A Review of Middleware for Networked Robots

Nader Mohamed, Jameela Al-Jaroodi, and Imad Jawhar

UAE University, P.O. Box 17551, Al Ain, UAE

Summary
Networked robotic systems consist of a collection of interconnected components (being multiple robots working together or robots working with external systems such as wireless sensor networks or servers). These components require cooperation and collaboration to achieve a common goal. However application development for such collaborative distributed systems composed of many robots with sensors, embedded computers, and human users is very difficult. Therefore, middleware services can provide advanced approaches offering many possibilities and drastically enhancing the development process and the overall functionalities needed for networked robotic systems. This paper provides an overview study of networked robot middleware in this domain. It discusses middleware roles in networked robotic systems, provides different criteria for evaluating networked robot middleware, and presents some representative middleware solutions specifically designed for networked robots. Open issues in that domain are also discussed.

Key words:
Networked Robots, Middleware, Networked Robot Applications, Ubiquitous Robots.

1. Introduction

Networked robotic systems have a great potential in many new applications and they offer practical and viable solutions to many application areas where human intervention may not be possible. Networked robotic systems refer to multiple robots, robot components and other external entities communicating and cooperating to achieve a common goal. The external entities can be sensor networks, computer systems, or human users. The common goal can be search-and-rescue (SAR) missions in dangerous environments or inaccessible terrains, human-assistance for the elderly or physically challenged, and medical/surgical operations. Robots can operate in environments that are equipped with wireless sensor networks and embedded computers communicating through wireless ad-hoc networks. These robots usually operate to achieve tasks in an efficient and accurate manner. The efficiency and accuracy of operations require robots to coordinate among themselves, act on information gathered from a wireless sensor network, utilize the services provided by other external computer systems, and cooperate with humans.

Those various functionalities and requirements make networked robotic systems complex distributed systems consisting of a number of integrated hardware and software components. Since these components are commonly heterogeneous including the different robots, sensors, wireless sensor networks, and special types of servers, it is necessary to find ways to mask this heterogeneity and offer some innovative solutions to develop their applications. Due to their high components heterogeneity and unique characteristics, networked robotic systems in general pose considerable impediment and make the development of networked robot applications non-trivial. In addition, the existence of such resources and components also requires advanced and efficient techniques for cooperation and collaboration among them to achieve the desired goals. Therefore, there must be new software services, middleware, that act as the glue to link everything together in an efficient manner, supporting concurrency-intensive operations, enhancing collaboration, and insuring efficiency and robustness.

Middleware for Networked robots should be customizable to different scenarios, applications and environments and it should be self-configuring, self-adaptive, and self-optimizing. Indeed the need for a middleware layer that fully meets the design and implementation of different challenges of networked robots technologies is a novel approach to resolve many of the open issues and drastically enhance communications among system components and the development of applications on such systems.

Some research efforts have been done on surveying different aspects of robotics. Some surveyed space robotics [24], while others focused more on robot programming environments features and evaluations [7][21]. Some also focused on surveying different vision models for mobile robot navigation [10], while [30] presented different robotic mapping techniques. In addition, some research efforts were conducted on surveying different middleware approaches for emerging technologies such as ad hoc networks [14] and wireless sensor networks [15]. However none of the existing work investigated the current state of research on the design and development of middleware for networked robotic systems. In this paper we explore

different characteristics and roles of relevant middleware platforms for networked robots and provide a discussion of these characteristics and roles. In addition, we provide different criteria to evaluate the characteristics of middleware for networked robots.

The remainder of the paper is structured as follows. Section 2 discusses some applications of networked robots. Section 3 outlines the most relevant roles of middleware for networked robots. In Section 4, we describe several middleware evaluation criteria for networked robotic systems. In this section we discusses different middleware platforms and approaches; however, due to paper length restrictions, extensive description will be reserved for some of the representative current middleware platforms undertaken towards these characteristics. Section 5 provides some discussions of the current middleware platforms and lists some open research issues and Section 6 concludes the paper.

2. Applications of Networked Robots

In real life, there are various situations where human intervention may be very difficult if not impossible. However, some work has to be done. Networked robots can do such tasks and assist humans in these situations. Some of the essential applications are the following:

Space Exploration: Networked robotics promises to be vital in different areas of space exploration (where humans cannot go or operate) including planetary surface exploration where multiple or even hundreds of small robots can provide extensive and up-close canvassing of a planet. In such application, the robots would have to coordinate their activities to make sure not to do redundant work and maximize the exploration capacity of the entire system. Another class of space applications include coordination of complex instruments on board of vehicles such as the space station or the space shuttles [3].

Domestic help: It is now becoming more and more feasible to have robots that help doing domestic chores. The robots can interact and coordinate with each other to accomplish certain tasks or to respond to human commands that are remotely communicated [26].

