and Associates Ltd. (MDA) and NSERC under Collaborative R&D Grant No. 342503-06.

system concept integrates Adaptive Information Fusion techniques with decentralized control mechanisms for Dynamic Resource Configuration Management (DRCM) and task execution management. Autonomously operating agents form an intelligent decision support system through distributed collaboration and coordination. The target platforms for this system are dynamically reconfigurable network architectures.

Operating under uncertainty in an adverse environment, as will be explained, the system design presented here strives for resilience through adaptivity, robustness and scalability, building on concepts of collaborative self-organizing systems inspired by biological processes and mechanisms [7]:

> *What renders this approach particularly attractive from a dynamic network perspective is that global properties like adaptation, self-organization and robustness are achieved without explicitly programming them into the individual artificial agents. Yet, given large ensembles of agents, the global behavior is surprisingly adaptive and can cope with arbitrary initial conditions, unforeseen scenarios, variations in the environment or presence of deviant agents.*

Border control and emergency response services deploy a range of mobile sensor platforms with heterogeneous sensor units that cooperatively perform surveillance and rescue missions in the vast coastal areas that constitute the *littoral zone*. Platforms include satellites; airborne vehicles (SAR helicopters, patrol aircraft and UAVs); coastguard vessels(frigates and various smaller boats); and land vehicles.

We distinguish cooperative search (e.g., locating a fishing boat in distress) from non-cooperative search (detecting illegal activities, e.g., smuggling operations). A typical mission involves various tasks and subtasks deploying multiple mobile platforms with diverse capabilities: sensor capabilities, mobility capabilities, extraction and transport capabilities. Situation awareness[1] entails flexible control mechanisms to respond to dynamic changes in *internal conditions* regarding mission requirements (priorities, search areas, medical conditions, time windows) and *external conditions*, such as adverse weather conditions and fading daylight, as well as often unpredictable changes in the operational status of platforms.

The NADIF system concept comprises three main parts: a Decision Support Engine, a Configuration Management Engine, and an Information Fusion Engine. We focus here on the design of a collaborative decision support and configuration management model. Section 2 discusses background concepts. Section 3 outlines the conceptual model, and Section 4 addresses Automated Planning and Tasking. Section 5 illustrates our System Reference Model. Section 6 concludes the paper.

[1] Situation Awareness, a state in the mind of a human, is essential for decision-making activities. It concerns the perception of elements in the environment, the comprehension of their meaning, and the projection of their status in the near future [8].

2 Background

This section presents background concepts relevant to the work presented here. We employ these to formulate a model that accurately captures the characteristics, functionality and requirements of the target system.

2.1 Cooperative Models

A Cooperative Multi-agent System in general consists of a number of agents collaborating with each other. Agents are able to use different types of communication in order to fulfill shared goals. Within cooperative systems, there is a wide range of different system architectures with each model defined by some main characteristics.

Different models have been studied (e.g., Swarms, Coalition, Collaboration and Clusters) in [9]. Among these models, *clusters* match the requirements of our system best. In such systems different agents can be combined together to aggregate their capabilities as a group. Clustering also increases the flexibility of the system through dynamic re-organization of the agents. This is beneficial for applications which work in a changing environment. Clustering results in a more complicated structure, which can be costly to manage, but the behavior of each node can be kept simple [10].

2.2 Self-Organizing Systems

The concept of self-organization is used in diverse research areas, such as Biology, Physics and also in social sciences such as Economics. In nature it can be seen, for instance, in flocking by birds and fish. The construction of physical structures by social animals (e.g., bees or ants) is another example. Self-organization is defined as the emergence of global level pattern in a system as a result of many low-level interactions which utilize only local information [11]. The concept of biologically inspired self-organization is becoming increasingly popular in automated systems. In Swarm Robotics, many simple physical robots communicate with each other while interacting with the environment. These communications produce feedback to the system and consequently initiate emergent global behavior in the system. Such behavior is called *Swarm Intelligence* [12]. Swarm behavior can be generated from very simple rules for individual agents.

