from The Rational Edge: The first in a two-part series, this article presents an overview of the Department of Defense (DoD) Architecture Framework (DoDAF) and describes its Operational View (OV) products. The author describes the relationships among the different views, and how the views can be leveraged both individually and collectively to add value in the modeling and design of complex systems.

The Department of Defense (DoD) Architecture Framework (DoDAF) defines a common approach for DoD systems architecture description, presentation, and integration for both war-fighting and business operations and processes. The intent of the DoDAF is to ensure that architecture descriptions can be compared and related to one another across organizational boundaries, including different military commands.¹

The DoDAF addresses this need by providing guidance on how to describe a complex systems architecture so that it can be evaluated and understood alongside other architecture descriptions developed according to this same guidance. Operational decision-makers can then use DoDAF-compliant reports to compare the architectures of alternative systems and manage the evolution of existing systems.

A compliancy report consists of model views that describe a systems architecture in sufficient detail to enable DoD management and the Congressional Budget Office (CBO) to evaluate the system for procurement purposes. Companies doing business with the DoD are tasked to comply with some or all of the DoDAF as they propose their systems.
In this paper, I discuss an approach to modeling complex systems architectures in conjunction with constructing DoDAF-compliant views. While exploring the DoDAF products, I explain how you can leverage the architectural model of the operational enterprise, Unified Modeling Language (UML) notation, and IBM Rational tools to help you generate complete, correct, and consistent DoDAF views in the context of a well-formed systems architecture model.

Leveraging the IBM Software Development Platform for DoDAF compliance

Architecting complex systems demands an extraordinary capacity for understanding and managing complex relationships. A thorough understanding of the enterprise architecture\(^2\) is crucial to effective design, implementation, deployment, and maintenance of evolving systems. A complete, consistent model of that architecture is the key to that understanding -- and is essential to mitigating risk and managing the system's complexity. DoDAF content provides us with a "window" into the architecture that we can leverage as we incrementally define the system.

DoDAF-compliant reports have typically been created to support the search for sponsorship and funding of major mission-oriented systems. However, systems engineering teams can realize so much more value from that investment by describing a systems architecture early in the system lifecycle. The sooner you can recognize integration challenges and operational dependencies, for example, the more effectively you can manage key decisions.

IBM Rational offers comprehensive support for DoDAF compliance in the form of integrated products, a proven systems engineering process (the Rational Unified Process® for Systems Engineering, or RUP-SE), and a robust enablement capability designed to facilitate discovery, description, implementation, and evolution of the kinds of complex enterprise architectures associated with DoD operational missions.

IBM Rational tools that explicitly support adherence to the DoDAF specification build upon the IBM Rational Eclipse-based modeling solutions, including IBM Rational Software Architect®, IBM Rational Software Modeler®, and IBM Rational Systems Developer™. Integrations with IBM Rational RequisitePro® for requirements management, IBM Rational ClearCase® for configuration management, IBM Rational ClearQuest® for change management, and other IBM Rational products empower the entire systems development team. Extended capabilities and plug-ins provided by Ready for Rational Partners further enhance capability with Systems Modeling Language (SysML) modeling and state-machine-based executable models.

An optimal approach to DoDAF compliance does not require divergent effort from the primary objective of developing the system. The IBM Rational approach incorporates DoDAF product generation into the overall architecture modeling effort, allowing the DoDAF views to represent an evolving enterprise architecture that is consistent with and traceable to the systems necessary to implement that architecture.

As with any complex activity, learning to create and maintain an enterprise architecture using DoDAF requires the skilled application of systems engineering principles, along with DoDAF-specific knowledge. IBM Rational is ideally positioned to offer enablement services to optimize
your efforts. The remainder of this article introduces you to the DoDAF and illustrates how you can address DoDAF compliance requirements in the context of describing an enterprise architecture.

**Key DoDAF elements**

The DoDAF focuses on modeling the relationships between significant architectural elements of the operational enterprise. The core elements of a DoDAF-compliant model are nodes, needlines, services, and information exchanges. Collectively, these entities describe the structure and allocation of significant activity in the operational enterprise.