Healthcare: Healthcare professionals will increasingly rely on networked robots to carry out diagnostics as well as perform surgery. Networked robots are expected to be used in tele-medicine to replace today's stand-alone devices [23]. In addition, several surgical operations can be performed by networked robotics to assist surgeons.

Military operations: At war times, it would be beneficial not to send humans to danger zones and this is where networked robots could become very handy. For example the military uses un-manned vehicles to canvass parts of enemy territory or geographic sites that are hazardous to humans in order to gather intelligence information about enemy location and movement [23].

From a control point of view, two types of networked robots are identified: (1) Tele-operated: In this category, humans send commands to the robot network and receive feedback. This type can be used in different domains such as research support, education, healthcare and public awareness. (2) Autonomous: In this category robots and associated sensors use the network to exchange information and coordinate activities to complete tasks. Using the network extends the sensing range and increases the effectiveness and robustness of the system.

Wireless Sensor Networks (WSNs): This is another important area where networked robots are expected to play important roles in sensor deployment especially in hard-to-reach areas. To increase effectiveness and efficiency of data collection, the robots would need to coordinate their activities and collected information. Some applications may involve ocean surface exploration, measuring humidity variations in forests, as well as chemical composition of air and water in various environments. Coordinated mobility, whether it is automated or human controlled, becomes an important factor in allowing the same sensor to collect data from different locations [3][17][25]. In addition to the above mentioned tasks, robots can be used in many other applications related WSNs including the important area of communication support. In this role, robots can collect information from the WSN as well as help in bridging connectivity between two or more partitioned segments due to sensor failures. Furthermore, a proper deployment of robots can be used to reduce the energy dissipation of the sensors by using the robot in long range transmissions at critical parts of the network. This can effectively increase the useful life of the WSN. Additional benefits can be drawn from the use of robots in WSNs including enhancing reliability, load balancing, congestion control, quality of service (QoS) support, user interface, safety, sensor/actuator fusion, and localization.

3. Middleware Roles

Middleware can play an important role in developing and operating networked robots. As with many other distributed systems, middleware can provide abstractions, hide heterogeneity, facilitate applications development, and provide several value-added functionalities. While the older generations of robots were designed to achieve specific tasks and manufactured as one unit, the new

generations of robots are diverse and ubiquitous. New robotic applications are composed of multiple robots and other devices that are connected through wireless networks. These robots and devices are usually controlled by software modules developed by different manufactures using different programming languages. The robots and other devices may use different communication mechanisms. Software modules are also needed to process sensor information and control actuators for performing computational, vision and cognitive tasks like planning, navigation, and user interaction.

Although utilizing networked robots for some applications have many efficiency and accuracy advantages, it raises some integration issues such as communication, interoperability, and configuration. These issues could be solved by including a middle layer - middleware. In general, middleware systems are used in distributed systems to reduce development time and cost. This is achieved by providing well-structured and well-tested services for often-needed functionalities. In addition, it provides some value added functions that cannot be added to the operating system such as reliability, security, and abstraction. However, the design and development of a successful middleware for networked robots is not trivial. It needs to deal with many challenges dictated by the robots characteristics on one hand and the applications needs on the other. Generally, middleware for network robots have several roles:

Simplify the development process: Application development is not easy for multiple robots. A middleware solutions should simplify the development process by providing higher-level abstractions with simplified interfaces (API) that can be used by robotic applications developers. In addition, the middleware should promote for software integration and reuse.

Support communication and interoperability: Robots and other devices are designed and implemented by different manufactures. Efficient communication and simple interoperability mechanisms are needed. Therefore, networked robots middleware should provide these functions. In addition, multiple robots may be arranged in ad hoc manner in which they cannot directly communicate with each other. In this case, communication support for ad hoc networks is needed. This type of support can be provided by middleware [14].

Provide collaborative operations among robots: When a robotic system relies on multiple robots to achieve a specific task, these robots must be able to work together efficiently to achieve that task. Therefore, a middleware solution must provide some functionalities and high-level abstractions to facilitate the development of the collaboration mechanisms. Tools and APIs are necessary to support the development and utilization of specialized collaboration functionalities.

Provide heterogeneity abstractions: Any networked robotic system contains many heterogeneous hardware and software components, communication and cooperation among these components is an important aspect. Commonly the abstraction role is played by a middleware which acts as a collaboration software layer among all involved components, hiding the complexity of the low-level communication and the heterogeneity of the components.

Provide integration mechanisms with other systems: New types of robots such as ubiquitous robots need to interact with other systems such as wireless sensor networks and high-end servers. Most of these interactions should be done in an abstract way and in real-time. Hence, middleware should provide real time interaction services with other systems.

