dCIM: An Agent-Based Distributed Common Information Model for Teams of Mobile Robots

Maksims Fiosins[1], Björn Zeise[2], Björn Gernert[3], Sebastian Schildt[3], Paul Fritsche[2], Ramin Safar Manesh[1], Jörg P. Müller[1], Bernardo Wagner[2], and Lars Wolf[3]

[1] Department of Informatics
Technische Universität Clausthal, Clausthal, Germany
[maksims.fiosins|rsm13|joerg.mueller]@tu-clausthal.de

[2] Institute for Systems Engineering
Leibniz Universität Hannover, Hannover, Germany
[zeise|fritsche|wagner]@rts.uni-hannover.de

[3] Institute for Operating Systems and Computer Networks
Technische Universität Braunschweig, Braunschweig, Germany
[gernert|schildt|wolf]@ibr.cs.tu-bs.de

Abstract. Mobile robots are used in a variety of applications including manufacturing, logistics and disaster recovery. In these domains, there is often a requirement to act autonomously, but cooperatively. In this case, autonomous agents and multi-agent systems are useful approaches to represent and execute individual robot decision-making as well as robot coordination and cooperation. A central problem in multi-robot systems is how to store and organize the knowledge individual robots acquire from their sensors or from other robots, and to create an adequate common representation of the environment (including robots' beliefs, goals, and commitments). In this paper, we present the architecture of a Distributed Common Information Model (dCIM), which can be used as a knowledge base for intelligent agents controlling mobile robots. We describe the concept and use of the so-called Information Integration Interface (3I); also the architecture covers methods for reliable communication. Additionally, we depict and validate the advantages of the proposed architecture using the example of high-level plan adaption. Finally, we discuss open research challenges to be faced in the further development of dCIM.

1 Introduction

Over the past few years, mobile service robots have been widely used in different applications, such as logistics [6], manufacturing processes [5], or household [12]. Although in some cases robots can be controlled by operators, the requirement to act autonomously without direct human control is common for these domains. Another important requirement for robots is to operate in teams, which should

form and coordinate autonomously. Thus, there is a trend towards using teams of cooperating robots, as exemplified by KIVA Systems' multi-robot fulfilment system[4]. Another example is the coordinated exploration of large-scale territories by multiple robots [9][13]. Some of the advantages of multi-robot systems are:

- broader sensor coverage and faster information acquisition,
- decentralized/redundant information storage,
- gain in reliability and robustness towards failure, and
- high flexibility in task execution.

The requirements of the above-mentioned applications imply a space-and-time-variant environment model as presented in [7]. In analogy to the human memory, the memory in this environment model is organized in three partitions: sensory memory, short-term memory, and long-term memory. While the raw sensor information is stored in the sensory memory, the short-term memory holds (possibly abstracted) environment information deemed relevant for the robot. Information that is further classified as useful is transferred (and further abstracted) into the long-term memory.

Besides the property of space and time variance, in robotic applications often there is the requirement to model, process and represent arbitrary information. In this field, notable work has been done in the *RoboEarth*[5] project, in which the *KnowRob* knowledge processing system [20] has been developed. KnowRob, which is based on ontologies, offers some relevant design considerations regarding knowledge processing systems. Important considerations mentioned are that those systems must operate effectively and efficiently, must provide the robot with self-knowledge, and need to provide methods for automatically acquiring and integrating information from different (sensor) sources. Although dCIM will not be based on ontologies, all of these considerations will be taken into account in the development process.

Lee [8] distinguishes three main methods for designing information models. First, there is a relational approach, where entities are described and related through attributes and stored in a relational database. Second, there is a functional approach, which represents information flow though a database network. The third approach is object-oriented, which extends the relational approach by encapsulation and modeling of behaviour. The approach presented in this work can be classified as functional, because we developed our architecture information-centered, considering the information evolution from raw data to high abstracted knowledge.

In [4], we presented an approach to autonomous team-based exploration in disaster scenarios. We demonstrated the feasibility of robot teams that perform reconnaissance missions enabling them to share information and to perform dynamic task scheduling. We introduced an architecture called dCIM, on whose detailed description the focus of the presented work lies.