2.3 Command and Control

In our previous work [13], the DRCM (Dynamic Resource Configuration Management) architecture for the current system was introduced. The topology of the resource configuration network is based on the Command and Control (C2) Hierarchy which is broadly used in the military domain [14]. In such a hierarchy, nodes represent different resources in the network (e.g., platforms, sensors, services and units) and the edges connecting nodes can be seen as relationships

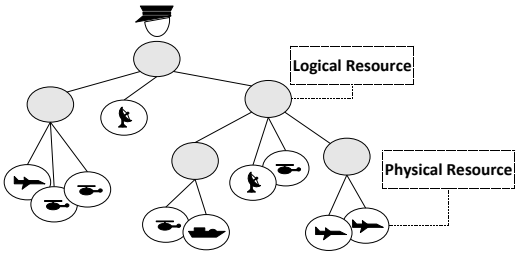

Fig. 1. Organizing Resources: Hierarchical Network Architecture

(command and control from parent nodes) or connectivity (e.g., link, TCP/IP, radio). In Fig. 1, missions are introduced into the system through the commander. As illustrated, the C2 hierarchy has a decentralized control structure where a parent node (commander) assigns tasks to a group of subordinate nodes and subsequently, local decisions are made in the lower layers.

3 Collaborative Self-Organization

This section introduces the conceptual model and basic design principles shaping the DRCM core, specifically the decentralized control processes that dynamically configure mobile resources into resource clusters and continuously monitor and control the operations and tasks being performed by the cluster components.

Operating under uncertain mission requirements in an unstable physical environment calls for flexible adaptation to new situations. Building on collaborative self-organizing system concepts naturally facilitates reconfigurable applications and dynamic reorganization in response to internal changes in resource requirements and external changes affecting the availability of resources.

The complex command and control structure is realized in a distributed and hierarchical fashion by means of a dynamic ensemble of autonomously operating *control agents* interacting with one another and with their local environment. Intuitively, each agent is associated with an individual resource representing a concurrent control thread of the decentralized system. Agents are created or eliminated at run-time as resources are added to or removed from the system.

3.1 Resource Hierarchy

Missions represent complex tasks and goals that normally exceed the capacity and capabilities of any individual resource; hence, they need to be decomposed into subtasks and subgoals in such a way that the resulting tasks and goals can be performed co-operatively by a resource cluster that has the required capacity and also matches the capabilities. This process is performed by the planning component in several iterative steps until all the resulting tasks are executable tasks. When such tasks are ready to be executed the tasking component then allocates resources depending on resource availability and task priority.

In order to simplify mapping the constituent tasks of a mission onto resources that execute them, the network architecture hierarchically organizes resources in clusters. Control agents are nodes in the network architecture; their organization into clusters is stated by undirected edges (see Fig. 1). Referring to distinct roles of resource entities, there are two different types of nodes:

- *Physical resources* refer to real-world resource entities in the form of mobile sensor platforms. In the hierarchy, only the leaf nodes represent physical resources. Depending on the level of abstraction at which a distributed fusion system is considered, a physical resource may refer to a group of mobile sensor platforms, to a single mobile platform, or even to an individual sensor unit on a given sensor platform. [2]
- *Logical resources* refer to abstract resource entities formed by clustering two or more physical and/or logical resources, each with a certain range of capabilities, into a higher level resource with aggregated (richer) capabilities needed to perform more complex operations. A logical resource identifies a cluster of resources. All non-leaf nodes represent logical resources (Fig. 1).

Resource clusters form collaborative self-organizing command and control units that are configured at run-time to perform specific missions and tasks, where their resource orchestration is subject to dynamic change. For increased robustness and to reduce control and communication overhead, logical resources operate semi-autonomously, making their own local decisions on the realignment and reorganization of resources within a cluster. DRCM policies govern the migration of resources between clusters based on common prioritization schemes for resource selection, load balancing and organization of idle resource pools. Resources may join or be removed from a cluster on demand depending on their sensor capabilities, mobility capabilities, geographic location, cost aspects and other characteristics. The underlying design principles resemble those for improving performance and robustness in mobile ad hoc networks.

3.2 Organization Principles

Specific challenges arise from complex interaction patterns between logical and physical resources and the dependencies between the operations and tasks to be performed in a collaborative fashion. The following organization principles outline some of the aspects that need to be addressed.

- *Resource Clustering Principles* control the arrangement of resources into resource clusters. Composition rules defined over resource descriptors specify the clustering of resources so as to form composite resources with richer behaviors. A resource descriptor is an abstract representation of resource attributes such as physical capabilities (e.g., sensor capabilities, mobility and time constraints), geographic position and workload information.

[2] Henceforth, we identify physical resources with mobile sensor platforms.