Offer often-needed robot services: A great deal of effort is spent writing new implementations of existing algorithms and control services for networked robotic applications multiple times. The same algorithms/services may be rewritten several times due to changes in the robot's hardware, the development of new applications, changes in the operating systems, changes of technical staff, or just for adding new functionalities. These often-needed robot services should be provided by networked robotic middleware which allows for reuse of the modules offering these functionalities.

Provide automatic recourse discovery and configuration: Networked robotic systems are considered dynamic systems due to the mobility of robots and the changes of their environment. For example, external devices can be dynamically available/unavailable for a robot's use. These external resources can be utilized by networked robots to enhance processing power and accuracy when they are available. Hence, automatic and dynamic resource discovery and configuration is needed. In addition, it should support mechanisms for the robots to be self-adapting, self-configuring, and self-optimizing.

4. Middleware Evaluation Criteria

As mentioned above, there are many roles for networked robots middleware. Some of these roles are implemented by existing middleware platforms for networked robots. In this section, we describe several middleware evaluation criteria for networked robotic systems. In addition, we will discuss different examples of middleware platforms and approaches.

4.1 Standard/Nonstandard Communication Model

Middleware for networked robots can be based on standard or nonstandard communication models. For example, some middleware are based on a standard distributed object model, CORBA. The main motivation of using the distributed object model is to improve the software development process for robotic systems and to enable the interaction among robots and other systems. Objects can be developed by different vendors to control different robots and other external entities. The communication, coordination, and collaboration can be done among robots and other devices through object interactions. Objects can be written using a single programming language or multiple programming languages and operated on homogeneous or heterogeneous operating systems. In addition, middleware following a standard communication model can easily utilize libraries implemented based on the same standard of the communication model.

One of the middleware examples using the standard distributed object model is Miro [11][32]. Miro is an object-oriented middleware for robots developed by University of Ulm, Germany. The main features of this middleware are improving the software development process for mobile robots and enabling the interaction between the robots and enterprise information systems. Miro is designed and implemented by applying object-oriented design and implementation approaches using the common object request broker architecture (CORBA) standard. This allows inter-process and cross-platform interoperability for distributed robot controls. Miro was implemented using multiplatform libraries for easy portability. Examples of these libraries are the CORBA based adaptive communication environment (ACE) [27] for providing object-oriented abstraction layers for many operating systems and communication primitives and the CORBA Notification Services [16] for providing the event-based communication functionality.

Another middleware example based on CORBA is RT (Robot-Technology) - Middleware [4]. This middleware was developed by the collaboration of The Japanese Ministry of Economy, Trade and Industry (METI), The Japan Robot Association (JARA), and National Institute of Advanced Industrial Science Technology (AIST). The main goal of this middleware is to build robots and their functional parts in a modular structure at the software level and to simplify the process of building robots by simply combining selected modules. Another important goal is to make robots more intelligent by distributing their necessary resources over a network.

RT-Middleware provides the necessary services to enable implementing robotic applications that need these types of distributed systems. One example of these applications is a network distributed monitoring system for the human assistance robotic system [18]. This application was developed to improve the interaction among the users and local robotic systems. In addition, it enables a remote user to better monitor the local human and the environment. Another application is the development of home integration systems [13]. In this project, multiple home devices and appliances interact with the robotic system. There are some efforts to standardize the architecture of RT-Components in the Object Management Group (OMG) [5]. These efforts will enable fast integration and configuration among robot components implemented by different manufacturers.

Some middleware platforms were implemented without following a standard communication model. This allows for providing some advanced functions that are specifically needed by the networked robotic applications. Example of a middleware that provides a nonstandard model is the PEIS Kernel [8]. The PEIS Kernel is based on a collaborative research project between the Electronics and Telecommunications Research Institute (ETRI), Korea, and The Centre for Applied Autonomous Sensor Systems, Sweden. The PEIS Kernel provides a shared memory model and supports heterogeneous devices. This middleware is designed toward the concept of Ecology of Physically Embedded Intelligent Systems, or PEIS-Ecology, in which many robotic devices, pervasively embedded in everyday environments such as in our homes or offices, cooperate in performing some tasks in the service of people. In this approach, complex robotic functionalities are not achieved via the implementation of extremely advanced robots, but rather through the cooperation of many simple robotic components.

The main aim of the PEIS Kernel is to provide a common communication and cooperation model that can be shared among robotic devices such as mobile robots, static sensors or actuators, and automated home appliances. With this middleware, any robot device with software controls in the environment is defined as PEIS. Each PEIS is a set of inter-connected software components developed to control sensors or actuators. All PEIS are connected by a uniform communication model, which allows for the exchange of information among PEIS. All PEIS can cooperate using a uniform cooperation model. In this model, each participating PEIS can use functionalities from other PEIS in the ecology in order to complement its own. For example, in a home environment, an autonomous vacuum cleaner (PEIS) can use a localization function provided by an overhead tracking system (PEIS) to know its position.

