論文 / 著書情報
Article / Book Information

論題	
Title	Basic Study on Element Administration in Robotics Middleware -2nd Report: An HTML5 based GUI for cross-middleware elements integration-
著者	Ceron Lopez Arturo E., 福島 E. 文彦
Author	Arturo E. Ceron Lopez, Edwardo F. Fukushima
掲載誌/書名	, , No. 13-2,
Journal/Book name	Proceedings of the 2013 JSME Conference on Robotics and Mechatronics, , No. 13-2,
発行日 / Issue date	2013, 5
URL	http://www.jsme.or.jp/publish/transact/index.html
権利情報 / Copyright	本著作物の著作権は日本機械学会に帰属します。
Note	このファイルは著者（最終）版です。 This file is author (final) version.

Basic Study on Element Administration in Robotics Middleware
2nd Report: An HTML5 based GUI for cross-middleware elements integration

Arturo E. CERON LOPEZ and Edwardo F. FUKUSHIMA

Tokyo Institute of Technology, aceron@robotics.mes.titech.ac.jp, fukusima@mes.titech.ac.jp

The Intelligent Cross-Platform Interface (ICPI) provides linking and administrative services to the state of the art robotics middleware platforms by making use of the "role definitions" concept. This work introduces an HTML5 based GUI which provides an accessible and intuitive way of using the services currently supported by the ICPI. Validity was proven through experiments for controlling a robot.
Key Words: *Service Robot, Robotics Middleware, Role Definition, GUI.*

1. Introduction

Service robots require integration of a heterogenic set of hardware and software elements, and for this reason the use of middleware, e.g., *Robotics Technology Middleware* (RTM) [1], *Robot Operating System* (ROS) [2] and *Microsoft Robotics Developer Studio* (MSRDS) [3], is getting increasingly importance.

Robotics middleware is the "glue" joining distributed software programs in order to make them work in a parallel and cooperative way. The authors are developing a new interface called Intelligent Cross-Platform Interface (ICPI) [4] which provides linking and administrative services to the state of the art middleware platforms; the ICPI is based on the client-server model. The ICPI use the "role definitions" concept, from where the base infrastructure of the interface is derived. Various modules contained in the server give its functionality; this is the case of the previously proposed Administration by Roles module, which enables the linking among Software Elements in a classified and automatic way.

In this article, an HTML5 based GUI module is proposed, which provides a visualization and manipulation tool to give the user an intuitive way of interacting with the proposed ICPI, and consequently, with its connected middleware platforms.

2. GUI implementation

2.1 Advantages of HTML5

Most popular and complete middleware platforms up to date require the user to install the platform's main features on a machine, including their supported GUI for monitoring and manipulating the system and platform; e.g. ROS' *rqt_console*, *rqt_graph* and *Gazebo* [5], OpenRTM-aist's *RT System Editor* (RTSE) [6] and MSRDS' *Visual Programming Language* [7]. This task becomes daunting for users that are not completely familiarized with robotics middleware, representing a high cost in terms of time and human resources. For these reasons, in this project it is proposed that the setup phase for a service robot development framework should be as transparent as possible to the user, at the same time its GUI should be able to work from any machine without depending on a full platform installation and/or the OS.

With the recent HTML5 specification [8], new possibilities for dynamic web applications have been brought. The HTML5 has gained wide acceptance and is now being supported by the major web browsers, which can run on a PC, Tablets and Smartphones with nearly same behaviors. For this reason, the authors have decided to make an implementation of the ICPI-GUI based on HTML5; this is to provide an intuitive tool where the user only requires of basic informatics knowledge.

2.2 ICPI-GUI Specification

The ICPI-GUI is a server module with access to the ICPI features and to the participating middleware platforms (Fig. 1.). By this, a new way of interacting with Robotics Middleware is proposed. Access to Software Elements and data is given in the same way for every middleware platform participating in the ICPI; an example of how the proposed abstractions are applied to middleware platforms is shown in table 1.

2.2.1 Core

The core makes requests to the server and process server's responses. All requests are triggered by events. The ICPI Server offers a series of functions for interacting with the Software Elements (SE) and Structured Data sets (SD), among the relevant ones the following are included: Initialize, Discover, Connect, Disconnect, Activate, and Deactivate.

Events include the following: on device input (e.g. mouse click or textual command), on startup, on connection, on disconnection. Events are mapped with

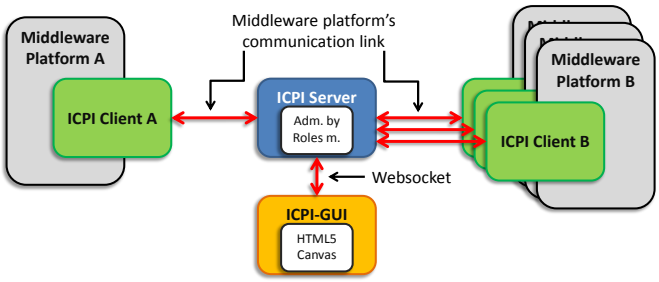

Fig. 1 Conceptual diagram of ICPI implementation.

