

Scale-free Enterprise Command & Control *Unified Command Structures*

Jay Bayne, PhD [§]

Echelon 4 Corporation
1045 W. Glen Oaks Lane
Suite 202
Mequon, WI 53092-3477
+1.262.240.2956
jbayne@echelon4.com

[§] Principal Contact

Raymond Paul, PhD

Department of Defense
OASD/NII
4502 7th St. NE
Washington, DC 20017
+1.703.607.0649
raymond.paul@osd.mil

Abstract

We are interested in the structure of enterprise governance in federated systems capable of supporting simultaneous, unified and time-bound objectives of self-directed (unilateral) and group-directed (multilateral) decision and control. Our solution requires a set of scale-free *joint enterprise command and control* (JEC2) services that provide allied teams of commanders, planners and operations personnel with collaborative, grid-based and real-time situation assessment, plan generation, and plan execution services. By *scale-free* we are referring to the ability of a system or service to scale from small to large applications – a design that is essentially independent of the scale of its deployment. The foundation of our unified JEC2 system depends on a coherent and scale-free view of an *enterprise* and characteristics of its underlying dynamic structure. Characteristics of unified JEC2 must, in addition, identify specific roles and responsibilities of the principal enterprise management *actors*. This paper, a companion of other ICCRTS papers¹, introduces our JEC2 *enterprise command framework* (ECF), a scale-free C2 system supporting unilateral and multilateral (collaborative) behavior among distributed federated systems [of systems].

Keywords

Unified Command; Joint Command; Enterprise C2

Introduction

In our treatment an *enterprise*² is modeled as *value production unit* (VPU), a system that is both self-directed and is able to collaborate (interoperate) for mutual benefit. Benefit may be realized along supply or asset *value chains*. Asset chains are defined by the *chain of command*, policy domain or accountability hierarchy of a *federation*³ that is responsible for the allocation of and associated returns on assets deployed in operating their value-adding supply chains. Supply chains are defined by the production and consumption of goods and services among allied enterprises and the marginal benefits derived there from.

¹ 8th CCRTS paper entitled "Performance Measurement for C2 Systems" and 9th CCRTS paper entitled "An Enterprise Command and Control Engineering Model," a 10th CCRTS paper entitled "Policy-based C2"

² An *enterprise* is an arbitrary unit of organization charged with production of a specific and quantifiable measure of value. The term *value production unit* (VPU) and *enterprise* are used interchangeably.

³ Sometimes referred to as a *policy or control domain hierarchy*

Figure 1 is a typical, albeit simplified, view of the *command* or *accountability* hierarchy of the DOD enterprise. It reaches from the highest level ("L5") of the President to the lowest level ("L0") of a warfighter or semi-autonomous piece of warfighting equipment. The highest levels are primarily focused on policy (i.e., strategy), the middle levels on operations, and the lowest levels on mechanism (i.e. tactics).

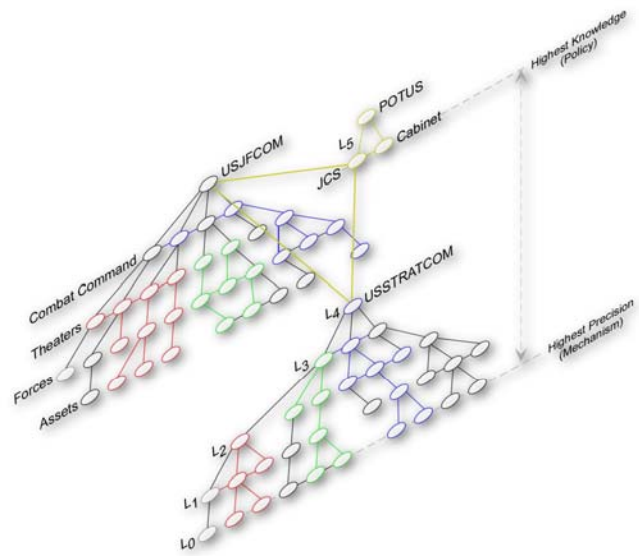


Figure 1 – DOD Command Axis

Figure 2 locates an enterprise subsystem within the DOD policy domain. Denoted as VPU[k,I], the enterprise operates within a specific community of interest (COI) and is bound into its associated value chains by producer-consumer relationships. Index "k" denotes the VPU's position along its supply chain, and index "I" denotes the VPU's position along its command chain. The lattice formed around VPU[k,I] is shown as planar, with single connections to its neighbors. Typically a VPU will simultaneously support fan-in and fan-out to multiple neighbors in each direction and serving multiple asset and supply chains, thus allowing it to operate concurrently in a complex mesh.

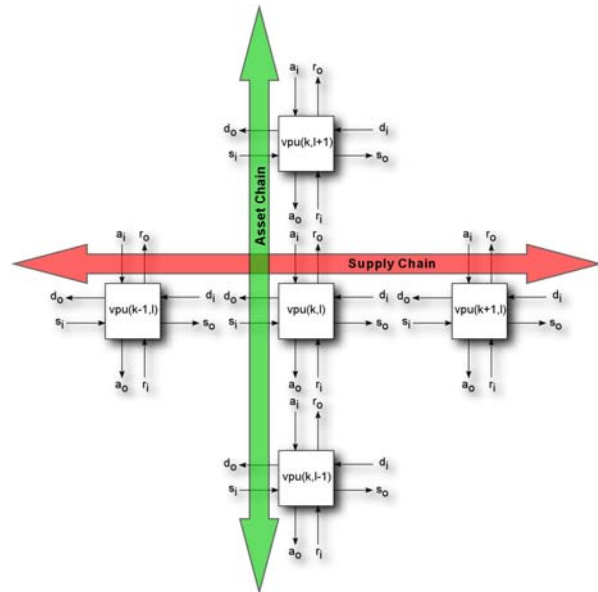


Figure 2 – COI Lattice

Timing in C2 Systems

An important requirement of scale-free systems is the necessity to operate in various time domains. This requires that temporal properties of the system be understood to the extent they may be parameterized and adjusted to accommodate end-to-end service completion time requirements, as determined at a given level of command (e.g., L3 in Figure 1). Such timing regimes can be quite complex and are central to our design. As such, our presentation begins with timing issues as a way to introduce the principle enterprise services and actors, and their roles in managing the "pace of play" for their respective VPUs. This will lead naturally to a discussion of the corresponding timing aspects of a JEC2 system as a whole.

