AC 2010-992: DEVELOPMENT OF A GENERIC COMMUNICATION SERVICE BETWEEN PROGRAMMABLE LOGIC CONTROLLERS AND PERSONAL COMPUTERS USING MICROSOFT ROBOTICS DEVELOPER STUDIO FOR DATA COLLECTION IN AUTOMATED AND SEMI-AUTOMATED MANUFACTURING PROCESSES

Jose Gutierrez, Oregon Institute of Technology
Bachelor of Science in Mechatronics, ITESM, MX

John Anderson, Oregon Institute of Technology

David Culler, Oregon Institute of Technology

Development of a Generic Communication Service Between Programmable Logic Controllers and Personal Computers using Microsoft Robotics Developer Studio for Data Collection in Automated and Semi-Automated Manufacturing Processes

Abstract

During the period of 1950-1990 productivity in United States industries increased 50 percent due to technological innovation. High-value-added products were a consequence of more efficient manufacturing processes and data processing equipment. It is predicted that in the next decades productivity increases will be largely due to the ability to add flexibility and improve infrastructure through the collection and management of product data. By achieving the goals set forth in this project, a valuable tool for educating future students will be added to our program and we will also continue to improve our laboratory facilities for applied research, education and industrial partnerships.

Microsoft Robotics Developer Studio (MRDS) is a Windows based environment used to easily create robotics applications across a wide variety of hardware already in the market and hardware yet to be built. These characteristics can help to face the major challenge that industry is facing: an industrial environment that brings together a wide range of computer systems, data acquisition systems, technologies, system architectures, operating systems, and applications. This variety makes it increasingly difficult for people and machines to communicate with each other, especially when they describe and format data differently. MRDS and its' compatibility with other Windows products can help with the task of bringing together large amounts of data and formatting it so it can be shared easily.

The primary motivation for working on this project is the lack of affordable open-architecture alternatives for teaching, trying out new ideas and interfacing existing and new devices and equipment into the current workplace. Furthermore, it focuses on the creation and development of a generic communication service for Programmable Logic Controllers (PLC) using Microsoft Robotics Developer Studio**,** which is free for personal, academic, and development use. Thus, it is an important added value to engineering academia, because it opens the door to other research areas and innovation that can be converted into highly efficient applications in manufacturing.

MRDS and Education: The benefits of the Know-How in the creation of generic services

This paper is justified for the actual tendency of all major computer companies, including BEA, Google, IBM, Intel, Microsoft, Oracle, SAP and Sun Microsystems that have adopted and supported the SOC computing paradigm, its technologies, and its features. [1][4][5][6][7]

Service-Oriented Robotics Computing is important not only in application software development but also in education. SOC-based robotics programming is easier than traditional robotics programming. Sponsored by the US Department of Education, a SOC-based robotics-computing curriculum is developed for high schools in United States. [2]

VPL provides an easy way to define how data flows between services. For neophyte programmers it is useful and straightforward to implement algorithms.

The opportunity area offered by MRDS is the capability to provide the CCR (Concurrency and Coordination Runtime) and DSS (Decentralized Software Services) components to develop and perform generic services for different types of devices like sensors, motors, displays, PLC's, etc. When VPL application starts, it can be find on the left side of the main window, all the services already developed and that are included ones you download the free version such as joysticks, simulation places, generic encoders, generic motors, analog sensors, etc. Fig. 1

