

Cyberspatial Mechanics

Jay S. Bayne, *Senior Member, IEEE*

Abstract—In support of a generalization of systems theory, this paper introduces a new approach in modeling complex distributed systems. It offers an analytic framework for describing the behavior of interactive cyberphysical systems (CPSs), which are networked stationary or mobile information systems responsible for the real-time governance of physical processes whose behaviors unfold in cyberspace. The framework is predicated on a cyberspace-time reference model comprising three spatial dimensions plus time. The spatial domains include geospatial, infospatial, and sociospatial references, the latter describing relationships among sovereign enterprises (rational agents) that choose voluntarily to organize and interoperate for individual and mutual benefit through geospatial (physical) and infospatial (logical) transactions. Of particular relevance to CPSs are notions of timeliness and value, particularly as they relate to the real-time governance of physical processes and engagements with other cooperating CPS. Our overarching interest, as with celestial mechanics, is in the formation and evolution of clusters of cyberspatial objects and the federated systems they form.

Index Terms—Command and control, cybernetics, cyberspace, cyberspace-time, distributed computing, enterprise governance, real-time.

I. INTRODUCTION

CYBERPHYSICAL systems (CPSs) are internetworked information systems responsible for governance of physical processes. Examples include aircraft autopilots, air-traffic control, industrial-process control, civil-emergency management, and military command and control systems. The identity and behavior of such systems are defined at the intersection of information, geophysical, and social domains—in a word, cyberspace. Cybernetics [4], [8], [33] is a systems science of long standing that describes the autonomic behavior (via feedback-control models) of interdependent natural and synthetic systems. Furthermore, we extend the scientific basis of cybernetics with addition of a cyberspatial reference model. We regard this approach as foundational in the emerging services science [31] arena.

Our interest in CPSs is motivated by our work on large-scale distributed real-time industrial-process control, terrorist- and natural-disaster incident management (infrastructure protection), and military command and control systems. In particular,

Manuscript received September 18, 2007; revised January 4, 2008. This work was supported in part by the Air Force Research Laboratory (AFRL), Rome, NY, under Contract FA8750-04-C-0084. Elements of the cyberphysical systems model presented here derive from work published in the author's *Creating Rational Organizations—Theory of Enterprise Command and Control*, Café Press, 2006, available at www.cafepress.com/mcsi. This paper was recommended by Associate Editor H.-X. Li.

The author is with Meta Command Systems, Inc., Mequon, WI 53092 USA (e-mail: jbayne@metacomsys.com).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TSMCB.2008.916309

we wish to improve the efficacy (e.g., quality and velocity) of enterprise governance services, services that are dependent on geospatial, infospatial, and sociospatial considerations. Our perspective is new and innovative to the degree that we formalize the definition of cyberspace and its principal actors, integrating concepts from the fields of physical sciences, management, industrial automation, computation, control, communications, and cybernetics sciences. This multidisciplinary approach follows previous efforts in developing integrated dynamic system-modeling frameworks.

Albus [2] introduced a theory of intelligence, integrating information and classical control theory. Conant [10] presented a unified view of information theory related to complex systems. Whitman and Huff [32] applied natural-systems concepts to the modeling of man-made enterprise systems. Forrester [12] combined engineering and business disciplines in establishing systems dynamics at the MIT Sloan School of Management. Moreover, Beer [9] developed management cybernetics at Manchester University as a formal scientific discipline, applying operations research and cybernetics to the automation and control of large-scale socioeconomic systems. To these and related efforts, we add a unified cyberspatial framework and an operational focus on real-time CPSs.

This first of several planned papers is organized into four main sections. Section I introduces the basic concept of operations in cyberspace. Section II presents fundamental issues related to time, clocks, and timeliness properties of real-time systems. Section III introduces core elements of cyberspatial mechanics, including messages, end-to-end transactions, performance metrics, and service dynamics. Section IV introduces metrics for establishing the time-based value of cyberspatial services. Section V concludes with a summary and way forward, introducing several key topics for further research and associated companion papers.

A. Cyberspace

Cyberspace is the operating environment in which actors, both human and synthetic, form socioeconomic networks (enterprises or value webs) for the purpose of survival and growth (viability) through exchange of information, goods, and services. The environment is simultaneously physical (tangible and real) and present in geospace (G), informational (logical and virtual) and present in infospace (I), and social (organizational and political) and present in sociospace (S).

Within cyberspace, a CPS provides services through one or more rational agents [34] or cyberspatial objects (CSOs). Messages are the means for one CSO to exert forces on another. Messages flow through infospace to affect decisions at rational sociospatial endpoints governing the states and behaviors of

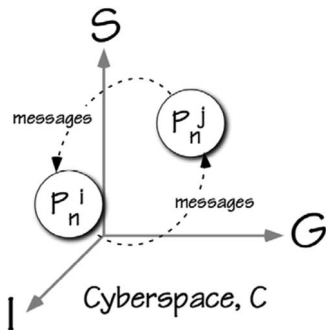


Fig. 1. Interacting CSOs.

geophysical processes. In Fig. 1, two CSOs operating on behalf of CPS_n , labeled as P_n^i and P_n^j , interact in a region of cyberspace. They affect each other's behavior through sending and responding to messages. Each CSO occupies a specific location $\{G, I, S\}$ at a point in time. Each has a unique identity and sustains itself by offering services characterized by quantifiable value propositions.

Relations among cooperating (and competing) CSOs are established (bound) either statically (*a priori*) or dynamically (on demand) as required. Dynamic binding is accomplished through a trading protocol [29], through which consumers (clients) ask for services and producers (suppliers) respond with bids. The consumer subsequently accepts a bid that satisfies the value proposition underwriting its service request. Following acceptance, the producer completes the request by delivering to the client a result satisfying the ask. The value of the result to the client is defined in the price element of the accept order. The (marginal) value to the producer is the difference between that price and its cost of producing the result. Negotiation (iterative ask-bid cycles) may take place prior to an accept.

Describing the behavior of individual CSOs (and their CPS containers) requires a set of performance measures. Comparing behaviors of interacting CSOs requires that these metrics be generalized and scalable, applicable to a potentially wide range of services and their underlying value propositions. Relative performance (e.g., throughput yield) involves both structural (organizational) and functional (process) considerations.

Fig. 2 shows the internal governance structure of CPS_n and its K_n subordinate CSO services. This cybernetic model, which was originally introduced by Beer [8] as the viable systems model (VSM), mechanized his theory of management cybernetics. The VSM was subsequently refined by this author and applied to military, governmental, and manufacturing enterprise command and control (EC2) requirements [5], [6]. EC2 theory [7] considers a CPS as a sovereign enterprise, governed by one or more rational actors (its management team or flight crew) organized in a collaborative command structure as shown. Governance structures may vary according to regional, economic, political, and social norms, but if viable (i.e., interactive, sustainable, and accountable), they share key operational characteristics.

As diagrammed, a CPS is composed of potentially many (embedded, encapsulated, and subordinate) CSOs, each offering a specific service or product. In the face of dynamic and probabilistic demand, achieving a degree of optimal (e.g., cost

effective) performance requires that each CPS be governed by some form of command structure accountable for its behavior. It is customary, logical, and intuitive to define governance structures in terms of roles and responsibilities of three primary actors [20], here, labeled as E5, E4, and E3. E5 (executive) represents the highest level of authority (accountability); E4 (navigator) provides strategy, analysis, and planning; and E3 (operator) attends to tactical execution activities.¹

Agile and adaptive CPSs are necessarily both proactive and reactive, maintaining dynamic stability (balance and homeostasis) through supervisory controls. As in natural systems, homeostatic control [4], [7], [8], [14] is achieved through two juxtaposed and counterbalancing feedback loops, shown in Fig. 2 as the sympathetic (E3–E2–E1–E3) and parasympathetic (E3–E3*–E0–E1–E3) circuits. Furthermore, individual CSOs are governed through tactical regulatory feedback control loops (E0–E2–E1–E0). In a recursive fashion, each E1 actor (CSO director) represents the command function accountable for the next lower level of value production. Consequently, E1 at level n in the management command hierarchy represents E5–E4–E3 at level $n-1$.

Each $E0^i$ actor (CSO production process) represents a specific unit of value production ($P_n^i, i = 1, \dots, K_n$). Service-oriented² CSOs are accessible through specific service access points (SAPs). CSOs within a CPS may be stationary or mobile in each of the three cyberspatial dimensions, independently or in unison. If mobile, their velocities and accelerations may also vary in each dimension.

The schema shown in Fig. 3 defines cyberspace as a 9-D hypercube. By including time, our cyberspace-time model provides CSOs with 10 DOF. This definition of cyberspace integrates three historically and semantically distinct coordinate systems. To allow them to form a proper hyperspace $\{G, I, S\}$ supported by a rationalized distance metric, we require a common unit of distance measure—a cyberspatial measurement unit.

Our solution is based on the following two assumptions: 1) in each dimension, a coordinate (index) may be interpreted as an abstract address object and 2) the three primary and three secondary dimensions are orthogonal. At both indexing levels (ref. Fig. 4), address objects are 3-tuples: $\{G, I, S\} = \{\{x, y, z\}, \{g, s, a\}, \{f, p, c\}\}$. This approach rationalizes addresses by converting each 3-tuple to a standard integer format augmented with domain-specific metadata, the details of which are the subject of a future paper. Each cyberspatial address component (e.g., the infospatial service-point index, “ a ” in $\{g, s, a\}$) is defined as a 64-bit integer.³ Consequently, each 3-tuple defines three components of a 192-bit address object on which uniform address arithmetic (supporting intra- and interspace distance metrics) may be computed. Fig. 4 shows the schema as a traditional Cartesian coordinate system.

¹ Depending on the size and complexity of an enterprise, one person (e.g., a pilot) may serve the combined E5, E4, and E3 function or several people may serve in a team for each individual function.

² Service-oriented architecture defines network-centric software-system models for providing specific services [11], [16].

³ A programming language-specific longword, *Cint64_t*, or Java long in the range $[0, \dots, 2^{64} - 1]$ or, if signed, $[-2^{63}, \dots, 2^{63} - 1]$.

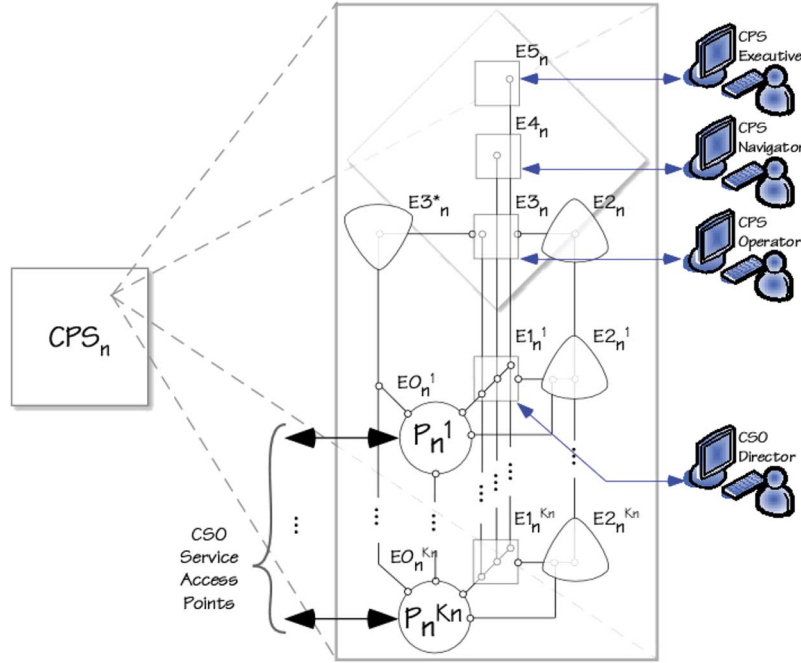


Fig. 2. CPS governance structure.

