Agent Standard Protocols for Building Multi-Robot Applications

Enric CERVERA [a] and Veysel GAZI [b]
[a] *Robotic Intelligence Lab, Jaume-I University, Spain*
[b] *TOBB University of Economics and Technology, Ankara, Turkey*

Abstract. A practical solution based on multi-agent protocols for the development of real-world multi-robot applications is presented. FIPA standard protocols implemented by the JADE library provide the standard functionality for a number of tasks. Robot behaviors are built upon the Player middleware. Such components provide off-the-shelf tools which allow a straightforward implementation of indoor localization and navigation tasks for a team of mobile robots. Such integration combines proven mobile robot algorithms with a distributed infrastructure, and extends the capabilities from a robot alone to a whole team of robots, thus allowing the development of cooperative applications. As a proof of concept, an auction-like goal assignment task is presented: the robot team is given a goal, and each robot proposes an estimated cost for achieving it, then the best proposal is selected. Most of the control flow is automated by the standard interaction protocols. Experimental evaluation demonstrates the advantages of combining both frameworks, for a practical yet sound development of multi-robot applications.

Keywords. Networked Robots, Multi-Robot Systems

1. Introduction

The aim of our research is the integration of a robot middleware and a multi-agent system, which paves the way for the straightforward development of multi-robot cooperative applications. Both components are complementary: the robot middleware provides generic interfaces with the robot hardware (base and sensors) and a handful of sound algorithms for single robot localization, planning and navigation; the multi-agent system adds the necessary infrastructure to extend the robot capabilities to a team of robots, based on standard protocols.

The efficacy of the integrated system undoubtedly derives from the choice of its components, Player [7] and JADE [2], whose capabilities and weaknesses will be highlighted in the context of the proposed system architecture. Besides their philosophy or design, they have been chosen mainly due to *practical* reasons: their widespread use, rich community of users, constant development and support, and overall usefulness in our previous experience in teaching and research of Robotics and AI. Last but not least, both platforms are open-source and free.

As a result, we develop two simple yet effective multi-robot applications which showcase the benefits of both platforms: extending the functionality of the sophisticated planning and navigation robot algorithms to the team of robots, thus allowing the realization of cooperative tasks. The synergy between the components is emphasized by their seamless and elegant integration, as well as for the complementary strengths.

2. System architecture

The proposed architecture consists of two platforms: a middleware for robot control (Player[1]) and a multi-agent development environment (JADE[2]).

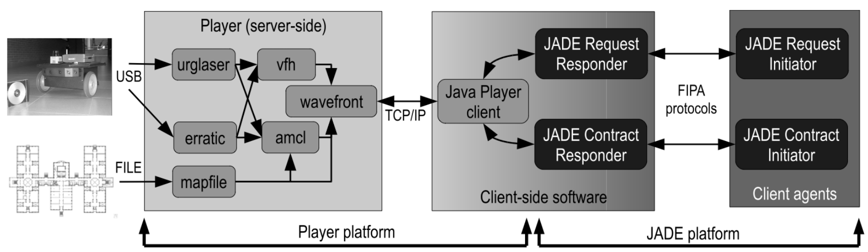


Figure 1. System architecture.

Robotic middlewares are playing an increasingly important role in robotics research, particularly for the development of architectures for mobile robots. They allow portability of code and enhance reusability, thus making the application independent of the robot hardware. Many popular middlewares are available, see e.g. [4] for a recent survey. However, such middlewares focus on solutions for *single* robot applications, and the extension to multi-robot applications, while possible, is not straightforward, since it may require significant communication and programming resources.

On the other hand, multi-agent platforms have been widely studied in the last decade [5], and successfully applied to many domains, including cooperative robotics [8]. But they are not designed with robots in mind, and again, a significant effort to interface with the robot hardware may be needed to perform a successful application.

We claim that the fusion of such complementary platforms may boost the development of cooperative multi-robot tasks. In the following, we introduce our choices of components, Player and JADE, and we describe their synergy and ease of use for practical multi-robot application development.

Figure 1 depicts the system architecture, namely Player and JADE platforms, with the different modules used in each of them, which are thoroughly described in the following.