4.2 Interoperability Flexibility

Different robotic software components, sensor nodes, sensor networks, and other external computer systems are designed to use different communication mechanisms. Usually some of these components are old and some of are new. Integrating these components under a single networked robotic system is a very complex process. Some efforts were made to provide middleware that provides flexible interoperability among the software components to enable a simplified integration process. One of these middleware is MARIE (Mobile and Autonomous Robotics Integration Environment) [9].

MARIE is a middleware framework created for developing and integrating new and existing software components for robotic systems. MARIE aims to create a flexible distributed components system that allows robotic systems developers to share, reuse, and integrate robotic software programs for rapid robots application development. MARIE middleware provides some services that allow the adaptation of different communication protocols and applications which make it very flexible. The integration aspect of MARIE uses the Adaptive Communication Environment [27] (ACE) communication framework. A variety of software components can be connected in MARIE using a centralized component. In addition, there are four functional components: application adapters, application managers, communication adapters, and communication managers. Application adapters act as proxies between the central component and the applications. The data exchanged among application adapters is translated by communication adapters, while communication managers create and manage the connections. Finally, application managers instantiate and manage components locally or across distributed processing nodes. MARIE follows the mediator design pattern in which it provides mediator interoperability layers among adapters and managers. The key features of MARIE are the interoperability and reusability of robotic application components.

Another example is a Middle Layer for incorporations among ubiquitous robots [19]. This layer is developed by Korea Advanced Institute of Science and Technology to enable communication among ubiquitous robots which are usually of different types. These types can be software robots, mobile robots, and embedded robots. Software robots are similar to mobile agents while mobile robots are usually hardware robots controlled by software. This middle layer is mainly designed to allow software robots and mobile robots to communicate even when they use different communication mechanisms. The middle layer consists of two mappers: the sensor and the behavior mapper. The senor mapper helps software robots get physical sensor information from mobile robots; while the behavior mapper helps software robots make physical behavior using the actuators of the mobile robots.

4.3 Automatic Discovery, Configuration, and Integration Support

Automatic discovery, configuration and integration are important features that a networked robotic middleware can have. The automatic discovery and configuration mechanisms are appropriate for dynamic computing environment such as ubiquitous robots. Mobile robots can discover the existence of external devices and can configure themselves to interact with them. These devices can be cameras, sensor networks, and controllable electromechanical devices. One robotic middleware example that provides self-configuration is the PEIS Kernel [8]. The PEIS Kernel provides a simple dynamic model for self-configuration and introspection. All PEIS are connected by a uniform communication model that allows dynamic joining and leaving of PEIS.

Some other efforts were conducted to follow the Universal Plug and Play (UPnP) architecture [31] for automatic discovery, configuration, and integration. UPnP was developed to offer peer-to-peer network connectivity among PCs, wireless pervasive devices, and intelligent appliances[31]. The UPnP has automatic discovery and configuration mechanisms. One example of these efforts is UPnP Robot middleware [1][2]. This middleware was developed by Korea Institute of Science and Technology to utilize the Universal UPnP architecture for dynamic robot internal and external software integrations and for ubiquitous robot control. UPnP mechanisms are utilized to configure robot components and to allow ubiquitous robots to discover and interact with other devices like cameras, sensor networks, and electromechanical devices.

Using UPnP mechanics robots are able to configure their internal components to interact with external devices based on the specific goals or services they should provide. This is an essential feature for the implementation of intelligent robotics. The intelligence component can be internal or external since software components can cooperate with each other regardless of their location. This approach provides a simple scheme for building intelligent robots with a lot of hardware and software components. It solves some of the implementation issues currently facing the robotic field.

4.4 Specific/Expandable Services Middleware

Some middleware platforms for networked robots are designed to provide specific high level abstracted services. These services for example are to get abstracted information from wireless sensor networks or deal with powerful fixed servers that provide some services for mobile robots. These middleware platforms are just

designed to provide specific services and they have some limitations in the flexibility of expandability. On the other hand, some middleware platforms are designed to allow adding different new services. These platforms are usually constructed using multilayer or multi-tier architecture to provide flexibility to add new services for often needed functions for networked robotic applications and to support heterogeneous robots and other devices.

One example of middleware that provides specific services is Sensory Data Processing Middleware [29] that was developed at The University of Tsukuba in Japan. This middleware provides abstracted services for accessing sensor information to support service mobile robots. Two types of services were implemented to provide obstacle information and to localize the robot position using landmark observations from multiple external sensors. This middleware provides a unified model for different configurations of external sensors on a service mobile robot. The unified model abstracted from sensors can be used in any service mobile robot application independent of the configuration of the sensors. Developed services can be reused in multiple applications without dealing with individual sensors.