[4] http://www.kivasystems.com/
[5] http://roboearth.org/

In the context of multi-robot systems, the Multi-Agent System (MAS) paradigm has been successfully applied [11][21][14]. On the one hand, MAS provides an intuitive conceptual model and algorithms for multi-robot systems, because it supports (i) the notion of autonomy including restricted local states, local preferences, motivations, and capabilities, (ii) the notion of communication and coordination between autonomous entities; (iii) a unified view of human and automated agents (e.g. mixed human-robot teams); and (iv) models and mechanisms that allow an integrated study of the relationship between individual and collective reasoning and decision-making, including multi-agent planning, game-theoretic models, auctions and market mechanisms, and models of computational social choice (see e.g. [2]). On the other hand, mobile robots provide agents with a physical body, allowing them to observe the environment using physical sensors and to manipulate it using physical actuators.

Robots collect information using different types of sensors. Based on this information and tasks allocated to the robots, they create plans for task execution, which are sequences of actions (commands to actuators). However, they only have a partial and noisy view of the environment. Besides, they often are unable to perform all the tasks by themselves. Therefore, it is beneficial to work in teams.

In teams, the robots are able to coordinate their behavior. This can be done in several ways and on different levels of abstraction: sharing information about the environment, exchanging tasks, coordinating task execution sequence and plans of task execution, coordinating single actions etc. Usually, changes on one of these possible levels requires also changes on other ones. Typically, modifications of information in lower levels of abstraction lead to changes at higher abstraction levels.

In this paper, we present a conceptual information model for teams of mobile robots, which we call dCIM. The architecture describes generic interfaces, an approach to data structure representation, and algorithms allowing cooperation between robots. There are two main contributions in this work: First, a general system layout including a description for information organization is presented. The system is designed to run standalone on every robot. The robots have the ability to exchange information, as long as communication between them is available. The second contribution is the so-called 3I which represents the foundation to deal with a variety of problems in information fusion, such as information synchronization, information tracking and information incest. To show the benefits in 3I some experiments are made for evaluation. At the end of this paper, upcoming challenges related to dCIM are outlined.

2 dCIM Architecture

A major problem in designing intelligent agents is the construction of an adequate information model, which supports storing information about the environment, processing it and providing the decision making module with sufficient information for efficient behavior. Another important aspect is the information

exchange between the agents, which allows the agents to have more exact view of the environment as well as to coordinate their behavior.

There are some attempts to describe generic information models for intelligent agents. For example, [3] describes a generic model, which supports scheduling, resource allocation and actions based on dynamic temporal infprmation. However, this model is mostly oriented to master-slave robot architectures. Another approach is the Agent Academy architecture [19], a generic platform for design and deployment of intelligent agents, which combines the information storage, information processing and agent training. However, the Agent Academy architecture is mostly oriented to the local models of agents, not on information exchange.

In the robotics domain, there are specific requirements for information models, which are not usually covered by generic architectures. The first one is the fact that one of the central data structures in robotics is a map of the environment. The map can be more or less detailed depending on the quality of sensors. The amount of raw data from sensors is comparatively big, which makes its processing and conversion to more compact but less detailed formats necessary. For that reason it is sensible to consider a hierarchical information model structure. Due to the fact that maps of the environment can be very dynamic and change over time, a layer-based approach is applicable. The second specific requirement is the need of information exchange and cooperation. In this context, the information exchange on all levels of the information model may be necessary. Therefore, the development of a homogeneous but customizable information model, which may operate on heterogeneous robots, is needed. Additionally to these two requirements, the information acquired by sensors and received from other robots is highly dynamic, partial and uncertain, which should be taken into account in the design of the model.

dCIM is the distributed knowledge base of the overall robotic system. Every robot maintains its own dCIM-instance. One single dCIM-instance offers a best-effort global world view to only the robot itself, while the combination of all instances represents the best global view of the whole system. With today methods of information fusion, processing information of one robot's sensors is a well-studied problem. The difficulty in the presented approach lies in the arbitrary number of robots whose information shall be merged.

When designing the dCIM layout and information model, the following main questions need to be answered:

- How does the dCIM architecture look like (Section 2.1)?
- What are the principles in storing information of a robot in a dCIM-instance (Section 2.2)?
- What kind of information should be exchanged between the robots and how to deal with conflicts in the information when two robots communicate (Section 2.3)?
- How to avoid faulty communication between robots (Section 2.4)?
- How is information processed in a dCIM-instance and how are the individual robots' plans adapted depending on updated dCIM information (Section 2.5)?

