18th ICCRTS
"C2 in Underdeveloped, Degraded and Denied Operational Environments"

Interacting with Multi-Robot Systems using BML

Topics
Autonomy
Experimentation, Metrics, and Analysis
Data, Information and Knowledge

Names of Authors
Thomas Remmersmann, Alexander Tiderko, Dr. Ulrich Schade
Fraunhofer Institute for Communication, Information Processing and Ergonomics FKIE

Point of Contact
Thomas Remmersmann
Fraunhofer Straße 20, 53343 Wachtberg, Germany
+49 228 9435 752
Thomas.Remmersmann@fkie.fraunhofer.de

ABSTRACT

Using a multi-robot system in military operations poses many control problems. The ability to formulate clear and unambiguous commands for the robots thus is extremely desirable. As numerous autonomous functions have been developed both for single robots and for multi-robot systems, we want the operator to express just what needs to be done, whereas the robots have to figure out how to do it. Therefore, we use the Battle Management Language (BML) to command multi-robot systems. As BML has been developed by the military research community, it is – as required – designed to express short, unambiguous orders readable by both humans and machines. This article presents an overview over the BML commands that can be given to the robots as well as an overview over the BML reports generated by robots and presented to the operator.

1. Introduction

There are many operations in which a multi-robot system (MRS) can be deployed to support the human forces, e.g., for reconnaissance tasks. Controlling a MRS in operations, however, is a complex and demanding task, especially if the MRS in question has to be controlled by a single operator in order to free her fellow soldiers for other tasks. The operator can be disburdened by giving the robots some autonomy. She then would be empowered to give orders on a more abstract level and to let the robots handle details themselves. However, this raises the question as to how those tasks assignments can be defined and exchanged. To formulate machine and human readable commands, NATO developed Battle Management Language (BML) in its research groups MSG-48 [1] and MSG-85 [2, 3]. This formal and controlled language allows expressing orders and reports in a clear and unambiguous way. BML has been developed to exchange communications between C2 systems and simulation systems. In addition, the use of BML for the interaction with robots has been considered early [4]. As a consequence, Germany's armed forces (the Bundeswehr) supported extensive experiences in controlling robots by giving BML orders which will be presented and discussed in this paper.

The remainder of the article is organized as follows: First, we give a brief overview of the communication architecture used in order to give BML orders to the MRS and in order to receive reports back from the robots (section 2). Section 3 is about BML. First, we describe the grammar of BML orders in general. Then, we will discuss the specific tasks that can be assigned to the MRS (sub section 3.1). Sub section 3.2 presents a discussion of the BML reports used to send the information collected by the robots back to the C2 system. The article ends with a conclusion section 4.

2. Communication Architecture

In order to command a MRS we developed a specific BML-GUI. The GUI supports the formulation of BML orders and displays the content of the robots' reports. It has its origins in the C2LG-GUI we originally developed for the NATO MSGs to command simulated units, cf. [5], section 4.2, [6], section 8.2.

In our experiments and showcases, we used multi-robot systems that included aerial as well as ground vehicles. The aerial vehicles were the so-called "UAV Psyche 1000", MD4-1000 drones, developed by Micro-Drones and modified by Siegen University. The ground vehicles include the so-called "RTS-HANNA", a car like UGV based on an off-the-shelf Kawasaki

Mule 3010 Diesel chassis which had been modified and adjusted by Hannover University, a quad bike based UGV called AMOR built by the university of Siegen, and two smaller UGVs modified and adjusted by Fraunhofer FKIE.

As the different types of vehicles use different operating systems, middlewares, and communication protocols, it was decided to use the Robot Operating System (ROS) [7] developed and maintained by Willow Garage, as communication standard among the vehicles. ROS offers well-defined data types for most kinds of data. It is open source and hence free available, and it has a constantly growing community contributing to the software repository.

The integration of ROS into the corresponding middlewares is not in the focus of this paper, but see [8] for detailed information on the integration topic as well as for details about some vehicle used in the experiments. In order to connect the vehicles' ROS systems with the BML-GUI, a ROS component called BMLConnector has been developed in Python. The BMLConnector receives BML commands and translates them to a defined ROS message, called BmlTask. The BmlTask message is then published as a ROS topic. For position reports, the respective ROS messages are translated back to corresponding BML position reports which then are sent to the BML-GUI so that the robots' positions can be displayed on the GUI's map. The robots also report about their operational state, the task status and detection of suspected enemy units. Other sensor data, in particular binary data such as images, video, or lidar data, is currently not directly translated into BML. The respective ROS messages are converted to an XML format instead and sent to another GUI, where they are displayed. A schema of the communication is shown in Fig. 1.

