To:	P. K. Sowell	Date:	05 June 1997
From:	R. F. Hilliard, T. B. Rice	Memo No.:	D510-M-013
Subject:	Comments on *C4ISR Architecture Framework*		
Copies To:	Distribution		

This memo documents comments on CISA–0000–104–96, *C4ISR Architecture Framework* (version 1.0), dated 7 June 1996, available from http://www.cisa.osd.mil/. References to that document appear in parentheses, below.

General Comments

The premise of the CISA effort is that "... common terms of reference, common definitions, and a common Framework for documenting architectures will significantly improve DOD's ability to achieve a seamless, integrated C4ISR environment" (ES-6).

While we concur with this goal, the document does not achieve this. As it stands, it is incomplete. Definitions are not complete or consistent. Key concepts and distinctions crucial to architectural practice are not delineated therein.

The spirit of this work is contrary to the policy direction of recent years which has been to move the DOD from its own standards base (MIL specifications and FIPS) to that of industry (IEEE, ISO, and de facto industry standards). This policy was enacted to better position DOD to take advantage of emerging commercial technologies, including, we would observe, those pertaining to architecture. It seems particularly ironic that in 1997 DOD would seek to create its own architectural "standards" while other Military standards are being "obsolesced" in favor of commercial and international standards. There is substantial literature and emerging practice in architecture from the commercial world, industrial, research and standards communities. The CISA effort would do well to examine these ongoing activities. References to some of these are provided below.

In a related vein, the trend in acquisition has moved toward contractor-defined products derived from the contractor's corporate processes; the CISA document goes against that trend, too, in that it defines government-specified architecture products.

This issue is exacerbated by the products that are selected. The proposed products as described in this document are summary in nature and suited at best for management purposes, not architecture or engineering activities; whereas the products from a streamlined acquisition approach are fundamentally constructive[1] in nature, since contractors' products focus on what needs to be built not what needs to be managed.

To satisfy its stated goal to "provide a basis from which the community can work collectively to evolve a framework and promulgate it as DOD direction via appropriate DOD policy directives and guidance instructions" (Preface), the CISA document should describe the intended process for its revision, maintenance, and future evolution. In particular, instructions for submission of comments should be provided for readers.

Terminology

The foundation of an effort such as this should be a strong set of definitions. At present, the proposed definitions are (1) incomplete, (2) not consistent with one another, and (3) not compatible with existing community practices.

"**Framework**" as it appears in the title of this document is not defined. This is a particularly confusing term in the field of architecture because of its various uses, as in: "implementation framework," "conceptual framework," "object-oriented framework," etc., to refer to a commonly used technique for implementing a system architecture or its infrastructure.

Recommendation: Change the name of the document to *C4ISR Architecture Documentation Practices*, to avoid confusion inherent in the term "framework."

"**Architecture**" as used in the CISA document, reduces the notion to virtual meaninglessness. It appears to be used in a manner synonymous with "model"—which is a perfectly good existing term, if that is the intent here. A unified DOD modeling "framework" is perhaps a good thing. If that is the intent, **we recommend** removing all references to "architecture."

The IAP's "agreed to" definition of architecture (ES-3) mistakenly cites IEEE Std 610.12 as its source. On the contrary, the inadequacy of IEEE's present definition, which actually reads,

> "**architecture**. the organizational structure of a system or component,"

has led to an ongoing IEEE effort, the Architecture Working Group. Their baseline definition of architecture reads,

> "An architecture is the highest-level concept of a system in its environment."

The rationale for this definition together with many other relevant terminological considerations is discussed in the IEEE AWG paper at ICECCS'96.[2]

[1] Constructive in the context of the stakeholders of any particular system. This does not discount maintainers and O&M stakeholders.

Architecture is Situated

Architecture, as an activity, is located within systems' life cycles, the operational and development environments, etc. Architectural products have multiple audiences drawn from multiple system stakeholders. These products should relate to those stakeholders' needs, in the context of other products they need to do their jobs (e.g., Statements of Need, Operational Concepts Documents, System Specifications, Top-Level Design Documents, Strategy, and Policy Memoranda, etc.). Products also have ownership and other roles.