Another example that provides high-level abstracted specific services is the Data Centric Model that is provided in the middleware of AWARE [12]. This platform is a data centric middleware for the integration of wireless senor networks and mobile robots developed by University of Seville, Spain and University of Stuttgart, Germany. The main aim of this middleware is to provide simplified mechanisms for integrating information gathered by various types of sensors including wireless sensor networks (WSN) and mobile robots. This type of integration is needed for applications where robots are used to obtain and process data from their environment through a WSN. This data can be temperature, light level, or humidity for example. Another application is to allow a robot to locate itself in an environment where a GPD (Geographic Positioning Device) is not available. This middleware provides data-centric capabilities in which users can access data in an abstract way. Any user of the network can make references to objects that exist in the environment, such as a fire, a car, or an animal. The user has to provide the conditions that define the targeted object. These conditions can be for example high temperature for a fire object. Then, the user can address any specific object in the environment in order to obtain data from it. In this platform, the middleware components are executed on all sensor and robot nodes. TinyOS operating system, which is designed for small devices with limited resource, is used for sensor nodes.

Examples of middleware platforms that provide flexibility in adding new services are Player/Stage system [20], Miro [11], and MARIE [9]. These middleware platforms are implemented using multi-tier/multi-layer architecture. The Player/Stage system [20] provides infrastructure, drivers and some algorithms for mobile robotic applications. This middleware has two major components: Player and Stage. Player is a device repository server for actuators, sensors, and robots. Each device in Player is composed of a driver and an interface. Interfaces are the part used by the client to write new applications that get information from a sensor or control an actuator. Drivers implement algorithms that receive data from other devices, process it, and then send it back. Stage is a graphical simulator that models devices in a user defined environment. A driver can also generate arbitrary data when needed.

The Player/Stage system is implemented as a three-tier architecture in which the first-tier is the clients which are software components developed for specific robot applications, the middle-tier is the Player which provides common interfaces for different robot devices and services, and the third-tier is the actual robots, sensors, and actuators. Various client-side libraries exist in the form of proxy objects for different programming languages to access the services provided by the Player platform. Clients can connect to the Player platform to access data, send commands, or request configuration changes to an existing device in the repository. Examples of client programming languages supported are C, C++, Java, and Python. The Player serves as an interface to different types of robot devices and provides drivers for hardware. Some of the main features are the platform-, programming language-, and transport protocol-independence; open source; and modularity. Player's modular architecture makes it flexible to support new hardware. Player/Stage system started at the University of Southern California then moved to Source Forge.

Both Miro and MARIE are implemented using three layers to provide expandable platforms for new services. Miro's layers are the device, the service, and the class framework [11]. The device layer provides object-oriented interface abstractions for all sensor and actuator devices. This layer is platform-dependant. The service layer provides abstractions for devices via CORBA interface definition language (IDL). The class framework provides a number of often-needed services such as mapping, self localization, behavior generation, path planning, logging, and visualization facilities. The layered architecture and object-oriented approach make Miro very flexible and expandable to support new devices and new services for new robot applications [22]. MARIE is also implemented in three layers[9]. The Core layer consists of services for communication, low-level functions, and distributed computing functions. The Component layer is used to add components for often-used services and support domain-

specific concepts. The Application layer contains useful services to build and manage integrated applications.

4.5 Added Communication Services

Middleware can add some communication performance features in terms of reliability, availability, or QoS (Quality of Service) on top of networks that do not provide these features. These features are usually needed by critical networked robotic applications. One example of middleware for robotics that provide QoS support is RSCA (Robot Software Communication Architecture) [32]. RSCA is a QoS-Aware middleware for networked service robots developed by Seoul National University. The key strength of RSCA is the real-time support. RSCA provides a standard operating environment and development framework for robot applications.

The operating environment consists of a Real-Time Operating System, communication middleware, and deployment middleware. The operating system is compliant with the PSE52 in IEEE POSIX.13. It provides an abstraction layer that makes robot applications both portable and reusable on different hardware. The communication middleware is compliant to minimum CORBA and RT-CORBA v.1.1 [28] and provides mechanisms for distributed heterogeneous components to communicate in real-time. The deployment middleware provides frameworks for programs to be executed in distributed environments, a dynamic program deployment mechanism, real-time support, QoS, and management capability for limited and heterogeneous resources.