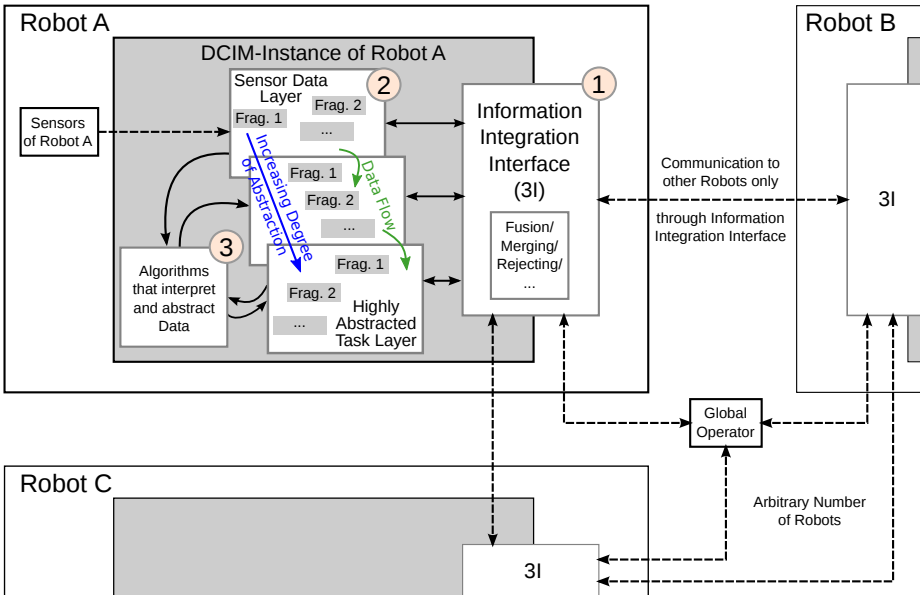


Fig. 1: dCIM scheme

In the following sections, we will provide an overview to dCIM. Accordingly, we will first present the general system layout. After that, an approach for creating information models as well as the communication interface called 3I will be explained.

2.1 General Layout

The main focus in the development of dCIM lies on decentralization. For that purpose, there is a standalone dCIM-instance on every single robot. dCIM can be seen as the knowledge base of the robotic system. The general layout of dCIM is depicted in Figure 1. A dCIM-instance consists of three main parts: (1) the communication interface that incorporates information from other robots into the own robot's information storage, (2) the layers for storing information and (3) several algorithms that interpret and abstract information.

The information of a robot is composed of two different kinds of resources. First of all, a dCIM-instance relies on its own robot's sensor inputs. In large environments explored by only a few robots it is possible that there is only a small number of rendezvous (the robots that meet and communicate with each other). For that reason, the own sensor inputs of a robot are the primary source of information for the respective dCIM-instance. As can be seen from the scheme there is a second source of information for a dCIM-instance: a robot can receive external information from an arbitrary number of other robots in communication range. This way, a robot is able to gather information in addition to its own

sensor data. To exchange information, the robots use the so-called 3I. This is a communication interface that operates on every single abstraction layer and is the only way to exchange information between the robots and the (human) operator. In this manner, it is possible to approach information fusion problems such as information incest. 3I will be explained in detail in Section 2.3.

The information storage in a dCIM-instance divides into several layers. These layers differ in their level of abstraction, which means that there will be at least two layers – one with raw sensor data and one with highly abstracted and interpreted information. Between those layers there can be as many other layers as necessary. Since the information layers can be defined very individually, nearly every type of information can be stored in a dCIM-instance. Since dCIM is designed to be the knowledge base for mobile robots, the information stored will be of a kind that is useful for robot navigation, i.e. odometry and laser scan data, occupancy grid maps or trajectories for path planning. The information flow between the different layers is unidirectional (from low to high abstraction level).

In order to facilitate information collection and distribution, we split the organization structure into two dimensions. The first dimension is represented by the use of different information layers. Each layer of information supports a specified level of abstraction; the information of the next level can be inferred from the previous one using corresponding algorithms. The second dimension is represented by the use of information fragments. On each layer the information is divided into fragments, which are usually predefined at design time. They represent spatial or logical parts of the information, which are independent from each other.

The information flow in the storage layers of a dCIM-instance is accomplished by algorithms that interpret and abstract information. The input information of these algorithms is always of a lower abstraction level than the output information. One typical example of such an algorithm would be a Simultaneous Localization and Mapping (SLAM) algorithm with odometry and laser scan data as inputs and e.g. occupancy grid maps as output.

The decentralized nature of information requires a corresponding decentralized organization in dCIM. In multi-agent and robotic research, attention is usually paid to hierarchical organization of actions. In our approach, we will concentrate on hierarchical organization of information, and couple it with hierarchical organization of actions.