Issues related to C2 timing in distributed systems arise from many sources and are treated from many perspectives. Contributors to the subject come from academia, industry, military, space science, communications and computing communities, to name a few. This paper is not intended to be a survey. Excellent references to the subject are available⁴, and

⁴ <http://www.real-time.org>; <http://shay.ecn.purdue.edu/~isorc05/>;
<http://asusrl.eas.asu.edu/srlab/activities/words05/words05.htm>;

new results appear nearly daily on a global basis⁵. For this presentation, we approach the subject from the perspective of the DOD's transformation to Network-Centric Warfare (NCW)⁶ requirements, in part defined by the communications paradigm of *task, publish, process, and use* (TPPU)⁷.

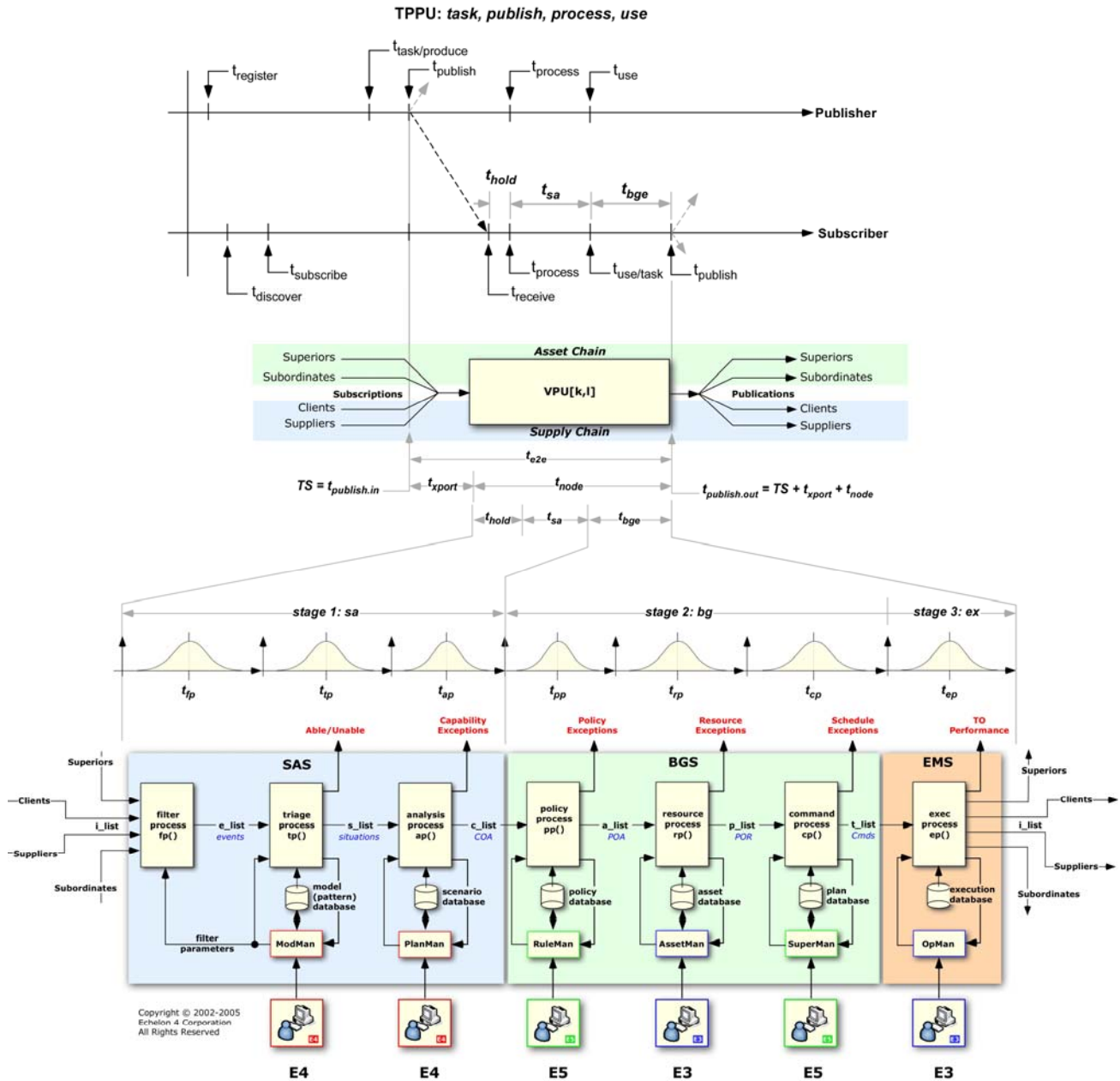


Figure 3 – C2 Timing Considerations

⁵ <http://cs-www.bu.edu/pub/ieee-rtts/Home.html>;
<http://www.springerlink.com/app/home/journal.asp?was=320e3xk1wq7qplaabw1h&referrer=parent&backto=linkingpublicationresults,1:100334,1>; <http://www.cs.york.ac.uk/rtts/>;
⁶ <http://www.dodccrp.org/research/ncw/ncw.htm>
⁷ <http://horizontalfusion.dtic.mil/about/net-c.html>

The TPPU protocol supports the DOD's move to packetized publish-subscribe communications throughout the global information grid⁸ (GIG) that allow organizations to more easily and effectively communicate both inside and outside of their own agency's *information silos*. The TPPU protocol also allows those communications to take place with a greater degree of parallelism and with correspondingly fewer delays resulting from the isolation of information sets. Furthermore, systems coupled by TPPU communications are potentially able to achieve greater agility, relying on their own timing properties and their own interpretation of raw data within their own local contexts.

Figure 3 introduces and summarizes several important temporal aspects of agile and scale-free C2, including 1) the general nature of TPPU protocol timing; 2) the relation between TPPU timing and the C2 processing stages in our enterprise VPU model; and 3) the nature of the information flows through these core processes. We describe each aspect briefly in the sections to follow.

TPPU Timing

The TPPU paradigm, as diagrammed at the top of Figure 3, generally unfolds as follows. Two entities, a service provider (the *publisher*, top line) and a service client (the *subscriber*, bottom line) meet in cyberspace. This meeting involves the publisher registering, at time $t_{register}$, its web services. The subscriber looks to a web services directory and, at time $t_{discover}$, finds the service it requires. The subscriber then subscribes, at time $t_{subscribe}$, to the desired service, gaining access to information products as they are periodically or aperiodically produced by the publisher. The publisher executes its internal tasks and produces its information products at time t_{task} , subsequently publishing these products at time $t_{publish}$. The publisher may continue to utilize these products in its process step to create further value ($t_{process}$) with other uses (t_{use}), perhaps to be published at a later time.

In the mean time, the subscriber receives the published products, after some arbitrary delay, at time $t_{receive}$. This delay can arise from transport and/or subscriber delays. Once received, the subscriber may also hold the information products prior to their use, modeled in Figure 3 as t_{hold} . Following any such hold time, the subscriber begins its TPPU consumption process step at $t_{process}$, eventually completing its own processing of the information products at its TPPU *use* step, beginning at t_{use} . Eventually the subscriber produces its own information products that may subsequently be published at time $t_{publish}$.

