Intelligent Agent Based CLAS Drift Chamber High Voltage System Controls

Vardan Gyurjyan, Tanest Chinwanawich, and Amrit Yegneswaran
Physics Division, Thomas Jefferson National Accelerator Facility, Newport News, Va 23606
April 29, 2004

This note details the design of the intelligent agent[1]-based, network-distributed CLAS drift chamber high voltage (DCHV) system control (Syscon), which at the agent level of abstraction, considers agents as atomic entities that communicate to implement Syscon's functionality, builds on distributed control and application entities that collaborate dynamically to meet physics experiments' control objectives, and creates a real time, auto-extensible system in which completely heterogeneous processes coexist and communicate with each other as peers – simplifying organization of algorithmic control and feedback mechanisms to achieve safe and efficient control of the experiment. Agents' engineering aspects are addressed by adopting the domain independent software standard formulated by the Foundation for Intelligent Physical Agent (FIPA).

The CLAS drift chambers (DC) are powered by three SY527 CAEN power supplies (CPS); each CPS has ten modules; modules have twenty-four channels, each of which powers ~150 DC wires.

CPS can be accessed by the front-panel keypad, the terminal (using a menu-driven CAEN native software), or by the workstation that hosts Syscon. CPS communicate with the workstation either via RS232 or CAENET[2] (Fig. 1).

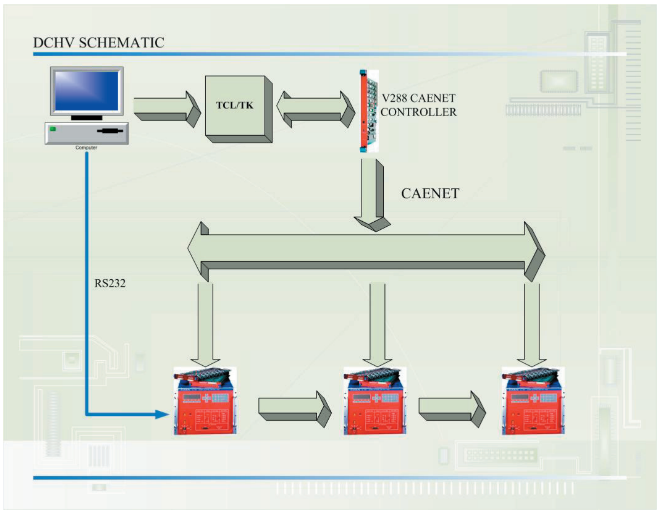

FIG. 1. System Configuration of present control system.

The current Syscon has a two-tier client server architecture. Client, written in TCL/TK interfaces the controls and monitoring system (Cosmos) on a Unix workstation with the V288 CAENET VME controller, a VME module PPC (dccntrl). The CAENET server, written in C, which resides in the PPC – a single board computer that runs VxWorks RTOS, interfaces the controller with the CPS. The communication protocol between client and server is CODA TCL/DP. Except for the CAEN firmware, running inside the CPS, all components were developed locally – simplifying program maintenance.

Operations parameters such as maximum current, voltage, and current resolution are downloaded on to the CPS through the VME interface using the communications port.

The current version of Syscon is stable and meets requirements. However, the architecture is inadequate to meet new CLAS demands and to support features such as distributed control, distributed monitoring, and dynamic histogramming; to be able to do this requires using the new control hardware developed by CAEN and a rewrite of Syscon using progressive programming paradigms and architectures.

The deliberated design is simple and provides a clear separation between application and presentation layers. Another advantage is that client and server operate in the same Unix environment, which allows use of advanced communication protocols capable of exchanging entire data objects, and which is tolerant to programs and system failures.

The proposed PC, open architecture[3], and intelligent agent based system will integrate the data acquisition (DAq) system and the DCHV Cosmos in the homogeneous multi-agent environment and will be able to cope in real time with dynamic reconfigurations of the control environment by auto-generating or specializing specific software agents, whenever new controls are added or control relationships are changed.

Software agents have their own thread of control, localizing code and state, and self-defining when and how to act. In an open and distributed, agent-based, integrated control environment, specifications are needed for ensuring interoperability of the autonomous agents; minimum requirements are:

- common communicative mode for agents to exchange information and delegate control tasks
- facilities by which agents can locate each other
- unique method for agent identification
- method for interacting with users
- method for migrating from one platform to another

FIPA, the most promising standardization effort in the software agent world [4] was selected to provide the normative framework within which agents can be deployed and can operate. FIPA specifications establish the logical reference model for the creation, registration, location, communication, migration, and retirement of agents. FIPA standards only specify the interface necessary to support interoperability between agent systems; the standards do not prescribe the internal architecture of agents or how they should be implemented. The FIPA agent platform (AP) suggests mandatory components or normative agents:

- Directory facilitator (DF) – provides "yellow pages" services to other agents. Agents may register their services with the DF or query the DF for information on other agents. An AP can have multiple DFs providing the possibility for creating software agent communities or domains of agents/virtual clusters with their specialized function. DF's can register with each other forming a federation of domains.

