SDU International Journal of Technologic Sciences
Vol. 3, No 1/Special Issue, February 2011
pp. 95-100

DDS-BASED HETEROGENEOUS ROBOTS COMMUNICATION MIDDLEWARE

Ekrem AKSOY*, Selçuk CANBEK*, Nihat ADAR*

Abstract

OMG (Object Management Group) DDS (Data Distribution Service for Real-time Systems) is an open international communication middleware standard for real-time and embedded systems. In this paper, a communication middleware model based on OMG DDS for multi-robot environments is presented. With the help of DDS QoS mechanism and Publish/Subscribe paradigm, proposed communication middleware abstracts different robot implementations sharing same domain (i.e. heterogeneity).

Keywords: DDS, middleware, heterogeneous robots, communication model, QoS

1. Introduction

OMG Data Distribution Service for Real-time Systems (DDS in short) is an open standard for communication middleware that is mostly used in mission-critical infrastructures [1]. The main viewpoint of DDS is data-centricity whereby establishing Publish/Subscribe paradigm. Thus, data-centric communication with the help of Quality-of-Service policies preserves consistency among Publishers and Subscribers.

Multi-robot systems consisting of robots with different capabilities (i.e. heterogeneous) are prone to real-world deployments because of synergistically enriched capabilities. Therefore, many researchers are interested in heterogeneous robot systems like [2] and [3].

On the other hand, heterogeneity conserves many new issues like communications and computing for different capabilities to be handled as stated in [4]. One of the most important issue is communication. As stated in [5], many researches focused on ad-hoc communication technologies for robotics, while some studies focus on middleware like MiRO [6] and HERM [6]. Moreover, from Multi-Agent viewpoint several studies reveal middleware and frameworks like JADE [7] and Open Agent Architecture [8].

In this paper, creating a data-centric real-time (or near real-time) communication middleware independent from robot capabilities is studied. The study offers a new communication model based on Quality-of-Service policies including and defined for capabilities to handle heterogeneity whereas extensible metadata and mission definitions are proposed.

2. OMG DDS Model

OMG DDS communication model provides a Global Data Space where data objects are addressed by Domain, Topic and Key. In this communication model, subscriptions are decoupled from publications, and contracts are established by means of QoS. In addition,

* Osmangazi University, Computer Engineering Department, Eskisehir, TURKEY, E-mail: eaksoy@ogu.edu.tr, selcuk@ogu.edu.tr, nadar@ogu.edu.tr"

DDS provides automatic discovery and configuration mechanisms [9]. From architectural viewpoint, DDS provides peer-to-peer un-brokered service model. Unlike other models like RMI or JMS, this model eliminates brokerage point of failure.

2.1. DDS Communication Model

Communication model is an abstract model of how applications interact. There are several common characteristics of communication models in use today:

a) Remote Method Invocation
b) Message Queuing
c) Publish/Subscribe Data-centric
d) Replication
e) Distributed Transactions

With all these forms, communicating parties could form one of the following:

a) Point-to-point
b) Client/Server
c) Many-to-many
d) Replication

In RMI communication model, a method on a remote system is abstracted on local system. RMI on Java, CORBA or Web Services are examples of this communication model. The main disadvantages of this model are cascading failure nodes and tightly coupling of systems.
On the other hand, Publish/Subscribe model (with Message Queuing or Replication) constructs a decoupled system and isolated failure nodes.

Although message distribution or replication could be performed within publish/subscribe model, there are differences among message distribution, replication and publish/subscribe models. In the case of message queuing, there is only one reader/consumer at a time whereas publish/subscribe model has multiple deliveries. Likewise, messages include update to data model in data distribution where there is no concept of "data" but message in publish/subscribe model.

Thus, in DDS, true publish/subscribe model is used which brings in high performance and reliability. The DDS communication model consists of publishers and subscribers forming up a Global Data Space. In system view, one party could both be a publisher and subscriber. The overall communication is moderated with QoS policies. Thus, publishing and subscribing are decoupled, and overall system has automatic discovery and configuration.

2.2. DDS Architectural Model

Publish/Subscribe system architectural models are either categorized as brokered or peer-to-peer. Brokered pub/sub models are also classified as centralized or segmented or federated.
The centralized broker pub/sub model consists of a single central moderator where all messages are going through it. Therefore, any party publishing or subscribing a message should access this server.

The segmented broker pub/sub model has a grid of moderators where some publishers and subscribers are assigned to at least one of these moderators. In this model, a publisher should communicate with as many as needed moderators to transmit messages meanwhile a subscriber should do the same for receiving messages.