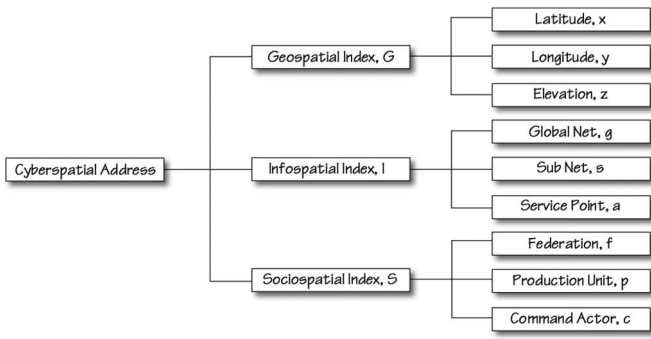


Fig. 3. Cyberspatial address schema.

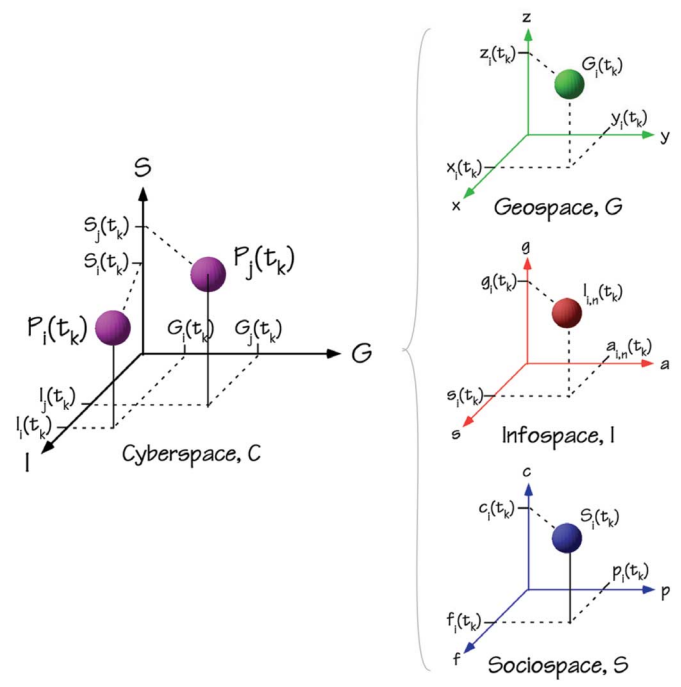


Fig. 4. Cyberspatial coordinates.

In the geospatial dimension (e.g., latitude, “x” in $\{x, y, z\}$), we have employed indexing relative to traditional geocentric (spherical) coordinates. Alternatively, we could have utilized indexing related to a digital Earth reference model (DERM) [17], where $\{x, y, z\}$ refers to the location of a hexagonal region defined by a tessellation on the Earth’s surface. One of our next goals of this work includes development of a standard DERM-compliant CSO directory structure.

Cyberspace is assumed Euclidean and compact, which is defined in relation to its orthogonal axes. In our construction, orthogonality has two complementary and equally important meanings. The first derives from traditional mathematical concepts, where orthogonal Euclidean 3-space vectors produce zero dot products. The second is a software design principle resulting from the desire to isolate system behaviors in order to realize compact functional designs.

From the software design perspective, orthogonality is an important property in making complex designs concise (compact). In a purely orthogonal design, operations have limited consequences; each action, whether a service or a macro invocation

or a language or protocol operation, changes just one thing per invocation without affecting others, thus producing minimal or no side effects. There is one and only one way to change a property of whatever object you are controlling.

The schema defines the relative cyberspatial position of objects as shown in Fig. 5. Let the cyberspatial position of CSO_i at time t_k be given by $P_i(t_k) = \{G_i, I_i, S_i\}(t_k)$. The cyberspatial distance between objects CSO_i and CSO_j is then $d_{i,j}^C(t_k) = \sqrt{d_{i,j}^G(t_k)^2 + d_{i,j}^I(t_k)^2 + d_{i,j}^S(t_k)^2}$, where $d_{i,j}^X(t_k)$,

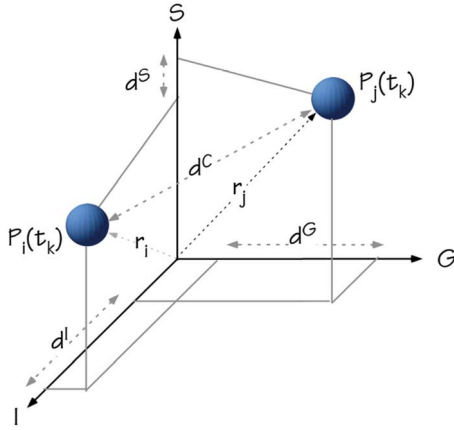


Fig. 5. Cyberspatial position.

for $\chi \in \{G, I, S\}$, are distance metrics for each of the three subordinate dimensions.⁴

B. Geospatial Dimension

Geospace provides a framework for specifying the physical location of an object. Let $P_i^G(t_k) = \{x_i, y_i, z_i\}(t_k)$ be the Earth-centered geospatial location of CSO_i at time t_k . This representation presumes that time is measured uniformly along each axis. We postpone discussion of relativistic effects (e.g., Lorentz transformations) in situations when velocities of geospatial objects (G_i) approach vacuum light speed. While this situation is unlikely in geospace and sociospace, it is likely in infospace due to the performance of optical communication systems.

As is standard practice, the geospatial distance between CSO_i and CSO_j at time t_k is

$$d_{i,j}^G(t_k) = \sqrt{d_x(t_k)^2 + d_y(t_k)^2 + d_z(t_k)^2}$$

$$d_x(t_k) = x_i(t_k) - x_j(t_k)$$

$$d_y(t_k) = y_i(t_k) - y_j(t_k)$$

$$d_z(t_k) = z_i(t_k) - z_j(t_k).$$

We note for both practical and historical reasons that geospatial objects also have georeferenced information SAPs, which are referred to as postal addresses and landline voice and video circuit (a.k.a., last mile) addresses. Through these physical addresses, real mail, video, and voice are sent and received. Increasingly nontangible goods traffic is carried via infospatial circuits in the form of digital voice, video, and data (e.g., web content and e-mail). We include postal addresses and analog communications circuits as geospatial addresses since we can map $\{x, y, z\}$ coordinate references to these more traditional forms of physical address.

C. Infospatial Dimension

Expanding on the work sponsored, in part, by the Air Force Research Laboratory [15], infospace provides a framework in which to specify the locations of an object's SAPs, which are

⁴ $\{\dots\}$ represents a list or vector of items and $\{\dots\}(t)$ denotes a list whose elements are functions of time.

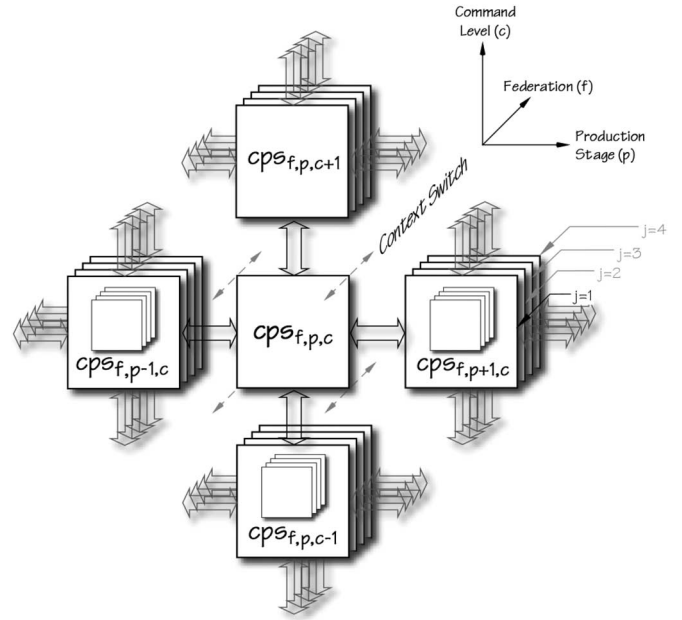


Fig. 6. CPS sociospatial structure.

communications ports on the object through which it interacts with other objects about their respective states, goods, and services (value propositions). Following the reference model defined in the Internet Protocol version-six (IPv6) standard [28], we define a generalized infospatial service port address for object P_i as the 3-tuple $r_i^I = \{g_i, s_i, a_i\}$, where g designates the global network address (nominally, 48 bit), s designates the subnetwork address (16 bit), and a designates the subnetwork's particular SAP (64 bit).

Let $P_{i,n}^I(t_k) = \{g_i, s_i, a_{i,n}\}(t_k)$ be the infospatial location of CSO_i 's n th SAP at time t_k . We define the infospatial distance between access point n on CSO_i and access point m on CSO_j at time t_k as

$$d_{i,j}^{n,m}(t_k) = \sqrt{d_g^{n,m}(t_k)^2 + d_s^{n,m}(t_k)^2 + d_a^{n,m}(t_k)^2}$$

$$d_g^{n,m}(t_k) = g_i^n(t_k) - g_j^m(t_k)$$

$$d_s^{n,m}(t_k) = s_i^n(t_k) - s_j^m(t_k)$$

$$d_a^{n,m}(t_k) = a_i^n(t_k) - a_j^m(t_k).$$

D. Sociospatial Dimension

Sociospace is a framework for specifying the location of an object with respect to its operational role within one or more federated systems. As with the intra-CPS governance structure of Fig. 2, the inter-CPS operational structure shown in Fig. 6 is based on an enterprise model developed in [7]. Sociospatial value webs are 3-D, with index f designating a specific federation, p designating the enterprise's position along the federation's horizontal production (supply chain) axis, and c designating the enterprise's position along the federation's vertical accountability (command chain) axis. $CPS_{f,p,c}$ belongs to at least one "root" or "home" federation $\{f = 1\}$. At its creation and until proactively altered, the root context of a CPS is that of its parent (superior).

Vertically in a given federation, $CPS_{f,p,c}$ is a subordinate (child) to, and therefore dependent upon (accountable to), a

single superior (parent) CPS $_{f,p,c+1}$ and is superior to (a parent of), and therefore responsible (accountable) for, its subordinates (children) CPS $_{f,p,c-1}$. Horizontally, CPS $_{f,p,c}$ is a supplier to (producer for) its clients (consumers) CPS $_{f,p+1,c}$ and a consumer (client) of its suppliers (producers) CPS $_{f,p-1,c}$. The vertical axis defines a federation's chain of authority (command); the horizontal axis defines its logistics (supply) chain. As shown in Fig. 6, each enterprise typically holds membership concurrently in several ($f > 1$) federations, requiring its governance system to maintain situation awareness and sufficient agility to context switch among its roles in multiple federations.

Operationally, this technical definition of a sociospatial value web requires the enterprise governance system to treat enterprises and their activities in much the same way multiprogrammed computer operating systems (specifically, their kernels) treat applications and their running processes (threads)—essentially, assigning to each a virtual machine (i.e., governance structure) that is allocated, by some scheduling policy sufficient resources (e.g., CPU, memory, and time) to allow its tasks to run to time-bounded completion. Our enterprise governance system (Fig. 2) provides the function of an OS kernel, implementing an enterprise operating system (EOS) that maintains separation between and integrity of multiple contexts supporting the supply and command axis tasks (CSOs) for each federation in which it is a participant.

Let $P_i^{S^{n,m}}(t_k) = \{f_i, p_{i,n}, c_{i,m}\}(t_k)$ be the sociospatial position of enterprise object CSO $_i$ at time t_k . Within any single federation f , a given CSO typically, interacts with multiple concurrent service providers (producers), clients (consumers), and subordinates enterprises. The n and m indexes identify the location and role of a particular neighbor (n) on the producer–consumer axis and (m) on the superior–subordinate axes. For practical and philosophical reasons, as noted in the figure, our model assumes a single superior within each federation. To simplify notation, we omit the n and m indexes in the following discussion.