2.1. Player robot platform

The Player framework interfaces with the underlying hardware (mobile robot base and sensors). In addition, it provides ready-to-use software drivers which implement localization and (local and global) navigation functionalities [6,3,1].

[1]http://playerstage.sf.net
[2]http://jade.tilab.com

2.2. Multi-agent protocols

JADE is a software framework which simplifies the implementation of multi-agent systems through a middleware that complies with the FIPA[3] specifications and through a set of graphical tools that supports the debugging and deployment phases.

Two FIPA-standard protocols [2] are used. In the first case (FIPA-Request) the initiator will ask for one or some robots to attain a given goal. In the second case (FIPA-Contract-Net) the initiator will ask for proposals of the cost estimated by each robot for achieving the goal, then selecting one or some of them to proceed to the goal.

It should be taken into account that the JADE framework provides the domain-independent source code for implementing the above protocols, thus freeing the developer from all the flow control and message processing burden. She only needs to concentrate on the particular details of the application, i.e. in our case multi-robot planning and navigation.

2.3. Interfacing Player and JADE

The client/server design of Player decouples the implementation language of each side, as long as they can communicate by TCP/IP. In our architecture, the localization and navigation drivers run natively on the server-side of Player. The client-side is programmed in the Java language[4] (see Fig. 1), thus allowing a seamless integration with JADE, which uses the same language.

As a result, both components are highly decoupled, since the agent will communicate with the wavefront planner driver. Interaction with the other drivers is indeed possible, yet not needed in our applications.

3. Applications

In the experiments, we have used three mobile robots, as shown in Fig. 2. Each one is endowed with a small laser rangefinder for localization and obstacle avoidance, and an onboard PC with wireless networking. Besides the operating system, each computer is running the Player drivers for controlling the hardware elements, those drivers which implement the localization, planning and navigation algorithms, and the Player/JADE responder agents which listen either for requests or contracts.

3.1. Multi-robot navigation based on request

The first example is a straightforward extension from a single-robot behavior (go to "goal") to a team of robots. The user asks some robots to go to a particular location in the map. Those robots which are available, will agree; then each robot will compute its path, and move towards the goal. If any robot fails to achieve the goal, it will send a failure message to the user; otherwise, an inform message will be sent.

Figure 3(a) depicts a screenshot of a special JADE agent, named *sniffer*, which allows the traffic of messages in the application to be monitored. In this example, all the robots have been requested to go towards the goal, they all have agreed, and successfully informed of the accomplishment.

---

[3]http://www.fipa.org

[4]http://java-player.sf.net/

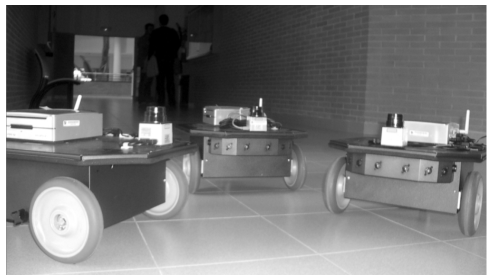
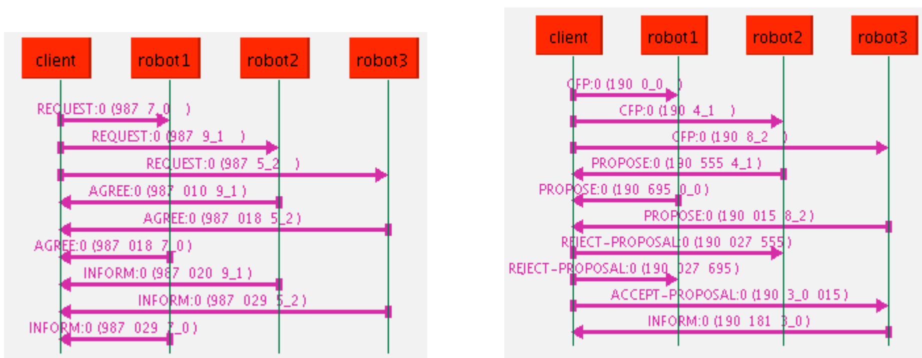


Figure 2. Team of robots used in the experiments. On top of each one, a compact PC is running all the control software, and a Hokuyo URG laser detects obstacles and provides range data for localization.