- *Resource Distribution Principles* refer to the spatio-temporal distribution of mobile resources in the geographical environment. Position information and projections of resource trajectories provide important input for grouping resources into clusters (e.g., keeping resources of the same group in close proximity to each other) and also to satisfy communication requirements (e.g., moving a resource in order to act as a communication proxy).
- *Task Decomposition Principles* define the decomposition of complex tasks (including entire missions) into subtasks based on common patterns and schemes for mapping tasks onto resources. This concept entails an abstract characterization of tasks for specifying their resource requirements and the required orchestration of resources for performing the tasks.

4 Intelligent Decision Support

In order to support the decision making of system operators, it is vital for the system to be able to simulate the entire process of decision making. The focus here is on Automated Planning and Automated Tasking.

4.1 Automated Planning

Generally speaking, any non-trivial objective will require a planning process in order to decide upon the best set of actions to follow. In the domain of Marine Safety and Security, missions introduced by command personnel are stated at an abstract level, establishing a goal without defining a specific manner in which it must be achieved. Expert knowledge and institutional policies can then be invoked in order to select which tasks are likely to be successful. This can be seen as an iterative process, as high level plans are used to generate subplans until an actionable set of tasks is found.

A hierarchical perspective captures this mechanism well: abstract tasks are successively broken down into more refined subtasks. Subtasks requiring more refinement before they can be implemented are also broken down. This continues until all of the resulting subtasks are executable. The set of executable tasks generated this way constitute the elements of the plan. The Hierarchical Task Network (HTN) approach [15] is a prime example of this kind of planning. HTNs use substitution rules called methods to select the right subtasks for each abstract task, generating a tree-like network through this process. The network not only shows the relationship between tasks and subtasks, but also any constraints that exist between tasks. If executable subtasks can be found for all tasks in the network, a solution has been constructed for the problem at hand.

Fig. 2 shows a simplified HTN plan. The top task, *Capsized Boat*, corresponds to the mission introduced to the system. It is too general as stated to be fulfilled, so a method associated with it is used to find three subtasks. In turn, these subtasks require further refinement and other methods are used to find appropriate subtasks for them. Two kinds of constraints are illustrated: precedence and shared resource. The first defines a partial order in which the tasks

must be completed, and the second forces a reasonable limitation on the choice of resources to complete the tasks. The leaf nodes of the network correspond to the executable tasks that make up the final plan.

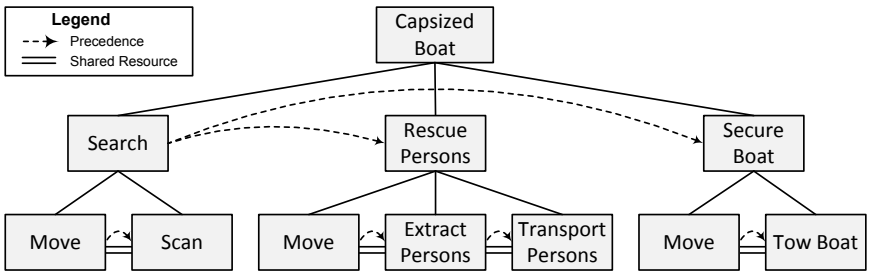

Fig. 2. Hierarchical Task Network (HTN) Example.

A planning system that interacts with the real world must have the ability to replan in order to adapt task choice to deal with changing conditions or new information. Replanning should cause as little disruption as possible to the existing plan, since tasks may currently be underway. Local replanning aims to do this by making changes in the task network as close to the problem as possible [16]. This entails the requirement to be able to evaluate the current world state in order to recognize when replanning is needed, and what in particular needs to change.

4.2 Automated Tasking

Once a plan has been decided upon, either in whole or in part, it is still necessary to see that it is successfully completed. This is due to factors outside of the control of the system, as well as imperfect knowledge gathered from sensors. Appropriate resources must be found for the tasks, and they must be performed only when appropriate. Tasks in the midst of execution must be monitored, since they may provide data that must be considered immediately. Tasks can end prematurely, due to resource failure or environmental interference, for example. In these cases the problem may be solved by finding a new resource to assign to the task, or it may be necessary to change the existing plan. The Tasking component in our system is responsible for handling these issues. It also serves as a nexus of interaction between the other components. In this way, it serves as a buffer and conductor for asynchronous events, enabling the system to act in a robust manner in real-time.

