Intelligent Ground Vehicle Competition – JAUS Challenge Entry 2012

Christopher D. Granz
University of Massachusetts – Lowell
1 University Ave. Lowell, MA 01854
chrisgranz@gmail.com
978-473-3712

ABSTRACT
In this paper, the Joint Architecture for Unmanned Systems (JAUS) implementation designed to work with the Robot Operating System (ROS) and run on the University of Massachusetts – Lowell's robot ("Stark") is described. An overview of the implementation utilizing the open source JAUS++ library is given. This includes a brief overview of the JAUS standard and the Intelligent Ground Vehicle Competition requirements, which are subset of the complete JAUS standard. The ROS side of the implementation is also discussed, as the implementation functions as a ROS node.

Author Keywords
Interoperability, JAUS, ROS, IGVC.

INTRODUCTION

One of the challenges for this year's entry in the IGVC is the implementation of JAUS on the University's robot, "Stark." This project represents the attempt to fulfill that requirement. As it has been decided upon to use ROS as a framework for the entire system running on the robot, the goal was to produce a working ROS node which provides all of the JAUS functionality required by the rules of the IGVC. In order to test this functionality, the project also included a JAUS test program based on the description given in the IGVC rules. This test program is outside of ROS and may be used to test the ROS node and it's implementation of the required JAUS interface. This work builds on previous work to increase standards compliance in unmanned systems [2].

PROJECT DESCRIPTION

What was developed for this project was essentially two pieces of software, both intended to run on Linux systems. First and foremost was the main interface layer between ROS and JAUS. This was developed using the open source JAUS C++ library JAUS++. This greatly facilitated the implementation of this software by allowing the development of the actual interface to be focused on.

In order to test the main component of the project, a second piece of software developed as a stand-in for what the judges' at the competition will be using to test the JAUS interface of the robot (the JAUS-ROS node of this project). This was designed based on the requirements given in the IGVC rules.

JAUS is an important standard developed and supported mainly by the Department of Defense [1], and has found its way into the private research and academic communities (although not overwhelmingly.) In some ways, JAUS is an alternative to ROS, but it does not provide an actual code base to work from like ROS, but rather simply a standard which must be implemented.

An initial effort was made on the part of the author to implement the JAUS standard directly using the standard Linux socket API. This proved to be very time-consuming and unnecessary with the discovery of the JAUS++ library. This is a fairly straightforward library to use with tutorials available on using the API.

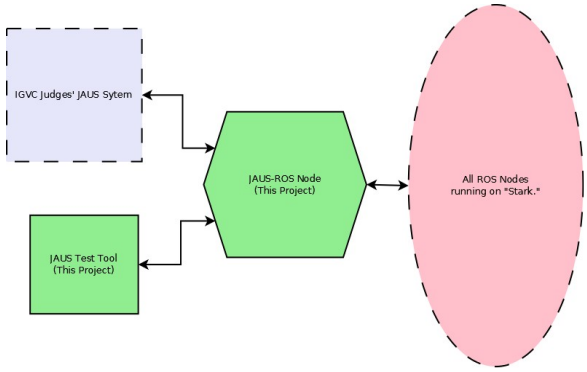

Figure 1. High-level system architecture.

A high-level system diagram is shown in Figure 1. The blocks labeled “JAUS-ROS Node” and “JAUS Test Tool” are the software created as part of this project. The block labeled “All ROS Nodes running on 'Stark.'” represents the rest of the system (ROS nodes) which complete the University's 2012 IGVC entry. The block labeled “IGVC Judges' JAUS System” is the unknown test program used by the IGVC judges to evaluate the University's 2012 entry.

JAUS is built around UDP and message passing [2] and requires the implementation of “services” provided by “components”. The main service is the Management Component, which is what allows high-level control of a robotic system. This component allows the robot to be stopped/resumed and shutdown. In addition to this, Sensor Components, which provide readings and data about a robotic system are generally created. Each component is essentially discovered initially by the outside control system, and then data may be requested of it, or commands sent to it.

In addition to the Management Component and Sensor Components, the IGVC Challenge requires what is known as a Local Way-point List Driver Component be implemented. The service that this component provides is the ability to set a list of way-points and the then have the robot execute the list by driving to each way-point sequentially.

The structure of these components and the discovery and UDP communication is handled by the JAUS++ library. To implement special functionality and handling way-point list messages, special classes can be created which are child classes of the JAUS++ library classes. This is done in this project for the Local Way-point List Driver JAUS component, but is merely a stub because the ROS architecture to be used has not yet been defined.