Formally, the hierarchical organization of information can be defined as follows. Let $A = \{a^1, a^2, \ldots, a^N\}$ be a set of N agents. Each agent a^j owns a dataset D^j, which represents information that is currently available to it. D^j includes all information available to a^j, such as maps of the environment, locations of objects and other agents etc., including all layers and fragments.

Let $D^{j,l}$ be the information of the agent a^j on l-th level of hierarchy. We suppose that it consists of fragments, and define $D^{j,l,q}$ the q-th fragment of the information on l-th layer.

As we already mentioned, the information on the next layer can be obtained from the previous one using interpreting algorithms. Additionally, the fragments on higher abstracted layers depend only on the part of fragments on the previous layer. This provides us with a networked structure for information organization. Let $I^{j,l,q}$ be a set of fragments on the $l-1$-th layer, on which the fragment $D^{j,l,q}$ depends. Let $U^{j,l,q}$ be a function, which allows to calculate the fragment of the next layer from the previous one. Then the update of the layer information looks like

$$D^{j,l,q} = U^{j,l,q}(D^{j,l-1,q'}|D^{j,l-1,q'} \in I^{j,l,q}).$$

As we already mentioned, each robot is equipped with sensors, allowing it to sense the environment. Let us denote $O^j = \{o_1^j, o_2^j, \ldots\}$ an set of observations about fragments $1, 2, \ldots$, received by robot j. Note that O^j is a set of low-level, raw observations which will later be transformed to highly abstract knowledge of the robot.

When a robot receives an observation O^j, it updates its fragments on the lowest level $D^{j,1}$. For this purpose, there is an update function $U^{j,1,q}$, which updates information depending on a new observation O^j:

$$D^{j,1,q} = U^{j,1,q}(D^{j,1,q}, o_q^j).$$

One example of such an interpretation and abstraction method is SLAM, which produces grid maps and information about the robot's position out of raw sensor data. However, $U^{j,1,q}$ should take into account trustworthiness of new information O^j. For this purpose, the old information and the new observation are usually processed by a trust function, which determines possible conflicts:

$$U^{j,1,q}(D^{j,1,q}, o_q^j) = MLE^{j,q}(TR^{j,q}(D^{j,1,q}, o_q^j).$$

The update function $U(\cdot)$ can have different forms and is defined at design time. If some redundant information is stored at lower layer, the next layers can be estimated using hierarchical resampling [1].

2.2 Data Structure

As mentioned before, in dCIM information is stored in several layers. The purpose of these layers is to store and organize the information. The composition of all the layers of a dCIM-instance is therefore a information representation in the sense of a replacement of the original environment (see [18, p. 524]). There are many different ways to organize information in a data storage system. It is common to use one of the Knowledge Organization Systems (KOSs) *classifications*, *thesauri* or *ontologies* (as described in [18, pp. 635]) for this purpose. These KOSs have a hierarchical structure in common, but differ in the level of complexity. Complexity in this context means that there exist different kinds of relations between the information fragments in the KOS. Thesauri and ontologies for instance offer more sophisticated relations than simple classifications. Those complex relations are especially useful in systems that provide a high level of

artificial intelligence (e.g. automated reasoning). Since the robotic systems for which dCIM is designed will not need such complex relations, the data structures for the layers will be oriented towards classification-like KOSs. For this reason, information will be stored in a textual representation, namely the XML format.

Listing 1.1: Exemplary XML structure encoding an occupancy grid map

```
<OccupanyGridMap>
  <timestamp>100</timestamp>
  <source>robot01</source>
  <resolution>0.1</resolution>
  <width>100</width>
  <height>100</height>
  <origin>
    <x>451</x>
    <y>729</y>
    <z>1042</z>
    <roll>1.1</roll>
    <pitch>1.2</pitch>
    <yaw>1.3</yaw>
  </origin>
  <grid>
    <occ cell="1">50</occ>
    <occ cell="2">0</occ>
    <occ cell="3">100</occ>
  </grid>
</OccupancyGridMap>
```

One has to keep in mind that the data fragments in dCIM can be arbitrarily defined. We do not want to predefine any data structures in this paper, because dCIM has to be seen as the paradigm for an information model. That means that – as long as the other application-specific parts of the system (i.e. the 3I and the interpreting algorithms) can handle the user-defined data structures – one only has to consider using XML syntax and to pay attention to the presence of required meta data and payload. Listing 1.1 shows an exemplary dCIM information fragment – an occupancy grid map. While the first elements of the data tree represent classical meta data (which is very important for the 3I in order to compare data fragments), the actual data (payload) is stored in the $< grid >$ respectively the $< occ >$ elements. In dCIM, information of every abstraction level is stored and transmitted in a XML format as seen in the listing. dCIM allows to create information fragments for every desired purpose, as long as the 3I and the algorithms processing the information can handle the data structures.