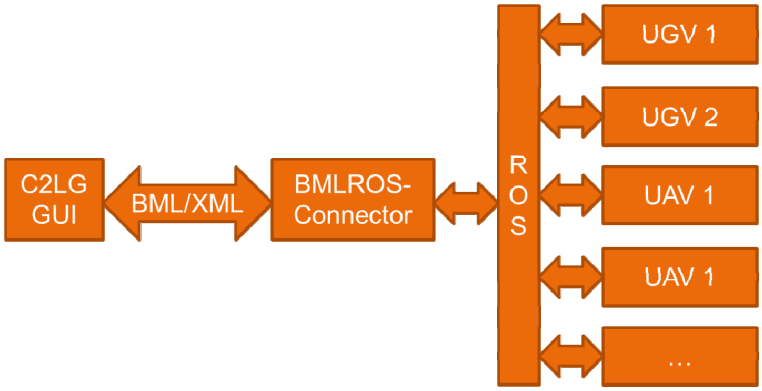

Figure 1. Communication between C2LG GUI and robots and formats used.

3. BML

Battle Management Language (BML) is an artificial, unambiguous, human-readable language and open standard used to express and to exchange orders, reports and requests among Command and Control systems (C2 systems), simulation systems and real units. In addition, BML also may be used to interact with robotic forces. In short, BML allows C2 systems and their users to interact with robot systems in the same way as with real units or with units simulated in simulation systems.

BML must be unambiguous to allow automatic processing. This unambiguousness is not self-evident for a language. For example, in natural English, the lexical term *bark* can refer to the sound a dog produces or to the skin of a tree. The interpretation of such ambiguous terms depends on the situational context and on the world knowledge of the listener. As such,

ambiguity can be handled by human (mostly) but not by artificial information processing systems.

In order to be unambiguous, BML has been designed as a formal language. A formal language is the set of all sentences generated by a formal grammar, cf. [9]. A formal grammar consists of a lexicon (the words of the language) and a set of rules (how to combine the words). In the case of BML, this grammar is the Command and Control Lexical Grammar (C2LG) [10, 11, 12]. To be more precise, BML's lexicon contains the attributes and values provided by the Joint Consultation Command and Control Information Exchange Data Model (JC3IEDM) (see https://mipsite.lsec.dnd.ca/ or [13]). This set of rules has been developed based upon the doctrines of ordering and reporting, e.g., STANAG 2014, and incorporates the idea of the 5Ws (Who, What, Where, When, Why) for individual BML expressions. With respect to orders, the central grammatical rules are those that assign a task to a unit. Thus, these rules are centered on the task expression (the What). Rule form (1) illustrates how these "tasking" rules are constructed. They consist of a task verb (TaskVerb) such as "advance", a reference to the one assigning the task (Tasker), a reference to the one who has to execute the task (Taskee) – in our case robots –, and, in some cases, depending of the type of task, a reference to something that is affected by the task. This is either an object (Affected) or another task (Action). In addition, a task assignment includes spatial and temporal constraints (Where, Start-When, and End-When), modifiers (Mod), and a reason why the task has to be executed (Why). The task assignment ends with a label that can be used in other expressions to refer to that task assignment.

(1) OB → TaskVerb Tasker Taskee (Affected|Action)
Where Start-When (End-When) Mod Why Label

3.1. Orders

In this sub section, we will present the types of BML orders we used to command the MRS. The operator can use the BML-GUI (see Fig. 2) to express these orders which cover the following types of tasks: "move," "patrol," "reconnaissance," and "observe." In addition there is the order by which an assigned task can be cancelled and two types of emergency commands, "stop" and "return to base." Figure 1 shows the GUI. The GUI has icons for each type of task in order to facilitate and to simplify expressing the orders. These icons are placed in the panel on the left side.