ANALYSIS OF RESULTS

To analyze the correctness of the implementation the results of running the JAUS-ROS node and communicating with it via the created JAUS test program was compared to the specifications given in the IGVC rules for the JAUS challenge. The ROS core software was run in the background with the standard ROS simulation known as stage and a simple ROS node which drives a robot around a track running. As the ROS architecture for the real robot is not completed at the time of this writing, this simulation acted as the ROS side of the system.

The results of this simulation were compared with those given in the IGVC JAUS Challenge rules and in particular the communication diagram shown in Figure 2.

In this diagram, what is labeled “COP” (for Common Operating Picture) on the left represents the judges' JAUS implementation. What is labeled “Entry” on the right represents the JAUS-ROS node developed for the project.

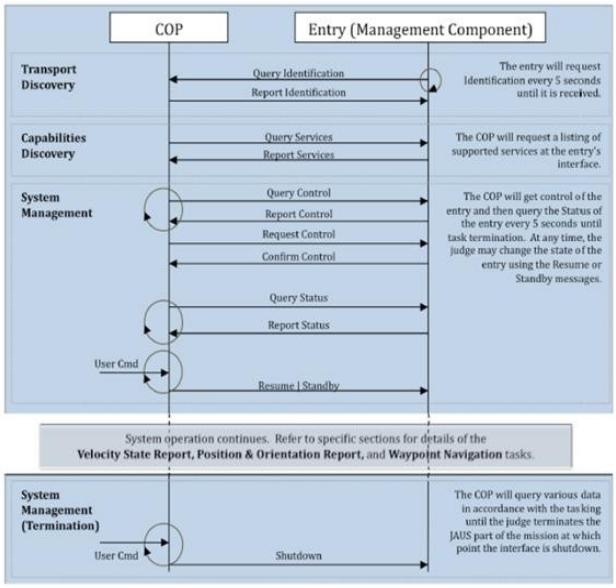

Figure 2. Communication Diagram.

The software created for this project meets the requirements as far as this author can tell. The main issue remaining is tying some of ROS side connections into the ROS architecture which has not yet been implemented by other individuals.

DISCUSSION

Although standards are invaluable for interoperability of unmanned systems, this author found JAUS to be unnecessarily complex, although it may simply be that it includes much more than is required for the IGVC challenge. The lack of a reference implementation makes the JAUS documentation seem even more convoluted.

Even with the complexity of the standard some have found it not sufficient for their purposes and have chosen to extend it [3]. This essentially negates the whole purpose of a standard in the first place.

This project was an important part of the University's 2012 IGVC entry project which has been developed by several researchers, and will hopefully result in a higher score at the competition. In hind-sight, this author may have chosen a different project to devote time to, as this turned out to be very frustrating and not terribly interesting or innovative.

As a necessary interface layer this project has value, and may be used as the basis for future entries in the IGVC by the University's team.

CONCLUSIONS

The author believes the project was a success, even if not glamorous to an outside observer. This was a project completed primarily out of necessity, not new research. It will be interesting to see what the future of the JAUS standard holds, as this author believes it may be replaced by

more popular systems, such as ROS, that have an open source code base to work from. None of the functionality defined by the JAUS standard cannot be provided directly by ROS, and the support for JAUS seems to be somewhat lacking in practice, based simply on research of publications related to JAUS by this author.

In future work, it may be helpful to implement a visual JAUS testing tool, perhaps which is cross-platform and does not use the JAUS++ library. By developing this type of testing tool, future IGVC entries can be appropriately tested and more quickly than the simple terminal based test program created as part of this project.

ACKNOWLEDGMENTS

The work described in this paper was conducted as part of a Spring 2012 robotics course, taught in the Computer Science department of the University of Massachusetts Lowell by Prof. Fred Martin.

REFERENCES

1. Huang, H., Albus, J., Kotora, J., and Liu, R. Robotic Architecture Standards Framework in the Defense Domain. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems 2003*.
2. Pacis, E.B., Everett, H.R., Farrington, H., Kogut, G., Sights, B., Kramer, T., Thompson, M., Bruemmer, D., and Few, D. Transitioning Unmanned Ground Vehicle Research Technologies. In *Proc. SPIE 2005*, Volume 5804, pp. 521-531 (2005).
3. Touchton, R., Kent, D., Galluzzo, T., Crane III, C. D., Armstrong II, D. G., Flann, N., Wit, J., Adsit, P. Planning and modeling extensions to the Joint Architecture for Unmanned Systems (JAUS) for application to unmanned ground vehicles. In *SPIE Defense and Security Symposium 2005*.