How to Build DoDAF 2-Based Architectures with UNICOM® System Architect®

Version 4
(for System Architect® 11.4.3 and later)

Authors
Lou Varveris
Chuck Faris
Mark Gregory
Publication information
April 2016
Information in this publication is subject to change. Changes will be published in new editions.

Copyright notice
System Architect® (the Program and associated materials) is a proprietary product of UNICOM Systems, Inc. – a Division of UNICOM Global. The Programs have been provided pursuant to License Agreement containing restrictions on their use. The programs and associated materials contain valuable trade secrets and proprietary information of UNICOM Systems, Inc. and are protected by United States Federal and non-United States copyright laws. The Programs and associated materials may not be reproduced, copied, changed, stored, disclosed to third parties, and distributed in any form or media (including but not limited to copies on magnetic media) without the express prior written permission of UNICOM Systems, Inc., UNICOM Plaza Suite 310, 15535 San Fernando Mission Blvd., Mission Hills, CA 91345 USA.
© Copyright 1997-2015 All Rights Reserved. UNICOM Systems, Inc. – a Division of UNICOM Global.
No part of this manual may be reproduced or transmitted in any form or by any means, electronic or mechanical, without prior written permission from UNICOM Systems, Inc.

Disclaimer
We cannot guarantee freedom from, or assume any responsibility or liability for technical inaccuracies or typographical errors. The information herein is furnished for informational use only and should not be construed as a commitment by UNICOM Systems, Inc. – a Division of UNICOM Global.

Trademarks
The following are trademarks or registered trademarks of UNICOM Systems, Inc. in the United States or other counties or both: UNICOM®, System Architect®
# Table of Contents

Operational Views .......................................................................................................................... 5
  Overview ......................................................................................................................................... 5
  How to Build OV's in System Architect -- Workflow Summary .................................................. 6
  Overview ......................................................................................................................................... 6
  OV-05a Operational Activity Decomposition diagram ............................................................... 9
  Creating an Activity Hierarchy on the Diagram ............................................................................. 9
  Creating an Activity Hierarchy in the Definitions .......................................................................... 9
  Specifying Hierarchical Numbering ............................................................................................ 11
  OV-05b Operational Activity Model ............................................................................................ 12
  Modeling Flow Between Activities ............................................................................................... 12
  Modeling Decomposition of Activities ....................................................................................... 12
  Creating a Chain of Parent/Child OV-5b's that Reflect the OV-5a ........................................ 13
  Specifying the Performer(s) of an Activity .................................................................................. 14
  Modeling Where and When a Performer Performs an Activity ................................................ 16
  Creating Operational Exchanges ................................................................................................. 18
    Operational Exchange Example ................................................................................................. 18
    Automatically Creating Operational Exchanges ...................................................................... 19
    Manually Creating/Visualizing Operational Exchanges on OV-06c ...................................... 20
    Manually Creating/Visualizing Operational Exchanges on OV-02z ...................................... 21
  Building the OV-2 Viewpoint ...................................................................................................... 26
    Creating the OV-02 ..................................................................................................................... 26
    Automatically Generating Needlines ....................................................................................... 27
  Systems Views .............................................................................................................................. 31
  How to Build SV's in System Architect -- Workflow Summary .................................................. 31
  Overview ......................................................................................................................................... 31
  SV-04a Systems Functionality Decomposition diagram ............................................................. 31
    Creating a System Hierarchy on the Diagram .......................................................................... 32
    Creating a System Hierarchy in the Definitions ....................................................................... 32
    Specifying Hierarchical Numbering ......................................................................................... 33
  SV-04b Systems Functionality Description Diagram ................................................................. 34
    Modeling Flow Between System Functions ............................................................................ 34
    Modeling Decomposition of System Functions ...................................................................... 34
    Creating a Chain of Parent/Child SV-4b's that Reflect the SV-4a ........................................ 35
    Specifying the Systems that Perform a System Function ....................................................... 36
    Modeling Where and When a System Performs a System Function ...................................... 37
  Creating System Exchanges .......................................................................................................... 38
    Automatically Creating System Exchanges ............................................................................. 38
    Optional: Visualize the System Exchanges on an SV-01z ....................................................... 39
    Optional: Manually Creating System Exchanges on SV-01z ................................................... 40
  Building the SV-1 Viewpoint ......................................................................................................... 45
    Automatically Generating System Resource Flows .................................................................. 45
    Modeling Systems of Systems .................................................................................................. 49
    Modeling Systems of Systems on the SV-1 Alternative .......................................................... 49
    Location of System .................................................................................................................... 49
    Location of System Performing Activity ..................................................................................... 49
  Building the SV-2 Viewpoint ......................................................................................................... 51
  Modeling Timing of System Exchange Between Systems .......................................................... 52
    SV-10c Systems Event Trace .................................................................................................... 52
    SV-10c System Functions Event Trace ..................................................................................... 52
    SV-10c Performer-Role Event Trace .......................................................................................... 53
  Measures .......................................................................................................................................... 55
    Measurement Sets, Measurement Types, Measurement Values .............................................. 55
    How to Create and Apply Measurements ................................................................................. 56
Operational Views