- Agent management system (AMS) – provides agent name services ("white pages") and maintains an index of all agents, who currently are registered with an AP, exerts supervisory control over access to and the use of an AP, and is responsible for creation, deletion, and migration of agents.
- Agent communication channel (ACC) – the message transport system, which controls all exchanges of messages within the platform, as well as messages to and from remote platforms.

All agent communication is performed through message transfer. Message representation is based on the Agent Communication Language (ACL) formulated by FIPA [5]. ACL has well-defined syntax, semantics, and pragmatics, and is based on speech act theory that has two distinct parts: communicative act and content of the message.

Communicative acts have a precise, declarative meaning independent of message content and extend any intrinsic meaning that the message content itself may have.

From a variety of FIPA specification implementations such as ZEUS, JADE, GRASSHOPPER, MOLE, RETSINA, FIPA-OS, JADE was selected because it simplifies agent-based application development, while ensuring standard compliance through a comprehensive set of FIPA services and agents [6].

The control agent (CA)[7] platform includes FIPA specified mandatory agents (ACC, AMS and DF) provided by JADE. JADE core Java classes are used to implement FIPA specifications.

The software architecture of the system, Fig. 2, is based on the coexistence of several Java Virtual Machines (JVM), which communicate with each other through Java Remote Method Invocation (RMI). CAs in the same domain share the single JVM, which plays the role of a basic agent container[8].

The system's architectural structure is based on hierarchies of agent containers dispersed over the network. CAs are dynamically created and grouped in virtual clusters that, as well as agent containers, can be created or destroyed as needed within the CA platform.

The Front-End is a special container running the FIPA normative agents and high level mediator agents, which take care of CAs' management and overall system coordination. The Front-End container maintains an RMI registry internally, Fig. 2, and is used by other agent containers to register them with the Front-End and to join the CA platform.

A special, lightweight, extensible markup language (XML) and resource definition framework schema (RDFS) developed by the WWW Consortium [9], based control oriented ontology markup language (COOL) has been developed to assist agents in sharing and annotating control specific information to support the heterogeneous nature of Cosmos and information resources – a language which enables agents to understand and develop domain specific services and devices'

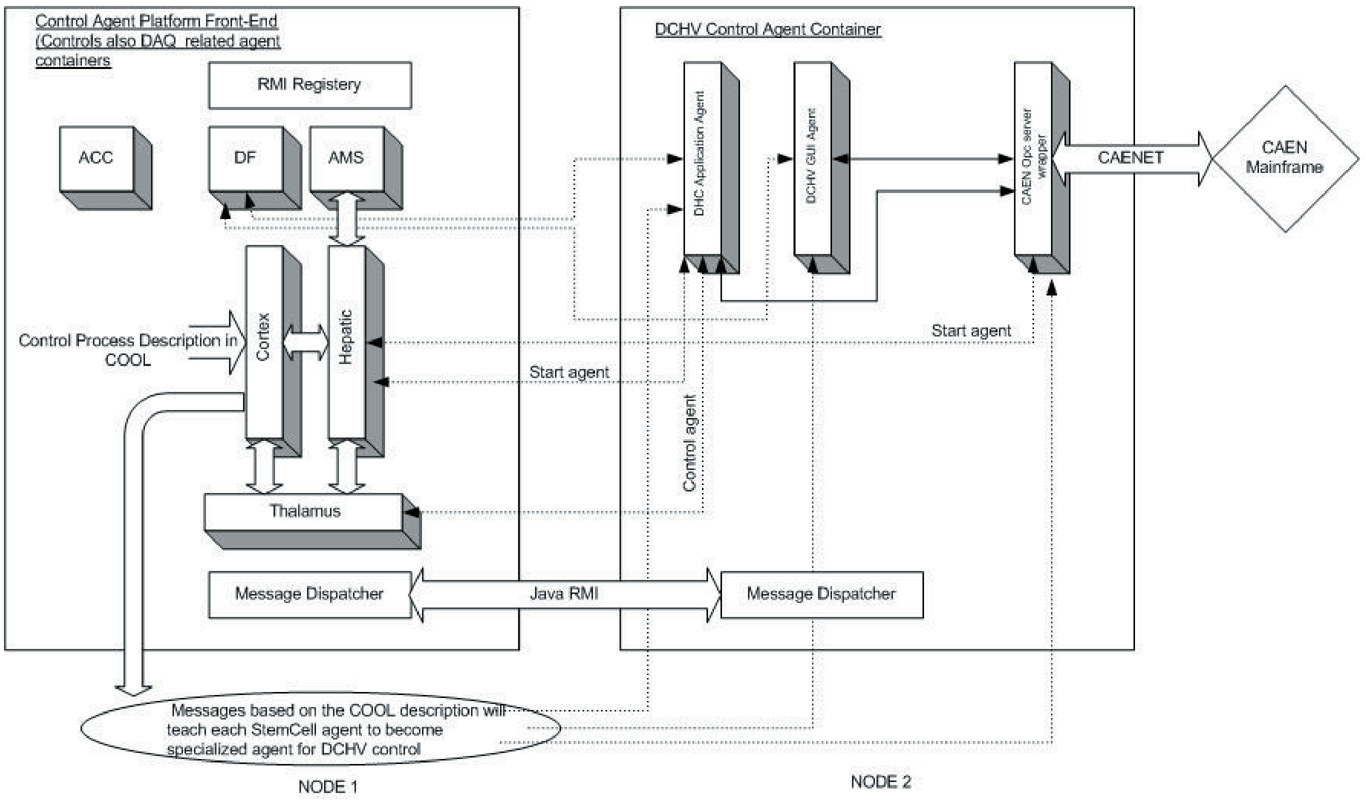
Control Agent Platform Front-End (Controls also DAQ related agent containers
RMI Registery
ACC
DF
AMS
Cortex
Hepatic
Control Process Description in COOL
Thalamus
Message Dispatcher
Java RMI
DCHV Control Agent Container
DHC Application Agent
DCHV GUI Agent
CAEN Opc server wrapper
CAENET
CAEN Mainframe
Start agent
Start agent
Control agent
Message Dispatcher
Messages based on the COOL description will teach each StemCell agent to become specialized agent for DCHV control
NODE 1
NODE 2