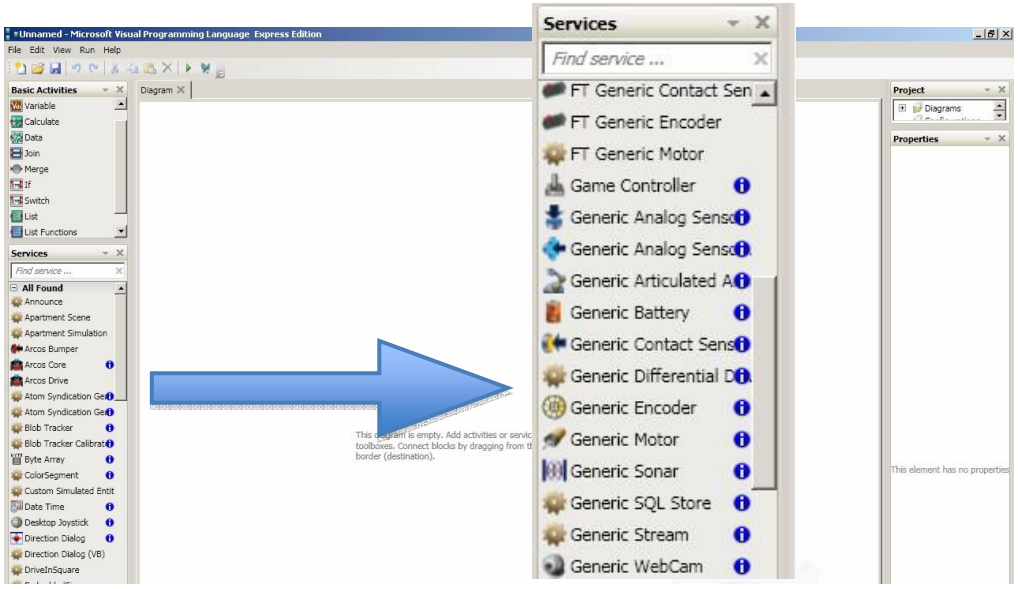

Fig. 1. Generic services in VPL

Because of the nature of MRDS and its proclivity in industry, is very important to start introducing to students an easy way to create generic services and how to use them in the VPL application. This can bring high schools and universities the huge possibility to create generic services for any kind of device (existing or new) in their workshops maintaining updated the laboratories and promoting the students to get familiarize with the new tendencies in software, automation and manufacturing processes.

The challenge in the development of this project is to have a guide and a good explanation of the Know-How in the creation of these services. Actually there are books and literature available in the web; however it is not simple for all the engineering backgrounds to understand how the architecture behind the services interacts or how to create one. Having a good description of the Know-How can bring to the academia and industry a new low-cost perspective in the implementation of automation.

Many robot manufacturers have moved their programming platform to VPL, including Coroware, irobot, kuka, LEGO NXT mindstorm, Parallax, Robosoft, Robotics Connection, Whitebox Robotics, etc. [3]

Brief introduction to MRDS

Microsoft Robotics Developer Studio (MRDS) uses Service-Oriented Computing (SOC) as the set of methodologies, concepts and principles to create robotics applications and services. The main components of MRDS are:

- The Concurrency and Coordination Runtime (CCR)
- Decentralized Software Services (DSS) comprise the run-time environment
- The Visual Simulation Environment
- Visual Programming Language (VPL)

.

The first two components are both manage libraries, so the robotics "services" that operate within their environments are also implemented using managed code. [3]

The Visual Simulation Environment is a 3D simulator with full physics simulation that can be used to prototype new algorithms or robots. The Visual Programming Language (VPL) is a graphical programming environment that can be used to implement robotic services. [3]

The Microsoft Visual Programming Language (VPL) is the application designed specifically to work with DSS services. The advantage of this application is that programs are defined graphically in flow diagrams.

CCR (the Concurrency and Coordination Runtime)

CCR (the Concurrency and Coordination Runtime) is a programming model for handing multi-threading and inter-task synchronization, whereas DSS is used for building applications based on a loosely coupled service model. [3]

Is a managed library that provides classes and methods to help with concurrency, coordination, and failure handling. The CCR makes it possible to write segments of code that operate independently. [3]

Basic Operations:

- When a message is received, it is placed in a queue, called a port, until can be processed by the receiver
- Enables segments of code to pass messages and run in parallel within a single process
- Each code can run concurrently and asynchronously
- Provides dispatchers, which determine what segment of code is currently running
- The number of threads is set according to the number of independent processors on the system, and code segments are scheduled when thread becomes available
- Eliminates many of the issues related to multi-threading, it uses its own threading mechanism, which is more efficient than the windows threading model. [3]

DSS (Decentralized Software Services)

The Decentralized Software Services (DSS) are responsible for controlling the basic functions of robotics applications. DSS is built on top of the CCR.