2.3 Information Integration Interface

In decentralized control architecture, no centralized conflict resolution rules can be applied. Moreover, each dCIM-instance needs to decide by itself which information it is going to accept and how to fuse and merge available information.

Two or more dCIM-instances are able to exchange sensor data, maps, tasks or any other information that can be extracted from any layer of the dCIM. Therefore, dCIM fragments, which contain the information, need to be processed in the 3I. As well as for the data structures, we do not want to predefine any specific rules of the 3I. Instead, we give some conceptual advise for the creation of such rules.

In general, the acquisition of information in a dCIM-instance is not as predictable as in the case of a centralized architecture. In case of a decentralized system, there is always the chance of correct information loss, because dCIM needs to decide by itself what is a relevant and correct information or what is wrong; hence, there is an evolutionary development inside the system. The difficulty of designing a dCIM-instance is the consideration of all relevant influences, systematizing them into layer based dynamic data structures. The kind of communication, the physical model of the robots and the environment need to be considered when designing the 3I.

The main task of the 3I is to integrate external information (information from other robots) into the own dCIM-instance. For that purpose, there is external communication between different robots on the one hand and internal communication between the 3I and the internal information layers of one particular dCIM-instance on the other hand (as seen in Figure 1). New information (dCIM fragments) are gathered through external communication. These fragments include meta data that can be used to establish a contextual connection between the new fragments and information that is already part of some dCIM information layer. Depending on parameters like acquisition time/location, origin or reliability the 3I has the ability to fuse, merge or reject the new information.

2.4 Communication

Many technologies can be integrated into robots to facilitate exchange of information in a team. A common low-cost and high-bandwidth solution is WiFi. Depending on the operating environment and requirements, other technologies are possible: For very small energy-constrained robots the WSN technology such as IEEE 802.15.4 could be an alternative. In dome deployments cellular networks might be an alternative for long range links.

However, independent from the used radio technologies the challenges when communicating in such dynamic scenarios are always the same: Continuous connectivity between all entities can not be guaranteed. Robots may move out of comunication range. The network might get divided into several distinct connected subsets. The communication stack needs to deal with these conditions and provide the best possible delivery of information, while some inevitable issues such as inconsistency of information and varying levels of available information need to be dealt with at the dCIM level.

To optimize information proliferation in a network with intermittent connectivity the system adopts a store-carry-and-forward technique: Instead of finding a stable route between arbitrary endpoints in the network, communication participants will store any received information until a new contact comes into

range. Thus, the network which runs dCIM is effectively a Delay Tolerant Network (DTN). The DTN abstracts the network discontinuities, making sure that data will arrive eventually even under high network strain. By using a DTN software stack the application does not need to deal with disconnections and broken links. However, even with a DTN data can not always be delivered: If the network is partitioned and it is physically impossible to transfer data to specific nodes within a predefined time, the dCIM needs to handle this situation.

Communication between several intelligent robots shall be possible on every layer of data abstraction. To share information with other robots, we define a send function $M_{j,i,l}(t) = Send_j(D^{j,l}, i, t)$, which decides whether and which information the robot j should send from its information layer l to robot i at time t. There are several important variants of the send function:

- The function may be independent on the parameter i. In this case we have a broadcast function, which sends equal information to all robots.
- The decision, whether to send the information, is dependent on parameter t and not on $D^{j,l}(t)$. In this case we have information sent periodically.
- The decision, whether to send the information, depends on the measure of difference $||D^{j,l}(t) - S^{j,l}(t-1)||$. That is why information is sent only in the case of changes.
- The message $M_{j,i,l}(t)$ is usually split into fragments, which correspond to the fragments of data.

All of these variants have to be covered by the 3I. Receiving the messages $M_{i,j}(t)$, robot j can integrate them into the information $D^{j,l}(t)$. For this purpose, an update function $U_j^{msg}(s, i, t)$ updates the information:

$$D^{j,l}(t+1) = U_j^{msg}(D^{j,l}(t), M_{i,j,l}(t))$$

The update function U_j^{msg} also combines maximum likelihood estimation (MLE) with trust in the message $M_{i,j}$:

$$U_j^{msg}(S^j(t), M_{i,j}(t)) = MLE^{msg}(TR^{msg}(S^j(t), O^j(t), i)).$$

Note that the trust function TR^{msg} includes an additional component – the index of robot i. This is because there can be different trusts to different robots.

