A Software Architecture for Control of Value Production in Federated Systems

Jay S. Bayne
Echelon 4 LLC
Senior Member IEEE, ACM, ISA
jbayne@echelon4.com

ABSTRACT

Federated enterprises are defined as interactive commercial entities that produce products and consume resources through a network of open, free-market transactions. Value production in such entities is defined as the real-time computation of enterprise value propositions. These computations are increasingly taking place in a grid-connected space – a space that must provide for secure, real-time, reliable end-to-end transactions governed by formal trading protocols. We present the concept of a value production unit (VPU) as a key element of federated trading systems, and a software architecture for automation and control of federations of such VPUs.

Keywords: Federated Systems, Cybernetics, Intelligent Automation, Grid Computing

1. INTRODUCTION

The value produced by a process is a relative measure of its performance. The marginal value, V, of a product or process (i.e., service) p at time t may be represented by $V[p,t]$. This marginal value is typically measured in terms of dollars per unit and is calculated as the difference between the fully burdened cost to produce $C[p,t]$ and the market clearing price $P[p,t]$ at the point (time and place) of sale. Production cost $C[p,t]$ is a function of several variables, including costs of raw materials, production capacity, innovation, capital, and burdened labor resources. Selling prices $P[p,t]$ are typically computed in terms of margin requirements, volumes, competitive situations (e.g., discounts), inventory and shipping costs, and market cycles. Recognizing the lag between production and sales (dt), the marginal value of a product or process may be expressed as

$$V[p,t+dt] := P[p,t+dt] - C[p,t]$$

The viability of a product p generally requires that it provide positive margins ($V[p,t] > 0$) over some finite interval T $>> dt$, stable or rising prices ($dP(p,t)/dt \ ¥ \ 0$), and falling production costs ($dC[p,t]/dt < 0$). Rising prices require continued investment in innovation to increase product features and benefits. Falling costs require lowering production costs, generally associated with gains in efficiencies resulting from investments in process or infrastructure. Both types of investment require deployment of financial assets derived from retained earnings or equity financing (e.g., borrowing or the sale of equity). It is not the purpose of this paper to discuss the complexity of marketing or pricing strategies or the financing of growth. Rather we seek to characterize the dynamics of the automation and control domain in which such trading systems must operate.

Products are created, produced and maintained by *value production units* (VPU). VPUs are dynamic, semi-autonomous logical entities that simultaneously serve two interdependent *value chains*:

- An *asset chain* where investors purchase equity in VPUs in exchange for gains from higher valuations resulting from increased production capacity or product capability, and
- A *supply chain* where producers and consumer trade in goods and services.

The interdependent activity of and control over of these two intersecting chains is the traditional purview of professional enterprise management. The primary goal of this work is to provide insights into the performance of this management "control domain" in order to implement higher levels of automation required to realize the promise of "real-time enterprise."

From this perspective, a VPU is a proxy for a generally complex, and often *ad hoc*, set of business activities spread among diverse and geographically dispersed organizational entities. These entities may be wholly owned, investment partners, or independent members of trading webs. In any case, a given VPU's role and relative position must be uniquely identified, as we discuss below. We propose a two-dimensional address space defined by its relative position in its respective asset and supply chains.

In the vertical (asset chain) dimension, investors at level $l+1$ allocate asset to VPUs at level l. In return, VPU's at level l provide returns on assets to investors at level $l+1$ and may invest in VPUs at level $l-1$. In the horizontal (supply chain) dimension, customers upstream at $k+1$ place demands for products from VPUs at location k. In return, VPUs at position k in the supply chain ship products to consumers at $k+1$ and may place upstream orders to VPUs at $k-1$. This cellular view defines a VPU as a four-port object, as diagrammed in Figure 1.
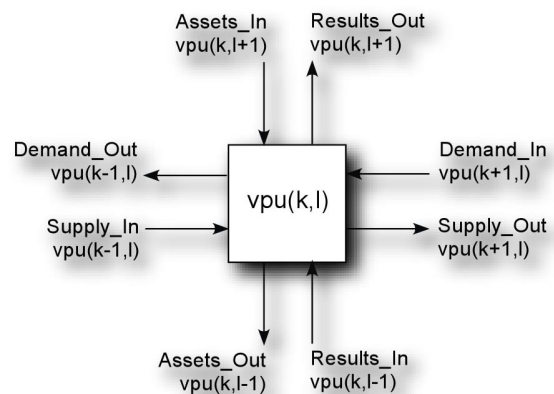
Figure 1 – Value Production Unit (VPU)

In this relative addressing scheme the marginal value generated by the sale of product p at time t produced by the VPU at cell

(k,l) may defined by the value generation function V[k,l,p,t]. Assets_In (e.g., equity financing) and Demand_In (i.e., sales of product) combine to produce Returns_Out (e.g., higher valuations) and Supply_Out (i.e., revenues resulting from product shipments). Figure 2 shows the value web in which VPU[k,l,p,t] functions.