The principal tasks of DSS are:

- Start and stop services
- Manage the flow of messages between services via service forwarder ports

DSS itself is composed of several services that perform the following tasks:

- Load service configurations
- Manage security
- Maintain a directory of running services
- Control access to local files and embedded resources such as icons
- Provide user interfaces through web pages.

DSS Protocol

DSS uses a protocol called DSS Protocol (DSSP). DSSP is a simple SOAP (Simple Object Access Protocol) based application protocol that defines a lightweight service model with a common notion of service identity, state, and relationships between services. DSSP defines a set of state-oriented message operations that provide support for structured data retrieval, manipulation, and event notification. DSSP uses TCP/IP or HTTP, both of which are reliable transports. [3]

Services can run anywhere on the network, so DSS provides:

- A communication infrastructure that enables services to transparently run on different nodes using all of the same CCR constructs that they would use if they were running locally.
- An environment that makes CCR easier to use
- Multi-tasking applications require a multi-threaded application using windows threads and mutual exclusions, or mutexes, to prevent two threads from simultaneously attempting to update the same variable. [3]

SOC (Service-Oriented Computing) and MRDS

In 2006, Microsoft released the SOC-based Robotics Studio and VPL (Visual Programming language), which mark a milestone in SOC and robotics. [2]

SOC (Service-Oriented Computing) is a set of concepts, principles, and methods that represent computing in three parallel processes: service development, service publication and application composition. As Object Oriented Computing (OCC) is based on imperative computing, SOC is

based on OOC. In fact, current SOC development requires that each service is defined as a class and the instance of a service is a remote object connected through a proxy. Typically, an OCC application is developed by the same team in the same language, while SOC application is developed using pre-developed services created by independent service providers. [2]

A service is a container of messages or building blocks with certain attributes depending of the function that it performs. CCR enables segments of code to pass messages and run in parallel within a single process. The DSS library extends this concept across processes and even across different machines. [2]

A group of services form a CCR application that in combination with DSS enables these services to run on completely separate computers and communicate via network. By standardizing these services, all the community of devices can share code and data more easily. [2]

SOC applications are less efficient than OOC applications because of the additional layer of standard interface, which makes it possible for language and platform independent communication and remote invocation implemented in Web services. Embedded systems and robotics applications often need real-time performance, and thus, SOC was considered not a suitable paradigm for developing embedded systems and robotics. [2]

The facts are, SOC does not have to be implemented over the Web, and thus, the standard interface could be simpler and faster. The remote services could be migrated into a local machine to reduce the communication cost. On the other hand, the benefits of applying SOC in embedded systems and robotics applications are significant, particularly, for the following reasons:

- Embedded systems/robots have limited capacity to carry programs that handle all possible situations
- Unforeseeable environmental situations can occur
- Faults can occur and on-site repair is often not available
- Users want to modify the system (requirements) without stopping the system. [2]

An SOC application can be made independent of devices that the application communicates with, and thus, the same application can be applied to drive different devices. [2]