5 System Reference Model

In this section, we describe the interactions between the components of the system in terms of an abstract generic scenario indicating their responsibilities and

relations. In order to model and analyze system scenarios, a standard notation, *User Requirements Notation (URN)* [17] is used. In 2008, the *User Requirements Notation* was approved as an *ITU-T* standard that combines concepts and notation for modeling goals and scenarios. A *scenario* describes a partial usage of the system. It is defined as a set of partially ordered responsibilities to be performed such that the system reaches its goals. Each scenario has *start points*, represented by filled circles and *end points* illustrated by solid bars. A scenario progresses along *paths* between these start and end points, and the *responsibilities* are represented by crosses on the path. The diamond symbol is called a *stub* and is a placeholder for a sub-scenario. We employ them in our model for complexity management by encapsulating some related and coherent responsibilities as a *subcomponent*. Beyond the above concepts and notations, there are other aspects supported by *URN*, but those are not used here.

5.1 Describing the Abstract Generic Scenario

This system is intended to be implemented in a distributed manner, with all five services running on each node in the C2 hierarchy. This allows for a truly decentralized control structure and improves the robustness of the network. We use *jUCMNav* for modeling different concrete scenarios of the system in various situations. The *abstract generic scenario* is the result of generalizing the common parts of these concrete scenarios. As shown in Fig. 3, the system has five components in addition to the Command and Control Center. This section defines the responsibilities of each component and also the communication among them. It is important to note that Fig. 3 shows the flow of *control* and *information*, and the duties of each element, so some parts of the path can be executed concurrently for different *missions* and *tasks*.

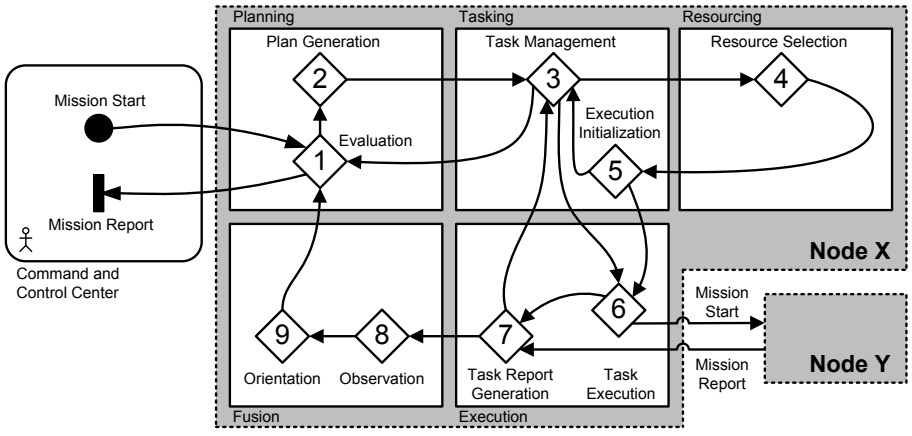

Fig. 3. Abstract Generic Scenario of the NADIF System

The Command and Control Center is outside the boundary of the system and is considered to be an actor on the system. This component is responsible for introducing *new missions* to Planning. In addition, it will receive a report of the finished mission, whether it is successful or not. The decision support components of each node are modeled as an interacting set of internal services. The responsibilities of each component and their respective subcomponents are:

Planning is responsible for generating new plans and also replanning previous cases when necessary.
- *Evaluation*(1): This subcomponent is responsible for evaluating finished tasks, as well as relevant situation information, such as the output of the Fusion component. If the results of a task compromise an active mission, it is sent back to Plan Generation for replanning. If instead a mission has finished (whether successful or not), this subcomponent issues a report to the Command and Control Center.
- *Plan Generation*(2): In this subcomponent, the current plan is decomposed into a set of tasks. Any tasks that are executable are sent to Task Management in order to wait for resource assignment and execution.

Tasking is responsible for managing tasks which are waiting for execution or that have just finished execution.
- *Task Management*(3): This subcomponent maintains the pool of waiting tasks. If a task needs a resource, a request is sent to Resource Selection. When a task is ready to execute, it is sent to Task Execution. It also checks if a task can no longer be executed, due to exceeding its time window, finished status (from the execution report sent by Task Report Generation), or if no resource assignments are possible (i.e., a rejection message from Execution Initialization). In these cases, the task will be sent back to Evaluation, which may result in replanning if necessary.
- *Execution Initialization*(5): Its main duty is to pick the best resources for executing the current task from the list provided by Resource Selection. The decision is based on different parameters such as resource availability, task priority, resource location, and other information. First, resources currently in use by tasks of higher priority are pruned from the list. If the resulting list is empty, this subcomponent sends a message to Task Management; otherwise, there are appropriate resources for executing the task. If the selected resources are idle, they are assigned to the current task, which is then sent to the task pool in Task Management. If any of the selected resources are in use, but the current task has a higher priority, a request to release these resources is sent to the Task Execution subcomponent. Once they have been released, the higher priority task obtains these resources and waits in Task Management for execution.