A possibility to improve network quality is allowing the distribution of additional networking elements by the robots themselves as a means to build a more dense network [16]. A standard way of implementing a DTN is the Bundle Protocol (BP) which is specified in RFC5050 [17]. Proven Bundle Protocol implementations suitable for running on robots are available. For example IBR-DTN [15] supports different communication technologies such as WiFi or IEEE 802.15.4 even in heterogeneous environments and runs on a variety of different hardware including embedded systems.

2.5 Task allocation and scheduling

In this section we will describe the interaction between different dCIM modules. We show how raw sensor data is processed in dCIM in order to obtain high-level tasks. Additionally, the principle of the 3I will be clarified.

The robots have a global task pool T. All tasks in this pool should be executed by the robots. In general, the robots should perform three interconnected activities in order to execute them:

- Task allocation/re-allocation
- Task scheduling/sequence planning
- Task planning/actions

The task allocation step is necessary to assign tasks to robots, i.e. to find subsets $T^j \subset T$ such that $\bigcap T^j = T$. That means that each task is covered by at least one robot. The costs of the subsets T^j depend on the sequence of task execution by each individual robot, so the cost $C(T^j)$ of the subset T^j is defined by the permutation $\pi(T^j)$ and individual costs of tasks $C^{ind}(t_k), t_k \in T^j$. Also these costs depend on datasets D^j, available to the robot j:

$$C^j(T^j) = C^j(\pi(T^j), C^{ind}(t_k), D^j | t_k \in T^j).$$

Finding the optimal permutation is called task scheduling; finding the optimal individual costs is connected with determining the action sequence for an individual task and is called task planning. Note that these tasks can be solved on different abstraction levels $D^{j,l}$. We used auction approach for this purpose; the details are described in [4].

Note that tasks from the set T are represented as high-level tasks. They may be formulated for example as "bring X from A to B". In order to calculate costs of such tasks and then to execute them, hierarchy of tasks is necessary. Such hierarchy represents tasks from higher level of abstraction until commands to physical motors.

It is convenient to coordinate hierarchy of information with hierarchy of tasks. Let us suppose that there are different levels of actions. Let $T^{j,l} = \{t_k^{j,l}\}$ be a set of possible actions of l-th level of j-th robot. We suppose that the cost of l-th level actions depend on information of l-th level and costs of tasks of $l-1$-th level, which are included into l-th level task (define this set as $J(t_k^{j,l})$).

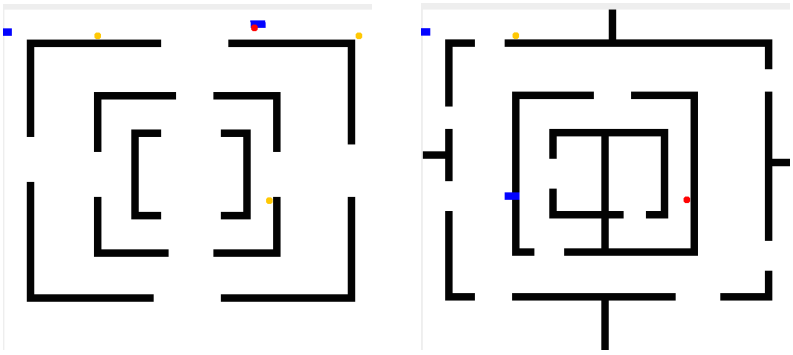
$$C^j(t_k^{j,l}) = C^{j,l}(t_m^{j,l-1}, D^{j,l} | t^{j,l} \in J(t_k^{j,l})).$$

3 Experiments

In order to illustrate the concepts of dCIM, we perform the following experiments. We consider a team of mobile robots, which act in a partially known environment. This means that the map of the environment, which is available to the robots (Figure 2a), differs from the actual map (Figure 2b).

In this experiment, the robots have to transport items (boxes) between particular locations in the environment. They receive messages from the boxes requesting a transport. The robots then decide decentrally, which one of them is going to perform the task.

During their operations the robots scan the real environment, detecting inaccuracy in their maps. Detected differences between the prior known map and



(a) Map available to the robots (b) True map of the environment

Fig. 2: Maps used in the experiments

the actual environment are transferred to other robots, which can update their plans accordingly. As a measure of the system's efficiency we calculate a total distance (in this case in abstract "steps" unit), which the robots have to travel to complete all the tasks. The experiment is repeated several times with the same map and an average distance is calculated.