4.6 Embedded Components & Low-Resources Devices Support

Robots in many situations use or interact with embedded devices that may have several limitations such as limited power, small memory, limited operating system functionalities and limited connectivity. Handling such resources is usually different from other regular resources; therefore, middleware should be able to provide special functionalities to handle these resources when needed. Examples of middleware that support embedded robots are the PEIS Kernal and Players. In the PEIS Kernel both tiny embedded devices and complex large robots can be supported [6]. Players can run on both embedded and regular Linux.

5. Discussion and Open Issues

In the previous section we surveyed different middleware evaluation criteria for networked robotic systems. Under each evaluation criteria, we discussed examples of related middleware platforms. The general observation is that all the platforms target some form of enhancement for the networked robots systems both at the development and the utilization levels. Table 1 includes a list of all networked robotic middleware platforms surveyed in this paper with a brief list of their characteristics and technologies/standards used.

As we investigated the different approaches we identified some lacking features and open issues that current middleware approaches did not sufficiently address. There are many issues, technical limitations and difficulties that need to be addressed to achieve advanced middleware solutions. Some of these are:

1. Current middleware systems provide very limited often-needed advanced services and collaboration mechanisms that can be used to simplify the development process and to enhance resources utilizations. In addition, many of the available services are not standardized, which make it difficult to achieve interoperability between different networked robots systems.

2. The availability of self-adaptation and self-configuration mechanisms is very limited. Since the target is to develop autonomous and ubiquitous robots, these mechanisms are very important to enhance the performance of the robot applications.

3. The security mechanisms within the middleware solutions for robotics are inadequately investigated. As the use of multiple robots, the need to secure their communication and collaboration becomes essential for their operations. Yet researchers mostly steer away from this issue.

4. There is very limited work towards providing high level abstractions for coordination and collaboration for multiple robots applications.

5. There is very limited work towards providing automatic middleware mechanisms for efficient utilization of the availability and the heterogeneity of multiple robots working on the same task. In some cases, tasks need to be distributed among the robots to be completed in parallel rather than being done by individual robots. While in others, the existence of multiple robots that have heterogeneous resources provides a great opportunity to redirect tasks to the robot with the most suitable resources.

6. Web services can provide great opportunities for enabling advance integration among networked robot components. However, there are very limited works in that domain. Web services can provides a flexible communication model as well as support for dynamic environments such as mobile multiple robots.

6. Conclusion

Middleware can play an important role in developing and operating networked robots. In this paper, we discussed the main roles of middleware platforms for networked robotic systems. These are: to simplify the development process of network robotics systems, to enable communication and interoperability, to provide collaborative operations among different robots, to provide abstractions among heterogeneous networked robotics components, to enable integration with other systems, to provide often needed robot services, as well as, to provide automatic resource discovery and configuration for dynamic robots and environments.

In addition, several evaluation criteria for networked robots middleware were discussed. These are: following a standard communication model; the level of flexibility in providing interoperability among components of networked robots; support for automatic discovery, configuration, and integration; expandability; the added communication services; and support for embedded systems. Several middleware platforms for networked robotic systems were discussed based on these criteria. Many middleware platforms have different objectives such as simplifying the development process, reusability, integration, flexibility, self-discovery, self-configuration, and supporting QoS. Furthermore we examined the current limitations and open issues in the middleware for networked robots. As a result we identified several open issues that need to be addressed. These include security, self-adaptation, collaborations support, and other advanced features for integration, discovery and higher level abstractions.

Table 1. A Summary List of the Middleware Platforms Discussed.

Platform	Standards/ Technologies followed	Service provided	Designed for automatic discovery & configuration	Flexibility of adding new services
Miro	CORBA, ACE	Generic	No	High
RT-Middleware	CORBA	Generic	Yes	High
UPnP Robot Middleware	UPnP	For automatic integration	Yes	Low
Player / Stage System	Three-tier architecture, Proxy objects	Generic	No	High
The PEIS Kernel	Uniform communication & cooperation models	Generic	Yes	Medium
MARIE	Mediator interoperability technology, ACE	Generic	No	High
RSCA	RT-CORBA	Generic and for QoS support	Yes	High
The Middleware of AWARE	TinyOS, TinySchema, Publish/subscribe	For sensory service	Yes	Low
Sensory Data Processing Middleware	N/A	For sensory service	No	Low
A Middleware Layer for Incorporation	Senor and behavior mappings	For incorporation among different robot types	No	Medium

7. Acknowledgement

References

[1] Ahn, S., J. Lee, K. Lim, H. Ko, Y. Kwon, and H. Kim, "Requirements to UPnP for Robot Middleware," in proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Oct. 2006.

[2] Ahn, S., K. Lim, J. Lee, H. Ko, Y. Kwon and H. Kim, "UPnP Robot Middleware for Ubiquitous Robot Control," in proc. 3rd International Conference on Ubiquitous Robots and Ambient Intelligence (URAI2006), Oct. 2006.