Resourcing is responsible for monitoring resources available to the current node. It also participates in resource configuration management.
- *Resource Selection*(4): This subcomponent acts as a filter to find the resources that satisfy the required capabilities of a task. In this manner, a list of resources that are able to perform the task is created and sent to the Execution Initialization subcomponent in Tasking.

Execution is responsible for managing and monitoring the tasks during their execution process.
- *Task Execution*(6): This subcomponent is responsible for monitoring the execution of current tasks. There are two different situations: 1) the assigned resources are *physical*, so the execution of the task is controlled by this subcomponent, or 2) the assigned resources are *logical*, so the task can be considered as a *new mission* to be sent to the corresponding node for execution (as shown in Fig. 3, in which Node X sends a task to Node Y as a new mission). In the new node, the new mission passes through the same scenario.
- *Task Report Generation*(7): Whenever task execution has finished, this subcomponent generates a report. In the case that the task has executed in another node (i.e., as a mission), its final report comes into this subcomponent. The report contains the results of performing the task and is sent to the Task Management subcomponent. This report will be used in Task Management to determine whether or not the task has effectively finished execution. Furthermore, Task Report Generation sends regular reports to Observation to provide data required for the Information Fusion process.

Fusion is responsible for the synthesis of high level information from low level data and information. Information fusion is a key enabler of Situation Analysis, a process which leads to Situation Awareness.
- *Observation*(8): This is concerned with getting all of the data produced from executing tasks into the same coordinate frame, i.e., aligning them in space and time, and presenting a coherent and consistent picture across several agents. The output of this effort are fused tracks. This is commonly referred to as Level 1 data fusion. Note that this is a processing function that is distinct from sensing.
- *Orientation*(9): Here conclusions are drawn from the tracks by reasoning about the relationships between objects and making inferences about their intentions, and ultimately analyzing the impact of those intentions on others. This stage is referred to as Level 2 data fusion.

Note that Observation and Orientation inside the Fusion component directly correspond to the first two steps of John Boyd's Observe-Orient-Decide-Act (OODA) loop [4]. The Act step is handled by the Execution component, while the rest of the system model is dedicated to the complexity of the Decision step. The OODA concept has wide acceptance in the military R&D community, and the mapping to an Agent concept emphasizes the distributed nature of the tasks.

5.2 Abstract State Machine Representation

We formally describe the detailed design specifications of the subcomponents comprising the system in terms of Abstract State Machine [18] models. These are executable in principle (for experimental studies) using the CoreASM tool environment [19]. ASM code for the Task Management subcomponent is included

in [9]. We present two rules here: (1) Execution Initialization establishes how a resource is assigned using a list of resource candidates provided by Resource Selection, and (2) Task Execution enables self-organization in the system by allowing tasks to be distributed in a recursive manner as missions among subordinate logical resources until they are assigned to a physical resource capable of executing them.

```
ExecutionInitialization(t : TASK) ≡
  let prunedList = {r|r in resourceCandidateList(t) with
                       priority(t) > MaxPriority(r, time(t))} in
    choose r in prunedList with cost(r, t) = MinCost(prunedList, t) do
      if busy(r, time(t)) then
        Release(r, time(t))
      resource(t) := r
      task(r, time(t)) := t
    ifnone do
      resourceCandidateList(t) := undef
      add "NO RESOURCE" to exceptions(t)
  add t to taskPool

TaskExecution(t : TASK) ≡
  let r = resource(t) in
    if PhysicalResource(r) then
      Active(t) := true
      Execute(t, r)
    else                              // r is a logical resource
      add t to missions(r)
```

6 Conclusions

Decision support systems have considerable potential in a range of application fields involving situation analysis processes: the examination of a situation, its elements, and their relations, to provide and maintain a state of situation awareness [20]. We contend that this is a sensible choice for Marine Safety and Security. For the considered scenarios the challenge is to manage complex coordination tasks under uncertain mission requirements, operating in a dynamically changing environment adversely affecting mission success. In the presence of uncertainty and frequent change, dynamic replanning and re-tasking constitute the norm.