We do not see that the CISA document addresses this context of Architecture. E.g., to distinguish architecture from any other modeling activity, it is important to locate architecture in the life cycle as distinguished from requirements and design. What is the relation of proposed architecture products to existing requirements documents? Are CISA architecture products intended to supersede, summarize, and enhance other products? How are these products kept in synch with each other and with other documents?

Another aspect of context is the location of architecture in the problem v. solution "spaces"—this distinction would be useful to understand the present CISA document. Most uses of architecture distinguish descriptions of the problem from prescriptions of the solution. E.g., the IEEE definition addresses this by making architecture the "highest level" of abstraction relative to design considerations.

A problem is that "architecture" is not defined relative to a subject area. We **recommend** that architecture be defined relative to a system, where the term "system" takes on its usual meaning which ranges over individual applications, product families, product lines, systems of systems, and whole enterprises (acknowledging that one person's system is another's component, or system-or-systems, or enterprise). To a first approximation, all of these entities may be viewed "systematically" and each such system may be regarded as having an architecture. The last 50 years of systems analysis and engineering seems to bear this out. (Unless otherwise noted, the term "system" in these comments is used to encompass all of these levels.)

Operational–Systems–Technical "Architectures"

However useful the "agreed to" IAP definition of architecture might be, it is not used any further within the CISA document for the subsequent definitions of the terms "operational architecture," "systems architecture," and "technical architecture." We fail to see how these terms, as defined, relate to the previous definition of architecture.

[2] Walter J. Ellis, Richard F. Hilliard, Peter T. Poon, David Rayford, Thomas F. Saunders, Basil Sherlund and Ronald L. Wade. "Toward a recommended practice for architectural description." In Proceedings of 2nd IEEE International Conference on Engineering of Complex Computer Systems, Montreal, Quebec, Canada, October 21-25, 1996.

We recognize it is important to model the operational, systems, and technical **aspects** of our C4ISR systems. However, it is contrary to conventional usage—and hence confusing—to call each of these aspects “architectures.” Architects do not to refer to a building’s “heating and ventilating architecture,” its “electrical architecture,” its “building code architecture” or its “occupant architecture.”

Recommendation: Reserve the term “architecture” for the whole system (with the above sense of “system”). Refer to the operational, systems, and technical **aspects** of a system of interest instead of operational, systems, and technical architectures.

Note: One might be tempted to call these aspects “views”—but this is not consistent with current practice in the commercial world and the standards communities. (See below.)

Recommendation: Define the information content of each of the aspects. E.g., figure A–1 is a useful start in modeling this information content of the Operational Aspect. Comparable models for the Systems and Technical Aspects should be provided.

Architectures Distinguished from Architectural Descriptions

Another problem with the definitions as presented is the confusion between architectures and their descriptions. In the traditional field of architecture, “the architecture” of a building is distinct from the drawings, blueprints, and models which represent it. The present draft makes this confusion between the definition of “architecture” and the operational and systems architectures which are called “descriptions.” This should be clarified for DOD-wide use. E.g., this confusion permeates the CISA document’s “Guiding Principles” (3–2, annex B), some of which are principles for architectures and some of which are principles for descriptions.

Recommendation: The definition of “architecture” refer to the concept of a system’s total architecture, its high-level structure, and related principles, as distinguished from any “architectural descriptions” of that [conceptual] architecture which are embodiments, portrayals, or engineering representations of that architecture. This change would clarify the apparent intended subject of the CISA document which is architectural description via “architecture products”—not the architectures themselves.

Time-phased Nature of Architectural Descriptions

Figure ES–1 recognizes that changes to technology (time-phased technical guidance) must be taken into account. Elsewhere, it is acknowledged that organizational relations change over time (4–3). In fact, virtually all aspects of architecture modeling needs to recognize time-varying structures; from mission and doctrine, which impact the operational and evolution of the systems aspects, through technology migration, etc. The present architecture products and process in the CISA document do not take this into account.