In the context of our enterprise C2 model, the subscriber's *process* step corresponds to the *situation assessment* (t_{sa}) phase of C2, and the subscriber's *use/task* step corresponds to the *behavior generation* (t_{bg}) and execution (t_{ex}) stages of C2. Taken together, command and control activities within VPU node [k,l] take a total time of t_{node} , the sum of t_{hold} , t_{sa} , t_{bg} , and t_{ex} .

In total, the end-to-end TPPU timing of a node in a C2 lattice, as shown in Figure 2, is given by

$$t_{e2e} := t_{xport} + t_{node}, \text{ where} \\ t_{node} := t_{hold} + t_{sa} + t_{bg} + t_{ex}; t_{sa} := t_{fp} + t_{tp} + t_{ap}; t_{bg} := t_{pp} + t_{rp} + t_{cp}$$

These relations contribute to the development of unilateral and multilateral plan completion time semantics for the COI members that collaborate on joint command and control.

⁸ <http://www.dtic.mil/whs/directives/corres/pdf2/d81001p.pdf>

Enterprise VPU Model

The central element of Figure 3 is our enterprise (VPU) node. Each such node operates on the two value production axes shown – the asset (or command) axis and the supply axis. The command axis couples superiors and subordinates to the VPU, while the supply axis couples the VPU to its customers and its suppliers. In a companion paper⁹ we present a more complete discussion of the VPU and its role in grid-based real-time C2. For our purposes here, it should suffice to note that a) VPUs are coupled in the grid (e.g., GIG) through publish-subscribe services operating under TPPU semantics, and b) management of their individual end-to-end performance is a key management objective of the VPU's management team.

C2 Process Timing

The more detailed base element of Figure 3 enumerates the core functions of C2, expanding and making more explicit the functions of VPU[k,l] in transforming its inputs and producing its outputs. This is our seven stage “process of doing C2.” The stages and their functions are summarized in Table 1. To effectively manage the end-to-end behavior of their enterprise, the VPU management team, i.e., its commander, navigator (aka, planner-analyst), and executive officer, must possess tools competent to govern the progress of their *situation assessment*, *behavior generation* and *plan execution* activities. Construction of such tools requires a formal yet operational model of each C2 stage, models that admit to specific degrees of control by specific actors. These degrees define important aspects of scalability in our JEC2 system.

Table 1 – CPF Stage Functions & Flows

CPF Stage	Step	Function & Flow
SAS	Filter Process	Receive all messages from valid subscriptions, decode and sort all messages into classes (orders, information, alarms, etc.), ordered by publication time and publisher ID, and produce an event list (<i>elist</i>) for input to the Triage Process
	Triage Process	Receive the <i>elist</i> and, based on the current situation and the currently active plans of record, determine which information and events apply to known situations and which are new. Selectively ignore non-critical new situations; create a situation list (<i>slist</i>) and send it to the Analysis Process
	Analysis Process	Receive the <i>slist</i> and look for preplanned scenarios with which to respond. If present, adjust the scenarios to the current conditions. If none exist, create a new scenario to handle the new situation. Send the list of feasible responses (<i>clist</i>) to the Policy Process in the form of one or more possible courses of action (COA).
BGS	Policy Process	Receive the <i>clist</i> and evaluate the plans for compliance with extant policies. If compliant mark the plan as viable, if not evaluate risk and/or adjust the plan to allow compliance, if possible. If not possible, abort the plan. Forward all viable plans to the Resource Process in the form of an actionable list (<i>alist</i>).
	Resource Process	Receive the <i>alist</i> and attempt to assign needed resources. If resource conflicts exist between the <i>alist</i> plans, or between <i>alist</i> and currently executing plans, create one or more resource assignment schedules that allow for the greatest potential utility to the VPU. Forward as new plans of record (POR) with appropriate schedules to the Command Process in the form of a plan list (<i>plist</i>).
	Command Process	Receive the <i>plist</i> and finalize the optimal schedule based on the current situation, the <i>plist</i> plans, and the current status of all resources. With a valid plan, authorize new tasking orders and issue them to the Execution Process in the form of a task list (<i>tlist</i>).
EMS	Execution Process	Receive the <i>tlist</i> and assign the task steps to capable subordinates, clients, suppliers and/or superiors. Continuously monitor the execution and adjust or issue new elements of the <i>tlist</i> as execution steps complete. Report on progress of <i>tlist</i> orders.
	Performance Measurement Process	Not shown in Figure 3. Introduced and discussed relative to Figure 8.

⁹ “Policy-Based C2”, 10th ICCRTS, C2 Policy Session

To emphasize the probabilistic nature of control processing, Figure 3 highlights per-stage completion time distributions that individually contribute to overall t_{node} timing. Each distribution represents a constraint and objective for the node's management team – the management of per-stage throughput (e.g., yield, productivity, and performance). In effect, management of these distributions (i.e., their first and second moments) is management of the “pace of play” of the VPU and the community (COI) in which it participates. While not discussed in this paper, our treatment of this topic is based on time-utility functions (TUF) and utility-accrual (UA) scheduling theory¹⁰.

Concepts introduced in Figure 3 help in explaining the roles and responsibilities of federated VPU management teams, especially those actors identified at the bottom of the figure as E3, E4 and E5. Our concept of scale-free C2 systems is predicated on the notion of a control model that scales in a manner supporting unification of command, regardless of whether it functions at the lowest tactical or highest strategic levels of the command hierarchy. We refer to such a control model as the *enterprise command framework* (ECF).

Enterprise Command

In our JEC2 model, the behavior of each VPU is governed through its own local autonomous enterprise command structure. Figure 4 is a diagram of our ECF command model, and introduces the principal actors responsible for guiding its behavior. These per-VPU actors include a single *commander* (or supervisor, denoted as *echelon five*, E5) representing the highest authority within the VPU, a single *navigator* (or analyst, denoted as *echelon four*, E4) responsible for modeling, planning and analysis functions (i.e., adaptation and change management), and a single *operator* (or operations executive, denoted as *echelon three*, E3) responsible for the execution of authorized plans of record.