We define the sociospatial distance between CSO $_i$ and CSO $_j$ at time t_k as

$$\begin{aligned} d_{i,j}^S(t_k) &= \sqrt{d_f(t_k)^2 + d_p(t_k)^2 + d_c(t_k)^2} \\ d_f(t_k) &= f_i(t_k) - f_j(t_k) \\ d_p(t_k) &= p_i(t_k) - p_j(t_k) \\ d_c(t_k) &= c_i(t_k) - c_j(t_k). \end{aligned}$$

II. TIME, CLOCKS, AND TIMELINESS

Our interest in cyberspatial mechanics requires that we clarify the notions of time and the temporal properties of communications among collaborating CSO. Temporal properties include the completion-time semantics of transactions, including the meaning of time, the synchronization of distributed clocks, and the concept of timeliness.

A. Time and Clocks

Understanding the dynamics of CSOs requires careful consideration of the general meaning of time, the associated

concept of timeliness, and, in the special case of CPSs, the classification real-time⁵ [18], [19]. In cyberspace, time is discrete and represented by incremental offsets from a reference clock (e.g., universal coordinated time), with $dt = t_k - t_{k-1}$ the interval between successive timestamps t_{k-1} and t_k . Our use of timestamps is consistent with the Network Time Protocol (NTP and SNTP) [25], [27], [30] used almost universally throughout infospace and, by inference, geospace. Clock-synchronization errors resulting from offset and oscillator drift are important issues in cyberspace, particularly for objects that are mobile and rely on time-sensitive services offered by other stationary or mobile objects and those associated with fast physical processes. For extremely fast processes, whose velocities approach the vacuum speed of light, relativistic effects (e.g., time dilation) may also be involved.

In the descriptions to follow, we assume that events unfold sequentially in time and that, for a given object, event a at time t_a precedes event b at time t_b if $t_a < t_b$. This “happens before” relation [22] establishes a partial ordering on events, which is denoted $a \rightarrow b$. Furthermore, the following conditions are assumed to hold.

- 1) If a and b are events in a given CSO (process) and a comes before b , then $a \rightarrow b$.
- 2) If a is the sending of a message by one CSO and b is the receipt of the same message by another CSO, then $a \rightarrow b$.
- 3) If $a \rightarrow b$ and $b \rightarrow c$, then $a \rightarrow c$. Two distinct events a and b are said to be concurrent if $a \not\rightarrow b$ and $b \not\rightarrow a$.

Let $\hat{T}_i(e)$ be a logical clock that assigns a timestamp⁶ to the occurrence of event e in CSO $_i$. For a system of logical clocks, the following two conditions must hold to enforce causal ordering: 1) if a and b are events in CSO $_i$ and a happens before b , then the $\hat{T}_i(a) < \hat{T}_i(b)$ and 2) if a is the sending of a message by CSO $_i$ and b is the receipt of that message by CSO $_j$, then $\hat{T}_i(a) < \hat{T}_j(b)$. For CPSs, these two conditions require the following: 1) each CSO $_i$ must increment its local clock \hat{T}_i between any two successive events and 2) each message issued by CSO $_i$ must contain a timestamp t_k equal to the time at which the message was sent and where subsequent receipt of that message timestamped t_k requires CSO $_j$ to advance its local clock \hat{T}_j to be later than t_k .

To achieve and maintain synchronization of physical clocks in CSOs, they must run at approximately the same rate. For physical clock $T_i(t)$, $dT_i(t)/dt \approx 1, \forall t$. To be synchronized with other clocks means that $T_i(t) \approx T_j(t), \forall i, j, t$. This requires the following two physical-clock conditions to hold: 1)

$$\exists \text{ cons } \kappa \ll 1 \text{ s.t. } \forall i : |dT_i(t)/dt - 1| < \kappa,$$

where for 100-MHz resonators, $\kappa \approx 10^{-7}$ and 2) the offset error ($\forall i, j : |T_i(t) - T_j(t)| < \varepsilon$) must be sufficiently small to provide the required application precision, yet be within the synchronization capabilities of NTP ($\pm 10^{-9}$ s). Fig. 7 shows a space-time view of a causally ordered sequence of events in

⁵A system is real-time to the degree (statistically) that it meets its deadlines.

⁶At this juncture, a timestamp assigned by $\hat{T}_i(\dots)$ has no inherent relation to physical (real) standard time.

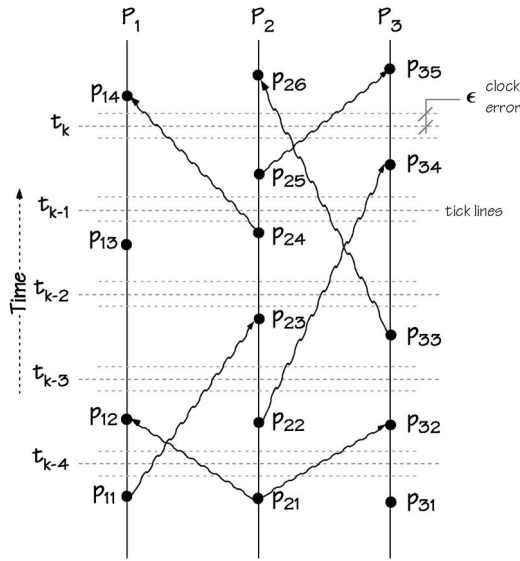


Fig. 7. Space-time diagram.

three processes, with their physical-clock “tick lines” aligned and the offset error ϵ shown.

B. Clock Synchronization

NTP is the protocol used for synchronizing a majority of server⁷ clocks distributed throughout the Internet. Its basic operation is client–server, where the time client sends an NTP message to one or more timeservers and processes the replies it receives. It recognizes a hierarchy of strata through which time-synchronization messages may flow from timeservers with high-resolution (cesium atomic) clocks outward through multiple levels of servers to progressively remote client end-systems. This cascading NTP client–server arrangement, comprising up to 20 strata, incurs transport delays. Additionally, both client and server internal clocks suffer drift due to electronic and mechanical (thermal) conditions.

NTP accounts for these sources of error in the following manner. With reference to Fig. 8 and [25], the time client issues a time-synchronization request to one or more NTP timeservers, affixing to its request the timestamp T_{i-3} . That message arrives at the server who affixes its arrival timestamp T_{i-2} , resulting in an inbound trip time of $a = (T_{i-2} - T_{i-3})$. The server interchanges addresses and ports in the request-message header, overwrites certain fields, recalculates the checksum, and returns the message immediately, affixing its transmit timestamp T_{i-1} . This return message arrives at the client at T_i , resulting in a return delay of $b = (T_i - T_{i-1})$.

Information included in the NTP message is then used by the time client to calculate the server’s time value with respect to its own local time and to accordingly adjust its local clock. Additionally, the message includes information to calculate the expected timekeeping accuracy and reliability of the local clock, as well as to select the best among possibly many NTP

⁷The majority of clocks in the Internet exist in client PCs whose internal clocks are typically synchronized at most daily with direct (i.e., non-NTP) requests to NTP-corrected timeservers such as NIST’s time.nist.gov and Microsoft’s time.windows.com.

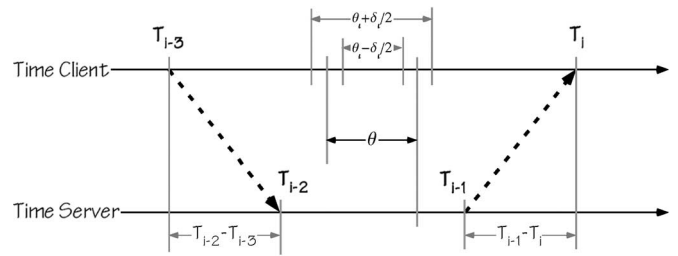


Fig. 8. NTP offset errors.

timeservers. Thus, with reference to Fig. 8, NTP allows the client to calculate, relative to the server

Clock offset:

$$\theta_i = (a + b)/2 = ((T_{i-2} - T_{i-3}) + (T_i - T_{i-1})) / 2$$

Roundtrip delay:

$$\delta_i = (a - b) = (T_{i-2} - T_{i-3}) - (T_i - T_{i-1}).$$

If θ (Fig. 8) is the true offset between the peer clocks, then it can be shown that offset errors $\theta_i = \theta(T_i)$ are bounded by $b \leq \theta \leq a$, or equivalently

$$(\theta_i - \delta_i/2) \leq \theta \leq (\theta_i + \delta_i/2).$$

The various arithmetic operations on 64-bit timestamps are constructed to avoid overflow while preserving precision. The only operation permitted on timestamps is subtraction, which produces signed 64-bit timestamp differences from 68 years in the past to 68 years in the future. The offset (θ_i) and delay (δ_i) calculations involve addition and subtraction of timestamp differences. To avoid overflow, these differences cannot exceed ± 34 years, a fundamental limit of 64-bit integer arithmetic.

Following a synchronization-message exchange, the current time, as perceived by a time client relative to that of a time-server, is

$$T(t) = T(t_0) + R(t - t_0) + 1/2D(t - t_0)^2$$

where t is the current time, and $T(t)$ is the time offset from the last measurement update at t_0 . $R(t - t_0)$ is the frequency offset and $D(t - t_0)$ is the clock drift due to resonator aging.

The NTP synchronization protocol estimates $T(t_0)$ and $R(t - t_0)$ at regular intervals τ and adjusts the local clock by averaging the last eight samples (typically) to minimize $T(t)$ in the future. In common cases, $T(t_0)$ and $R(t - t_0)$ can have systematic offsets of several hundred parts per million (ppm) with random variations of several ppm due to ambient-temperature changes. If not corrected, the resulting errors can accumulate to seconds per day. As a practical matter, for nominal accuracies of tens of milliseconds, this requires time clients to exchange messages with timeservers at intervals on the order of tens of minutes.

C. Timeliness

Synchronized physical clocks are necessary but not sufficient for guaranteeing that cyberspatial systems can provide services

that are, in a measurable sense, timely with respect to service-level commitments expected of their federation (sociospacial) partners. Quality clocks do assist in providing accurate timestamps and for supporting high-resolution scheduling of local resources, but in collaborative arrangements among distributed agents, group (i.e., end-to-end or transnode) timeliness requires additional facilities.

There is currently no generally accepted (let alone, standardized) mechanism to achieve end-to-end timeliness in distributed systems, particularly remote method invocations (RMI), allowing two or more distributed CPS processes (CSO) to rendezvous in cyberspace-time. There is, however, a significant body of contemporary work addressing the subject [18], including a thread-scheduling paradigm realized in real-time CORBA V1.2 [26] and compliant ACE/TAO [1] open-source middleware, the nondistributed Real-Time Specification for Java (RTSJ) [19], and its distributable-thread (DT) successor introduced in the Distributed RTSJ (DRTSJ) [3].

The paradigm essentially states that when CSO_i commits to certain application-level timeliness properties and, subsequently in the course of its execution, requires the services of a remote CSO_j , it must transmit its expected completion-time requirements (time constraints) along with its service invocation request as part of a distributable thread. In accepting this thread invocation request, CSO_j agrees to make best effort to adjust its local scheduling policies (e.g., priorities and resource commitments) to meet CSO_i 's completion-time requirements, or to reject the request. CSO_j 's mechanism for scheduling threads must therefore involve optimality conditions that balance the multitude of typically conflicting completion-time requests. To do so, [D]RTSJ utilizes “pluggable” application-level scheduling policies (e.g., a utility-accrual (UA) scheduling mechanism, as described in [23]).

The DT paradigm depends on as follows: 1) the transmission of end-to-end timeliness specifications along with service invocation requests and 2) a means of adhering to these specifications by recipients. These parameterized specifications effectively define an expected application-level quality of service (AQoS) explicitly or implicitly associated with the service-level agreement (SLA) defined in a CSO's published (discoverable) service profile.