[3] Akin, H. L., A. Birk, A. Bonarini, G. Kraetzschmar, P. Lima, D. Nardi, E. Pagello, M. Reggiani, A. Saffiotti, A. Sanfeliu, M. Spaan, "Two Hot Issues in Cooperative Robotics: Network Robot Systems, and Formal Models and Methods for Cooperation," A white paper, EURON Special Interest Group on Cooperative Robotics, April 30, 2008.

[4] Ando, N., T. Suehiro, K. Kitagaki, T. Kotoku, W. Yoon, "RT-Middleware: Distributed Component Middleware for RT (Robot Technology)," in proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3555-3560, Aug. 2006.

[5] Ando, N., T. Suehiro, K. Kitagaki, T. Kotoku, "RT(Robot Technology)-Component and its Standardization- Towards Component Based Networked Robot Systems Development-," in proc. SICE-ICASE International Joint Conference (SICE-ICCAS 2006), pp.2633-2638, Oct. 2006.

[6] Bordignon, M., J. Rashid, M. Broxvall, A. Saffiotti, "Seamless Integration of Robots and Tiny Embedded Devices in a PEIS-Ecology," in proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Oct. 2007.

[7] Biggs, G. and B. MacDonald, "A Survey of Robot Programming Systems," in proc. Australasian Conference on Robotics and Automation (CSIRO), Dec. 2003.

[8] Broxvall, M., B.S. Seo, W.Y. Kwon, "The PEIS Kernel: A Middleware for Ubiquitous Robotics," in proc. IROS-07 Workshop on Ubiquitous Robotic Space Design and Applications, Oct. 2007.

[9] Côté, C., Y. Brosseau; D. Létourneau; C. Raïevsky, F. Michaud, *Robotic Software Integration Using MARIE*, in The International Journal of Advanced Robotic Systems, Vol. 3, No. 1, pp. 55-60, March 2006.

[10] DeSouza, G. N. and A. C. Kak, *Vision for Mobile Robot Navigation: A Survey,* in IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 24, No. 2, pp. 237-267, Feb. 2002.

[11] Enderle, S., H. Utz, S. Sablatng, S. Simon, G. Kraetzschmar, and G. Palm, "Miro: Middleware for autonomous mobile robots," in proc. IFAC Conference on Telematics Applications in Automation and Robotics, 2001.

[12] Gil, P., 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.

[13] Hada, Y., S. Jia, K. Takase, H. Gakuhari, and T. Ohnishi, "Development of Home Robot Integration System Based on Robot Technology Middleware," in proc. 36th International Symposium on Robotics (IRS), Japan , 2005.

[14] Hadim, S., J. Al-Jaroodi, and N. Mohamed, "Middleware Issues and Approaches for Mobile Ad hoc Networks," in proc. IEEE Consumer Communications and Networking Conference (CCNC2006), Las Vegas, Nevada, Jan. 2006.

[15] Hadim, S. and N. Mohamed, *Middleware Challenges and Approaches for Wireless Sensor Networks*, in IEEE Distributed Systems Online, Vol. 7, No. 3, art. no. 0603-o3001, March 2006.

[16] Harrison, T. H., D. L. Levine, and D. C. Schmidt, "The design and performance of a real-time CORBA event service," in proc. OOPSLA'97, pp. 184-200, Oct. 1997.

[17] Johnson, D., T. Stack, R. Fish, D. Flickinger, R. Ricci, and J. Lepreau, "Mobile Emulab: A Robotic Wireless and Sensor Network Testbed," In Proc. of INFOCOM, 2006.

[18] Jia, S. and K. Takase, *Network Distributed Monitoring System Based on Robot Technology Middleware,* in International Journal of Advanced Robotic Systems, Vol. 4, No. 1, pp. 69-72, 2007.

[19] Kim, T., S. Choi, and J. Lim, *Incorporation of a Software Robot and a Mobile Robot Using a Middle Layer,* in IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 37(6) pp:1342-1348, Nov. 2007.

[20] Kranz, M., R. Rusu, A. Maldonado, M. Beetz, A. Schmidth, "A Player/Stage System for Context-Aware Intelligent Environments," in proc. System Support for Ubiquitous Computing Workshop (UbiSys), Sep. 2006.

[21] Kramer, J., M. Scheutz, *Development environments for autonomous mobile robots: A survey*, in Autonomous Robots, Volume 22, No. 2, pp. 101-132, Feb. 2007.

[22] Krüger, D., I. Lil, N. Sünderhauf, R. Baumgartl, P. Protzel, "Using and Extending the Miro Middleware for Autonomous Robots," in proc. Towards Autonomous Robotic Systems (TAROS), Guildford, September 2006.