Fig. 2 shows an example of a robotics application

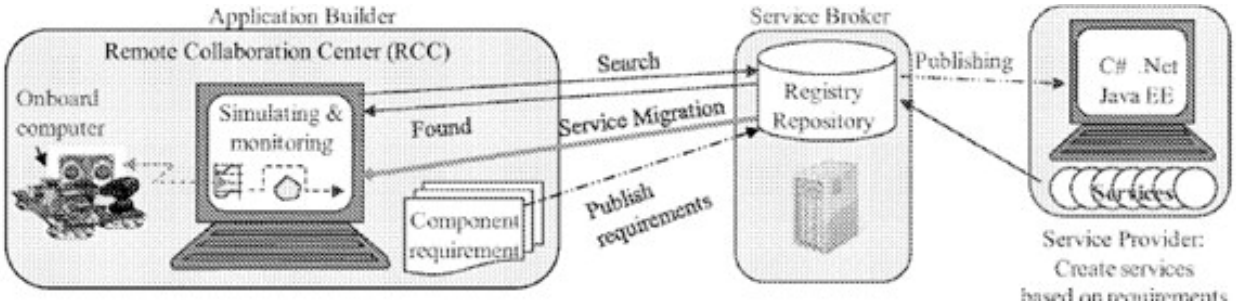

Fig 2. SOC-based robotics application development

Project Description

To create an application consisting in three contracts for a DL205 Programmable Logic Controller using Microsoft Robotics Developer Studio as the main communication platform. It is expected to demonstrate its functionality in an automated cell with the manufacturing of a product.

PLC & DSS application: The visual explorer application uses a PLC for communication with devices. The PLC is accessed via three generic contracts that describe the APIs (Application Programming Interfaces) for the Input Card, the Output Card and the Ethernet module. In fig. 3 it is shown the DSS application desired.

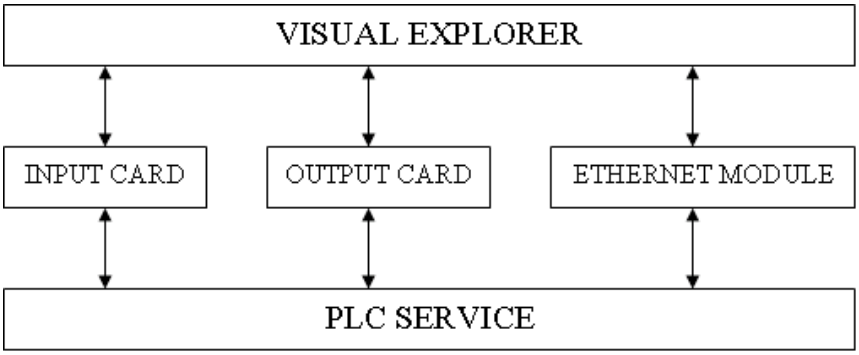

Fig. 3. Contracts for PLC service

Expected results

It is intended to develop two approaches:

Software:

- Microsoft Robotics Developer Studio as a platform for the development of a Generic Communication Service between Programmable Logic Controllers and Personal Computers. (Service Oriented Architecture approach)
- PLC programming (ladder diagram programming) for operation
- HMI programming (C-more panel programming) for monitoring and control
- Simulation with MRDS of the automated cell and control with services
- Generalized Control and Data Acquisition System/Data flow architecture

Hardware:

- Connection through a PLC and Ethernet module of an arm robot, conveyor, milling machine, vision system and bar code reader. This is with the objective to have all the devices communicating in a Local Area Network
- Basic electrical connections for operation and safety
- Design of the automated cell (location of devices)
- Feasibility of operation

The possible solutions for communication data flow between the PLC and the other devices of the automated cell (Fig. 4) are:

- Implementation of the devices available for the automated cell in a Local Area Network, with possible connection to other networks (like OIT network). Using TCP/IP protocols and wireshark software for monitoring of communication messages through the network. A switch will be the message repeater.
- Heart of data flow is the PLC & Ethernet switch.
- The possible implementation can be done using wireless connection instead of RJ-45 cable, following the new standard ISA100.11a for wireless in automation and monitoring processes.