We are interested in the class of VPUs that are capable of self-regulation and autonomic behavior – with their prime objectives being to attain and sustain viability over time. A system is viable to the extent that it is both reflexive (reactive) and driven (proactive), and is capable of evolving in order to adapt to a changing context. A viable system is real-time to the extent its survival depends on the timely execution of selected processes [JENSEN00]. Timeliness requires that end-to-end time constraints be met, and that system timing is predictable *a priori* within specified limits of accuracy.

Viability presumes VPUs are embedded in (typically dynamic) environments populated by other viable systems. The population must therefore know how to interoperate – to cooperate and to compete for finite resources. This requires existence of *rules of engagement* between systems, and between a system and the environment, sufficient to define an *operational commons*. This commons, in open *federated* environments, assumes Jeffersonian principles. It is sufficient to define the means for recognizing public and private resources (e.g., intellectual property) and standardized protocols (e.g., legal frameworks) for their acquisition, protection and use. VPUs operating in a federated system must, at minimum [BEER95, BAYNE96], exhibit the following characteristics:

- Be viable and identifiable members of a community;
- Be uniformly governed by the community's common laws; and
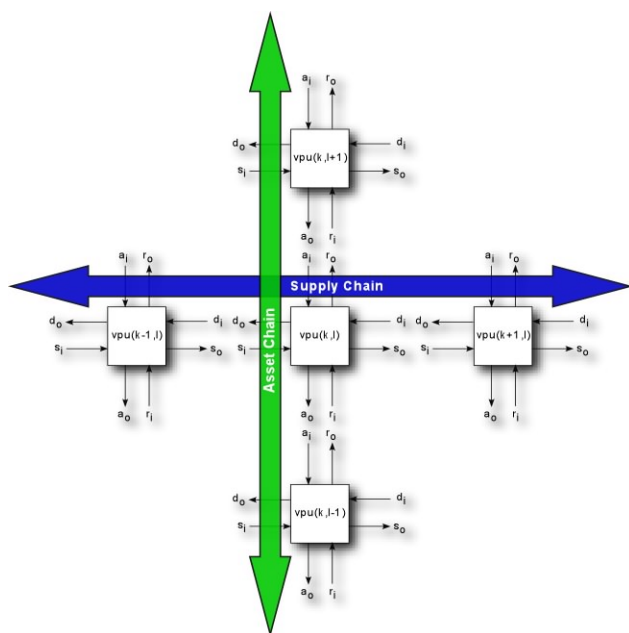- Provide their individual contributions to the good of the community.

Figure 2 – Value Production Web

A consequence of these a requirements is that the environment (i.e., metasystem) in which a VPU is immersed defines basic

standards of behavior and an interconnect structure (e.g., a computing grid) through which reliable communications may take place. At minimum, behavioral standards provide protocols for the creation, invocation, evolution and graceful termination (i.e., with minimal side-effects) of VPUs. Side effects, or aberrant behaviors, must be recognized and contained.

2. CYBERNETIC SYSTEMS

Internally, a VPU must contain logic that manages its production capacity and appetite for resources against often conflicting demands for product margins and asset returns. Key elements of this logic underwrite the accounting curricula of all business schools, but generally remain outside of the realm of automation. We propose mechanization of logic of value production based, in part, on the management cybernetics work of Stafford Beer and his associated "viable systems model" (VSM) [BEER95].

The VSM defines five echelons of control found in natural (e.g., biological) systems as exemplified in human neuro-anatomy. At the highest level (E5) is awareness and volition, where conscious behavior generation takes place. At E4 are innovation, development and planning, where the state of the external environment dictates forward-looking strategies for adaptation and evolution. E3 is internally focused and provides the operations function, where supervision and coordination of parallel lower level production processes is implemented. E2 are the regulatory controls, where the autonomic (i.e., sympathetic) behavior of individual production processes is managed. And at E1 exist the basic processes themselves.