In federated broker pub/sub model, moderators are interconnected with a software bus where publishers and subscribers are connected to their counterpart moderators. The software bus among moderators has its own communication model (e.g. peer-to-peer, multicast, etc.) independent of pub/sub communication model.

DDS has a peer-to-peer architecture where there is no moderator among publishers and subscribers. Thus, DDS eliminates single point of failure. Each participant has a local queue and communicates peer-to-peer.

Using peer-to-peer architecture has the advantage of using only one protocol over other brokered services models where they require two protocols in use: a client protocol and a service protocol.

2.2. DDS Object Model

As explained previously, DDS system forms a Global Data Space which is accessible to all participants. This Global Data Space is named a Domain and there could be several domains in a system. The participant named as Domain Participant and allows an application to access the domain.

The domain participant either publishes or subscribes to a group of objects called a Topic. The topic addresses that group of objects in a Global Data Space where each object is identified by a key.

The domain participant provides a Data Writer if it intends to publish topic or a Data Reader if it intends to subscribe a topic. These classes, in turn, provide type safe operations either to write or to read message objects.

Overall object interaction, or simply, communication is moderated with QoS policies. This QoS policies help on reliability, performance, durability and defines the characteristics of a system. The QoS policy also provides a mechanism to couple publishers and subscribers. Thus, a subscriber could subscribe to a publisher on a topic if it complies with QoS policy requested.

3. DDS-based Heterogeneous Robots Communication Middleware

In this proposed DDS-based middleware, each robot is both publisher and subscriber to several domains identifying missions. Beside, each robot provides at least two topics, one for metadata (capabilities and identifiers including published extended topics) and one for overall mission status (position, local timestamp, mission id and status). Each participant establishes communication based on offered/requested QoS policies. The architectural model of the middleware is proposed and implementation details of middleware and preliminary results are presented.

3.1. Assumptions

Heterogeneous robot environment is assumed in the proposed middleware. Heterogeneity of robots might occur both in physical capabilities and software framework. On the other hand, some assumptions are made for sake of practical implementations. First, an embedded

operating system having IP-networking with TCP/UDP support is under assumptions. Second, robots should have network line-of-sight.

Although there are simpler microcontroller based robotic architectures available today, these are mostly not capable of handling computing required. Thus, to handle required computing and to be able to communicate within a networking stack an embedded computer based robot is required. Indeed, most real-world robots contain at least one embedded computer on-board. This embedded computer runs an operating system (mostly Linux). An example of these systems is Pioneer DX-3 robots. In fact, these robots are planned to be used in this study in conjunction with several other available models.

In real-world applications a robot campaign consisting of robots differing in performance and capability is working on missions. In order a robot to have a situational awareness; it should be operated in a networked environment. Thus, network line of sight is under assumption.

3.2. System Configuration

Proposed middleware architecture has two domains: one for system configuration and one for mission management.

The system configuration domain provides several topics for forming the heterogeneous robot environment. Any robot intends to participate in a robot campaign or a group of robot to form a new campaign uses these topics to form the campaign. The topics in this domain are listed in the table below:

RobotMetadataTopic	Robot ID, Local Timestamp, Model/Make, Built-in Topics
CapabilityTopic	Capability ID, CapabilityName, CapabilityParameters, CapabilityResponse
HWStatusTopic	Power Status, HW Failure
LocationTopic	Coordinates, Baseline, INS, GPS data, Local timestamp
MissionQueueTopic	New Missions, Existing Missions, Coordinators, Teams
CommTopic	ID of robots in communication

Each robot publishes RobotMetadataTopic which includes the ID, Model/Make, and other Topics published. This topic is subscribed by every other robot to communicate with the publisher. The IDs are used in system.

Another important topic for configuration is CapabilityTopic. This is where heterogeneity is absorbed. Each robot publishes its capabilities, the parameters required to operate the capability and the response structure of a capability. This data will be used in mission management later on.

The robot publishes its hardware status with HWStatusTopic. This topic also includes very important information for mobile robots, the power status.
Many real world applications require location based formations. Thus, LocationTopic provides these information including global position, INS if exists and local timestamp for sync.

As stated in DDS architecture, there is no server in DDS model. Every participant provides a local cache. The mission data therefore is spread all over campaign. MissionQueueTopic provides the situational awareness of overall campaign.

Each robot publishes its missions, tasks, and its position in teams. When each message gathered together, the snapshot of campaign status could be taken.

Robots also provide IDs of robots they are in communication through CommTopic. With this information, a specific task or mission could be assigned to a specific team.