A different sense of "time-phased" also concerns us in the CISA document. It is emphasized at several points in the document that "operational architectures drive associated system architectures" (3–6), that "technical architectures are intended to transition the logical operational architecture to the physical systems architecture..." (3–8) and that "[t]he products are presented starting from the operational architecture products *upon which the systems products are based* and ending with the technical architecture products" (3–16). If the order implies some precedence in construction of architectures, this cannot be maintained. Operational aspects are not prior to systems or technical ones. In fact, this mind set, which mirrors the traditional top-down or waterfall development model, led to the stove-piped custom systems we are now trying to avoid. Attention to architecture is exactly the antidote, by giving equal priority to all three aspects within system development.

The Operational [Architecture] Aspect

It is stated that "...operational architectures present the functional or logical requirements for C4ISR support to the warfighter, ..." (ES-3). The intended or desired level of description is not clear (3–5): is the Operational Aspect intended to define the problem, or partially specify its solution? What activities, operational elements, **...** are "given" vs. what entities are part of the architecture? If the Operational Aspect "should not be system-dependent"— what is it bounded by? Why isn't activity based upon organizational realities, e.g. on organizational models and force structure? Either these are given or to be defined. As defined, how is this different from a requirements specification, a domain or enterprise model? Without some delineation of these it is difficult to interpret the "tenets" and characteristics of the Operational Aspect (3–5). E.g., are IERs really intended as requirements? If so, originating with whom? And requirements upon whom?

Recommendation: Distinguish requirements from architecture; requirements are a *critical input* to the architecture, and must be distinguished from it.

Limit the scope of the term "architecture."

The Systems [Architecture] Aspect

In the definition, "including graphics" confuses form with content (3–6). Similar questions arise here as those for the Operational Aspect. Which "key nodes, circuits, networks, warfighting platforms" are given, part of the problem space or environment, and which are part of the solution. These should be distinguished by the constructs in the CISA Document. Similarly, "performance parameters" should be distinguished as either required or proposed—part of the characteristics of the solution. Implication of the expanded definition is that the Systems Aspect is "constructed to satisfy" i.e., follows from the Operational Aspect.

Other Terminology Not Defined

Terms like architecture characteristics, operational elements, and capabilities are used frequently but not singly defined.

Architectural Views

In our experience, the architecture constructs defined by CISA (Operational, Systems, and Technical Aspects), are much too coarse-grained for planning, management or engineering purposes. Most architecture efforts do not fit cleanly into the categories of operational, systems, or technical aspects. Even much of the "architecture information" in the CISA document (3–10) itself does not fit cleanly into a single aspect.

What is needed is a more fine-grained construct. Such a modular, reusable construct is in line with the CISA document's principle 4 (B–2). The notion of a "view" as it appears in contemporary practice, appears to satisfy this.[3]

For example, in the Army's Sustaining Base Information Services, a number of engineering views of the architecture were adopted. These worked very well for communicating with the client, users, and system engineers. However, they do not map cleanly to the three "architecture types" of the operational, systems and technical aspects (see table). Notice no view (row) falls into a single aspect (column).

There is nothing special about these particular views—other architectures might draw from a large "palette" of views which can be found in the industrial and research literature, such as:

- behavioral
- dynamic
- functional
- information
- logical
- physical

[3] See e.g., Philippe B. Kruchten. The 4+1 view model of architecture. IEEE Software, 28(11):42-50, November 1995, David E. Emery, Richard F. Hilliard, and Timothy B. Rice. Experiences applying a practical architectural method. In Alfred Strohmeier, editor, Reliable Software Technologies - Ada-Europe '96, number 1088 in Lecture Notes in Computer Science. Springer, 1996.