Our first experiment demonstrates how the number of steps increases, if the robots have an inaccurate map of the environment. The results presented in Figure 3 demonstrate that – for the given environment – the robots with an inaccurate map require a 5 - 15 % higher travel distance.

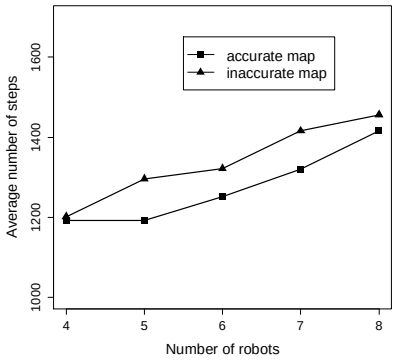
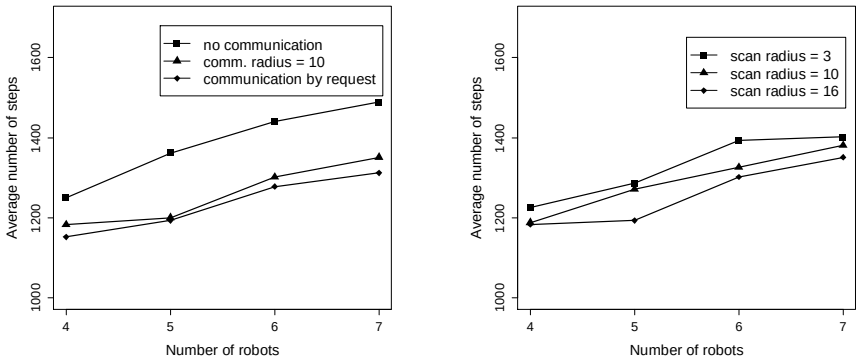


Fig. 3: Average travel distance depending on number of robots with incorrect (above) and correct (below) map of the environment

Our next experiment demonstrates how the communication influences the system's efficiency. Figure 4a depicts the average number of steps depending on the communication radius. Looking at it, we see that it is possible to reduce the number of steps to 5 - 15 % – and achieve the same efficiency as by correct maps.



(a) Average travel distance depending on the communication radius

(b) Average travel distance depending on the scan radius

Fig. 4: Results of the investigation of communication/scan radius influence

The last set of experiments allows us to understand the role of information collection for the system's efficiency. Figure 4b demonstrates how the scanning radius influences the travel distance. Here we see only a decrease of 3 - 5 % of traveling distance.

4 Open Challenges

In the current stage of development, dCIM offers conflict resolution only on a highly abstracted task layer. One upcoming challenge is to create an extension to the 3I, which is capable of solving conflicts on other (especially low-level) information layers. It is assumed, that these algorithms will have a performance-disturbing influence on the 3I since much more information has to be processed at low-level information layers.

Another topic that – in this work – only has been mentioned is the development of a method to merge dCIM-instances of different robots. This can be necessary in situations where the robots' communication fails or where the robots explore a very large environment travelling through the same areas at different time not getting into communication range. Such a fusion method could provide the best global view of the whole system/environment including as much information of all deployed robots.

As we described in section 2.2, in dCIM it shall be possible to define own data structures. In order to make dCIM more compatible for different kinds of robots, it is sensible to constrain this feature for some information. For very common information (e.g. map data) it is desirable to use standardized data structures. First efforts in this area were made by the IEEE Robotics and Automation Society Standing Committee for Standards Activities (RAS-SCSA), especially the Map Data Representation Working Group (MDR WG[6]). The standard aims to provide specifications for representing 2D metric and topological maps [10]. When the standard is available it should be reviewed whether it could be useful in dCIM or not.

5 Conclusions

In this work, we presented a software architecture called dCIM that forms the knowledge base for teams of mobile robots in various applications. We outlined the principle layout of dCIM and explained why XML-like data structures are most suitable for the architecture. Additionally, we introduced an interface to resolve conflicts in data integration that arise with e.g faulty communication or ambiguous observations.

We further explained the path data takes through a variable number of data layers, abstracting data from low-level to high-level. The influence of changes in the environment, which are propagated through dCIM and the 3I between different intelligent robots, was evaluated in several experiments. These experiments confirmed the usefulness of the proposed architecture.

Due to the fact that dCIM is still in an early stage of development, varieties of extensions and improvements are possible. Some of these open challenges were described within this paper.