The means adopted for describing completion-time requirements for remote service invocations involves CSO_i providing CSO_j with a service deadline in the form of a specification, such as a time utility function (TUF) [21]. A TUF is a parameterized expression describing the value of completing the service request as a function of time, or in the reverse direction, the time value of information returned from the service request. TUF specifications are therefore useful for describing liveness properties of data contained in a message, whether invocation orders or results. The message contents, thus, take on a “valid-while” predicate within a TUF-specified window. This is particularly useful in real-time applications where information quality may deteriorate as a function of time or distance or both.

Completion-time specifications may be described by an infinite number of possible time-value functions. For example, Fig. 9(a) shows a service request whose completion-time value

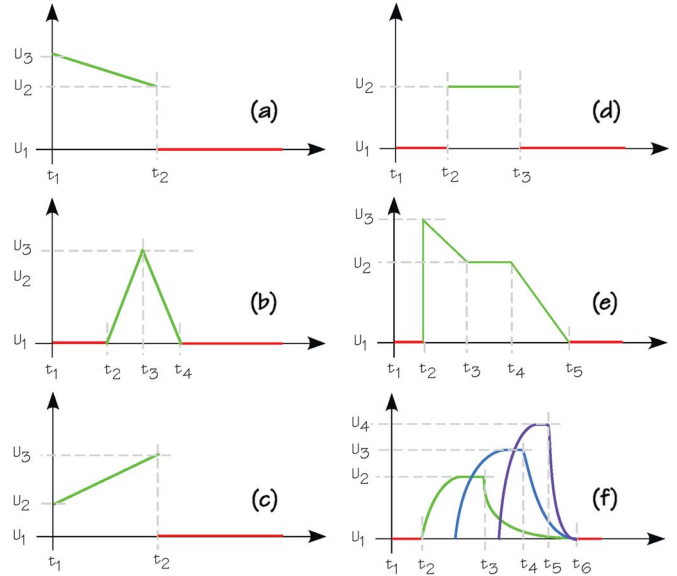


Fig. 9. Time utility functions (TUFs).

is maximum at t_1 (i.e., “as soon as possible”), deteriorates linearly until t_2 , and then goes to zero thereafter. Fig. 9(f) shows a service request, whereas time progresses the completion-time requirements are described by a sequence of increasingly narrower and higher value TUF specifications—indicative of phased and increasingly critical processes. Fig. 9(d) shows the simplest to specify and easiest to implement TUF specification—a step function or sequence of step functions.

Let $u(t) = \{\text{tuf}(t), \{t^{\text{start}}, t^{\text{end}}\}\}$ be a utility function specification carried along a DT (i.e., within an RMI message payload) specifying completion-time requirements for a given service. Thus, $u(t)$ defines a specific quantum of service

$$u(t) = \begin{cases} \text{tuf}(t), & \text{for } t^{\text{start}} \leq t \leq t^{\text{end}} \\ 0, & \text{otherwise.} \end{cases}$$

Here, $\text{tuf}(t)$ is a piecewise continuously differentiable function in the specified interval. A deadline (i.e., critical completion-time requirement) is defined as

$$u^{\text{max}}(\text{at } t^{\text{critical}}) = \max[\text{tuf}(t), \{t^{\text{start}}, t^{\text{end}}\}].$$

In summary, a service is deemed timely (i.e., real-time) to the degree it is able to respond to client-specified completion-time requirements, requirements carried along distributable threads in the form of TUFs. Given the earlier temporal considerations, we are now in a position to reason about the dynamics of objects whose behaviors unfold in cyberspace-time.

III. CYBERSPATIAL MECHANICS

As with celestial mechanics, we are interested in the formation and evolution of objects (stars) and the federations (galaxies) they form. Cyberspatial mechanics is concerned with positions, velocities, and accelerations of objects relative to each other and an observer's frame of reference. Referring to Fig. 10, CSO_i changes position between time t_{k-1} and t_k .

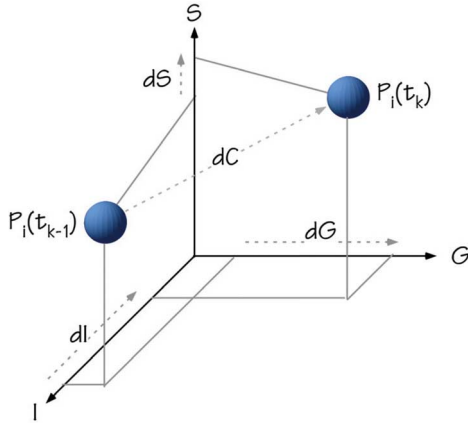


Fig. 10. Cyberspatial motion.

The differential distance dC involves translations in infospace (dI), geospace (dG), and sociospace (dS), where the distance is given by

$$dC = \sqrt{dG^2 + dI^2 + dS^2}.$$

If the object moves distance dC in time $dt = t_k - t_{k-1}$, its corresponding differential velocity is dC/dt .

The meaning ascribed to motion along cyberspatial axes is straightforward. The change in an object's geospatial position ($|dG| > 0$) implies that it is in physical motion and its geolocation has changed (e.g., a mobile phone or a satellite- or ship-based service). Change in infospatial position ($|dI| > 0$) implies that one or more of an object's access points has changed, either through acquiring new IPv6 addresses⁸ or eliminating, altering, or establishing the services available at a specific access point. Change in an object's sociospatial position ($|dS| > 0$) implies that an object's operational (socioeconomic) relationship to other objects has changed due to switching federation context or changes in the object's sociospatial role. For a given CSO, changes in its three cyberspatial coordinates may be uncorrelated, happening independently at arbitrary times, or changes may happen in a coordinated fashion within a given time period.

Velocities (v) and accelerations (a) of CSOs are described in the usual (i.e., Newtonian) manner

$$\begin{aligned} v_i^C(t) &= \dot{r}_i^C(t) = dC/dt \\ a_i^C(t) &= \dot{v}_i^C(t) = \ddot{r}_i^C(t) = d^2C/dt^2. \end{aligned}$$

Equivalently, component velocities and accelerations along each axis are given by

$$\begin{aligned} v_i^G(t) &= \dot{r}_i^G(t) = dG/dt; & a_i^G(t) &= \dot{v}_i^G(t) = d^2G/dt^2 \\ v_i^I(t) &= \dot{r}_i^I(t) = dI/dt; & a_i^I(t) &= \dot{v}_i^I(t) = d^2I/dt^2 \\ v_i^S(t) &= \dot{r}_i^S(t) = dS/dt; & a_i^S(t) &= \dot{v}_i^S(t) = d^2S/dt^2. \end{aligned}$$

From classical mechanics, Newton's second law of motion in geospace is $F^G = m^G a^G$, where F^G is the net force exerted on an object with mass m^G and acceleration a^G . This naturally

leads us to question the meaning and implications of infospatial ($F^I = m^I a^I$) and sociospatial ($F^S = m^S a^S$) analogs and their relation to a fuller development of cyberspatial physics. This line of inquiry is the subject of a planned subsequent paper. For the present discussion, we focus on messages the primary vehicle for exerting forces on cyberspatial systems.

A. Messages

In addition to the relative motion of CSOs, we are interested in their communication patterns and, in particular, the forces exerted by information in the form of messages flowing among them. The notion of infospatial force derives from the meaning (semantics) of a message, defined here relative to the capabilities of a recipient and the message's ability to act as selector of behavior (goal-directed activity) from among a CSO's states of conditional readiness [34]. For CPSs, in particular, messages flow through infospace to affect decisions at rational endpoints in sociospace that are accountable for governing states of geospatial processes.

Let $m_{i,j}(t_k) = \{i, j, n, p, t_k\}$ be a message containing a payload p with $n \geq 1$ service selectors sent from CSO_{*i*} to CSO_{*j*} at time t_k . Timestamp t_k is the time of a "message-sent" event in the sender CSO_{*i*}. The message payload $p = \{\{o_r, q_r, t_r, u_r\}\}$, for $r = 1, \dots, n$ contains one or more 4-tuples, each comprising an order o_r , a measure of the quality q_r of that order,⁹ a timestamp t_r identifying when the order was issued, and a completion-time or data liveness specification u_r .

o_r is an order (possibly including metadata describing goals, objectives, plans, and constraints) that acts as a selector of one of the recipient's available services. The meaning (semantics) of an order is determined by the sender and the receiver in advance of the message being sent. Typically, a service provider publishes its services and their invocation orders and parameters in its service directory.

q_r is an indicator of the quality of parameters in an order o_r . Parameters may suffer from a number of quality concerns, including precision, accuracy, pedigree (authenticity), liveness (age and history), and source (originator and route). These concerns when quantified in a quality indication, provide recipients with a means of invoking due-diligence or risk-management activities prior to utilizing information in the order.

t_r is a timestamp declaring the time at which the order and its associated quality metric were issued. In addition to providing its age, the timestamp establishes partial ordering (sequencing) of orders. t_r is the timestamp of the "order-issued" event, the time when a client CSO first issued the order to the server CSO.

u_r , as previously defined, contains the completion-time specifications (e.g., in the form of a TUF) for the invocation request contained in o_r . The timeliness parameters contained in u_r establish the service's expected time-dependent contribution to the requestor's value proposition.

⁹We include a quality metric expressly for situations where pedigree, validity, precision, or accuracy of orders (selectors) may be in question (e.g., a measurement provided by a sensor in need of calibration or one whose identity has not been verified).

⁸For example, executing the OS command "ipconfig/renew."

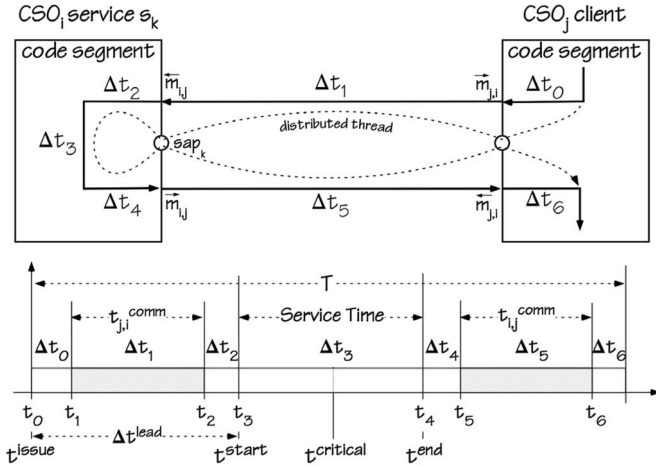


Fig. 11. Service timing.

B. End-to-End Considerations

Messages, once in transit, travel at the speed of the medium (link) through which they propagate. In infospace, contemporary media offer link speeds ranging from 64 kb/s ($\sim 10^4$ b/s) on “last mile” digital-voice telecom circuits (DS0), 1–40 Mb/s ($\sim 10^7$ b/s) on intermediate (DS1 through DS3) circuits, to 39.813 Gb/s ($\sim 10^{10}$ b/s) on OC-768 optical intermetro SONET circuits, and beyond.

Referring to Fig. 11, let $v_{i,j}$ and $v_{j,i}$ be the mean link speed,¹⁰ in bits per second, of circuits connecting server CSO_i and client CSO_j . The transmission of a service invocation request from CSO_j to CSO_i takes

$$\Delta t_1 = \text{Length} [m_{j,i}^s] / v_{j,i} \pm \varepsilon \quad (\text{in seconds})$$

where ε is the clock-offset error between endpoints. In the reverse direction, the transmission of service-invocation results takes

$$\Delta t_5 = \text{Length} [m_{i,j}^s] / v_{i,j} \pm \varepsilon \quad (\text{in seconds}).$$

For $\{\Delta t_1, \Delta t_5\} \gg \varepsilon$, clock error may be ignored. Total roundtrip service time, from sending a request to receipt of its returned results, takes $T_i^s = \sum_{i=0}^6 \Delta t_i$ (in seconds). In addition to the on-the-wire transmission times, T_i^s includes processing time on both ends, the dominant (controllable) element being the service time Δt_3 , which is the focus of TUF specifications. Other times, $\{\Delta t_0, \Delta t_6\}$ and $\{\Delta t_2, \Delta t_4\}$, represent the cost of EOS kernel scheduling and activation inherent in client and server host machines, respectively.