	Operational Aspect	**Systems Aspect**	**Technical Aspect**
Application View	Modularization to support split-base	System layering	Inter-layer APIs
Data View	Data element ownership	Legacy v. Standardize packaging	Data modeling and interface standards
Distribution View	Physical and computational distribution	Distribution design rules	RPC APIs
Security View	Physical and operational security policy	Secure IS, subjects, objects and access privilege strategies	APIs
Developer-Maintainer View	Installation, release	Shared APIs, division of developer responsibilities	Tools and methods

Table 1. *Relationship of Engineering Views to the three "Aspects"*

Views are more expressive than the proposed set of CISA architectural products. In our experience, there are a number of views that do not fit readily into the proposed set of architectural products and there are good reasons for differing views: each view type highlights a certain set of concerns not readily modeled by another view type; thus, different views support different types of architectural analysis.

Recommendation: CISA define, evolve and maintain a palette of views, identify their appropriateness for various uses in architectural descriptions, methods, and tools for analyzing those views.

Linkages among Architecture Types

While view-level analysis is invaluable, there is also much insight to be gained from between analyses. "The relationship among the architecture products is described to facilitate traceability of C4ISR solutions back to the operational warfighting and warfighter support requirements that they are aimed at satisfying." (ES-5) The linkages shown in figure ES-4 are insufficient to provide engineering traceability between products. Linkage among architecture types (views, aspects, in the sense above) requires strong traceability. Support for this is not evident in the present document.

Recommendation: More rigorous approach to linkages be developed against a full information model of required architectural information.

Community Practices

It is ironic that in the late 1990s DOD is devising its own “architecture framework” when there is substantial commercial, industrial, research, and standards activity in this area. There are a number of existing or emerging conceptual frameworks for thinking about and describing system architectures. Given the current orientation of DOD toward utilizing or promoting the growth of international and industry standards, we wonder why DOD is undertaking this effort in architecture, without investigation of these “commercial” frameworks. Some scholarship on the part of CISA in this regard, would benefit the present document. We are aware of the following efforts:

- IEEE Architecture Working Group
- Open Distributed Processing
- X/Open
- Object Management Group

Architecture Products

There are two issues with products: what is to be captured (content) and how (form). As proposed, the ontology of types of architectural information (figure 3–12) is fairly narrow: data, activities, users, nodes, etc. It is difficult to capture the full range of architecture information within this ontology. Is it intended that the proposed architecture products completely “cover” the space of possible architectural information, or is this an evolving set? In our experience, there are a number of possible views which we cannot correlate with these products. Also in our experience, several of these proposed products are in reality *inputs* to the architect—rather than outputs. For general-purpose use in DOD we would expect to see a much richer set of constructs, (behaviors, goals, processes, security objects and policies, APIs, etc.) and have those constructs representable within the proposed architecture products.

Of the proposed products listed, some appear to be primary, while others appear to be derived or secondary. For example, among the products for the Operational Aspect (4–1), High-level OCD, Command Relationship Charts, Activity Models appear to be primary, while IERs, Required Capabilities, and Node Connectivity Diagrams seem to be derived presentation from other more primary information.

Recommendation: Products be defined such that users of this document will know for each product: what is to be shown by a product, what are the allowable constituents, permissible relations between constituents, and what kinds of analyses may be performed on the resulting product.

Operational Aspect Products

The proposed products are insufficient for operational modeling needs. Key aspects of operation are: agents' goals, intentions, plans, actions, etc. These are very hard to model with activity modeling. The template for the Operational Concept Diagram (4–2) is so vague that one wonders why anyone would need to tailor it. However, due to its static nature, the Command Relationships Chart formalism (figure 4–2) as described, is insufficient to describe the dynamics acknowledged within the CISA document itself, "command and control relationships may differ under different circumstances, such as for various phases of warfare; these command structures may mean that activities are performed differently by different units. Different coordination relationships may mean that connectivity requirements are changed." (4–3)

The activity model examples (figures 4–3 and 4-4) are illegal in IDEF$_0$—all activity boxes should have at least one control arrow. Example figure 4–3 would be more convincing if real activities were identified.

