A Generic Information-Centric Architecture for Robotic Systems Derived from a New Theory for Adaptive Systems

Vasile Coman*, Michael Chervenic, Radu Moldovan Ph. D.
XCLSoft
Littleton, MA

***vcoman@xclsoft.com**
Tel: 978 265-4024

Abstract

Writing complex software is very difficult. Many if not all of the large enterprise applications sold in this multi-billion dollar industry took many years to be designed and were developed by thousands of developers and systems architects. Despite all the efforts, each one of them is built on a custom-based architecture that is sharing a very limited set of standardized components. This situation has been captured by Grady Booch in a recent interview with "Scientific American" (June 17, 2008): "*In other disciplines, engineering in particular, there exist treatises on architecture. This is not the current case in software, which has evolved organically over only the past few decades. All software-intensive systems have an architecture, but most of the time it's accidental, not intentional. This has led to the condition of most software programming knowledge being tribal and existing more in the heads of its programmers than in some reference manual or publicly available resource*."

Writing software for robotic systems is far more complex than writing enterprise applications. Features like adaptability, real-time interfacing with sensors and awareness of its environment are likely to be implemented using a custom design. This may lead quickly to costly changes and compromises resulting on limited functionality when upgraded. For the robotics industry, this situation is a catch-22. While the market is waiting for better applications, the cost of adding new features to the existing ones is very high. The only solution is to create a generic software platform shared by all robotics applications that will support all specific features like adaptability or real-time information processing from sensors. The architecture and the design of such platform is the main objective for this paper.

To succeed in accomplishing this goal, we propose a different approach. Instead of using a "connectionist" model—all components are designed with a role in mind, and then the integration is done through the calls between them—to design this architecture, we go back to the basics and find out what makes a system adaptive. By examining the adaptive systems we know best which are the biological organisms, we uncover a controlled hierarchy structure in which all its basic components, *the cells*, are playing two roles. The first role, operational, is very simple and is directly related to its individual existence. The

second role is where the complexity lies, and it is related to the overall biological organism viability. All cells are organized in a controlled hierarchy that routes information, most of the times with the help of the nervous system, only accordingly to some strict rules. These rules, together with the definition of some new concepts, form the foundations for the new adaptive system theory.

In this new theory, the most important concept added is called *information density*, and it links the logic contained in an information message to the complexity required by a system need to change. This addresses the most complex requirements when designing a robotic system, which are its ability to learn, and change its operations based solely on the logic embedded in the information messages received from its environment. This concept plays the same role for the logic embedded as the digitization plays for the physical aspect of the message. The higher the information density needed to be processed, the more capable a robotic system is required to be when comes to adapt its set of pre-programmed operations. And because the "information density" is not dependent on the instructions to the system itself to act on what is programmed to do, the components added to make it able to change can be standardized across all the robotics systems.

One other important factor in further defining the generic architecture for robotic systems is the context in which changes occur. For instance, in a robotic system changes to its operations, changes to its technology, and changes to its HMI are driven by different factors. As a result, all three are considered different contexts. They are captured by three different controlled hierarchies that interact through intermediate components playing the role of translators of logic.

The final architecture presented by this paper addresses few more areas, as *operational entities lifecycle*, and *operational space and time* that render the entire architecture *stateful*, *distributed*, *hierarchical*, and *dynamic*. It also proposes a way to measure the adaptability of a system based solely on its capability to process the logic embedded in the messages it receives.

With the help of Radu Moldovan, Ph. D. and post-doc at Carnegie Mellon, and his expertise in microorganism behavior, we are evaluating how well this model applies to one of the simplest, but well studied worm called *c. elegans*. This study is comparing how well our proposed model can describe the different roles nervous cells are playing in processing the information during different activities, including reacting and learning from mechanosensory input, chemosensory input, and thermosensory input.

We are closing our paper by proposing a high-level implementation for this generic architecture that uses open source components written in Java that run on a J2EE platform. This is drawn from our two decade-long experience in designing and building complex enterprise applications.

This generic architecture for robotic systems when applied can not only reduce by orders of magnitude the time to develop an architecture, but it can have a big impact when it is upgraded. This is because new features and changes can be added without the need to re-architect the system, which is the main cause of high cost and long development cycles.

Presenting author biography

Vasile Coman is a principal with XCLSoft. He has over 18 years experience in software development, six years as aeronautical engineer, and almost four years as management consultant. He held technical, consulting and management positions with companies like Spyglass, eSkill, Instrumentation Lab, Cigna, and PRTM. He received an M.S. in Aeronautical Engineering at the Polytechnic Institute of Bucharest, Romania. He published articles in the enterprise architecture and dynamic business applications field and made a presentation of the dynamically-stable enterprise concept at a conference in the complex systems. His interests include the design and the architecture of complex systems that are dynamic, including robots, machine vision, processing information from sensors, and system design and implementation.