By virtue of the VSM model, we can identify several key elements of a viable VPU. First, there are at least two E1 processes within any given VPU – the process responsible for participation in the value chain, and the process responsible for participation in the asset chain. Second, there must exist independent E2_xCP (where x := <A|C>) regulators of the two E1 processes. Third, there must be an E3:Ops directorate to coordinate (i.e., arbitrate between and synchronize) the two E2 regulators in their individual efforts at running the E1 processes. Fourth, for recognition of and adaptation to changing product and equity markets, the VPU needs an E4 function that drives innovation in the substance and manner of product and asset value creation, including evolution of the VPU's supporting infrastructure. And finally, the VPU requires a supervisory control function to maintain the VPU's identity, vision and mission and to supervise the ensemble behavior of the lower-level functions. The goal at E5 is maintaining homeostasis – dynamic equilibrium (i.e., viability) of the VPU.

3. VALUE PRODUCTION AS A VIABLE SYSTEM

By way of example, we present a simple commercial enterprise comprising a business area (BA) with its two subsidiary business units (BU). Figure 3 peers into one of the business units to uncover its VPU, as defined by its structure as a viable system. The figure identifies the essential internal objects that provide the dynamic behavior of value production as defined above. The two circles at the lower left contain the asset chain (ACP) and supply chain (SCP) processes responsible for engaging neighboring vertical and horizontal VPUs in their respective chains. These two echelon 1 components contain the fundamental operations of the viable system, the VPU's *implementation*. Notice that the circles do not include the E1

managements of these operations. They are considered services to fundamental processes and are included as part of the regulatory functions at E2.

With respect to generally accepted accounting practices (GAAP), a BU's ACP activities are recorded on its balance sheet, and its SCP activities are summarized on its income statement. The tradeoffs required in the management of these two views are the responsibility of the E5-E4-E3 management team. The `E1_ACP` management is focused on acquiring and deploying assets (e.g., infrastructure development), and the `E1_SCP` management is focused on the sale and delivery of products and services.
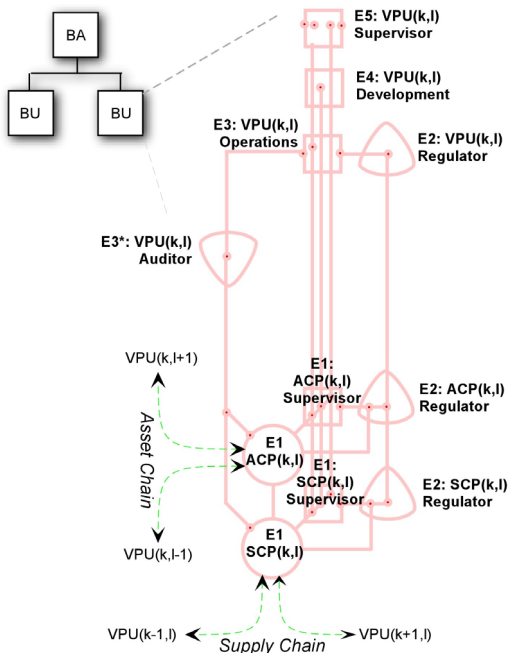

Figure 3 – Business Unit VPU(k, l)

There are eleven components identified in Figure 3 comprising the core objects of the VPU. They are `E1_ACP`, `E1_SCP`, `E1_ACP_Sup`, `E1_SCP_Sup`, `E2_ACP_Reg`, `E2_SCP_Reg`, `E2_VPU_Reg`, `E3_VPU_Ops`, `E3_VPU_Audit`, `E4_VPU_Dev` and `E5_VPU_Sup`. The lines connecting these objects represent grid-connections, allowing the objects to be distributed in arbitrary fashion. The message formats and protocols flowing between and among these objects are the subject of a subsequent paper, all defined in the spirit of work in community standards activities such as found at www.w3.org, www.jcp.org, www.omg.org, www.ietf.org and www.rosetta.org.

Figure 3 highlights several key elements of the dynamic behavior in a VPU. First, the regulatory loops for the ACP and SCP activities (e.g., `E1_ACP`, to `E2_ACP_Reg`, to `E1_ACP_Sup`, back to `E1_ACP`) define the need for a protocol machine and set of measurement and control messages. This control loop is patterned after classical feedback controllers, and is the subject of a separate paper.

Second, for predictability and stability reasons, regulatory loop timing requirements must be specifiable and consistent within a

VSM. The VSM within the `E1_ACP` process, for example, must adhere to timing requirements that do not conflict with those of its encapsulating `E2_ACP_Reg`. And those must not, in turn, conflict with the `E3_VPU_Ops` operations loop.