The CISA document should describe how activities are "keyed" to warfighter activities from the UJTL.

"Activities are assigned to nodes"—how does this relate/overlap with the IDEF mechanism construct? This is how such "assignments" are typically modeled in IDEF.

Rules, such as those implied on (4–4) should be clearly marked; e.g., we detect the following:

> Rule 1: "For any given operational [aspect], the activity hierarchy should start using one of the branches of the UJTL as the top of the tree."
>
> Rule 2: "The top-level task should be broken down into lower level tasks until the tasks at the lowest level can be clearly identified with a particular operational element or node responsible for accomplishing the task."
>
> Rule 3: The activity tree should be refined only to the level necessary to meet the needs of the particular operational [aspect] being developed.

An example of the suggested approach using activity model templates should be provided—since it is different from the usual practices of multiple models.[4]

We disagree that there is "a logical progression" from IDEF$_0$ to IDEF$_1$—both are principal elements of a model; IDEF$_0$ is not prior to IDEF$_1$. There is some confusion here also about IDEF$_1$ for designing data. That is not the role of the architect. However, IDEF$_1$ is quite useful also at the architectural level of discourse, but for a different purpose and scope. (4–5)

[4] See for example, C.A. Irvine and D.L. Morris, Requirements and design models: a case study. *Proceedings of the 4th International Conference on Software Engineering*, 1979.

Overlays to activity models using “dashed arrows” (figure 4–4) are not needed for the identification of “nodes that perform an activity”—this is already supported within standard IDEF_0 by mechanism arrows.

The example of costing with IDEF (figure 4–4) is somewhat misleading in its simplicity due to the well-known problem of *quantification* in IDEF. Imagine that the three activities to be costed are: A1, Make Plan; A2, Compare Alternatives; A3, Implement Plan. The “cost” of A1 might be dependent on a number of parameters of its inputs, the approach to carrying out A1, and the complexity of the output (i.e., the resulting plan). The “cost” of A2, too, will vary with the number of alternatives considered—which would not appear anywhere in the activity model—as well as other considerations. Furthermore, for each activity, one needs to know what processing resources (i.e., mechanisms) are available to perform that activity. The point being cost models based on activity models will have to take into consideration: not just factors for each activity box, but information about arrows, quantification laws for each box and arrow, and other information not available within the model itself.

Recommendation: The inclusion of a more representative example of costing from an activity model.

It would be useful for users if the entries in the Information Exchange Matrix (figure 4–5) were correlated with IDEF constructs, i.e.,

- Supported Operational Task—Activity
- Information Consuming Node—Mechanism
- Information Producing Node—Mechanism
- Element of Warfighter Information—Data Arrow

Isn’t the matrix simply a derivative of the full data dictionary associated with the architecture model? It is pointed out that “any attempt to present the entirety of the required capabilities in Node Connectivity Diagram form will likely detract from its readability and defeat its purpose of facilitating comprehension...” (4–9). Therefore, what guidelines or principles should be used or what constructs should be used instead?

Systems Aspect Products

The claim that “[t]he systems component products are largely derived from the operational component products” (4–10) is indicative of an approach that preserves the top-down approach to systems definition and requirements satisfaction which led to our stove-piped systems predicament. Indeed, this is the whole reason why architecture is a concern: to address systems decisions in a fashion which is not driven by operational (e.g., user) concerns, which yields single function systems.

A problem with overlays, as described, is that they are not particularly analyzable. The community has not found effective ways of analyzing models attributed with other information because such models usually lack the logic of combination. A more tractable approach is to use individual views to separate concerns, and associate with each view type one or more methods of analysis.

It is unclear whether items like System Performance Parameters (Matrix) are “inputs” or “outputs.” This is a pervasive problem throughout the framework: for a given piece of information is it an input to the architecture (i.e., a requirement on the system), or an output (i.e., a characteristic of the system resulting from an architectural choice)? These discriminations should be reflected in architectural summary or engineering information.