The NADIF system concept is characterized by adaptiveness, robustness and scalability and thus embraces change. Building on biologically-inspired computing principles, a self-organizing network of intelligent control agents forms the backbone of the Decision Support and Configuration Management engines. Collaboratively agents decompose and distribute complex operations across the network. By deferring decisions on how to operationalize mission requirements and by localizing decisions on resource alignments within a cluster, this organization enhances flexibility by avoiding the bottleneck of central control structures.

References

1. Natural Resources Canada, The Atlas of Canada – Coastline and Shoreline, http://atlas.nrcan.gc.ca/site/english/learningresources/facts/coastline.html (Visited: May 2010)
2. Marine Traffic Project, http://www.marinetraffic.com/ (Visited: May 2010)
3. Guerlain, S., Brown, D.E., Mastrangelo, C.: Intelligent Decision Support Systems. In: IEEE Intl. Conf. on Systems, Man, and Cybernetics (2000)
4. Li, Z., Leung, H., Valin, P., Wehn, H.: High Level Data Fusion System for CanCoastWatch. In: 10th Intl. Conf. on Information Fusion, Quebec City, Canada (2007)
5. Net-Enabled Adaptive Distributed Information Fusion for Large Volume Surveillance (NADIF), https://wiki.sfu.ca/research/NADIF/ (Visited: May 2010)
6. Royal Canadian Mounted Police, Marine Security Operation Centres (MSOC), http://www.grc-rcmp.gc.ca/mari-port/msoc-cosm-eng.htm (Visited: May 2010)
7. Biology-Inspired Techniques for Self-Organization in Dynamic Networks, http://www.cs.unibo.it/bison/ (Visited: May 2010)
8. Endsley M.R.: Theoretical Underpinnings of Situation Awareness: A Critical Review. In: Situation Awareness Analysis and Measurement, Endsley, M.R., Garland, D.J. (eds.), CRC Press (2000)
9. Glässer, U., Jackson, P., Khalili Araghi, A., Yaghoubi Shahir, H.: Intelligent Decision Support for Marine Safety and Security Operations. In: IEEE Intl. Conf. on Intelligence and Security Informatics, Vancouver, Canada (2010)
10. Arabie, P., Hubert, L.J., De Soete, G. (eds.): Clustering and Classification. World Scientific (1996)
11. Camazine, S., Deneubourg, J.L., Franks, N.R., Sneyd, J., Theraulaz, G., Bonabeau, E.: Self-Organization in Biological Systems, Princeton University Press (2003)
12. Beni, G., Wang, J.: Swarm Intelligence in Cellular Robotic Systems. In: NATO Advanced Workshop on Robots and Biological Systems, Tuscany, Italy (1989)
13. Farahbod, R., Glässer, U., Khalili, A.: A Multi-Layer Network Architecture for Dynamic Resource Configuration & Management of Multiple Mobile Resources in Maritime Surveillance. In: SPIE Symposium on Defense, Security + Sensing: Multisensor, Multisource Information Fusion, Orlando, USA (2009)
14. McCann, C., Pigeau, R.: Clarifying the Concepts of Control and Command. In: 5th Command and Control Research and Technology Symposium (1999)
15. Ghallab, M., Nau, D., Traverso, P.: Automated Planning: Theory and Practice. Morgan Kaufmann (2004)
16. Ayan,N.F., Kuter, U., Yaman, F., Goldman, R.P.: HOTRiDE: Hierarchical Ordered Task Replanning in Dynamic Environments. In: 3rd Workshop on Planning and Plan Execution for Real-World Systems (2007)
17. Mussbacher, G., Amyot, D.: Goal and Scenario Modeling, Analysis, and Transformation with jUCMNav. In: 31st Intl. Conf. on Software Engineering, Companion Volume, Vancouver, Canada (2009)
18. Börger, E., Stärk, R.: Abstract State Machines: A Method for High-Level System Design and Analysis. Springer-Verlag (2003)
19. Farahbod, R., Gervasi, V., Glässer, U.: CoreASM: An Extensible ASM Execution Engine. Fundamenta Informaticae 77(1-2), 71-103, (2007)
20. Bossé, É., Roy, J., Ward, S.: Concepts, Models, and Tools for Information Fusion. Artech House Inc. (2007)