Third, the `E2_VPU_Reg` regulator must be able to prioritize and preempt operations in the `E1_ACP` and `E1_SCP` VSMs. This capability requires that policies and mechanisms exist to support coordination and synchronization, assisting `E2_VPU_Reg` with its role in damping oscillatory and avoiding deadlocked behaviors that may result from contention over shared resources.
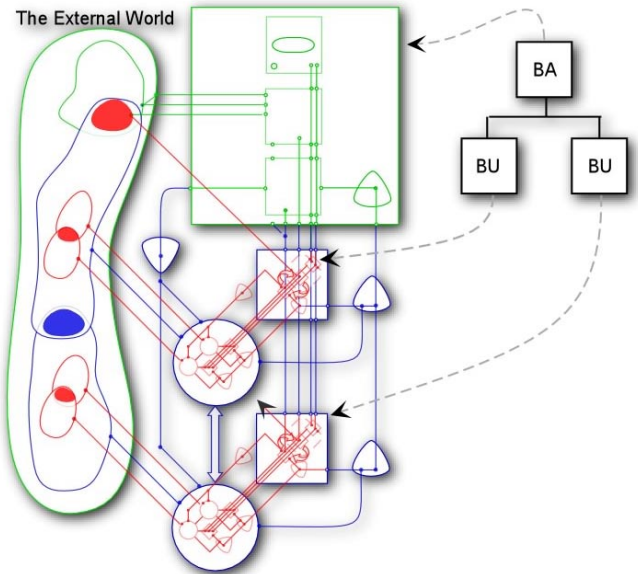

Figure 4 – Enterprise VPU

Finally, because the VMS is recursively defined, the object interfaces, protocols and message syntax must scale, and not be level specific. Figure 4 continues in the vain of Figure 3, expanding and coupling the VPUs of two BUs within the VPU defining the BA. Also shown are the equity and product market segments (environment) within which the BA operates, and how the three VPU's partition their views and orchestrate their responsibilities.

The figure shows the BU VSMs rotated 45 degrees in order to fit them into the BA VSM. Likewise, under a microscope we'd see the asset and supply chain VSMs within the BU's. And in a similar fashion, zooming outward, we'd find the BA embedded in a corporate VSM structure that may contain other BAs. This scoping works from the lowest enterprise VSM levels (e.g., a manufacturing cell within a factory within a BU up through alliances among corporations, entire vertical market segments, and national and global markets.

4. VPU OBJECT CLASSES

As noted above, there are eleven key components (i.e., `classes`) in the VPU object model, including `E1_ACP`, `E1_SCP`, `E1_ACP_Sup`, `E1_SCP_Sup`, `E2_ACP_Reg`, `E2_SCP_Reg`, `E2_VPU_Reg`, `E3_VPU_Ops`, `E3_VPU_Audit`, `E4_VPU_Dev` and `E5_VPU_Sup`. These components define the java "package

`com.echelon4.vpu."` In this section we outline the services of each class.

1.1 Asset Chain Process – `Class E1_ACP`

The asset chain process (ACP) is an enterprise object class (e.g., "enterprise java bean", EJB) defined by

- An *asset production model* deployed in a VPU
- A set of transactions (messages and protocols) that interface the model to an enterprise's underlying financial (e.g., ERP) systems
- A set of metrics (e.g., Six Sigma) for auditing the performance of the ACP
- A set of trading interfaces to the relevant financial markets

1.2 Supply Chain Process – `Class E1_SCP`

The supply chain process (SCP) is an enterprise object class (EJB) defined by

- A *supply production model* for products produced in and exchanged by the VPU
- A set of transactions (messages and protocols) that interface the model to an enterprise's underlying manufacturing (e.g., MRP or project management) systems
- A set of metrics (e.g., Six Sigma) for auditing the performance of the SCP
- A set of trading interfaces to the relevant product markets

1.3 Asset Chain Supervisor – `Class E1_ACP_Sup`

The asset chain supervisor provides direct administrative controls over the `E1_ACP`, and includes such services as

- Receive, interpret and execute commands from E3_VPU_Ops
- Formulated and send status of E1_ACP to E3_VPU_Ops
- Develop normative execution plans for E1 operations
- Supervise the regulatory actions of E2_ACP (e.g., operational set-point controls)

1.4 Supply Chain Supervisor – `Class E1_SCP_Sup`

The supply chain supervisor provides direct administrative control of `E1_SCP`, and includes such services as