A truly useful “evolution view” will require a finer level of detail than the proposed System Evolution Diagram (4–13). Evolution has to be managed at the level of individual components, their interfaces, the standards which apply to those interfaces and the underlying technologies. The typical “herringbone” charts are misleading because they imply that there is a convergence at the end into a single steady-state. This has never been true; and it is not likely to be in the future.

Technical Aspect Products

We fail to see how technical architecture products which are intended to provide “technical guidance that governs system implementation and operation” can be separated from the Systems and Operational aspects.

Core Information Products

This section is very confused apparently because (1) there is not a clear distinction made between data models of “architecture information” and “warfighter information”; and (2) the scope of the “core data model” is not defined.

Recommendation: Clarify scope of “core information” limiting it to architectural information. “Warfighter information” is totally out of the scope of architecture description; eliminate references to it here, it is only confusing. Develop an information model (IDEF$_{1X}$) of Architecture Information (annex A is a start); with the goal of explaining the information content of each architectural product, and use that information model to identify those critical architectural areas which are not covered by the proposed architectural products. Validate this Architecture Information model by looking at other approaches to capturing and modeling architectural information. Other “metamodels” to look at include: IEEE AWG work, ODP, noted above.

The Architecture Information model can then also be used to define the “interrelationships among product types” (B–3) which is a key characteristic of their development and use.

Recommendation: Strong form of traceability be used to govern and document those relationships.

Figure 4-14 should use IDEF_{1X} instead of old E-R syntax.

The relationship between interoperability and architecture is much more complex than the preliminary discussion in section 4.6. Standard descriptions are probably more misleading than helpful in this regard. How systems are constructed has little to do with interoperability potential—more to the point are whether they share certain protocols with respect to messaging, communication, processing, etc.

Guiding Principles

The principles in annex B confuse architectures with architectural descriptions, again. Insofar as they may be applied to descriptions, i.e., the architecture products, principle 2 seems to weigh against the current proliferation of highly redundant products as noted above. N2 charts, and matrices are neither modular nor reusable (principle 4). Principle 5 is counter to one of the few empirical principles of software engineering, Boehm’s Law: it’s worth spending effort up front in the development because the sooner you find defects, the cheaper they are to correct!

The Role of Products

This material should be improved and moved earlier in the document, to justify the various architecture products being proposed. As it stands it is not clear what the intended role of architecture products is relative to others. To the extent that this is addressed currently in section B–2, it appears “architecture development” is a side activity to real work rather than integrated into the life cycle.

We take exception to the notion that the architecture “products form a continuum from the Operational Concept Diagram through the various products associated with operational, systems, and technical [aspects]” (B–2).

Five-Step Procedure

The proposed “architecture development process” (3–17, B–4.1) is lacking both iterative and verification criteria. Rarely in engineering does one arrive at the complete solution the first time; therefore most modeling processes are iterative: steps are revisited to improve the eventual product. It is also useful to have exit criteria or a verification step to determine whether the product is sufficient to satisfy the initial conditions which led to the modeling activity.

The process is so general one wonders why it needs to consider tailoring at all (B–5).

Recommendation: Look at modeling processes with these characteristics to refine the proposed process.[5]

If you have any questions, please contact Rich Hilliard at Bedford extension 2724 or Tim Rice at 617-377-6839.

R. F. Hilliard
Lead Architect
Information Systems Architecture and Internet Technology

T. B. Rice
Group Leader
Information Systems Architecture and Internet Technology

RFH/TBR:cw

[5] See for example, above references on architecture, initial work on IDEF by Ross. etc.

Distribution:

H. A. Bayard
T. K. Backman
G. M. Friedman
E. J. Green
E. K. Kriegel
G. R. LaCroix
W. F. Lynch
J. W. Moore
T. F. Saunders
S. C. Schwarm
J. A. Surer
E. N. Skoog
J. A. Wilson