FIG. 2. DCHV Control system agent..

requirements, and standardize the description of the DCHV Syscon data processor. A node and arc diagram of the process description part of COOL is shown in Fig. 3.

Using COOL, information in Syscon can be expressed in a precise, machine-interpretable form so that the agents participating in the conversation understand meaning of terms that describe specific controls.

COOL uses RDFS to define hierarchical descriptions of concepts (classes) in a control specific domain and is extensible via XML-namespace and RDFS based modularization.

Properties of each class describe features and attributes of the control concept. Logical statements describe relations among concepts. COOL has a set of predefined instances to simplify knowledge base development.

A set of classes to define GUI components and their relations have been designed. Some of the Java swing classes are mirrored in COOL to simplify the GUI design for the specific control process. COOL descriptions of DCHV Cosmos, knowledge base, are integrated in DAq at runtime, without actual programming.

The created control knowledge base saved in RDFS format is accessed by the cortex, the specialized high-level mediator agent. The cortex agent is the interface between a specific Syscon designer and global Syscon, and enables the integration of the controls provided by non-agent software into a multi-agent control community. The cortex auto-generates wrapper agents and necessary ontology classes by parsing COOL specific Syscon descriptions – DCHV control processes. Agents on the control platform relay messages to the wrapper agent and have them initiate action over the CAENET network. Figure 4 shows the main design architecture components.

The Hepatic high level mediator agent is responsible for the AP management and recovery processes – creation, recovery, and removal of the agents, agent clusters, or entire containers, and resurrection of any fallen agents or containers, thus achieving AP stability and fault tolerance.

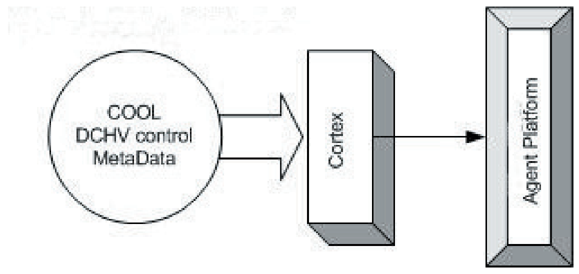

FIG. 4. Design architecture of the DCHV system.

Coordination of the agent cluster is accomplished by the Thalamus that ensures that partial, local solutions to control problems are integrated into the global framework.

To conclude, the new Syscon design based on intellegent agent technology will meet new CLAS demands. Currently, a prototype based on FIPA compliant JADE AP has been tested; COOL control process abstraction has been implemented, allowing description and integration into the general control environment of all control processes; high-level mediator agents specialized in AP management and system coordination have been developed, increasing the Syscon's reliability and fault tolerance and external software or systems integration based on COOL description files has been accomplished by the special mediator agent responsible for auto-generating wrapper agents on the platform. A new GUI is under development.