- **Nodes -- systems, actors, and workers.** The principle element of the DoDAF is the node, which represents a logical or physical entity (a worker, system, or subsystem), operating within or outside the enterprise, whose role is to interact in some manner with one or more elements of the enterprise. Nodes are the foundation for the architecture and design of the system-of-systems that comprise the operational enterprise. The architecture will tend to focus more on the relationships between the nodes, while the design deals more with the structure and behavior of the individual nodes. Accordingly, the DoDAF’s primary objective -- and the benefit of architectural modeling of the operational enterprise -- is a characterization of the manner in which nodes cooperate to fulfill the mission.

In DoDAF, we deal with three types of nodes: operational nodes, which are described in the Operational View (OV) and reflect combinations of actors, workers, and systems; systems, which are logical elements that implement the behavior of the operational nodes; and system nodes, which represent physical elements or localities that host logical systems or subsystems.

- **Needlines -- relationships and dependencies.** In the DoDAF, relationships between collaborating operational nodes are represented as needlines. Each needline represents the requirement for one node to provide one or more operationally necessary services and associated information to another node. Needlines are abstractions in that they may represent a single service/information exchange or a group of services/information exchanges. In either case, the needline illustrates that one operational node depends on another for service(s) or information and specifies the direction in which the service(s) or information flows.

- **Services -- significant operational functionality.** Services represent one or more operationally significant functions that one node renders to another node. Each service, implicitly or explicitly, also represents the transfer of information between nodes and may be characterized as a message or operation.

- **Information exchanges -- characteristics of information transferred.** Information exchange is associated with a set of functional and non-functional requirements characterizing the constraints under which information is captured, transferred, or used.

**Best practices for complex systems development**

By seamlessly incorporating production of the required DoDAF content into the overall process of elaborating the enterprise architecture (EA) and its associated requirements, you effectively negate the perceived burden of DoDAF compliance in the context of complex systems development. Moreover, you can leverage the invaluable engineering information captured in the DoDAF products to reduce risk to cost and schedule during systems development.
The IBM Rational approach to detailing the structure and behavior of an architecture is based on proven principles. These "six principles of systems engineering" are pragmatic guidelines that provide the foundation for a well-managed system evolution. They serve to highlight the critical areas on which organizations developing complex systems should put their focus. They also enable organizations to assess challenges and diagnose their causes.

- **Decompose systems, not requirements.** Develop each abstraction level before proceeding to the next lower level. Elaborate the use cases fully and captured behavior explicitly. Be sure to consider not only the logical architecture, but also the physical/locality-oriented aspects of the architecture. Discover and document the relationships between the logical and physical architectures for each level of abstraction addressed. Iterate to the next lower level of abstraction until the architecture is sufficiently captured to meet the needs of the development organization.

- **Enable both separation and integration of concerns.** Examine both black-box and white-box views for each level of abstraction addressed. Strive for balance between perspectives to avoid overcooking in either direction. Too much separation results in functional decomposition and its associated integration issues; too much emphasis on integration and you risk missing important functional issues.

- **Systems and components collaborate, so should development teams.** Developers of components and systems/subsystems that need to collaborate depend on thorough knowledge of dependencies. Without developer collaboration, you increase the risk that integrations will fail.

- **Specifications flow up and down the architecture.** You should understand the requirements at each level of abstraction and use them to derive the capabilities of the elements that collaborate at that level of abstraction.

- **Base the lifecycle on removing risk and adding value.** Minimize the obstacles to success when the resources are available to do so.

- **Development organization should reflect product architecture.** Optimal application of development team skills calls for shifting responsibilities from one role to another throughout iterations. Organizing teams with multiple complementary skills provides for more management flexibility and increases the overall capabilities of individuals to the organization.

Risk management drives the overall process of developing an enterprise architecture. Rigorous application of an iterative process and the use of a standard notation like the Unified Modeling Language (UML) results in a comprehensive visual representation of multiple perspectives of system structure and behavior at successively lower levels of abstraction. Recursively applying the principles to the level of subsystems definition and internal design results in a complete, consistent engineering model of the architecture. This, in turn, provides a foundation for design, implementation, deployment, management, and controlled evolution of a complex system.

**Organization of a DoDAF-compliant model**

The DoDAF structures architectural information around views. The All Views (AV) products are intended to provide an overall perspective of the subject system within the context of the operational enterprise, and address overarching concerns like Concept of Operation (CONOPS).
and critical mission objectives and strategies, as well as an integrated dictionary of architecturally significant terms.

The **Operational View (OV)** focuses on the externally visible structure and behavior of the subject system. This view describes operational nodes and their relationships, and identifies dependencies reflecting mission requirements, thus providing an overall context for enterprise definition and evolution.