References

1. Andronov, A., Fiosins, M.: Applications of resampling approach to statistical problems of logical systems. Acta et Commentationes Universitatis Tartuensis de Mathematica **8**, 63–72 (2004)
2. Brandt, F., Conitzer, V., Endriss, U.: Computational social choice. In: G. Weiss (ed.) Multiagent Systems, 2nd edn., pp. 213–283. MIT Press (2013)
3. Chang, A.: A computational data model of intelligent agents with time-varying resources. In: J.J.J.H. Park, L. Barolli, F. Xhafa, H.Y. Jeong (eds.) Information Technology Convergence, *Lecture Notes in Electrical Engineering*, vol. 253, pp. 783–791. Springer Netherlands (2013)
4. Gernert, B., Schildt, S., Wolf, L., Zeise, B., Fritsche, P., Wagner, B., Fiosins, M., Manesh, R., Müller, J.P.: An interdisciplinary approach to autonomous team-based exploration in disaster scenarios. In: 12th IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR2014), pp. 1–8. IEEE (2014)

[6] http://ieee-sa.centraldesktop.com/1873workinggrouppublic/

5. Giordani, S., Lujak, M., Martinelli, F.: A distributed multi-agent production planning and scheduling framework for mobile robots. Computers & Industrial Engineering **64**(1), 19 – 30 (2013)
6. Hentschel, M., Lecking, D., Wagner, B.: Deterministic path planning and navigation for an autonomous fork lift truck. In: 6th IFAC Symposium on Intelligent Autonomous Vehicles (IAV), pp. 102–107. IFAC (2007)
7. Hentschel, M., Wagner, B.: An adaptive memory model for long-term navigation of autonomous mobile robots. In: Journal of Robotics, vol. 2011. Hindawi Publishing Corporation (2011)
8. Lee, Y.T.: Information modeling: From design to implementation. Proceedings of Second World Manufacturing Congress (1999)
9. Liu, L., Shell, D.: Large-scale multi-robot task allocation via dynamic partitioning and distribution. Autonomous Robots **33**(3), 291–307 (2012)
10. Madhavan, R., Yu, W., Schlenoff, C., Prestes, E., Amigoni, F.: Draft standards development of two working groups [industrial activities]. IEEE Robotics & Automation Magazine **21**(3), 20–23 (2014). DOI 10.1109/MRA.2014.2334971. URL `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6894718`
11. Müller, J.P.: The Design of Intelligent Agents, *Lecture Notes in Artificial Intelligence*, vol. 1177. Springer-Verlag (1996)
12. Nakauchi, Y., Fukuda, T., Noguchi, K., Matsubara, T.: Intelligent kitchen: cooking support by lcd and mobile robot with ic-labeled objects. In: Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on, pp. 1911–1916 (2005)
13. Reid, R., Braunl, T.: Large-scale multi-robot mapping in magic 2010. In: Robotics, Automation and Mechatronics (RAM), 2011 IEEE Conference on, pp. 239–244 (2011)
14. Scerri, P.: Team oriented plans and robot swarms (extended abstract). In: Human Control of Bioinspired Swarms, Papers from the 2012 AAAI Fall Symposium, Arlington, Virginia, USA, November 2-4, 2012, vol. Technical Report FS-12-04, pp. 51–53. AAAI (2012)
15. Schildt, S., Morgenroth, J., Pöttner, W.B., Wolf, L.: IBR-DTN: A lightweight, modular and highly portable Bundle Protocol implementation. Electronic Communications of the EASST **37**, 1–11 (2011)
16. Schildt, S., Rottmann, S., Wolf, L.: Communication architecture, challenges and paradigms for robotic firefighters. In: Proceedings of the 5th Extreme Conference of Communication (ExtremeCom 2013). Thorsmork, Iceland (2013). URL `http://www.ibr.cs.tu-bs.de/papers/schildt-extremecom2013.pdf`
17. Scott, K., Burleigh, S.: Bundle Protocol Specification. RFC 5050 (Experimental) (2007). URL `http://www.ietf.org/rfc/rfc5050.txt`
18. Stock, W.G.: Handbook of information science. De Gruyter Saur, Berlin, Boston (2013)
19. Symeonidis, A.L., Kehagias, D., Mitkas, P.A.: Intelligent policy recommendations on enterprise resource planning by the use of agent technology and data mining techniques. Expert Syst. Appl. **25**(4), 589–602 (2003)
20. Tenorth, M., Beetz, M.: KnowRob: A knowledge processing infrastructure for cognition-enabled robots. The International Journal of Robotics Research **32**(5), 566–590 (2013)
21. Velagapudi, P., Sycara, K., Scerri, P.: Decentralized prioritized planning in large multirobot teams. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4603–4609. IEEE (2010)