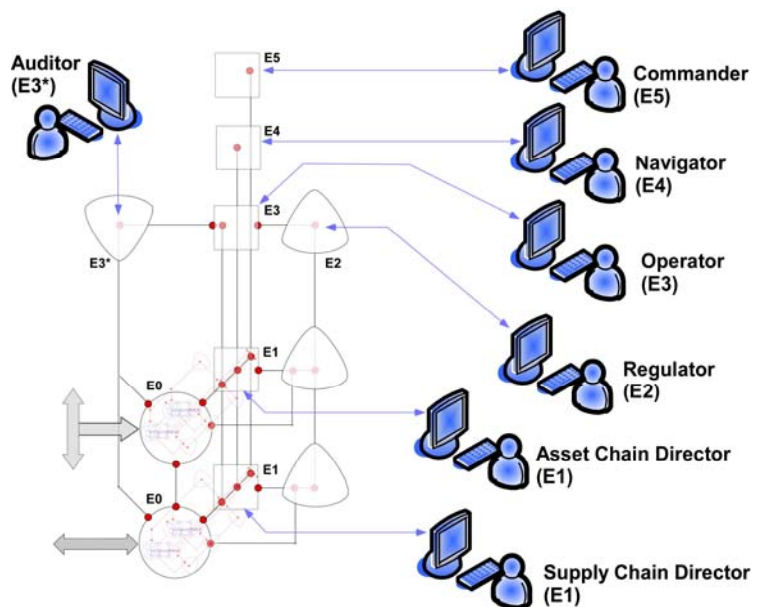


Figure 4 – Enterprise Command

Supporting these three principals are two or more subordinate *directors* (denoted *echelon one*, E1) of functional enterprise capabilities (embedded VPUs, at least one for the asset chain, and one for the supply chain), *regulators* (denoted *echelon two*, E2) responsible for the synchronization of subordinate VPUs in their execution of coordinated tasks that must rendezvous in time or synchronize on shared serially-reusable resources, an *auditor* (denoted *echelon three star*, E3*) responsible to E3 for continuously measuring and reporting on the performance of the subordinate VPUs, and the two or more embedded value production *processes* themselves (denoted as *echelon zero*, E0) that are managed by their respective E1 actors.

¹⁰ <http://scholar.lib.vt.edu/theses/available/etd-08092004-230138>

Although subtle and difficult to diagram, an important and distinguishing feature of this model is its inherent ability to scale through recursion, or self replication, to increasingly lower levels of enterprise operations. Careful inspection of the E1-E0 structures will reveal that the entire ECF structure is present within the embedded VPUs. This cybernetic and fractal model of control is motivated by and has counterparts in human neuro-anatomy, and mimics the manner in which network operating systems manage computational nodes, and multi-processor compute nodes manage executing tasks, and executing tasks are managed by threads. For many philosophical and practical reasons we believe this structure is a viable model of level- and domain-neutral, and therefore scale-free, enterprise governance¹¹. The following paragraphs further develop this argument.

To effectively scale, the ECF, in its support of the interactive processes of real-time situation assessment, plan generation, and plan execution among COI members, requires a set of generalized yet well-defined *protocols* between and among the ECF actors within and between VPUs in a COI lattice. These application-level communication protocols are implied by the lines in Figure 4 that terminate on specific objects associated with each actor. The figure also shows the individual actor user or client-side interfaces in the form of workstation icons.

Table 2 enumerates and summarizes the roles of each enterprise management actor.

Table 2 – Principle Enterprise C2 Actors

Echelon	Service Name	Enterprise Roles & Responsibilities
E5	Command	Goals, Objectives & Policy Domain Management
E4	Planning	Mission Capability Management
E3	Operations	Program & Capability Management
E3*	Audit	Plan (Process) Performance Assessment
E2	Regulation	Process (Task) Synchronization
E1	Director	Process (Task) Management
E0	Process	Value Production Process (Task) Under Control

Note: "*" designates a non-controlling role at a given echelon

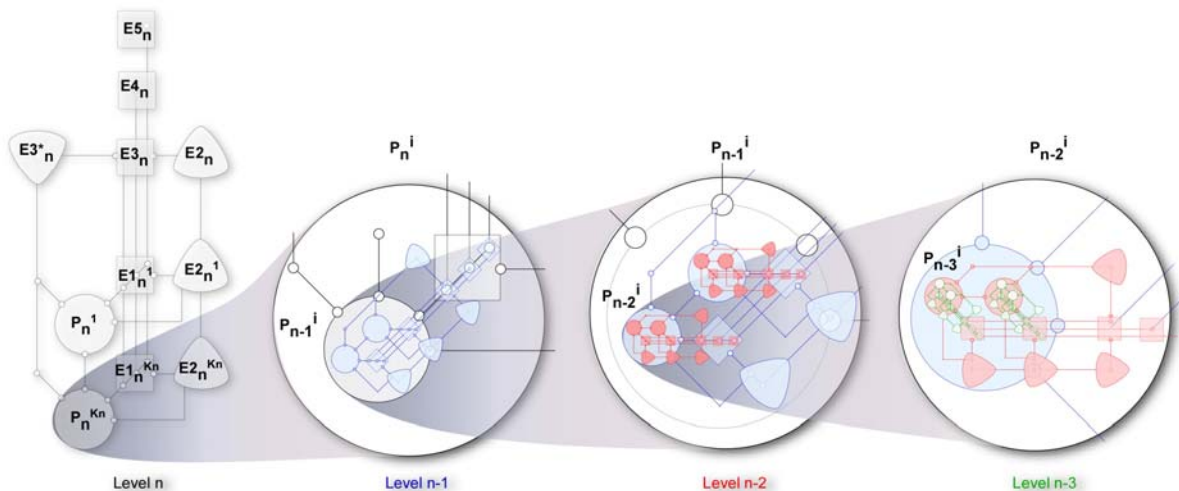


Figure 5 – ECF Nesting Levels – The Essence of Scalability

Figure 5 diagrams the essence of our scale-free argument, taking a closer look at the ECF structure's recursive properties. Command at level "n" is shown on the left to contain the

¹¹ [http://www.echelon4.com/content%20files/sci2003\(1\).pdf](http://www.echelon4.com/content%20files/sci2003(1).pdf)

K_n subordinate process $\{P_n^1 \dots P_n^{K_n}\}$ of $VPU[k,n]$. Looking specifically into $VPU^{P_n}[k,n]$ we see its level $n-1$ command structure, $VPU[k,n-1]$ for its i^{th} internal process P_{n-1}^i , and so on down to level $n-3$. At each level of recursion, the same basic ECF structure is used to describe governance. As a consequence of this symmetry, the triumvirate E5-E4-E3 at level n represents the function of an E1 *director* at level $n+1$, thus instantiating as control recurses the continuity and accountability of the command hierarchy depicted in Figure 1.