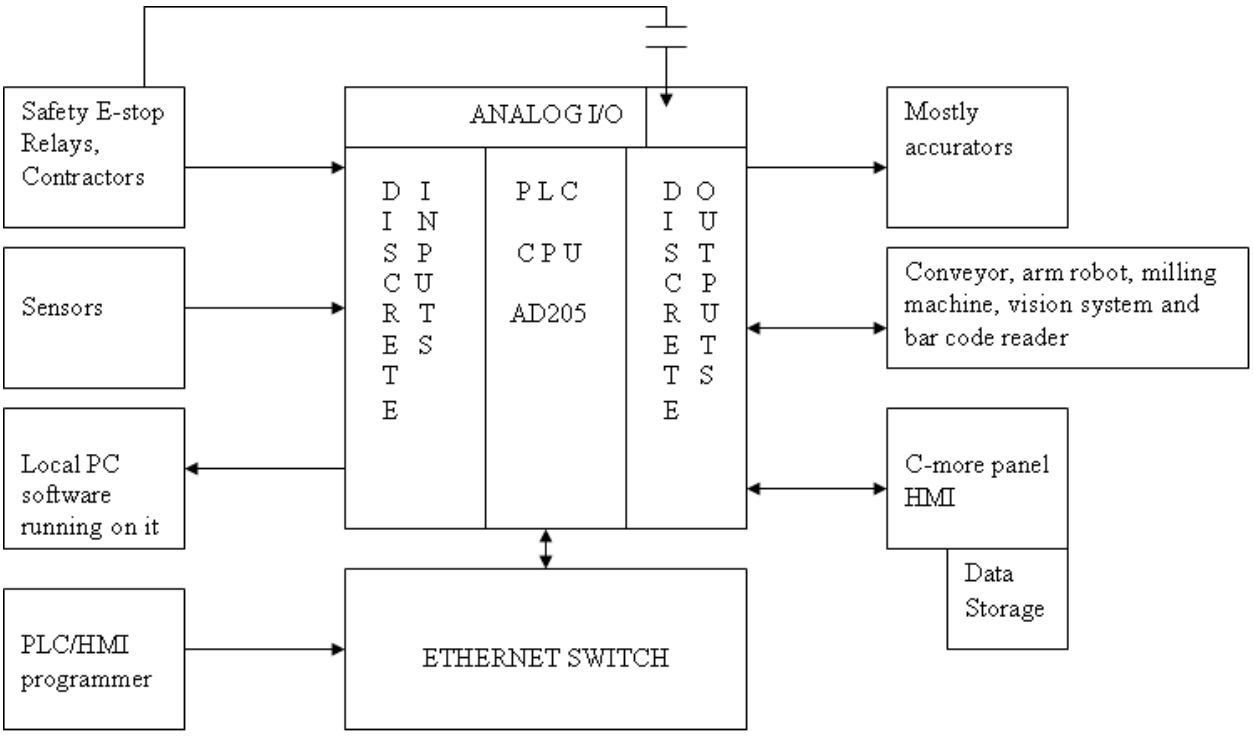

Fig. 4 Diagram of devices connections

So far the Local Area Network using a switch connected to the PLC and other devices is the actual solution for the implementation of the automated cell; however wireless communication can be an option once testing with the original idea works.

Conclusion

SOC is actually the tendency that big companies in information technology are using are supporting, so it is important for the academia to prepare students for the new trends n data communication technology. This attain several areas in engineering, for this reason is important for all the future professionals be familiarize with this kind of technology.

References

[1] M. Chang. J. He, Enrique Castro-Leon, Service-Orientation in the Computing Infrastructure, 2nd IEEE International Symposium on Service Oriented System Engineering, Shanghai, Oct 2006, pp 27-33
[2] Chen, Y. & Bai, X. (2008), On robotics applications in Service-Oriented Architecture, The 28th International Conference on Distributed Computing Systems Workshops proceedings; Beijing, China, 551-556
[3] S. Kant Vajpayee (1995), Principles of Computer-Integrated Manufacturing, Prentice Hall, pp 308-316
[4] Jason Clark, inside "indigo"--infrastructure for Web Services and connected applications, Microsoft Press, April 2005
[5] F. Heinemann & Rau. C. Web Programming with the SAP Web Application Server. SAP Press. 2003
[6] INTEL, Service Oriented enterprise, the technology path to business transformation. 2005, www.intel.com/business/bss/technologies/soe/
[7] C. McMurtry, Mercury, Marc & Watling Nigel. Microsoft Windows Communication Foundation: Hands on, Sams Press, 2006.