Again, messages and the orders they contain are selectors of behavior in recipients (servers). Typical of free-market dynamics, the value of a server’s timely response is determined by clients in their selection of TUFs accompanying their requests.

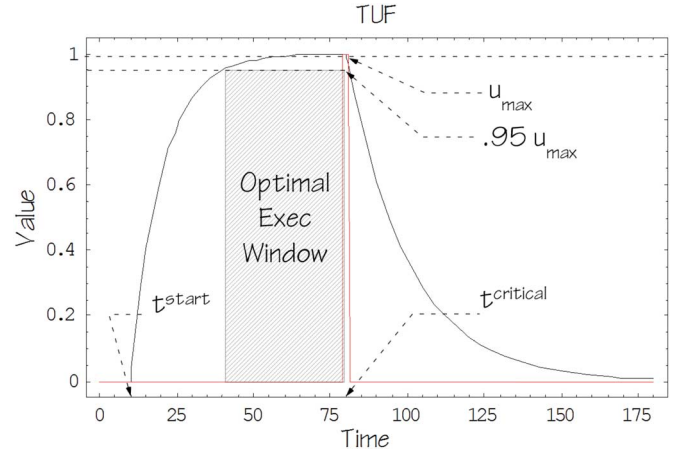


Fig. 12. Example TUF.

TABLE I
EXAMPLE TUF CHARACTERISTICS

Velocity (val/sec):	$\begin{cases} 0.271828 e^{-0.1t} & 10 \leq t \leq 80 \\ -2.72991 e^{-0.05t} & t > 80 \end{cases}$
Time-Value (val-secs):	$\begin{matrix} i \rightarrow c & c \rightarrow e \\ 60.0091 & 19.3961 \end{matrix}$
Avg val/sec:	$\begin{matrix} 0.750114 & -0.277086 \end{matrix}$
Mean Value:	0.56718

Fig. 12 shows a generalized TUF that exhibits several client-specified characteristics. For this example

$$\text{tuf}[t_{\text{issue}}, \text{start}, \text{end}, \text{umax}, \text{crit}] := \text{Piecewise}[\{ \{0, t < \text{start}\}, \{ \text{umax} (1 - \text{Exp}[-.1(t - \text{start})]), 0 \leq t \leq \text{crit} \}, \{ \text{umax} (\text{Exp}[-.5(t - \text{crit})]), t > \text{crit} \}, \{0, t > \text{end}\} \}];$$

The deadline (t^{critical}) is defined at the point in time when the utility of the request is maximized, $u^{\text{max}} = \text{tuf}(t^{\text{critical}}) = 1.0$.

For this example (ref. Table I), the area under the curve in the interval $[t^{\text{start}}, t^{\text{critical}}]$ is ~ 60 value-time units and in the tail interval $[t^{\text{critical}}, t^{\text{end}}]$ is ~ 19 value-time units, giving a total of ~ 79 value-time units. For this request, 75% of the value occurs prior to the deadline, within 70 s following start at $t = 10$. Value grows during the $10 < t < 80$ interval at a rate of 0.75 value/s and deteriorates at -0.28 value/s during the $80 < t < 175$ tail period. Late execution (i.e., missing the deadline) is clearly of marginal value. To achieve 95% of the maximum value requested, the service scheduler should respond in the $40 \text{ s} < t < 80 \text{ s}$ execution window shown in the figure. The execution window describes the period within which the server CSO_i should produce its response to client CSO_j ’s request. The window, therefore, provides a bound to the service time Δt_3 .

Given the timing profile shown in Fig. 11 and the utility function shown in Fig. 12, the following two issues present themselves: 1) the definition of the temporal scope of the TUF specification and 2) assumptions sender and receiver must make relative to their respective roles in achieving end-to-end commitments.

¹⁰If up- and downlink speeds are the same at both ends, $v_{i,j} = v_{j,i}$.

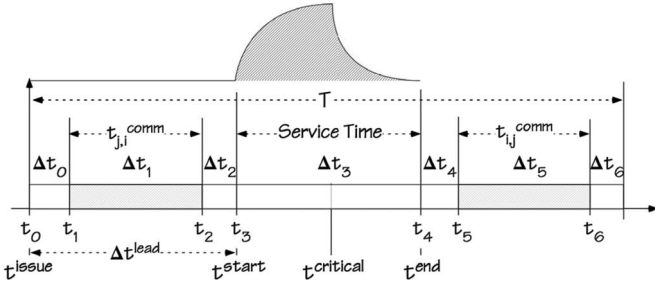


Fig. 13. TUF-specified service time.

Concerning 1) and with reference to Fig. 13, the client-specified TUF profile ignores effects of the service epilog period $[t_4, t_6]$. Except to note the time of issuance of the service request, the profile also ignores details of the prolog period $[t_0 = t_{\text{issue}}, t_3]$. Δt_2 , Δt_3 , and Δt_4 are clearly server-side matters, their specifications critical for a client to anticipate and meet its overall performance objectives. Server-centric TUF development requires the client to factor into the design both prolog and epilog timing consideration. This, in turn, requires service providers to specify, to the statistical degree possible, useful estimates (bounds) of their Δt_2 , Δt_3 , and Δt_4 processing times within their SLAs. These specifications, derivable in a fashion similar to estimating NTP delays, will necessarily qualify the efficacy of the ask–bid–accept trading cycle.

In a lightly loaded server, a single service request poses little difficulty. The challenge arises when a payload containing several service requests arrives from a single client or several requests arrive from different clients in close proximity, all demanding responsive service. Fig. 14 shows an example showing a set of four such requests (TUFs), their relation to one another in time and value, and their respective deadlines over a 200-s interval. As aforementioned, an example means of scheduling overlapping (competing) requests in a service supporting DRTSJ distributable threads involves UA scheduling mechanisms [23].

Having offered a mechanism for specifying temporal properties of transactions (e.g., RMI) within distributed applications, we define next how to measure overall application-level performance in a CPS, whose viability depends on achieving certain value propositions—propositions with real-time constraints. In short, we require a set of AQoS metrics consistent with TUF specifications.

C. Performance Metrics

In this paper, a CPS is considered a sovereign and rational free-market (Keynesian) entity whose sociospatial interactions require services predicated on quantifiable value propositions. Specific value propositions underwrite operating policies that, when properly formed and executed, are sufficient to sustain its existence, establishing and nurturing associated marketplaces (ecosystems) supporting exchange of goods and services. Achievement of individual and group values requires monitoring of self and group sociospatial, geospatial, and infospatial operations by each member. Furthermore, competitiveness in a given ecosystem demands continuous improvement in effectiveness (e.g., energy efficiency and capacity utilization)

of value production processes, individually within and collectively among cooperating CPS. Such monitoring necessarily requires a set of shared, normalized, and domain (federation)-neutral metrics—presumably defined and refined by the equivalent of a cyberspatial bureau of standards.

To that end, we offer the following six performance indexes, which are described in [7]: three primaries and three secondaries derived from the primaries for each service offered by a given CSO.

Actuality $\alpha_i^s(t)$: A measure of throughput yield [in messages per second (mps)] currently being achieved by service s in CSO $_i$ given its present level of resources.

Capability $\chi_i^s(t)$: A measure of throughput yield (in mps) of service s in CSO $_i$ possible given its current level of resources.

Potential $\pi_i^s(t)$: A measure of throughput yield (in mps) of service s in CSO $_i$ possible given its maximum (design) level of resources.

Latency $\lambda_i^s(t) \equiv \chi_i^s(t)/\pi_i^s(t)$: A measure (in percent) of the capacity latent in the potential of service s in CSO $_i$.

Productivity $\gamma_i^s(t) \equiv \alpha_i^s(t)/\chi_i^s(t)$: A measure (in percent) of utilization of the current capability of service s in CSO $_i$.

Performance $\psi_i^s(t) \equiv \alpha_i^s(t)/\pi_i^s(t) = \lambda_i^s(t) * \gamma_i^s(t)$: A measure (in percent) of the current utilization of the potential of service s in CSO $_i$.

The three primaries satisfy $\alpha_i^s(t) \leq \chi_i^s(t) \leq \pi_i^s(t)$ (in mps).

For an example that ignores scripting and normalizes the design potential of a given service to $\pi = 1$ (100%), suppose its current capability is $\chi(t) = 0.5$ mps (50%) and its measured actuality is $\alpha(t) = 0.35$ mps (35%). The result is a productivity index of $\gamma(t) = 0.35/0.5 = 0.7$ (70%), a latency index of $\lambda(t) = 0.5/1 = 0.5$ (50%), and an overall performance index of $\psi(t) = (0.35)/(1.0) = (0.7)(0.5) = 0.35$ (35%).

Armed with these six AQoS performance metrics, we are in position to discuss various dynamic properties of CSO and their services.

D. CSO Service Dynamics

As stated in the introduction, we require each CSO to be viable (i.e., self-sustaining). Viability is achieved through provision of one or more services deemed valuable to members of its ecosystem. As discussed, we assume that each CSO offers its services through an infospatial SAP. A CSO may also have “brick and mortar” service portals located in geospace. Services are invoked through messages addressed to the SAPs, as shown in Fig. 15 (refer to discussions related to Figs. 11 and 13).

Let $\vec{m}_{i,j}^s(t)$ and $\overleftarrow{m}_{i,j}^s(t)$ be messages carrying service orders (demands) from and responses to, respectively, clients CSO $_j$ of service s in server CSO $_i$ at time t . It follows that $d\vec{m}_{i,j}^s(t)/dt$ and $d\overleftarrow{m}_{i,j}^s(t)/dt$ are the corresponding messaging rates (in mps), into and out of service s .¹¹ For lossless channels, $\vec{m}_{i,j}^s(t) = \overleftarrow{m}_{j,i}^s(t)$ and $\overleftarrow{m}_{j,i}^s(t) = \vec{m}_{i,j}^s(t)$.

¹¹Henceforth, to simplify the notation without loss of generality, we drop the subscript “ k ,” distinguishing a particular service s_k .

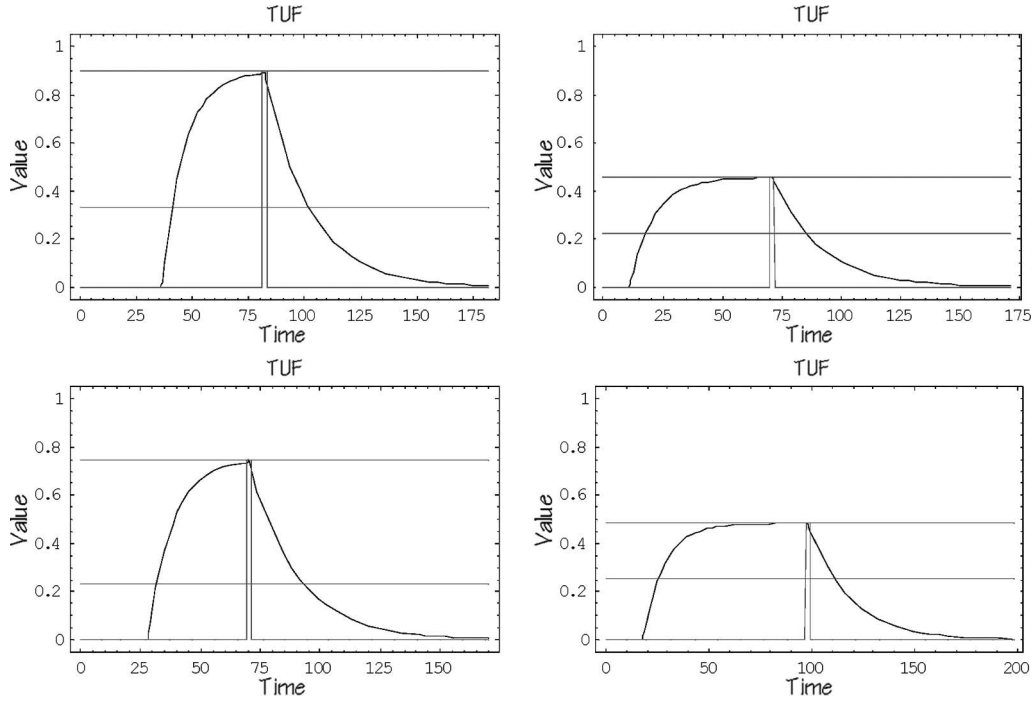


Fig. 14. Service-time utility set.