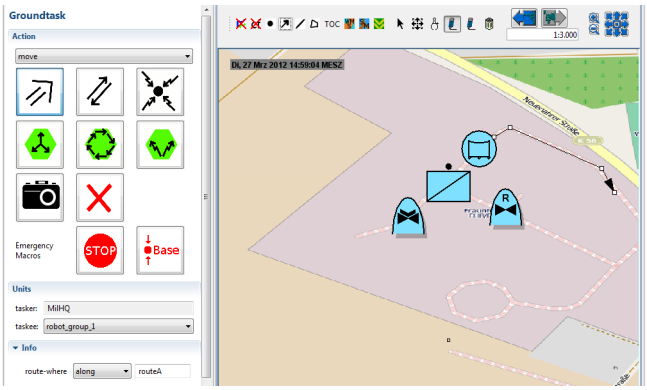

Figure 2: The figure shows the BML-GUI. Here, the action "move" had been selected by pressing the left upper button. Then the taskee "robot_group_1" was selected from the fields below the buttons. Finally, a route was created on the map and assigned as Where parameter ("routeA") to the move task.

Receiving a command, the MRS has to interpret it and must react accordingly. For all ordered tasks those reactions are constrained by principles that hold in general. For example, the execution of all movements is constrained by collision-avoidance systems which are implemented in the local navigation of each robot. This prevents the robots from colliding with each other and enables them to avoid other objects as well. As a consequence, it also helps them to reach their objective even if the direct way is blocked. Another constrain is that the robots do change detection to report about moving objects. This is done using 3D laser scanner data like the UGVs have or using images comparison on the UAVs.

In principle, the commands are sent to the MRS and interpreted accordingly. In particular, the commands are transformed into individual task assignments for every robot of the MRS for executing. If all robots have completed their individual tasks, the group task is considered completed.

For example a "move" command is split up as follows: first, the leading UGV receives its individual moving task which copies the "Where" part of the group's moving command. Each other UGV then receives a command to follow another robot, the second robot has to follow the lead robot and so on. This leads to a convoy of the UGVs. The UAVs continuously get individual move commands with the current position of the leading UGV as destination. As a consequence, the UAVs coordinate their movements among each other and fly in formation over the leading UGV.

An "observe" task has a point to observe as "Affected" parameter. The execution of an observe tasks start with a move, like mentioned above, towards this target. When the robots reach the target area each robot receives an individual observe task which includes a location where it should move to and the target object it should look at. One UAV receives a task to stay above the target, the other UAV is commanded to fly circles around the target.

A "patrol" task is executed by sending one UGV to each end of the patrol route with a move command. One of the other UGVs then is commanded to move back and forth the route and the UAVs are also flying along the given route.

The "recce" command allows to recce larger areas. The UGVs split up the previously known roadmap and move along the roads of the area in question to check if they are passable. The UAVs also split up the area and each UAV takes pictures of its part of area.

Other orders, we implemented, are "image intelligence gathering" to order to make a certain robot taking a picture with high resolution and "disengage" to cancel a previously given command.

The order of battle of the Robots, which for example determines who will be the lead robot in a move, is defined in an MSDL File. The GUI can read this file in order to show the user which robots are under his command.

3.2. Reports

We used BML reports to send information from the robots to the BML-GUI. Position information, the operational status, the task status and information about detected units are directly transformed into BML reports. The robots also report binary data such as images, video, or lidar data. These data cannot be translated into BML because BML is limited to textual messages. Instead, the data are converted into another XML format. After conversion,

it is sent to the GUI, and can be stored in a central data base so that other (superordinate) C2 systems can make use of it.

Since we would like to report about this data in a BML we defined a BML message which reports the link to the data in the database to the user of the MRS. This message is sent to the GUI. The user then can call the link so that the data is displayed in side panels of the GUI.

The following sections provide an overview of the types of BML reports that are used.

Positions of the robots (position reports)

In order to keep the positions of the robots on a map up-to-date the BmlConnector creates position reports at regular time intervals.

Example: *report own position Robot_1 at [Point A] at now eyeball completely reliable RPTFCT positionreport-12345;*

Operational status (status-gen reports)

Operational status reports indicate which robots are currently able to receive commands. We use the Code "OPR" (Operational) if a unit is in autonomous mode and "TNOPS" (Temporally not Operational) if the Robots are in any other mode, e.g., if they are controlled by joystick.

Example: *report own status-gen Robot_1 OPR at now eyeball completely reliable RPTFCT statusreport-13456;*

Task status (task reports)

The robots report about the current status of the task they execute. This includes "Tasked", "In Progress", "Complete" and "Abort". Task status reports are important since they give the operator feedback to which degree a course of action has been carried out.

Example: *task-report OPR ongoing at now eyeball completely reliable RPTFCT label-report123;*

Detection Reports (unit report/position report)

The robots use different sensors to detect changes. These changes are classified which may result in "suspected enemy forces" which is to be reported. These reports include a description of the object in question as well as the location where it was detected.