3.3. Mission Management

Missions are submitted through any participant in the environment. Additional interfaces to the campaign could be defined via extending DDS and implementing new publishers for mission management topics.

Mission management topics are defined in mission management domain. New missions are submitted by publishers of these topics. These topics are listed in the table below:

MissionMetadataTopic	Mission attributes, values
MissionCapabilityReqTopic	Min. capability requirements for mission
MissionStatusTopic	Mission completion rate, atomic task results
MissionTasksTopic	Atomic tasks and responsibilities.

Missions are defined with MissionMetadataTopic. This topic includes mission attributes and their values. These attributes are defined by context of the application.

The important part of mission management is forming the team for a mission. In order to accomplish this task, mission's minimum capability requirements are needed. As mentioned in previous section, each robot publishes its capabilities. Each robots decision mechanism, which is based on implementation, uses these two topics to either participate or form a mission team.

As a mission is being completed, task results and completion rate could be seen by MissionStatusTopic. Since DDS has Real-Time capability, this status could be used in further planning and capacity assignment jobs.

The last topic in mission management domain is MissionTasksTopic which a mission's atomic tasks are published through. Each task defined by a capability and other attributes like easiness rank, estimated completion duration, etc. These attributes could be defined by application developer or by robot sub-systems like planner with an AI capability like those on multi-agent systems.

Like many of middleware architectures, the proposed architecture could be extended where a developer might extend the topics based on the application of purpose.

4. Implementation

Proposed middleware architecture is under development. Development is now simulated on PC where on-robot tests will be accomplished after development completes.

4.1. Hardware

The proposed middleware mainly has two hardware platforms: a development platform and a test platform. The development platform is a PC with 1GB RAM and running Linux with 2.6 kernel version with AMD Athlon 3000+ processor. The test platform is a Pioneer DX-3 robot running ARCOS Linux OS and has Hitachi H8S 32-bit RISC processor. The robot platform also has Wi-Fi communication module.

On the other hand, DDS platform is available for both Linux and Windows operating systems. Thus, one could use the proposed middleware on other robotic platforms running these OS's.

5. Conclusions

This paper presented a communication middleware model based on OMG DDS for multi-robot environments. With the help of DDS QoS mechanism and Publish/Subscribe paradigm, proposed communication middleware was shown to abstract different robot implementations sharing same domain.

References

[1] N.Mohamed, J. Al-Jaroodi, I.Jawhar, "Middleware for Robotics: A Survey", Proc. of The IEEE Intl. Conf. on Robotics, Automation, and Mechatronics (RAM 2008), pp. 736-742, Sep. 2008.

[2] R. Simmons et al., "Coordinated Deployment of Multiple, Heterogeneous Robots," Proc. 2000 IEEE/RSJ Int'l Conf. Intelligent Robots and Systems, IEEE Press, Piscataway, N.J., 2000.

[3] I.Akyildiz, W.Su, Y.Sankarasubramaniam, and E.Cayirci, "Wireless Sensor Networks: A Survey", Computer Networks, 2002.

[4] W.Evan, B.A.MacDonald, F.Trepanier, "Distributed Mobile Robot Application Infrastructure", IROS 2003. Proc. v2, pp.1474-1480. 2003

[5] A. Gage, D. Murphy, D. Valavanis, and M. Long, "Affective task allocation for distributed multi-robot teams", Technical report TR2006-26 for Center for Robot-Assisted Search and Rescue (CRASAR).

[6] N. Miyata, J. Ota, T. Arai, and H. Asama, "Cooperative transport by multiple mobile robots in unknown static environments associated with real-time task assignment," IEEE transactions on robotics and automation, vol. 18, no. 5, 2002.

[7] Sugawara, K., Mizuguchi, T., "Multi-task allocation and proportion regulation of homogeneous agents," Advanced Robotics and its Social Impacts, 2005. IEEE Workshop on, vol., no., pp. 145-148, 12-15 June 2005

[8] P. Gil, I. Maza, A. Ollero, P. Marrón, "Data Centric Middleware for the Integration of Wireless Sensor Networks and Mobile Robots," in Proc. 7th Conference on Mobile Robots and Competitions, ROBOTICA 2007. April 2007.

[9] Zhang, T.; Hasanuzzaman, Md.; Ampornaramveth, V.; Ueno, H., "Construction of heterogeneous multi-robot system based on knowledge model," Robotics and Biomimetics (ROBIO). 2005 IEEE International Conference on, vol., no., pp.274-279