- Receive, interpret and execute commands from E3_VPU_Ops
- Formulate and send status of E1_SCP operations to E3_VPU_Ops
- Develop normative execution plans for E1_SCP operations
- Supervise the regulatory actions of E2_SCP (e.g., operational set-point controls)

1.5 Asset Chain Regulator – `Class E2_ACP_Reg`

The asset chain regulator provides the feedback controls that support reflexive (autonomic) controls over behavior of the asset chain process. Through `E2_ACP_Reg`'s four interfaces,

- It directly senses and responds to activities in the E1_ACP process

- It couples the normative supervisory control functions of E1_ACP_Sup to the E1_ACP production process
- It connects to and supports the regulatory functions of the VPU operations level through E2_VPU_Reg
- It connects to the supply chain regulator E2_SCP_Reg in order to coordinate with actions of the E1_SCP process

As such, this function participates in four control loops, and is a critical element in the VPU's ability to attain and sustain homeostasis.

1.6 Supply Chain Regulator – `Class E2_SCP_Reg`

Like its `E2_ACP_Reg` counterpart, t he supply chain regulator provides the feedback controls that support reflexive (autonomic) controls over behavior of the asset chain process. Through `E2_SCP_Reg`'s four interfaces,

- It directly senses and responds to activities in the E1_SCP process
- It couples the normative supervisory control functions of E1_SCP_Sup to the E1_SCP production process
- It connects to and supports the regulatory functions of the VPU operations level through E2_VPU_Reg
- It connects to the supply chain regulator E2_ACP_Reg in order to coordinate with actions of the E1_ACP process

1.7 VPU Regulator – `Class E2_VPU_Reg`

The VPU encapsulates and coordinates the behaviors of value and supply chain production. As such it is responsible for managing the resources and synchronizing the events that are required for this role. A key function in support of that responsibility is coordination of the two process regulators, `E2_ACP_Reg` and `E2_SCP_Reg`. This role is performed by `E2_VPU_Reg`, whose services include

- Provide real-time status to E3_VPU_Ops relative to E1 activities and VPU objectives
- Accept "override" commands (e.g., set-point changes) from E3_VPU_Ops in response to E1 exceptions
- Balance the real-time demands of the E1 processes against one another according to E3 policy

1.8 VPU Operations – `Class E3_VPU_Ops`

The VPU, as [a proxy for] a responsive (i.e., reflexive and adaptive) value production entity within an enterprise, must operate according to plans that undergo constant revision. Such plans are the result of combining history, operational pragmatics, current objectives, resource constraints and incremental developmental improvements. It falls to the operations directorate to continually assess and rationalize these aspects and produce executable programs for the E1-level directorates. To do so `E3_VPU_Ops` requires the ability to independently assess current activities in its E1 processes (`E3_VPU_Audit`), react to real-time E1 events (`E2_VPU_Reg`), and to participate in the planning of incremental change (`E4_VPU_Dev`). Through these interfaces, `E3_VPU_Ops` provides the following services.

- Continuously receive, filter interpret and respond, through the exception-reporting services of E2_VPU_Reg, the real-time behavior of E1 production systems

- Continuously interrogate, interpret, filter and report to E4-E5, through the auditing services of E3_VPU Audit, the current status of specific E1 activities
- Periodically develop, revise, deploy (to E1_SCP_Sup and E1_ACP_Sup) and monitor tactical operating plans received from E5_VPU_Sup and E4_VPU_Dev that achieve the [typically, near-term] objectives of the VPU
- Support incremental "reprogramming" of the VPU in order to implement innovations provided by E4_VPU_Dev and mission directives from E5_VPU_Sup.

1.9   VPU Auditor – `Class E3_VPU_Audit`

The ability to objectively assess the current state of a system requires a set of uniform and consistently applied metrics. The `E3_VPU_Audit` class provides this capability. This is not a control function, nor is it meant to be intrusive in the affairs of E1 or E2 activities. Its primary role is one of establishing a uniform frame of reference for `E3_VPU_Ops` in carrying out its responsibility for achieving milestones in its operating plans. As such, the `E3_VPU_Audit` class provides

- A defined set of operational performance metrics (e.g., Six Sigma)
- A defined set of measurement methods
- A schedule for performing the methods within the E1 domains
- A reporting protocol for communicating the results to E3_VPU_Ops and to the E1_ACP and E1_SCP processes.