Figure 6 diagrams details of the embedded ECF C2 structure within our VPU (ref. Figures 2 and 3) and introduces its internal and external communications ports and associated operational databases. There is more detail here than we will discuss in this short paper, but the detail should provide the reader with a sense of our engineering of the ECF structure, and our claims of deployment scalability. A careful reading of Figure 6 will reveal the following important features:

1. Each VPU contains two or more embedded VPUs
 - a. At least one supply chain process (VPU^S)
 - b. At least one asset chain process (VPU^A)
2. Each VPU contains a regulator (E2) responsible for
 - a. Regulating (synchronizing with) peers through its "C" port
 - b. Regulating (synchronizing its) subordinates through its "F" and "G" ports
3. Each VPU communicates with its view of the "outside world" through its "U" and "V" ports
4. The navigator (E4) perceives the global context for the VPU through its "P" and "R" ports
5. Coupling the VPU commander (E5) to its superior is via the "A" and "B" ports (command axis)
6. Embedded VPUs synchronize with their peers (collaborators) through their "H" and "J" ports

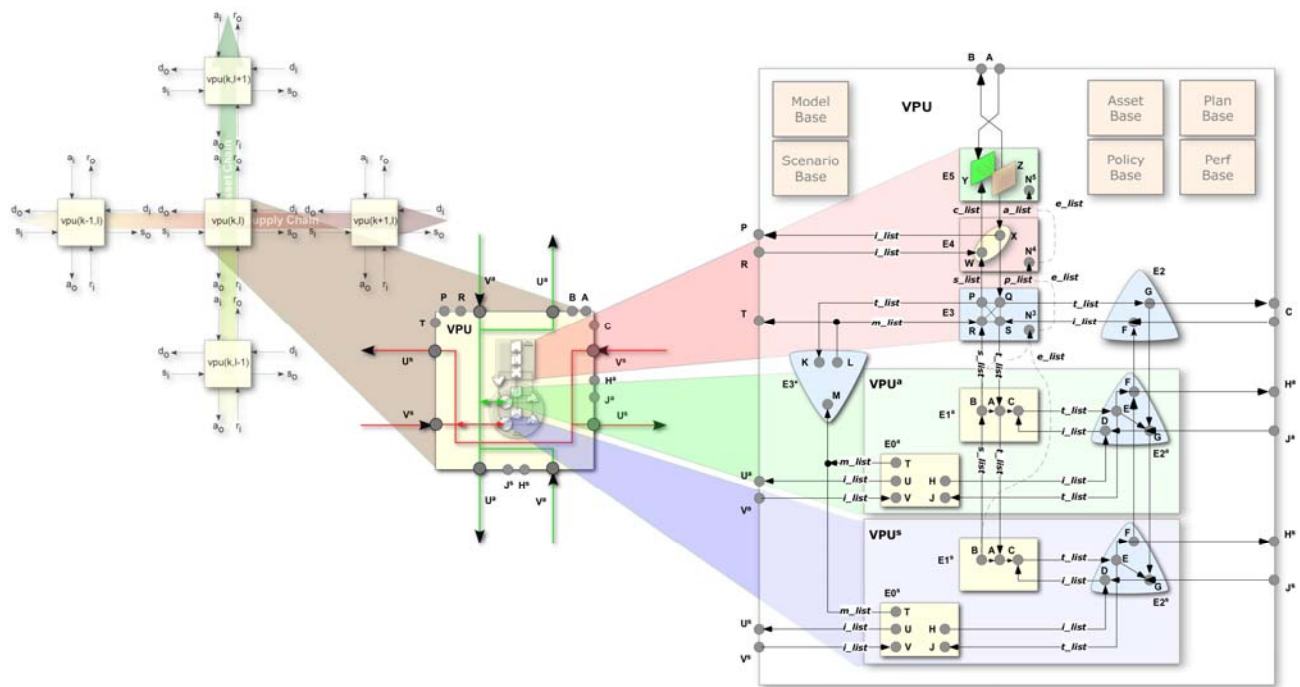


Figure 6 – VPU Command Framework Details

To demonstrate behavior of the ECF we offer the three operational views. The first describes receipt of and response to a tasking order from a higher-level command along the command axis. The second describes the introduction of the new tasking order into a running system. And the third describes the role of the E2 regulator in managing a value production process. All three scenarios presume the presence of the CPF application services as depicted at the base of Figure 3.

Operational View 1: Receipt & Processing of a New Tasking Order

With reference to Figures 6 and 7, our description begins with the arrival of a new tasking order resulting from a communication between the VPU[k,l] commander and his VPU[k,l+1] superior along their interconnected *command axis* (port A-B). The VPU[k,l] is likely busy processing previously scheduled activities – some self-generated, some resulting from collaboration with COI peers (clients, suppliers), and some from demands of superiors or subordinates. We make no assumptions at the outset how busy (under-loaded, over-loaded) the VPU might be. Our model does presume, however, that the VPU's management team does in fact know what its capacity is at all times. Our design provides for this knowledge in a fully scalable manner through the services of its *performance measurement framework* (PMF).