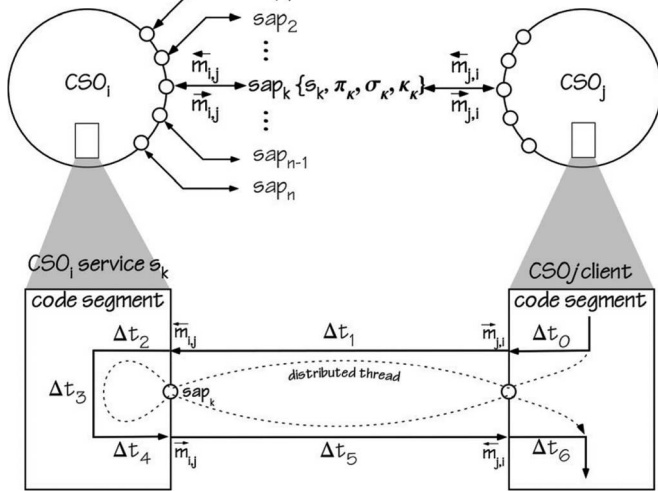


Fig. 15. CSO service model.

The bidirectional partial and total message volume of service s in CSO_i at time t is the sum of all inbound requests and outbound responses,¹² respectively

$$\begin{aligned}\mu_{i,j}^s(t) &= \overleftarrow{m}_{i,j}^s(t) + \overrightarrow{m}_{i,j}^s(t) \\ \mu_i^s(t) &= \sum_j [\mu_{i,j}^s(t)] = \sum_j [\overleftarrow{m}_{i,j}^s(t) + \overrightarrow{m}_{i,j}^s(t)].\end{aligned}$$

¹²We do not distinguish between meaningful and meaningless (i.e., spam) messages since both require some degree of processing. If we did, spam would likely be recast as a noise source.

The corresponding message rates (in mps) through service s in CSO_i at time t are, respectively

$$\begin{aligned}\dot{\mu}_{i,j}^s(t) &= d\mu_{i,j}^s(t)/dt = d\overleftarrow{m}_{i,j}^s(t)/dt + d\overrightarrow{m}_{i,j}^s(t)/dt \\ \dot{\mu}_i^s(t) &= d\mu_i^s(t)/dt = \sum_j [d\overleftarrow{m}_{i,j}^s(t)/dt + d\overrightarrow{m}_{i,j}^s(t)/dt].\end{aligned}$$

Symmetry and lossless channels require that¹³

$$\begin{aligned}d\overleftarrow{m}_{i,j}^s(t)/dt &= d\overrightarrow{m}_{j,i}^s(t)/dt \\ d\overrightarrow{m}_{i,j}^s(t)/dt &= d\overleftarrow{m}_{j,i}^s(t)/dt.\end{aligned}$$

During the period $[t_0, t]$, the number of messages processed by service s is

$$N_i^s[t_0, t] = \int_{t_0}^t \mu_i^s(t) dt = \int_{t_0}^t \sum_j [\overleftarrow{m}_{i,j}^s(t) + \overrightarrow{m}_{i,j}^s(t)] dt.$$

Over the same period, the number of messages processed at the two endpoints is

$$\begin{aligned}N_{i,j}^s[t_0, t] &= \int_{t_0}^t \mu_{i,j}^s(t) dt = \int_{t_0}^t \overleftarrow{m}_{i,j}^s(t) dt + \int_{t_0}^t d\overrightarrow{m}_{i,j}^s(t) dt \\ N_{j,i}^s[t_0, t] &= \int_{t_0}^t \mu_{j,i}^s(t) dt = \int_{t_0}^t \overleftarrow{m}_{j,i}^s(t) dt + \int_{t_0}^t d\overrightarrow{m}_{j,i}^s(t) dt.\end{aligned}$$

We define the actuality metric (actual throughput, in mps) for service s in CSO_i as the message-processing (i.e., service completion) rate in mps measured at its associated SAP

$$\alpha_i^s(t) = \dot{\mu}_i^s(t).$$

¹³An infospatial form of Kirchhoff's electrical current law.

Similarly, the partial actuality metric (partial throughput, in mps) for services s in CSO_i , with respect to requests from CSO_j , is defined as

$$\alpha_{i,j}^s(t) = \dot{\mu}_{i,j}^s(t).$$

Definitions of capability and potential are somewhat less intuitive. A given cyberspatial server¹⁴ (i.e., a platform or host) may be capable of supporting multiple services. That server's resources are assigned to services according to policies concerned with marginal utility, mean service time, server load, criticality of service request (e.g., TUF parameters), hardware platform capacity, availability, return on capital investment, etc.

Over its lifetime, a server's capacity may evolve through the installation of a progressively larger fraction, within design limits, of its total physical resources (processor, memory, disk, network adapters, personnel, capital, etc.).¹⁵ Services allowed to run on that server will share its current capacity (i.e., its increasing potential), which is a full hardware complement representing the server's full capacity. If that capacity were assigned to a single service, the service would achieve its full potential (relative to that server, at least). If, on the other hand, the server's full potential were allocated (by some allocation policy) to all executable services, then each would have some measurable capability, but still the service does not achieve its individual full potential on that server.

Let Γ be a server's maximum potential measured in instructions per second (ips)¹⁶ when configured with its full complement of hardware resources. Let $\omega\Gamma$ be the server's capability (in ips) when operating at a fraction ($0 < \omega \leq 1$) of its maximum potential. With reference to the service model shown in Fig. 15, let $\pi_k(t)$ be the maximum (potential) service rate in mps for service s_k at access point sap_k when running alone on the full potential (i.e., dedicated) server. Let κ_k be the number of instructions per message (ipm) required by service s_k to react to a given message, with $\bar{\kappa} = 1/n \sum_1^n \kappa_i$, the average number of ipm for all n concurrent services. The server is thus capable of an average of $\omega\Gamma/\bar{\kappa}$ mps.

Let $\sigma_k(t)$ be the fraction of server capacity available to service s_k (by policy) when the server is shared and running with all other services, such that

$$\begin{aligned} \sum_i \sigma_i(t) &= 1 \\ \sigma_i(t) &\geq 0 \\ \sum_i \sigma(t)_i \pi_i(t) \kappa_i &= \omega\Gamma. \end{aligned}$$

Given that the server's maximum potential Γ is achieved at $\omega = 1$, it follows that service s_k achieves its maximum potential of $\pi_k(t) = \omega\Gamma/\sigma_k(t)\kappa_k$ (in mps) when running alone on a

dedicated server, with $\omega = \sigma_k(t) = 1$. When running with the other $n - 1$ services, s_k 's capability is

$$\begin{aligned} \chi_i^{s_k}(t) &= \sigma_i^{s_k}(t) \pi_i^{s_k}(t) \\ &= \omega\Gamma/\bar{\kappa} - \left[\sum_{j=1}^{j=k-1} \sigma_i^{s_j}(t) \pi_i^{s_j}(t) + \sum_{j=k+1}^{j=n} \sigma_i^{s_j}(t) \pi_i^{s_j}(t) \right]. \end{aligned}$$

Let $\hat{\pi}(t) = \max(\pi_i(t))$, where $i = 1, \dots, n$, be the potential of the service having the greatest capability and $\hat{\pi}_i = (\pi_i/\hat{\pi}) \leq 1$, $\forall i$ be the resulting normalized potential of each service.

Then, the normalized capability index for a service is

$$\begin{aligned} \hat{\chi}_i^{s_k}(t) &= \sigma_i^{s_k}(t) \hat{\pi}_i^{s_k}(t) \\ &= \omega\Gamma/\bar{\kappa} - \left[\sum_{j=1}^{j=k-1} \sigma_i^{s_j}(t) \hat{\pi}_i^{s_j}(t) + \sum_{j=k+1}^{j=n} \sigma_i^{s_j}(t) \hat{\pi}_i^{s_j}(t) \right]. \end{aligned}$$

Again, for clarity and generality, we drop the subscript k .

The normalized productivity index for service s is

$$\hat{\gamma}_i^s(t) = \alpha_i^s(t)/\hat{\chi}_i^s(t) = \dot{\mu}_i^s(t)/\sigma_i^s(t)\hat{\pi}_i^s(t).$$

The normalized latency index for service s is

$$\hat{\lambda}_i^s(t) \equiv \hat{\chi}_i^s(t)/\hat{\pi}_i^s(t) = \sigma_i^s(t)\hat{\pi}_i^s(t)/\hat{\pi}_i^s(t) = \sigma_i^s(t).$$

The normalized performance index for service s is

$$\hat{\psi}_i^s(t) \equiv \alpha_i^s(t)/\hat{\pi}_i^s(t) = \dot{\mu}_i^s(t)/\hat{\pi}_i^s(t).$$

By definition, these six performance indexes are independent of an object's cyberspatial location. They are applicable, intentionally, to any object (agent process and service) regardless its sociospatial role (parent, child, producer, consumer in Fig. 6). Consequently, the indexes provide a scale-free means (along both horizontal production and vertical command axes) of comparing the performance of two or more collaborating (or competing) enterprise objects.

IV. SERVICE VALUE PROPOSITIONS

We conclude our introduction of cyberspatial mechanics with a discussion of the value of a unit or quantum of service. Value, as introduced in [5] and [6], is an intrinsically sociospatial notion, typically associated with the idea that the viability of an autonomous system depends on the degree its factories (CSO) are profitable—able to produce products and services predicated on marketable value propositions. Value propositions are predicates (cost-benefit constraints) governing how factories convert payloads (orders), the raw material in messages, into results that are of utility to clients. Value is in the proverbial “eye of the consumer.”

¹⁴A cyberspatial server may be a small embedded or standalone computational device, a network of such devices, a business unit, corporate, civil, or military agency, etc.

¹⁵Equivalently, imagine servers being replaced periodically with higher performance servers.

¹⁶Measures of ips are typically specified in terms of performance against “SPECint,” “SPECfp,” etc., test suites.

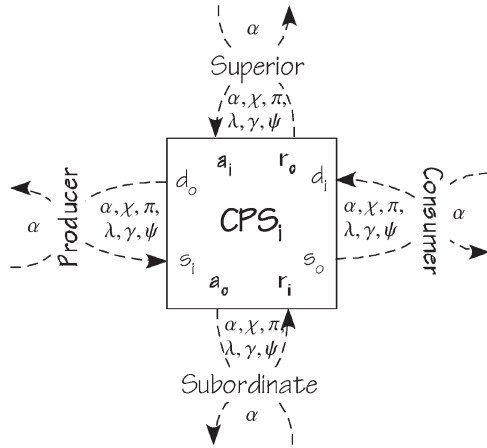


Fig. 16. CPS_i service interfaces.