Realization of internal structure and behavior is the focus of the **Systems View (SV)**, which incorporates a rigorous allocation of functional and non-functional requirements (from the Operational View) to both logical and physical system elements and interfaces.

Standards constraining the operational architecture of the enterprise are reflected in the **Technical Standards View (TV)**, and address both current and future states of the system(s).

The OV is the focus of this month's article. Next month, in Part 2, I'll describe the SV and TV. Figure 1 illustrates the relationships among the various DoDAF views.

![Figure 1: Relationships among DoDAF views](image)

**How the DoDAF views relate**

Consistency within and among DoDAF views is critical. Optimal derivation of DoDAF views necessitates consistent modeling at multiple levels of abstraction (i.e., systems decomposition). As we drill down in an architectural model, recursively applying a rigorous systems architecture discovery process to successive levels of abstraction of an enterprise, we learn more about an element and may employ alternative means to represent its characteristics. For example, we may initially represent a complex system that satisfies the needs of its users by way of a use-case or context diagram. As we learn more of the supporting activities (system white-box behavior), we may add class, activity, and/or sequence diagrams to reflect the additional detail. **Nodes**, portrayed as actors in one diagram, may be more appropriately represented as classes or objects in other diagrams. **Services** may be implemented or realized by collections of class operations making up a subsystem.
In determining how best to model each of the core DoDAF elements, you must first understand the essential semantics underlying that element, along with any applicable constraints, and then apply the appropriate notation given the context of the overall engineering effort. This context includes risk, complexity, tools, notation, and objective(s) for the modeling effort.

The overall process of producing DoDAF views is both iterative and incremental. As more breadth and depth of architectural information is captured, evolution of the All Views (AV-1 and AV-2) progresses. Using the AV-1 as a foundation, the architectural interactions of the operational enterprise and the subject system are examined, resulting in discovery of the high-level interactions between the system and the operational nodes. Full characterization of these high-level relationships is the focus of the Operational View.

Only after you fully understand the external systems behavior (at the enterprise level) do you proceed with the elaboration of the Systems View. This is where we begin to design and organize the internal behavior and subsystems interactions that provide the foundation for full-scale development. At this point, we also reconcile multiple viewpoints that allow us to deal with the necessary physical and logical realization of operational behavior through joint realization practices and use-case flowdown.

**All Views products**

The following table briefly describes the All Views products and the order in which you would typically create them.

<table>
<thead>
<tr>
<th>Product</th>
<th>Title</th>
<th>Description</th>
<th>Representation</th>
<th>Creation Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>AV-1</td>
<td>Overview and Summary Information</td>
<td>Textual document describing scope, purpose, intended users, and operational environment for the subject system. Provides an overall understanding of the nature of the enterprise and how it interacts with the subject system. Supports the strategic vision for system usage.</td>
<td>Model-referenced text document.</td>
<td>1</td>
</tr>
<tr>
<td>AV-2</td>
<td>Integrated Dictionary</td>
<td>Definitions for all terms used to describe the architecture. Provides a set of standard reference terms to maintain consistency of meaning to all consumers of the architecture.</td>
<td>Model resident, repository-based text; exportable to XML.</td>
<td>Ongoing</td>
</tr>
</tbody>
</table>

The DoDAF All Views (AV) products provide a summary of the environment in which the subject systems are to be developed, deployed, and managed during their evolution. The summary describes mission objectives, strategies, operational concepts, and the general context for operations, along with relevant terminology.

**AV-1: Overview and summary information**
The AV-1 is a textual summary of the operational environment and the mission capabilities to be exercised in the context of the evolving systems. Its focus is on the subject system or enterprise you need to establish in that context. Relevant Concepts of Operations (CONOPS) and strategies are presented at a level of abstraction appropriate to executive leadership in order to facilitate decision-making. The content of the AV-1 represents the guidance or vision that has captured essential business drivers and the need for the subject system under development.

The acquirer or a development organization may prepare the AV-1; although, as with all DoDAF view products, substantial interaction with Subject Matter Experts (SMEs) who have extensive operational experience will be necessary. In the approach described here, you can produce the AV-1 document using a word processor and associate reference links with the model containing the visual DoDAF products.

**AV-2: Integrated dictionary**

The AV-2 represents a simple but essential concept to systems and software development. The need for consistency and clarity of meaning is substantially met through establishing a single, centralized glossary of architecturally relevant definitions and potentially ambiguous terms.