[23] Kumar, V., G. Bekey, and A. Sanderson, "Networked Robots," Assessment of international research and development in robotics. P:73-80. http://www.wtec.org/robotics/. NASA, 2006.

[24] Pedersen, L., D. Kortenkamp, D. Wettergreen and I. Nourbakshk, "A Survey of Space Robotics," in proc. 7th International Symposium on Artificial Intelligence, Robotics and Automation in space (i-SAIRAS-03), 2003.

[25] Sanderson, A., "Distributed Robotic Systems: Network Robotics, Multi-Robot Systems, Sensor Networks, and Environmental Applications", WTEC Robotics Workshop, Aug. 2004.

[26] Sanfeliu A., N. Hagita, and A. Saffiotti, *Network Robot System*, Elsevier Journal of Robotics and Autonomous Systems, Vol. 56, No. 10, 793-797, Oct. 2008.

[27] Schmidt, D. C., "ACE: an object-oriented framework for developing distributed applications," in proc. 6th USENIX C++ Technical Conference, April 1994.

[28] Schmidth, D. C., A. Gokhale, T. Harrison, and Parulkar, *A higher-performance endsystem architecture for realtime CORBA*, in IEEE Communication Magazine, Vol. 14, Feb 1997.

[29] Takcuchi, E. and T. Tsubouchi, "Sensory Data Processing Middlewares for Service Mobile Robot Applications", in proc. International Joint Conference SICE-ICASE, Oct. 2006.

[30] Thrun, S. "Robotic Mapping: A Survey," Technical Report CMU-CS-02-111, School of Computer Science, Carnegie Mellon University, Feb. 2002.

[31] UPnP, www.upnp.org

[32] Utz, H., S. Sablatng, S. Enderle, G. Kraetzschmar, *Miro–Middleware for Mobile Robot Applications*, in IEEE Transactions on Robotics and Automation, 18(4):493-497, Aug. 2002.

[33] Yoo, J., S. Kim, and S. Hong, "The Robot Software Communications Architecture (RSCA) QoS-Aware Middleware for Networked Service Robots," in proc. International Joint Conference SICE-ICASE, pp:330-335, Oct. 2006.

Nader Mohamed joined the College of Information Technology, UAE University in Fall 2006. Before that, he was an assistant professor of Computer Engineering at Stevens Institute of Technology in New Jersey, USA. He worked at the University of Nebraska-Lincoln, Nebraska, USA as a research assistant from May 2001 to May 2004 in National Science Foundation (NSF) supported research and development projects. His current professional interest focuses on communication software and middleware, Internet computing, sensor networks, and cluster, Grid, and cloud computing. He published more than 60 refereed articles in these fields. His research was published in international journals such as IEEE Transactions on Parallel and

Distributed Systems (TPDS), Journal of Parallel and Distributed Computing (JPDC), International Journal of High Performance Computing Applications, and IEEE Distributed Systems Online. He has 8 years of industry experience in the information technology field. He obtained his Ph.D. in Computer Science from The University of Nebraska-Lincoln, USA in 2004.

Jameela Al-Jaroodi is an assistant professor at the College of Information Technology, UAE University, UAE. Prior to joining UAEU, She was a research assistant professor at Stevens Institute of technology in New Jersey, USA. Dr. Al-Jaroodi received her Doctor of Philosophy Degree in computer Science from The University of Nebraska-Lincoln, USA in 2004. Currently, her research interests involve middleware, distributed collaborative systems, information systems, and mobile, ubiquitous and pervasive computing. Her research generated over 50 refereed articles in international Journals and conferences. Al-Jaroodi received the Research Excellence Grant from Sun Microsystems, Inc. She also received several research grants from Research Affairs at UAEU. In addition, several areas of her research were also supported by the United States National Science Foundation (NSF), Nebraska Foundation, and the National Center for Information Technology in Education (NCITE).

Imad Jawhar is an assistant professor at the College of Information Technology at UAE University. He has a BS and an MS in electrical engineering from the University of North Carolina, Charlotte, an MS in computer science, and a Ph.D. in computer engineering from Florida Atlantic University. He has published many papers in international journals, conference proceedings and book chapters. His research interests are in the areas of wireless networks and mobile computing, sensor networks, routing protocols, distributed systems, multimedia, and software design. He served on numerous international conference committees. Imad Jawhar worked at Motorola as engineering task leader involved in the design and development of IBM PC based software used to program the world's leading portable radios, and cutting edge communication products, providing maximum flexibility and customization. He was also the president and owner of Atlantic Computer Training and Consulting, a company based in South Florida that trained thousands of people, and conducted numerous classes in the latest computer system applications. Its customers included small and large corporations such as GE, Federal Express and International Paper. Imad Jawhar is a member of IEEE, ACM, and ACS organizations.