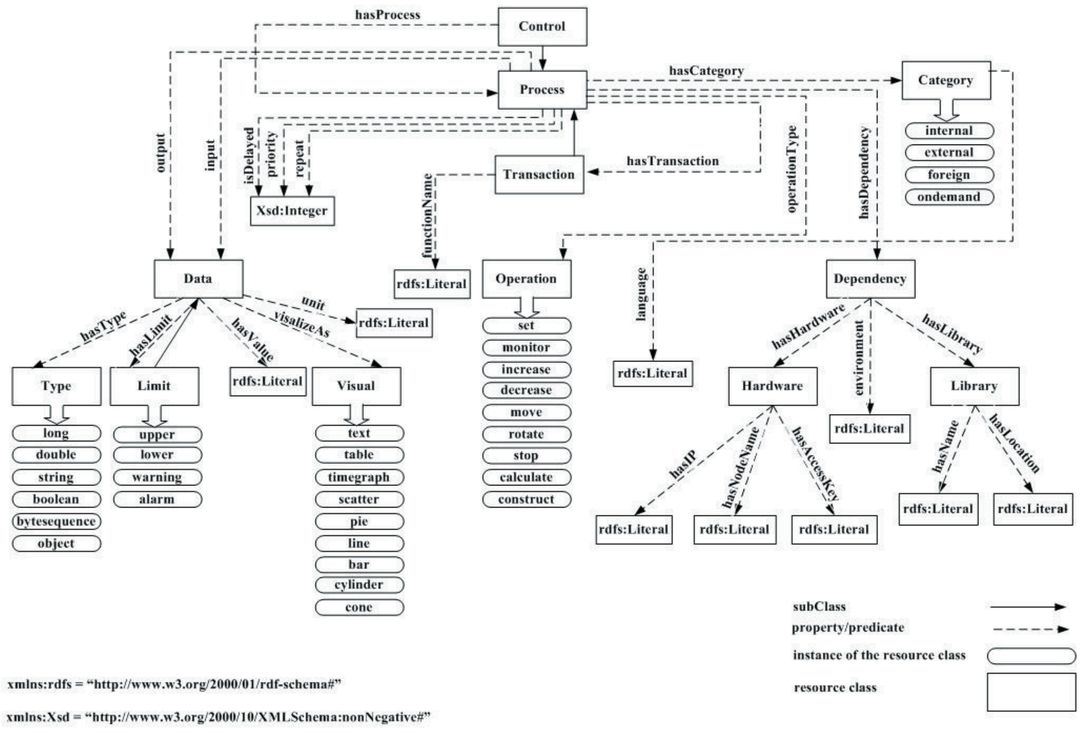

FIG. 3. COOL schema and taxonomyplatform.

[1] An agent is a software entity capable of acting intelligently on behalf of a user to accomplish a given task. Agents are capable of addressing both knowledge processing and control specific actions simultaneously in a real-time distributed environment. A society of agents can combine their efforts to achieve a goal.

The characteristics of agents are:

• autonomy

• proactive intelligence (agents do not simply act in response to their environment, but are able to take initiative)

• temporal continuity (they are continuously running processes)

• mobility, rationality/benevolence (agents don not have conflicting goals)

• adaptive intelligence (agents have the ability to learn)

[2] CAENET allows up to 100 crates to be daisy-chained, permitting 24,000 channels to be controlled and monitored by a single VME V288 CAENET interface.

[3] Presently, Syscons in high energy and nuclear physics experiments are becoming more heterogeneous and diverse. Increasingly proprietary industrial Syscons with their specific hardware are being used and proving to be reliable. PC use, as it is inexpensive and extensible, is accepted and supported as is development in the MS Windows environment. Cosmos and DAq problems with interoperability, scalability, and standard user interface are resolved by the open architecture design – an open architecture design adds flexibility by shifting the focus from hardware to software.

[4] Foundation for Intelligent Physical Agents. Available at http://www.fipa.org

[5] FIPA ACL Message Structure Specification. Available at http://www.fipa.org/specs/fipa00061

[6] Java Agent DEvelopment Framework. Available at http://sharon.cselt.it/projects/jade

[7] Control agents (CAs) are active objects, having more than one behavior and can engage in multiple activities simultaneously. To minimize the number of threads required to run the AP, each CA implements a scheduler, who carries out a round robin non-preemptive policy among all behaviors registered with the agent. Agent behaviors can be added or removed at run time.

[8] An agent container provides a complete run time environment for agent execution and allows agents to concurrently execute on the same host. Each agent container is a multi-threaded execution environment composed of one thread for every agent.

[9] K. Ahmed, D. Ayers, M. Birbeck, J. Cousins, D. Dodds, J. Lubell, M. Nic, D. Rivers-Moor, A. Watt, R. Worden, A. Wrightson. “Professional XML Meta Data”.Wrox Press Ltd.