The IBM Rational approach incorporates continuous evolution of this integrated dictionary within the model repository managed via IBM Rational's Eclipse-based modeling tools, including IBM Rational Systems Developer, IBM Rational Software Architect, and IBM Rational Software Modeler. As you create model elements, you incorporate them into the engineering information within IBM Rational's Eclipse-based modeling tools, from which you can extract an AV-2 at any time. All graphical model elements associated with DoDAF stereotypes are automatically captured in this manner. You will need to add textual references manually or, alternately, access them via some other tool, such as IBM Rational RequisitePro.

**Operational View products**

The DoDAF Operational View is composed of various products that provide multiple perspectives on the external structure and behavior of the subject system in the overall enterprise context. Within these views, we characterize the interactions between the system and its actors, the mission objectives required of the system, and the necessary dependencies and interactions for achieving those objectives.

The focus of the OV is on those requirements and capabilities that affect the mission. The Systems View (SV) describes how the OV is realized. The following table briefly describes the OV products and suggests an order in which to create them.
<table>
<thead>
<tr>
<th>Product</th>
<th>Title</th>
<th>Description</th>
<th>Representation</th>
<th>Creation Order</th>
</tr>
</thead>
<tbody>
<tr>
<td>OV-1</td>
<td>High-Level Operational Concept Graphic</td>
<td>Graphic abstraction of the operational concepts supporting the mission of the enterprise.</td>
<td>High-level abstract graphic, Enterprise Context Diagram, Enterprise Use-Case Diagram</td>
<td>1*</td>
</tr>
<tr>
<td>OV-2</td>
<td>Operational Node Connectivity Description</td>
<td>Operational nodes, activities, connectivity, and information flow.</td>
<td>Enterprise Context Diagram with Needlines and IO Entities</td>
<td>4**</td>
</tr>
<tr>
<td>OV-3</td>
<td>Operational Information Exchange Matrix</td>
<td>Information exchanged between nodes and the attributes of the information.</td>
<td>Model resident text matrix; exportable to XML</td>
<td>4**</td>
</tr>
<tr>
<td>OV-4</td>
<td>Command Relationships Chart</td>
<td>Command, control, and coordination relationships between operational organizations.</td>
<td>Freeform diagram with organizational elements</td>
<td>2**</td>
</tr>
<tr>
<td>OV-5</td>
<td>Activity Model</td>
<td>Activities, relationships between activities, I/Os, constraints, and mechanisms that perform activities.</td>
<td>Activity Diagram(s) for each Enterprise Use Case</td>
<td>2**</td>
</tr>
<tr>
<td>OV-6a</td>
<td>Operational Rules Model</td>
<td>Identification of business rules and process constraints that affect the operational activities.</td>
<td>Model constraints (OCL/SysML), model-referenced functional, and non-functional requirements</td>
<td>2**</td>
</tr>
<tr>
<td>OV-6b</td>
<td>Operational State Transition Description</td>
<td>Identification of relationships between events and operational sequences.</td>
<td>State Transition Diagrams</td>
<td>4**</td>
</tr>
<tr>
<td>OV-6c</td>
<td>Operational Event/Trace Description</td>
<td>Identification of externally visible operational sequences and actions that trace to scenarios or critical activities.</td>
<td>Sequence Diagrams</td>
<td>3</td>
</tr>
<tr>
<td>OV-7</td>
<td>Logical Data Model</td>
<td>Structural relationships of data supporting operational exchange of information.</td>
<td>Class Diagram indicating IO Entities and their relationships</td>
<td>5</td>
</tr>
</tbody>
</table>

* OV-1 content is started first, but the OV-1 graphic cannot be completed until OV-2 is complete.
** These products are not serially dependent and can be created in either order; or they may be co-dependent and developed jointly.
*** State Transition Diagrams are optionally used to model critical real-time responses to complex events requiring special treatment.

The order in which products are likely to be generated is shown in the activity diagram in Figure 2. The proposed order is based on an architectural discovery process founded on the six principles of systems engineering touched on above. By following this order, you can generate DoDAF-compliant products efficiently, without detracting from the primary task of defining the enterprise architecture.
Figure 2: Suggested order for generating DoDAF AV and OV products

**OV-1: High-level operational concept graphic**

The OV-1 clearly and concisely communicates the scope of the subject system within the context of the operational enterprise. The graphical depiction of the OV-1 is typically an artist-rendered product reflecting content derived from multiple sources. The primary information sources for the OV-1 are the AV-1 Overview and Summary document, an Operational Context Diagram, and an Enterprise Use-Case Diagram. We construct the Enterprise Use-Case Diagram starting with the subject system and identify any external systems and organizational entities that interact with that system. We characterize these interacting elements as actors or roles. Use cases are then added to the diagram for each operational goal or objective attributed to actors. UML «communicates» stereotyped associations are added where appropriate.