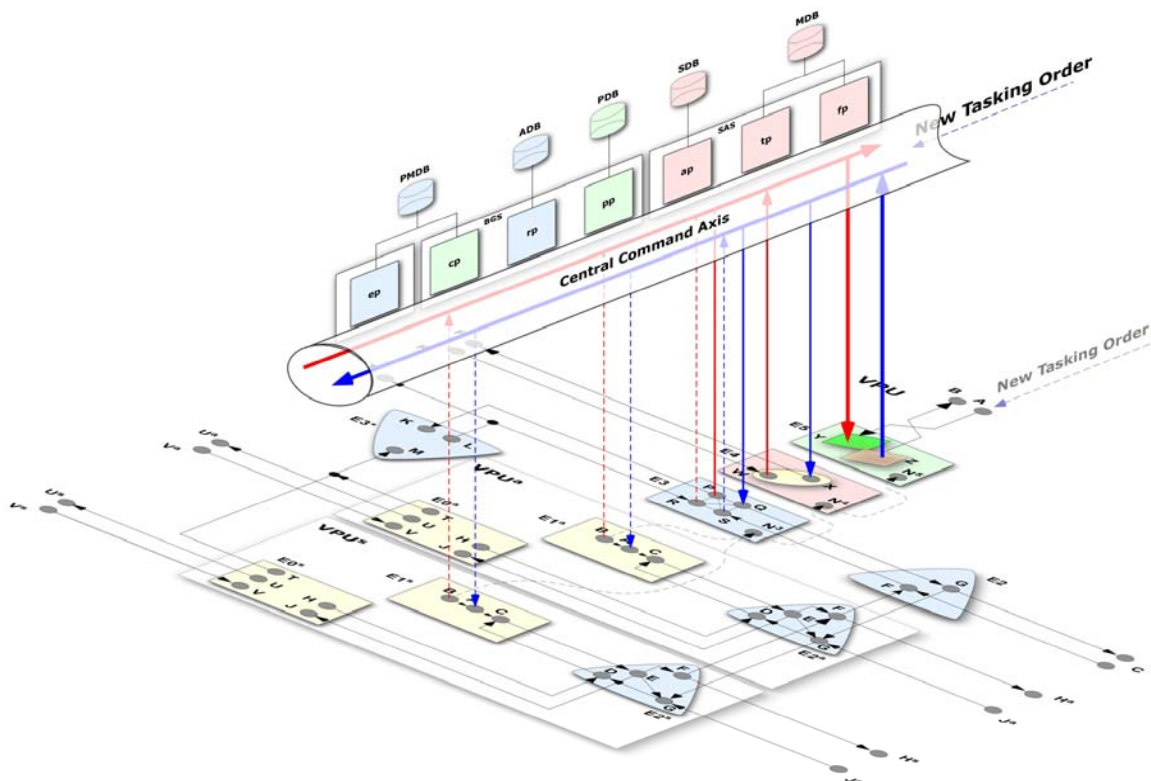


Figure 7 –Command Axis ECF-CPF Processing

Imagine an enterprise commander's operational *dashboard* containing real-time performance indicators as represented on the dial faces in Figure 8.

The figure shows three primary (top) and three derived (bottom) VPU level- and domain-neutral (scale-free) performance measures. The three primary measures are *potential*, *capability* and *actuality*. The three derived values are *latency*, *productivity*, and

performance. For each VPU, PMF services compute the derived indices as shown in the example in Figure 9.

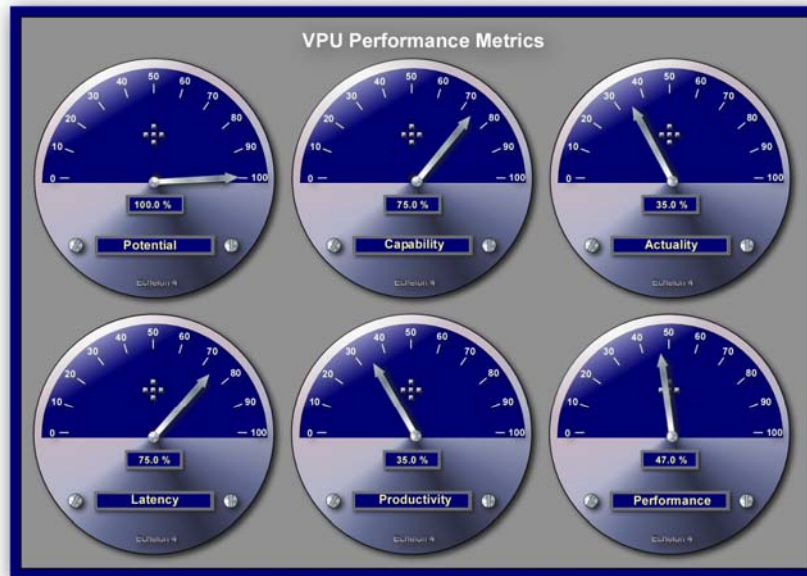


Figure 8 – VPU Dashboard Performance Meters

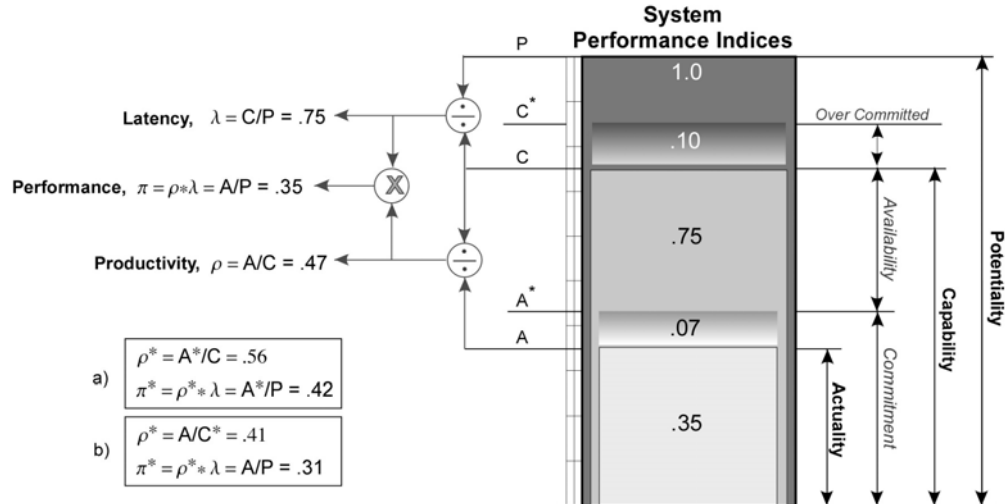


Figure 9 – VPU Performance Indices

Figure 8 and 9 show a VPU operating at an *actuality* of 35% of potential. The VPU has been “resourced” to a *capability* of 75% of potential, so its *latency* is 75%, its *productivity* with the resources it now holds is 47%, and its absolute *performance* is 35%. Clearly, there is capacity available to absorb more work.

Figure 9 also expresses two examples of the value of these metrics in E4’s planning activities:

1. Note the VPU's current "commitment" level (denoted A^*) of 42% (.35+.07), resulting in an "availability" level of 33% (.75-.42). In a) the impact of the commitment would be to raise *productivity* to 56% and raise *performance* to 42%.
2. There is a potential "over commitment" of 10% (denoted C^*) above *capability*. The effect of over commitment b) would be to lower *productivity* to 41% with a corresponding lowering of *performance* to 31%.

Returning to ECF and recalling Figure 3, a new tasking order in the form of a *tlist* is issued by the superior VPU[k,l+1], emerging from its internal "cp()" process step. All or some portion of that tasking order is dispatched by VPU[k,l+1]'s E3 and is received at the VPU[k,l]'s A port in Figures 3 and 7. The arrival enters VPU[k,l]'s "fp()" process and wends its way through its *situation assessment* and *plan generation* processing to emerge as a set of internal (i.e., derivative) tasking orders. The nature of this processing is outlined below.

1: Receipt of new tasking order (time: t_0)

2: Situation Assessment (start time: $t_0 + t_{\text{hold}}$)

E4 answers the question "What is our current situation and do we have the capacity to handle this new order?" "Do we have a plan (COA) that is appropriate, and if not, do we have a COI partner that does?" "Does the order require we collaborate with other VPUs?" and if so, "Do we have collaboration agreements in place, perhaps in the form of SLA, MOU, or MAA¹²?" If required, E4 is responsible for developing new SLA and proposing appropriate COA, updating its scenario database, and issuing COA to E5 as candidate plans of action for policy review.