Example:
report unit unknown_001 vehicle at now eyeball completely reliable RPTFCT positionreport-12345;
report hostile position unknown_001 at [Point A] at now eyeball completely reliable RPTFCT positionreport-12345;

4. Conclusion

Our work has demonstrated that BML can be used for communicating with robots in the intended way. This is at least in part to be credited to ROS for allowing inter-service communication. From the military point of view, many missions can profit from using multi robot systems. For example, the robots can scout the area a convoy has to pass through for enemy forces or IEDs. Using BML allows commanding the MRS by one single controller using a mobile devise, e.g., a tablet. Thus, the MRS can be commanded directly from the convoy itself which helps to facilitate the information flow and the reaction time in the case the robots run into the enemy. Besides, loosing robots is preferable to human casualties.

References

[1] K. Heffner, N. de Reus, L. Khimeche, O.M. Mevassvik, M. Pullen, U. Schade, J. Simonsen & R. Gomez-Veiga, *NATO MSG-048 C-BML Final Report Summary*. 2010 Fall Simulation Interoperability Workshop (10F-SIW-039), Orlando, FL, September 2010.

[2] M. Pullen, D. Corner, R. Wittman, A. Brook, O.M. Mevassvik & A. Alstadt, A., *MSDL and C-BML Working Together for NATO MSG-085*. 2012 Spring Simulation Interoperability Workshop (12S-SIW-045), Orlando, FL, March 2012.

[3] T. Remmersmann, U. Schade, L. Khimeche, B. Grautreau & R. El Abdouni Khayari, *Lessons Recognized: How to Combine BML and MSDL*. 2012 Spring Simulation Interoperability Workshop (12S-SIW-012), Orlando, FL, March 2012.

[4] C. Blais, M.R. Hieb & K. Galvin, *Coalition Battle Management Language (C-BML) Study Group Report*. 2005 Fall Simulation Interoperability Workshop (05F-SIW-041), Orlando, FL, September 2005.

[5] M. Pullen, S. Carey, N. Cordonnier, L. Khimeche, U. Schade, N. de Reus, N. LeGrand, O.M. Mevassvik, S.G. Cubero, S. Gonzales Godoy, M. Powers & K. Galvin, *NATO MSG-048 Coalition Battle Management Initial Demonstration Lessons Learned and Follow-on Plans.* 2008 Euro Simulation Interoperability Workshop (Paper 08E-SIW-064), Edinburgh, UK, June 2008.

[6] M. Pullen, K. Heffner, L. Khimeche, U. Schade, N. de Reus, O.M. Mevassvik, R. Gomez-Veiga & A. Brook, *An Expanded C2-Simulation Experimental Environment Based on BML.* 2010 Spring Simulation Interoperability Workshop (10S-SIW-049), Orlando, FL, April 2010.

[7] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. B. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, *ROS: an open-source robot operating system*, International Conference on Robotics and Automation, 2009.

[8] M. Langerwisch, M. Ax, S. Thamke, T. Remmersmann, A. Tiderko, & B. Wagner, Realization of an Autonomous Team of Unmanned Ground and Aerial Vehicles. International Conference on Intelligent Robotics and Applications (ICIRA 2012), Montreal, Canada, October 2012.

[9] B. H. Partee, A. ter Meulen & R. E. Wall, Mathematical Methods in Linguistics, Dordrecht, The Netherlands: Kluwer 1990.

[10] K. Rein, U. Schade & M.R. Hieb, *Battle Management Language (BML) as an Enabler,* IEEE International Conference on Communications, ICC 2009, Dresden, Germany, June 2009.

[11] U. Schade & M.R. Hieb, *Formalizing Battle Management Language: A Grammar for Specifying Orders*. 2006 Spring Simulation Interoperability Workshop (06S-SIW-068), Huntsville, AL, April 2006.

[12] U. Schade & M.R. Hieb, *Battle Management Language: A Grammar for Specifying Reports*. 2007 Spring Simulation Interoperability Workshop (07S-SIW-036), Norfolk, VA, April 2007.

[13] M. Gerz & U. Schade, *Das Joint Consultation Command and Control Information Exchange Data Model*, in J. Grosche, & M. Wunder, (Eds.), *Verteilte Führungsinformationssysteme,* Heidelberg, Germany: Springer 2009.