Many actors/roles collaborate within organizational elements in order to meet mission needs. The aggregation of actors/roles to organizational elements results in identification of operational nodes, which are captured using a class diagram, designated the Operational Context Diagram. Systems architects and other SMEs then collaborate with the graphic artist in rendering the Operational Context Diagram in an OV-1 graphic (see Figure 3), tailored for an executive-level audience. This diagram provides the foundation for structuring the externally visible architecture of the operational enterprise as it relates to the system under development. The diagram’s content will evolve as further information is captured and additional DoDAF products are generated.
Where multiple actors represent activity within operational nodes, you may need to aggregate the roles associated with those actors. Subsequently, the interactions between actors and the subject system become represented by the collective interactions, or needlines, between the operational nodes (actor aggregation) and that system. The IO Entities associated with those actors transitively become associated with the specified operational node.

**Figure 3: OV-1 high-level graphic**

**OV-2: Operational node connectivity description**

The OV-2 identifies and models critical operational dependencies between operational nodes. The DoDAF defines these dependencies as needlines. There are two primary approaches to determining needlines:

1. Identify the nature of the dependency represented in each «communicates» association in the Enterprise Use-Case Diagram and specify a corresponding needline. Give the needline a directional component so that it is navigable from the consumer (for that relationship) to the supplier of the service or information.
2. Wait until you begin to detail the use-case flows and scenarios and capture them in the OV-6c sequence diagrams (see below). At this point, you can then identify specific object/role interactions, which can be "rolled up" to representative needlines.

The first option is a manual process, since some level of engineering/architectural analysis is required. The second option allows you to leverage some capabilities of IBM Rational's Eclipse-based modeling tools to automatically populate the needlines (and OV-3 Information Exchange Requirements, or IERs) from content in manually produced sequence diagrams. This latter approach has the additional advantage of guaranteeing consistency between OV-2, OV-3, and OV-6c, since they will all be derived from identical model information.
A needline may represent many information exchanges or service dependencies. Accordingly, once you have identified a needline between any two context diagram elements, it is inappropriate to add further needlines that point in the same direction. Figure 4 illustrates needlines for a sample OV-2.

Figure 4: Sample OV-2 with needlines
Click to enlarge

**Note:** UML 2.0 introduces a new classifier, the Collaboration. The semantics associated with the Collaboration offer the potential for you to characterize relationships more robustly. You can specify relationship roles, patterns, templates, and associated parameters. You can also instantiate the information associated with collaborations as collaboration occurrences, further specifying each potential IER. Augmenting the minimal set of DoDAF representations with class and composite structure diagrams (referencing collaborations and collaboration occurrences, respectively) may prove worthwhile. The *UML Language Reference Manual*\(^4\) provides a thorough discussion of these UML elements.

**OV-3: Operational information exchange matrix**

The OV-3 is a matrix of IERs that collectively represent the needlines of the OV-2. The OV-3 can be generated automatically using the IBM Rational Systems Developer design and development tool by referencing the OV-6c content. Each row in the OV-3 matrix represents an IER, which is made up of characteristics of the data transferred between roles and objects in an interaction from the OV-6c sequence diagrams. The matrix identifies a distinct IER for each pair of objects or roles that interact and exchange information. Specific IER characteristics are typically associated with non-functional requirements or design constraints. Each IER’s content represents an instantiation of an OV-6c IO Entity class (see below), where the attributes represent the data characteristics
that the DoDAF requires. Accordingly, each information element in the matrix should trace to the Logical Data Model, OV-7.

The OV-3 emphasizes the logical and operational characteristics of the information exchanged within the architecture. It is not intended to exhaustively capture all the details of information exchange, but rather acts as a mechanism to help you understand the most important aspects of significant exchanges. Figure 5 illustrates the appropriate level of detail. This content would typically trace to supplemental or non-functional requirements.