Table 1. ICPI abstractions applied to middleware platforms

ICPI	RTM	ROS	MSRDS
Software Elements	RT-Components	Nodes	Activities (DSS Services)
Structured Data	Ports as data sources	Topics	Messages

requests to these functions (e.g. mouse click on Discover button will trigger Discover request to the server). Requests are sent through a Websocket connection.

The program flow is as follows:

a) On startup: Connect to Websocket server

b) On connection: Send Initialize request to server

c) On device input event from screen:

c.1) Match event with mapped request(s).

c.2) Send request(s) to the server.

c.3) Wait for server's response:

c.3.1) On reception of SE, SD and connection list: Update internal GUI's database.

c.3.2) On reception of debug event: Store information temporally.

c.4) Buffer the screen with graphical (SE, SD and connection diagrams) representations and textual information (for debug messages).

d) On disconnection: Attempt to reconnect to server and buffer reconnection message on screen.

e) Display: Screen buffer.

f) Loop program: From c), or from b) if disconnection event happened.

2.2.2 Communication

By using Websocket connections, a full-duplex communication using a single socket is possible, which enables the development of real-time and event-driven applications [9]. The ICPI-GUI uses Websockets for gaining access to the ICPI Server. In order to format requests and responses between the server and the GUI, a sub-protocol is defined. A first version of it is represented as a character string and is constructed as follows:

Sender + Process ID + Request/Response Name + Args

Where *Sender* is the name of the entity sending the request/response (e.g. *icpigui*); *Process ID* is the label of the involved process (e.g. *main0!id!*); *Request/Response Name* is the label of the pertinent function or variable (e.g. *ACTIVATE*); *Args* are the related arguments to that function or variable, which are indexed with numbers in parenthesis in case of being a vector of values (e.g. *args(1)1(2)3*); and the plus "+" operator represents a string concatenation function. An example of a request:

"icpiguimain0!id!ACTIVATEargs(1)1(2)3".

2.2.3 Screen

The GUI uses the Canvas element [10]; with this, many graphical entities are drawn dynamically into a webpage. Also input events are detected (e.g. mouse and keyboard).

Screen buffer is drawn into the Canvas. In this implementation five panels have been provided to interact with the ICPI as well as debugging its status: *Menu panel, Software Elements panel, Structured Data panel, Diagram View panel and Console View panel*.

3. Testing and evaluation

For testing, three RT-Components from the RTM platform were used; one for reading a joystick, one for multiplying the joystick data by a predefined value (i.e. 0.5) and other for controlling the motors of a robot. Transferred data was formatted in character strings. The test was performed on a Smartphone (Android 2.3/Opera), and on a standard PC (Win7/Firefox) (Fig. 2).

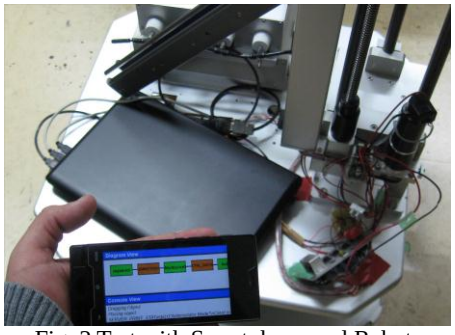

Fig. 2 Test with Smartphone and Robot.

The ICPI Server and ICPI Client for RTM were executed. The Administration by Roles module made all connections automatically inside the RTM platform. Then the ICPI-GUI followed the program flow previously described. The user made requests to the server by clicking on different objects on the GUI screen (e.g. clicking on the Joystick icon to activate or deactivate it). The requests and responses were sent using the Websocket and the sub-protocol previously defined.

4. Conclusion

In this article an HTML5 based GUI was proposed for interacting with the ICPI in an intuitive way. Such GUI is able to work in a variety of devices, increasing its accessibility. Testing was made for building and manipulating a system that controls a robot with successful results. Future works include giving full support to the "role definitions" concept, as well as reducing communication overhead.

Acknowledgement

The first author acknowledges support from CONACyT-I^{2}T^2 and Roberto Rocca Education Program through scholarships for graduate studies at Tokyo Institute of Technology.

References

[1] Ando, N.; Suehiro, T.; Kitagaki, K.; Kotoku, T. & Woo-Keun Yoon; "RT-middleware: distributed component middleware for RT (robot technology)", 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp.3933-3938, Aug. 2005.

[2] Quigley, M. et al.; "ROS: an open-source Robot Operating System", ICRA Workshop on Open Source Software, 2009.

[3] Microsoft Robotics Developer Studio 4, http://www.microsoft.com/robotics

[4] Ceron Lopez, A., Fukushima, E.: Proposal of Intelligent Cross-Platform Interface for Robotics Middleware, Proceedings of the 2012 IEEE International Conference on Cyber Technology in Automation,Control and Intelligent Systems,pp.382-387,May 2012.

[5] Gazebo, http://gazebosim.org/wiki/Main_Page

[6] OpenRTM-aist, http://www.openrtm.org/

[7] Microsoft Robotics: Visual Programming Language (VPL), http://msdn.microsoft.com/en-us/library/bb483088.aspx

[8] HTML5.1 Nightly, http://www.w3.org/html/wg/drafts/html/master/

[9] HTML5 Web Sockets, http://www.websocket.org/quantum.html

[10] HTML Canvas, http://www.w3.org/html/wg/drafts/2dcontext/html5_canvas/