1.10   VPU Development – `Class E4_VPU_Dev`

For a VPU to remain viable through adaptation to the changing environment in which it operates it must objectively sense the state of that environment, sense the state of its own performance, and develop plans of adjusting to the changes these measurements suggest. This process of innovation must address three dimensions of VPU behavior:

- Its vision and mission (i.e., the objectives that result from its E5 belief system, and how it manages innovation through its E4 development programs)
- Its core capabilities as defined in its E1_SCP and E1_ACP processes
- Its infrastructure (i.e., how it functions through the services of E2 and E3), and

These dimensions must be captured in a coherent model of the VPU itself, against which the competitive environment is measured and adaptations are applied. These requirements are supported by E4 services that include

- VPU modeling and simulation
- Competitive environment measurement and assessment
- Recommendations to E5 on adjustments to vision and mission objectives
- Recommendations to E3 on operational improvements (e.g., infrastructure development)
- Recommendations to E1 on product and process changes

1.11   VPU Supervisor – `Class E5_VPU_Sup`

At any level of recursion, the ultimate point of accountability for the viability of the VPU rests with its superior authority, in our case the VPU supervisory level, E5_VPU_Sup. The responsibility of this authority includes

- Establishing and maintaining the identity, vision and mission objectives of the VPU (i.e., its reason for existence)
- Deriving and enforcing the policies (doctrines) that derive from the mission objectives
- Representing the VPU in the affairs of the metasystem in which it is an element (recursively defined)

The `E5_VPU_Sup` (together with the `E4_VPU-Dev` and `E4_VPU_Ops`) is, at the next higher level, the `E1_x_Sup`, where x is defined in the context of that level.

5.   SCHEDULING AND RESOURCE MANAGEMENT

For a system to be responsive to internal and external stimuli and to be capable of adjusting its behaviors in a direction that increases its survivability requires that its internal processes be "timely" in their computations. In this context, timeliness implies *able to meet completion time requirements* and that the timing characteristics of [selected] activities are *predictable*, and can be determined *a priori* to a specified degree of accuracy. In distributed enterprise systems this is tantamount to requiring that VPUs, and their supporting transactional computing systems (e.g., ERP, DBMS, MRP, CRM, etc), be able to schedule resources and processing activities in a manner that includes explicit timing requirements (e.g., start times, completion times or acceptable execution windows). The current state of technology does not in general support such semantics, and as a consequence the model of enterprise automation and control presented here requires new technology.

That technology, distributed real-time programming, is emerging in Java Community Program project JCP-50 Distributed Real-time Specification for Java (DRTSJ) [CLARK02] and in the Open Management Group (OMG) Real-time CORBA efforts [OMG02]. The specification and implementation of the eleven classes discussed in Section 5 is utilizing this new technology. In addition, the Global [Information] Grid Forum is progressing important work on distributed resource scheduling that is also critical to successful implementation of federated enterprise systems that are grid-connected. While the GridForum has yet to encompass real-time programming semantics, its current level of specification is acceptable for implementation of "best effort" style enterprise systems.

6.   CONCLUSIONS

We have introduced the viable system model of a value production unit, a recursively defined component (class) of an enterprise whose fundamental mission is the creation of value through coordinated engagements in asset and supply chains. This work is part of a larger effort at designing and developing grid-based automation and control software for distributed real-time enterprise. The technology and resulting products are primarily intended for use by enterprise development organizations (e.g., R&D, business development, venture capital, and M&A firms) seeking tools and techniques for

designing, constructing and operating real-time federated enterprise systems.

7. REFERENCES

BAYNE96   Bayne, J., *Metasystems*, ISA Conference Proceedings, 1996, www.echelon4.com

BAYNE98   Bayne, J., *Automation and Control in Large-Scale Interactive Systems*, IEEE International Symposium on Object-oriented Real-time Computer Systems (ISORC), Conference Proceedings, 1998, www.echelon4.com

BEER95   Beer, S., Brain of the Firm, Wiley, 2nd Edition, 1995

CLARK02   Clark, R., et al, *Distributed Real-time Specification of Java: A Status Report*, Revised 2002, www.real-time.org

GIG02   Global Grid Forum, 2002, www.gridforum.org

JCP02   Java Community Project 50, Distributed Real-time Specification for Java, 2002, www.jcp.org

JENSEN00   Jensen, E.D., *Utility Functions: A General Scalable Technology for Software Execution Timeliness as a Quality of Service*, Proc. Software Technology Conf., Utah State Univ., Apr 2000, www.real-time.org

OMG02   Open Management Group, *JCP RTSJ and Real-time CORBA Synthesis, Initial Submission*, OMG Document realtime/2002-06-02, 2002, www.omg.org