<table>
<thead>
<tr>
<th>Needline Identifier</th>
<th>IER Identifier</th>
<th>Information Element Description</th>
<th>Producer</th>
<th>Consumer</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Information Element Name and Identifier</td>
<td>Sending Op Node Name and Identifier</td>
<td>Receiving Op Node Name and Identifier</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Content</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Scope</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Accuracy</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Language</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Needline Identifier</th>
<th>IER Identifier</th>
<th>Nature of Transaction</th>
<th>Performance Attributes</th>
<th>Information Assurance</th>
<th>Security</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Mission Scenario UJTL or METL</td>
<td>Periodicity Timeliness</td>
<td>Access Control Availability Confidentiality Dissemination Control Integrity</td>
<td>Accountability Protection (Type Name, Duration, Date) Classification Classification Caveat</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Transaction Type Triggering Event Interoperability Level Required Criticality</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 5: Sample OV-3 Information Exchange Matrix**
[Click to enlarge](image)

**OV-4: Command relationships chart**

The OV-4 models the relationships between organizational entities that affect the operational architecture of the enterprise and its systems. Specific organizational elements are likely candidates for the roles; i.e., instantiations of operational nodes in the interaction diagrams comprising the OV-6c (see below). The OV-4 is represented by a freeform diagram in which organizational elements may be candidates for the instantiation of operational nodes in the OV-6c sequence diagrams.
**Note:** Some implementers have elected to create this diagram but show little, if any, mapping between the OV-4 and the remainder of the DoDAF views.

**OV-5: Roles and responsibilities diagram**

The OV-5 clarifies roles, responsibilities, and order of execution with respect to accomplishing key mission objectives in the context of the operational enterprise. The OV-5 is a graphical presentation of the externally visible behavior of the operational enterprise, represented by flows of activities allocated to component systems. Significant data flow associated with those activities is also provided in order to develop a strong sense of coupling between behavior and supporting data. The OV-5, combined with the textual content of requirements and use-case specifications, significantly enhances the systems engineering team's ability to ensure completeness, clarity, and consistency within an operational perspective of the enterprise architecture and the manner in which it supports the mission.

**OV-6a: Operational rules model**

The OV-6a captures constraints upon the operational processes used to achieve mission results within the context of the operational enterprise and the subject system. Information is captured in text and produced in document form. A template would typically be provided and tailored to the organizational audience. Decision points in the OV-5 activity diagrams should reflect the instantiation of those rules. Some of this content may lend itself to being expressed using SysML or Object Constraint Language (OCL) and used to validate architectural artifacts generated by the modeling tool. However, the primary product for this view is a document.

**OV-6b: Operation state transition description**

When the behavior of one or more key architectural elements is event-driven, modeling with State Diagrams can be especially useful in understanding that behavior. Where this approach is warranted, the OV-6b is produced.

**OV-6c: Operational event/trace description**

The OV-6c describes externally visible behavior; i.e., behavior that is visible from the perspective of the subject system, for each flow and scenarios associated with the enterprise use cases (see above). You capture this information using sequence diagrams that focus on operational nodes (actors) interacting with the subject system via messages. These messages represent requests made to the subject system by associated operational nodes, or requests made by the system to one or more of those nodes. Any information exchanged as part of those requests (e.g., parameters) is represented by an instance of an IO Entity class.

Having identified node-system relationships and associated information content, you can automatically generate content necessary for the OV-2 and the OV-3. Add needlines to the Enterprise Context Diagram (see above) by parsing the interactions and parameters identified between message sender and receiver until you have identified each dependency relationship.
Figure 6 illustrates an OV-6c product.

![Figure 6: OV-6c Operational Event Trace Description](image)

**OV-7 Logical data model**

The OV-7 reflects the structure and flow of key information being used to achieve the functionality expressed in the Enterprise Use Cases. The content of this product should be directly attributable to the IO Entities identified during construction of the OV-6c.

**Next month**

In this month's installment, Part 1, I've presented an overview of the DoDAF and described the Operational View products. In next month's installment, Part 2, I'll go on to explore the Systems View products.

**Notes**


2. The Office of Management and Budget (OMB) Circular A-130 defines an enterprise architecture as "the explicit description and documentation of the current and desired relationships among business and management processes and information technology."

3. For more information on the six principles of systems engineering, see "Hardware/software co-development using a model-driven systems development (MDSD) approach" by Murray Cantor and Gene Roose in the December 2005 edition of *The Rational Edge.*