A. CPS : Value-Production Units

Clearly, there are aspects of cost, particularly capital assets, which are derived from a CPS 's geospatial and infospatial structure. These are typically platform or hosting (factory) costs, distinct from the value derived of services running on the platform and costs typically depreciated over very long periods with respect to the duration of requested services. While not insignificant, we leave this command axis investment (i.e., balance sheet) aspect of the discussion to a future paper.

Within a given ecosystem, a cyberspatial system participates through offering one or more services. Recalling our discussion related to Fig. 6, a CPS (i.e., each of its $CSOs$) communicates with its neighbors in its ecosystem via specific SAPs. Each CPS supports at least four SAP, one for each of the four principal operational roles it plays. As shown in Fig. 16, it may operate as 1) a superior to other subordinate CPS , 2) a producer (supplier) to other consumer CPS , 3) a client (consumer) of other producer CPS , or 4) a subordinate to other superior CPS . A CPS will typically play a given role in one federation while concurrently playing a different role in other federations. Notions of value are thus dependent on the role it is playing at a given time in a given federation (i.e., its operational context).

Operational roles are defined through a set of services and associated protocols present at each CPS application program interface (API), ref. Fig. 16.

Superior API:

- a_i Assets in—demand orders and accompanying assets issued by a superior to its subordinate CSO .
- r_o Returns out—results returned by a subordinate CSO to its superior in response to demand orders and asset allocations.

Subordinate API:

- a_o Assets out—demand orders and accompanying assets issued by a CSO to its subordinates.
- r_i Returns in—results returned by subordinates related to CSO demand orders and asset allocations.

Consumer API:

- d_i Demand in—supply demand orders issued by clients of a CSO .
- s_o Supply out—supplies issued to clients of CSO in response to their demand orders.

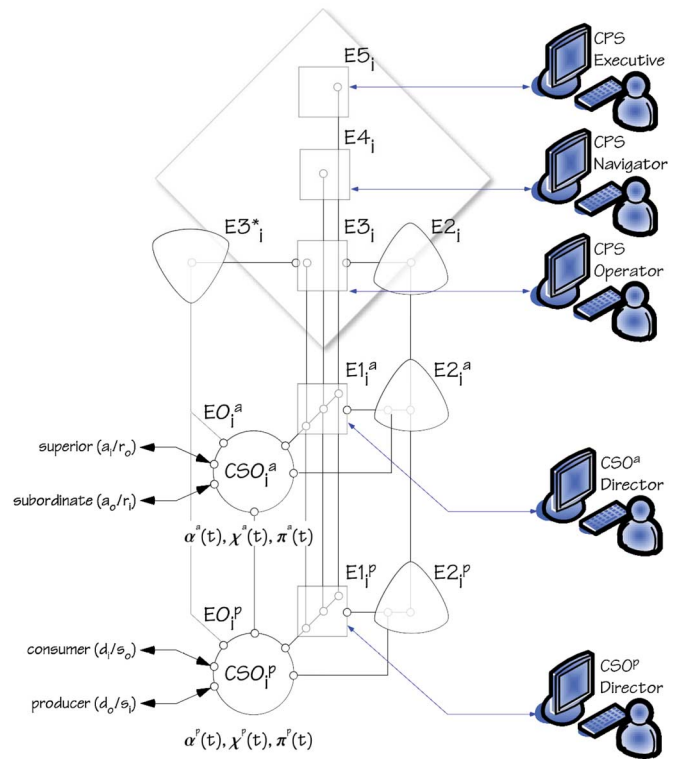


Fig. 17. CPS_i governance structure.

Producer API:

- d_o Demand out—supply orders issued by a CSO to its suppliers.
- s_i Supply in—supplies issued by producers in response to CSO supply orders.

As previously defined, one or more orders are carried within the payload of a message, each order acting as a selector of one of possibly many behaviors enabled by a recipient's (server's) state of conditional readiness. The value of an order is defined in terms of a sender's (client's) expected benefits from a service invocation. The client is therefore responsible for encoding its value proposition in the TUF parameters accompanying the order.

In our cybernetic model, CPS governance (Fig. 17) is implemented through two complementary and concurrent sets of services, one (CSO_i^a) dedicated to providing services to superiors and subordinates along its vertical asset (command) chain and one (CSO_i^p) dedicated to serving consumers and producers along its horizontal production (supply) chain. The performance of CSO_i^a is characterized by $\{\alpha_i^a(t), \chi_i^a(t), \pi_i^a(t)\}$ and CSO_i^p by $\{\alpha_i^p(t), \chi_i^p(t), \pi_i^p(t)\}$. These CSO are necessarily coupled to achieve balance (homeostasis) among competing demands flowing through the CPS . Their coupling is both direct (CSO_i^a to CSO_i^p), as diagrammed, and indirect through the CPS ' command structure (E5–E4–E3–E1).

B. Adaptive Production Control

The regulatory control of a service (e.g., CSO_i^p) and its associated E0–E2–E1–E0 loop is embedded in and administered by the E3–E1–E2–E3 supervisory control loop. These two

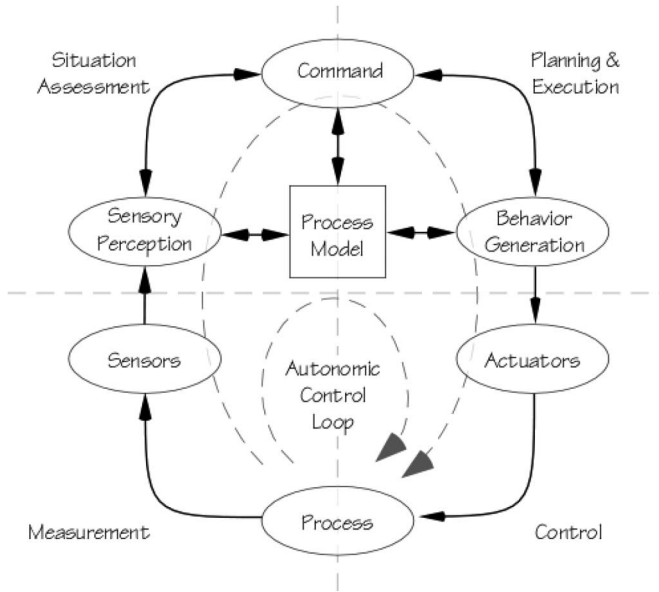


Fig. 18. Cyclic-feedback control loop.

complementary control loops are the cyberphysical analog of the human autonomic nervous system, providing a CPS with its reflex arc, its motor controls. The cybernetic feedback control cycle governing these two loops is shown in Fig. 18. Sensors observe running processes; assessment rationalizes sensor measurements to form perceptions; process models are updated and maintained; action plans governing present-state-to-next-state transitions are formulated; commands are issued to actuators; and the process is controlled. Oversight of this lower level autonomic activity is provided by a supervisory (command) function that allows human or synthetic (e.g., artificial intelligence) intervention of perception and planning.

At the regulatory level (Fig. 19), the production axis supports communications among producers and consumers in regulating flows of goods and services in the supply chain. The figure shows two concurrent supply chain value production cycles under control and influenced by the aforementioned thread-scheduling parameters. In Fig. 19(a), a message carrying a client order arrives at the d_i service point. The request enters CSO_i^p , goes into production, and ends with results returned to the client through service point s_o . In a complementary fashion, Fig. 19(b) shows CSO_i^p issuing a supply order from its d_o service point following execution of a production loop that presumably hits an inventory reorder point. The supply order is followed by receipt of supplies (results) at its s_i service point. Similar scenarios take place in CSO_i^a on behalf of command-chain tasking and asset-request orders.

Coordinating demands on these four SAPs requires proactive and reactive supervisory control. Fig. 20 shows a supervisory-control cycle attempting to synchronize the four concurrent threads arising from a subordinate via CSO_i^a and a client via CSO_i^p . In the situation depicted in Fig. 20, CPS_i may be in the process of “outsourcing” a client’s supply-chain service order to a subordinate (asset chain) factory. Interactions between supply and asset chains are varied, but their real-time coordination requirements are quite common.

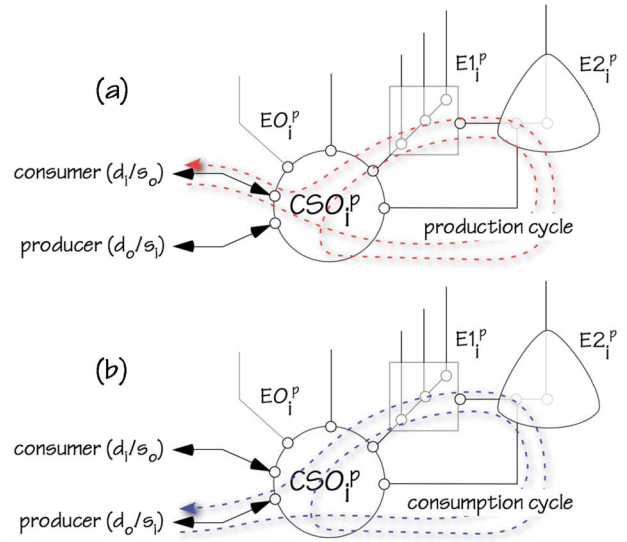


Fig. 19. CPS_i regulatory control cycles.

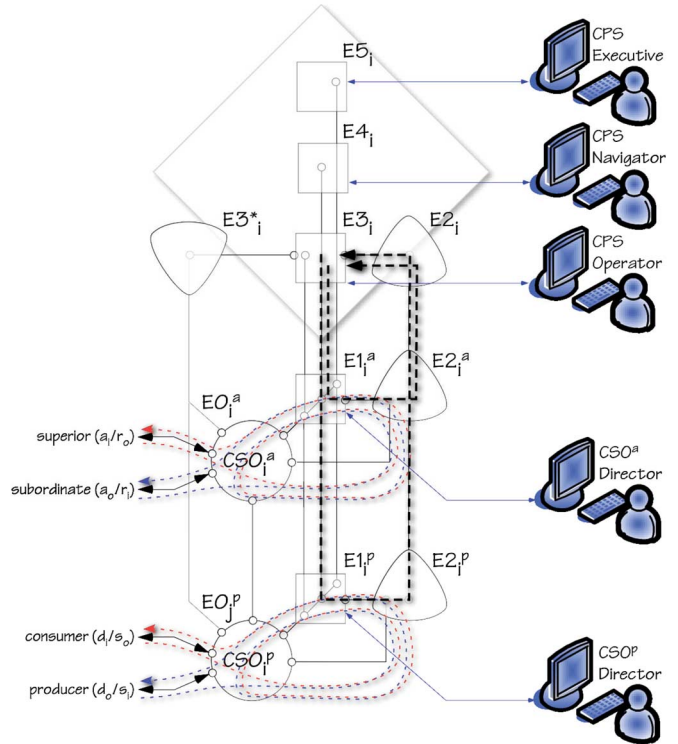


Fig. 20. CPS_i supervisory control.

C. Value Metrics

The effectiveness of CPS governance is measured by externally and internally visible metrics. Externally (ref. Fig. 16), clients can see the level of service they receive by measuring their partial actuality $\alpha_{i,j}^s(t) = \dot{\mu}_{i,j}^s(t)$. Unless they all get together and compare their partial actualities, they cannot discern CPS_i ’s total actuality $\alpha_i^s(t) = \dot{\mu}_i^s(t)$ nor assess its internal capability, potential, latency, productivity, and performance indexes unless CPS_i chooses to publish its performance indexes to its federation affiliates.

We defined $v(o_r, t) = \text{tuf}_r(t)$ to be the value of an order at time t , as defined by the client’s TUF specifications.