(a) Request Protocol.

(b) ContractNet Protocol.

Figure 3. Traffic monitoring of messages in the 3-robot application using the each protocol (vertical axis is not scaled in time).

3.2. Multi-robot navigation based on contract net

In this second example, the robots will not move right away to the goal, but compute a cost (the length of the path) and answer to the initiator with a proposal. The initiator will select some of the robots, usually the shortest path, and will send an acceptance message to that robot, and reject messages to the rest, as presented in the protocol before. Figure 3(b) depicts another screenshot of the JADE *sniffer* agent when asking for proposals to attain a goal. In this example, the proposal of robot 3 is accepted and it successfully informed the initiator of the accomplishment; the others are rejected.

4. Conclusion

To this end, what makes the combination of Player and JADE so appealing for cooperative robotics? JADE is a mature and widely-used agent platform, which lacks the interface to *real things* like e.g. mobile robots. Player is an extremely popular software for mobile robots, a de-facto standard; despite its distributed client-server model, it is by no means designed for multi-robot applications.

Combining Player and JADE brings together the best of both, resulting in an extremely useful framework for the development of real, cooperative robotics tasks. The examples presented, though simple, demonstrate on one side how straightforward is the translation from a generic protocol to a real robot application, and on the other side how simply behaviors can be shared and combined by the robot team.

Comparison with equivalent frameworks is our next target in the near future. From the developer's point of view, one possible measurement of the strength of each platform could be the length of the source code for the same task. Our responder agent based on the contract-net protocol has only 134 lines of Java code, distributed almost at 50% between generic agent code and specific robot application code. This is indeed a very short program, yet it performs remarkably well and it scales to any number of robots (as long as communication infrastructure allows).

More complex protocols (iterated contract net, english auction, dutch auction, brokering, recruiting, subscribe) do exist which could suit well to other robotic tasks. Simple yet effective problems could form the basis for a benchmark in cooperative robotics.

Acknowledgements

This work has been partially funded by the EU-VI Framework Programme under grant IST-045269 - "GUARDIANS" of the EC Cognitive Systems initiative, and by the Spanish Ministry of Science and Education (MEC) under grant DPI2005-08203-C02-01.

References

[1] J. Barraquand, B. Langlois, and J. C. Latombe. Numerical potential field techniques for robot path planning. *IEEE Transactions on Systems, Man and Cybernetics*, 22(2):224–241, 1992.

[2] Fabio Bellifemine, Giovanni Caire, Agostino Poggi, and Giovanni Rimassa. JADE: A software framework for developing multi-agent applications. lessons learned. *Information and Software Technology*, 50(1–2):10–21, 2007.

[3] Dieter Fox, Wolfram Burgard, Frank Dellaert, and Sebastian Thrun. Monte Carlo Localization: Efficient position estimation for mobile robots. In *Proceedings of the AAAI/IAAI*, pages 343–349, 1999.

[4] James Kramer and Matthias Scheutz. Robotic development environments for autonomous mobile robots: A survey. *Autonomous Robots*, 22(2):101–132, 2007.

[5] A. Sloman. What's an AI toolkit. In *Software Tools for Developing Agents: Papers from the 1998 Workshop. AAAI Press, Technical Report WS–98–10*, pages 1–10, 1998.

[6] I. Ulrich and J. J. Borenstein. VFH+: reliable obstacle avoidance for fast mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, volume 2, pages 1572–1577, 1998.

[7] Richard T. Vaughan and Brian P. Gerkey. Really reusable robot code and the Player / Stage project. In D. Brugali, editor, *Software Engineering for Experimental Robotics*, Springer Tracts on Advanced Robotics. Springer, 2007.

[8] K. Wieczerzak, G.; Kozlowski. Agents that live in robots: how are successful applications built? In *Proceedings of the Fourth International Workshop on Robot Motion and Control, 2004. RoMoCo'04.*, pages 97–102, 2004.