Note: As shown in Figure 3 (bottom), exceptions to a given situation (e.g., tasking order) may be "thrown" at each stage of CPF processing.

3: Behavior Generation (start time: $t_0 + t_{\text{hold}} + t_{\text{sa}}$)

E5 answers the question "Given our operating policies, does this tasking order violate or otherwise conflict with or compromise our rules of engagement?" If they do "Do we have authority to either suspend or override such policies?" "What is the risk (price) for doing so in this situation?" "Can we acquire from our superiors and/or peers requisite authorization to proceed in the face of such violations?" E5 issues a plan of action (POA) based on these considerations.

E3 receives the POA and answers the question "Do we have the requisite resources to support this order as expressed in the POA?" And if so, "When are they available to be assigned?" "Does such an assignment meet the completion time requirements of the tasking order?" or "Do we have to preempt running tasks to reassign their resources?" "What is the cost of preemption?"¹³ Following resourcing, E3 issues a "funded" plan or record (POR) to E5 for authorization.

E5, assuming the policy and resourcing questions have been appropriately answered, authorizes the execution of the new tasking order and its derivative plans, possibly also authorizing the suspension and cancellation of existing plans affected by the new POR.

¹² *Service Level Agreements; Memoranda of Understanding; Mutual Aid Agreements*

¹³ The issue of the costs associated with introducing new activities into a running system is the basis for our use of *time-utility functions* and *utility-accrual scheduling* – the subject of our companion paper at this 10th ICCRTS conference entitled "Policy-based C2."

E3 then dispatches to its subordinate VPUs the derivative POR their associated tasking orders (ref. Figure 8).

Operational View 2: Execution of a New Tasking Order

4: Plan Execution (start time: $t_0 + t_{\text{hold}} + t_{\text{sa}} + t_{\text{bg}}$)

In this phase of responding to the new tasking order E3 is required to execute the order by "fitting" it into its running system. As implied by the figure, there are many possible scenarios. We shall present scenario 1 (nominal) here, leaving the others to the interested reader.

E3 delivers to its subordinate directors (E1s of VPU[k,l]) elements of the new tasking order. This is denoted as the (S-A) link in Figure 9. Note the symmetry and recursion with E5 receiving its original order in Figure 7. With E1's acceptance of the order (after its internal CPF processing!), E3 programs its E2 regulatory agents to monitor the E1-E0 execution loop (to be discussed shortly with reference to Figure 10) for this task so that

- a) E3 can continuously monitor the progress of the task, and
- b) To allow for synchronization with other VPUs executing related task steps or to coordinate the sharing of serially reusable resources

This programming takes place on the (Q-G) link. E2's monitoring on E1's behalf (i.e., *regulatory control*) is defined by the (H-D-C-E-J). E2's monitoring activity on E3's behalf (i.e., *supervisory control*) is defined by the (E-F-S-Q-G) links. As can be seen, E2's role is critical, operating on behalf of both E1 and E3, effectively coupling the supervision (E3-E4-E5) with operations (E1-E0).

Operational View 3: Coordination of Executing Tasking Orders

Figures 10 and 11 present in greater detail three key operational aspects of our scale-free solution to enterprise command processing, *supervisory*, *regulatory* and *synchronization* control over VPU workload execution. Again, the specific treatment of the fourth key aspect, critical time and schedule control, is the subject of a companion paper.

Supervision

A standard feature of all control systems supporting human intervention is the notion of "supervision" of the underlying automatic (autonomic) controls. These autonomic controls are regulated by subordinate controllers, as we shall discuss in the following section. Our first task is to show how E3 provides *supervisory control* of the subordinate E1-E0 VPUs.

With reference to Figure 10 (A), our newly minted tasking orders are delivered to the VPU's subordinate directors (E1) via their respective S-A command axis connections. Upon their acceptance, E3 delivers to its E2 regulators (E2, E2^a, and E2^b) their components of the plan that provide for supervision, regulation and synchronization. Their components include plan start-up, shared resource hold and abort logic, thread rendezvous points and shutdown logic. As plans execute, the supervisory loop among E1-E2-E3 operates as indicated. E1 reports through its C port to E2, which in turn reports to E3 through its F port. If E3 determines a need for supervisory intervention, it does so directly to E1 via the S-A port as before.

Regulation

Automatic (autonomic) controls are implemented through an appropriate form of closed loop "feedback control," traditionally referred to as *regulatory control*. Regulatory controls are usually analytically or heuristically designed to operate without human intervention as long as the process under control remains inside some well-defined envelop of behavior for which the regulator was designed. Outside that regime, typically during start-up, shutdown, faults and overload conditions, supervisory controls are engaged.

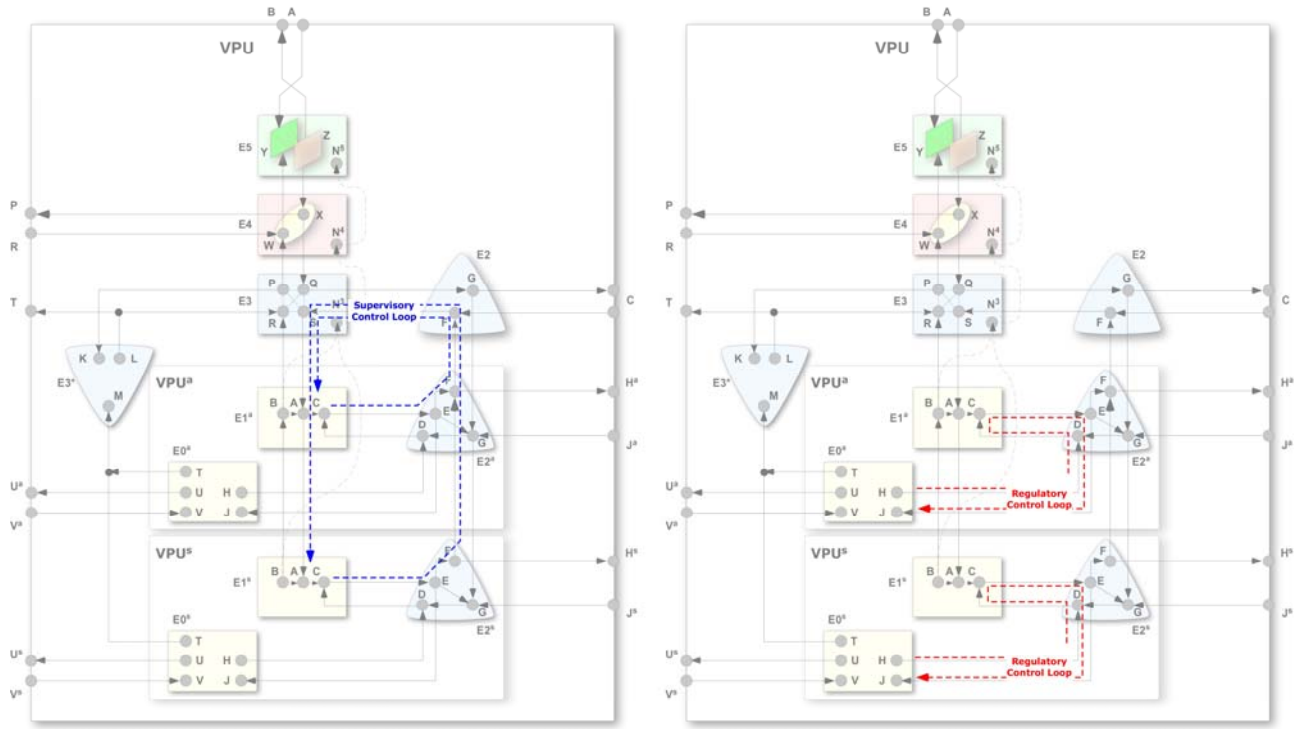
With reference to Figure 10 (B), the regulatory controls reside in E2 and function in the E0-E2-E1 loop. In this capacity, the regulator is functioning on behalf of E1, not E3, and used to maintain the dynamic stability (i.e., *homeostasis*) of the embedded VPU. Notice that through this regulatory loop the E2s are responsible for continuously reporting running estimates of their respective VPU's *actuality* measure, as reported in Figure 8. Likewise, the E1s maintain measures of their respective VPU's *capability*; and E3 in turn maintains the aggregate measures of VPU *potential*.