Let $\bar{v}(o_r)$ be the mean value (utility) of an order

$$\bar{v}(o_r) = \frac{1}{(t_r^{\text{end}} - t_r^{\text{start}})} \int_{t_r^{\text{start}}}^{t_r^{\text{end}}} \text{tuf}_r(\tau) d\tau.$$

Let $\hat{v}(o_r)$ be the maximum value of an order

$$\hat{v}(o_r) = \max [\text{tuf}_r(t), \{t^{\text{start}} \leq t \leq t^{\text{end}}\}].$$

Let $\tilde{v}(o_r, t)$ be the actual value achieved by a server in completing an order at time t

$$\tilde{v}(o_r, t) = \frac{1}{(t - t_r^{\text{start}})} \int_{t_r^{\text{start}}}^t \text{tuf}_r(\tau) d\tau, \quad t^{\text{start}} \leq t \leq t^{\text{end}}.$$

Let $\bar{C}_i^s(o_r, t)$ be the expected cost in server CSO_i for execution of order o_r issued at time t

$$\bar{C}_i^s(o_r, t) = k_s \bar{v}(o_r) + \frac{k_l}{(t^{\text{start}} - t)}, \quad \text{for } t < t^{\text{start}}$$

where k_s is a unitless pricing ($k_s > 1$) or discount ($0 \leq k_s \leq 1$) strategy, and k_l (in dollars times seconds) is a penalty for clients issuing orders with short lead times ($\Delta t^{\text{lead}} = t^{\text{start}} - t$).

Let $\hat{C}_i^s(o_r, t)$ be the cost in the server for achieving the client's maximum utility $\hat{v}(o_r)$ of an order issued at time t

$$\begin{aligned} \hat{C}_i^s(o_r, t) &\geq \bar{C}_i^s(o_r, t) \\ &= \bar{C}_i^s(o_r, t) (1 + M_i^s(t)) \end{aligned}$$

where $M_i^s(t)$ is the margin (fee) charged by the server for completing an order issued at time t

$$\begin{aligned} M_i^s(t) &= k_\chi \lambda_i^s(t) + k_v \frac{\tilde{v}(o_r, t) - \bar{v}(o_r)}{\hat{v}(o_r)} \\ &= k_\chi \frac{\chi_i^s(t)}{\pi_i^s(t)} + k_v \frac{\tilde{v}(o_r, t) - \bar{v}(o_r)}{\hat{v}(o_r)} \end{aligned}$$

where k_χ is the weight given in maintaining sufficient latent potential (service capacity) to execute new orders, and k_v is the weight given to the server's success in realizing the client's maximum utility. In this model, the cost to the client is based on the mean value $\bar{v}(o_r)$, while the marginal incentive in the server is to exceed the mean value. Achieving less than the mean results in the margin being negative and, therefore, reduces the cost to the client.

The client's cost (i.e., the server's bid price) $C_j^s(o_r, t)$ for execution of order o_r issued at time t is, therefore

$$\begin{aligned} C_j^s(o_r, t) &= C_i^s(o_r, t) (1 + M_i^s(t)) \\ &= \left(k_s \bar{v}(o_r) + \frac{k_l}{t^{\text{start}} - t} \right) (1 + M_i^s(t)) \\ &= \left(k_s \bar{v}(o_r) + \frac{k_l}{t^{\text{start}} - t} \right) \\ &\quad \times \left(1 + k_\chi \frac{\chi_i^s(t)}{\pi_i^s(t)} + k_v \frac{\tilde{v}(o_r, t) - \bar{v}(o_r)}{\hat{v}(o_r)} \right). \end{aligned}$$

Above the mean, the server gains additional revenue; below the mean, the client gets a discount. In the case (ref. Fig. 14) where a server is faced with executing several orders whose maxima are all clustered around the same deadline, the margin calculation provides a strong bias. Server CSO_i 's goal, simply stated, is to maximize the productivity ($\gamma_i^s(t)$) of its assets and its margin ($M_i^s(t)$) while minimizing the cost ($C_i^s(o_r, t)$) of its service.

V. SUMMARY

In this introduction to cyberspatial mechanics, we have outlined several features of a general theory of CPSs, including the following:

- 1) a functional definition of cyberspace;
- 2) the notion of sociospatial objects (rational agents), including an intra-CPS (inter-CSO) governance model;
- 3) interobject distance metrics and a means of rationalizing cyberspatial distances;
- 4) the role of clocks, synchronization, and timeliness in reasoning about the dynamics of CSOs;
- 5) the structure of sociospatial federations and a federated (inter-CPS) organizational model;
- 6) timeliness and the essential role and characteristics of an interobject message model;
- 7) a uniform, scalable, and application-neutral (i.e., canonical) set of performance metrics;
- 8) a model for expressing the cost performance basis of CSO value production processes.

Furthermore, this introduction suggests a number of important topics needing further research, including the following:

- 1) development of military and civil command and control (C2) strategies (e.g., natural- and terrorist-disaster and infrastructure management) based on cyberspatial references;
- 2) further development of cyberspace-time physics;
- 3) in relation to a DERM, development of a CSO directory structure and associated structured query language;
- 4) specification of a (standards-compliant) cyberspatial addressing mechanism (IPv6 + GPS);
- 5) development and specification of an EOS and associated CPS "virtual machine" execution logic;
- 6) modeling and analysis of stability in the CPS sympathetic and parasympathetic control loops;
- 7) the role of TUFs in achieving enterprise performance (SLA) objectives;
- 8) further development of science and technology related to collaboration (e.g., via trading protocols) among viable federated (autonomic and sovereign) systems.

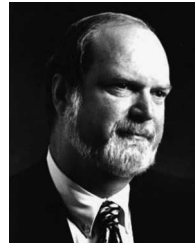
On a broader academic front, this paper suggests new analytic and pedagogical approaches in the fields of business, economics, information, computer, social, and geophysical sciences, including the following:

- 1) expanding the domain of systems science (cybernetics) to include unification of cyberspatial reference frames;
- 2) expanding engineering disciplines to include CPSs analysis and design concepts;

- 3) expanding (geo)physical sciences, including DERMs, to include location, interrelationships, and behaviors of intelligent-agent-based CSOs;
- 4) expanding (business) management, economics, and social sciences with formal concepts related to services science and its associated requirements for operational (technical) enterprise governance systems;
- 5) expanding computer and information sciences with concepts of EOSs (execution environments) and associated business-process engineering that integrate geospatial, infospacial, and sociospatial dynamics.

REFERENCES

- [1] *ACE/TAO*. [Online]. Available: <http://www.cs.wustl.edu/~schmidt/TAO.html>
- [2] J. Albus, "Outline for a theory of intelligence," *IEEE Trans. Syst., Man, Cybern.*, vol. 21, no. 3, pp. 473–509, Jun. 1991.
- [3] J. S. Anderson and E. D. Jensen, "Distributed real-time specification of Java (DRTSJ)—A status report (digest)," in *Proc. JTRES*, Paris, France, Oct. 11–13, 2006, pp. 3–9.
- [4] R. Ashby, *Introduction to Cybernetics*. London, U.K.: Chapman & Hall, 1957.
- [5] J. S. Bayne, "A software architecture for control of value production in federated systems," in *Proc. World Multi-Conf. Syst., Cybern. Inf.*, Orlando, FL, Aug. 2003, pp. 1–6.
- [6] J. S. Bayne, "A software architecture for control of value production in federated systems," *J. Syst., Cybern. Inf.*, vol. 1, no. 8, pp. 60–65, Aug. 2003.
- [7] J. S. Bayne, *Creating Rational Organizations—Theory of Enterprise Command and Control*, 2006, Café Press. [Online]. Available: www.cafepress.com/mcsi
- [8] S. Beer, *The Brain of the Firm*. Hoboken, NJ: Wiley, 1994.
- [9] S. Beer, *Decision and Control*. Hoboken, NJ: Wiley, 1988.
- [10] R. C. Conant, "Laws of information which govern systems," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-6, no. 4, pp. 240–255, Apr. 1976.
- [11] T. Erl, *Service-Oriented Architecture*. Englewood Cliffs, NJ: Prentice-Hall, 2005.
- [12] J. Forrester, *Collected Papers*. Bala Cynwyd, PA: Pegasus Commun., 1975. [Online]. Available: <http://www.systemdynamics.org/>
- [13] [Online]. Available: <http://jcp.org/en/jst/detail?id=50>
- [14] [Online]. Available: http://uk.encyclopedia.msn.com/encyclopedia_761577951/Homeostasis.html
- [15] [Online]. Available: <http://www.infospherics.org>
- [16] [Online]. Available: <http://www.oasis-open.org>
- [17] [Online]. Available: <http://www.pyxisinnovation.com/pyxwiki/index.php?title=Handbook>
- [18] [Online]. Available: <http://www.real-time.org>
- [19] [Online]. Available: <http://www.rtsj.org/>
- [20] E. Jaques, *Requisite Organization*. Green Cove Springs, FL: Cason Hall, 1992.
- [21] E. E. Jensen, "Utility functions: A general scalable technology for software execution timeliness as a quality of service," in *Proc. Softw. Technol. Conf.*, Apr. 2000. [Online]. Available: <http://www.real-time.org/docs/dagstuhl1.pdf>
- [22] L. Lamport, "Time, clocks, and the ordering of events in a distributed system," *Commun. ACM*, vol. 21, no. 7, pp. 558–565, Jul. 1978.
- [23] P. Li, "Utility accrual real-time scheduling: Models and algorithms," Ph.D. dissertation, Virginia Polytechnic State Univ., Blacksburg, VA, 2004.
- [24] K. Merchant and W. Van der Stede, *Management Control Systems*. Englewood Cliffs, NJ: Prentice-Hall, 2003.
- [25] D. L. Mills, "Internet time synchronization: The network time protocol," *IEEE Trans. Commun.*, vol. 39, no. 10, pp. 1482–1493, Oct. 1991.
- [26] Object Management Group (OMG), Real-Time CORBA Specification. V1.2. [Online]. Available: <http://www.omg.org/cgi-bin/doc?formal/05-01-04>
- [27] RFC1305, *NTP Standard*. [Online]. Available: www.ietf.org/rfc/rfc1305.txt
- [28] RFC2460, *IPv6 Standard*. [Online]. Available: www.ietf.org/rfc/rfc2460.txt
- [29] RFC2801, *IOTP Standard*. [Online]. Available: www.ietf.org/rfc/rfc2801.txt
- [30] RFC4330, *SNTP Standard*. [Online]. Available: www.ietf.org/rfc/rfc4330.txt
- [31] J. Spohrer and D. Riecken, "Special Issue: Services Science," *Commun. ACM*, vol. 49, no. 7, pp. 30–34, Jul. 2006.
- [32] L. Whitman and B. Huff, "The living enterprise model," in *Automation and Robotics Research Institute*. Arlington, TX: Univ. Texas, 2000.
- [33] N. Wiener, *Cybernetics*. Cambridge, MA: MIT Press, 1948.
- [34] M. Wooldridge, *Reasoning About Rational Agents*. Cambridge, MA: MIT Press, 2000.



Jay S. Bayne (S'69–M'75–SM'96) received the B.Sc. degree in electrical engineering, the M.Sc. degree in electrical engineering and computer science, and the Ph.D. degree in electrical engineering and computer science from the University of California, Santa Barbara, in 1970, 1971, and 1976, respectively.

From 1973 to 1984, he was a Professor of computer science with California Polytechnic State University, San Luis Obispo, where he founded his first company, Protocol Solutions, Inc. He has since held VP Technology and Strategic Marketing positions

for global engineering companies involved in the design and application of distributed fault-tolerant real-time control systems for automation of transportation, power production and distribution, pulp and paper, chemical, pharmaceutical, petroleum, and related continuous manufacturing processes. He is currently an Adjunct Professor of computer science with the University of Wisconsin, Milwaukee, a Consultant to the Office of the Assistance Secretary of Defense (OSD/NII), Chief Executive of his second company, Meta Command Systems, Inc., and Executive Director of the Milwaukee Institute (www.mkei.org). He has published numerous papers and a recent book on the subject of the theory and technology of real-time enterprise governance.