Synchronization

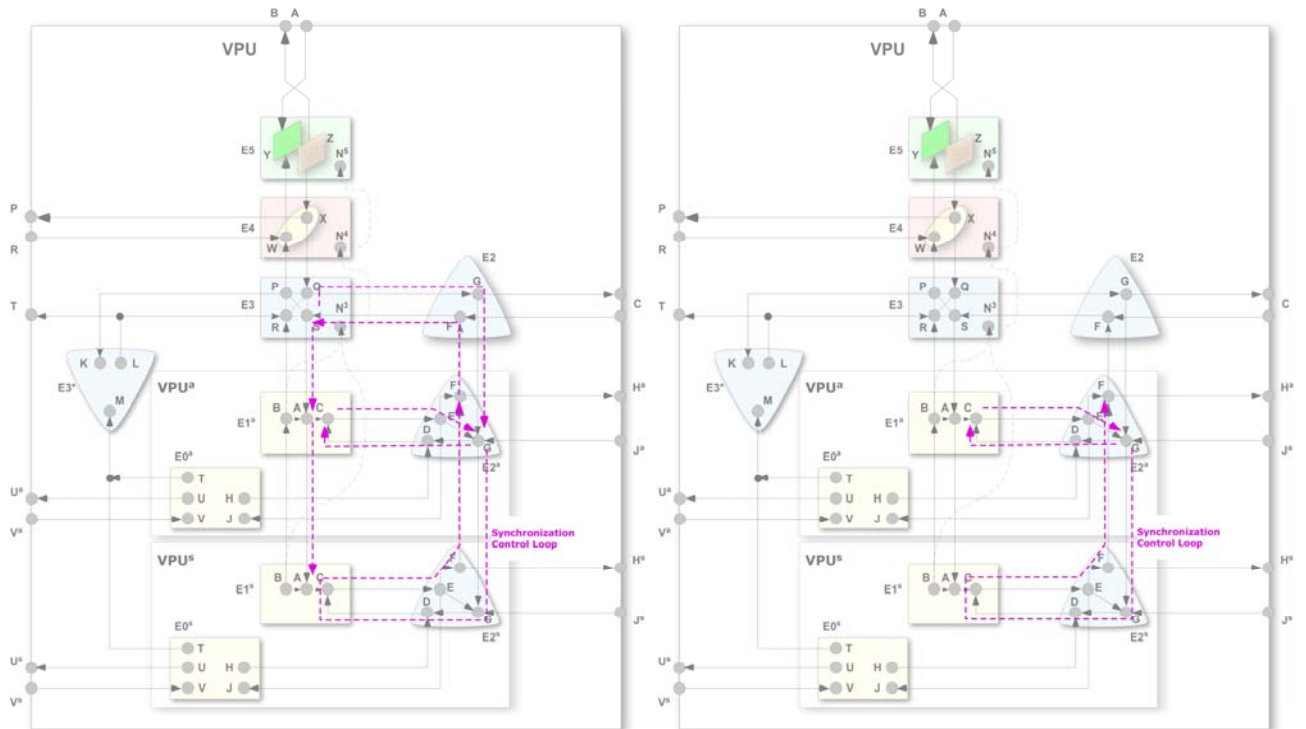
Governance of VPU behavior is most challenging when subordinate processes must be synchronized in time and with respect to the sharing of resources. This issue is well known in computer science, and in particular, in the management shared and distributed information resources. Our ECF treatment of this classic problem is consistent with resource reservation protocols, via mutex and semaphore lock mechanisms, familiar to the network and operating system design communities.

With reference to Figure 11, synchronization involves supervised (A) and unsupervised regulatory (B) controls to manage resources and rendezvous in JEC2 systems. Supervision involves the initial scheduling of task execution and its demands on establishing initial resource reservations. Regulation involves the automatic resource abort and rescheduling that takes place automatically during execution through our use of Time-Utility Functions (TUF) and Utility-Accrual (UA) scheduling.

Note that in Figures 10 and 11 we have not discussed the supervision, regulation and synchronization outside of the VPU among its COI allies. These mechanisms are handled in a consistent manner through the identified VPU boundary ports.



(A) (B)
 Figure 10 – VPU Execution Control Loops



(A) (B)
 Figure 11 – VPU Synchronization Control Loops

Summary

Our exposition of the mechanics of scale-free C2 has been necessarily brief. However, we believe the treatment contains sufficient information for the interested reader to explore many related operational issues. This work forms the basis for our current implementation of JEC2 system software, and our specific emphasis on applications related to command structures for Homeland Defense and Emergency Services. We are also working to define implementation guidance for specific Air Force and the Naval C2 requirements. This work continues to be the source of considerable input to and guidance from OASD-level discussions related to "unified command", the Unified Command Structure, and policy-based controls.

Acknowledgments

This work is partially funded by the Air Force Research Laboratory (AFRL) at Rome, NY and Navy Space and Warfare Systems Center (SPAWAR) at San Diego, CA (via their support of the National Institute for System Testing and Performance, NISTP, University of South Florida, Tampa) through their combined funding of AFRL Grant FA8750-04-C-0084.