Overview

The operational view establishes a logical view of the architecture -- what is going on, who is doing it, how it is getting done, under what conditions -- without specifying the physical details of the systems involved and the actual locations of those systems (that is done in the Systems or Services view).

At the core of the operational viewpoint you are modeling:
- The Activities that are being performed,
- The flow of Resources between Activities. A Resource can be Information, Data, People (Person definition type), Organizations, Systems, or any kind of Performer (including Services, Ports) (see figure below). An Activity transforms a Resource into another state or Resource.
- Who is performing those Activities -- where 'who' is in general terms a Performer that can be for example, a logical organization or role (people perform roles that usually have something to do with their job title, but sometimes they perform a role outside of their job responsibilities),
- Precisely what Performer performing what Activity sends Resources to what other Performer performing what Activity -- this is called an Operational Exchange. The notion of Operational Exchange enables you to specify measures on how two performers exchange information or resources.
- What Performers need to communicate with each other (send Resources to/from each other) based on the Activities that they perform. This is an indirect relationship -- if Performer A performs Activity A, and Activity A sends resources to Activity B, and Performer B performs Activity B, then there is a need for Performer A to communicate with Performer B. This relationship is called a Needline -- it is a logical channel of Resource passage; a grouping of Operational Exchanges.

You use the following diagrammatic views to show this information:
- The OV-05a Operational Activity Decomposition diagram to show the hierarchy of Activities -- visualizing a tree of Activities and their sub-Activities.
- The OV-05b Operational Activity Model to show the flow of Resources between Activities. An ActivityResourceOverlap relationship line is used to show the flow of Resources, and within the relationship, you model the list of Resources that go from one Activity to another.
- Optionally, the 'fit-for-purpose' OV-02z to diagrammatically formulate Operational Exchanges.
- The OV-02 Operational Resource Flow diagram to show Performers and the Needlines between them.
How to Build OV's in System Architect -- Workflow Summary

You may build the operational views in System Architect by adding information and relationships through definitions, or by building/visualizing diagrams, or by adding information and relationships through matrices.

Overview

The overall workflow to create Operational Exchanges and Needlines is described by the picture below, and detailed in the steps below.

1. **Model the hierarchy of Activities** and their sub Activities using the OV-05a Operational Activity Decomposition diagram.

2. **Model Resource flow between Activities** using the OV-05b Operational Activity Model. Draw an ActivityResourceOverlap relationship line between Activities; within it you model the Resources passed between Activities.

3. **Specify what Performer performs each Activity**, using the ActivityPerformedbyPerformer relationship. You can do this on the OV-05b diagram, or, if you wish to keep that diagram clean with only Activities on it, then you can use the OV-02 Alternative diagram.

4. **Create Operational Exchanges** using the fit-for-purpose OV-02z diagram. On this diagram you model ActivityPerformedbyPerformer relationships as nodes, and draw Operational Exchanges between them. Remember that OperationalExchange is the exchange between a particular Performer performing a particular Activity, and another particular Performer performing a particular Activity -- in other words, a relationship between two ActivityPerformedbyPerformers. An Explorer Relationship Report provides automatic guidance on what Operational Exchanges you should create.

5. **On each OperationalExchange, point at the ActivityResourceOverlap line that carries the Resources between the two Activities involved**. An Operational Exchange has a reference to one and only one ActivityResourceOverlap. The same Explorer Relationship Report provides automatic guidance on what ActivityResourceOverlap you should relate to each Operational Exchange.

6. **Model Needlines between Performers on the OV-02 diagram**. An Explorer Relationship Report provides automatic guidance on what Needlines to create and the Operational Exchanges they should carry.

7. **Generate OV-3 report**.
OV-5a hierarchical diagram breaking out into child OV-5b diagrams for each level of hierarchy, with Activities decomposed to lower-level child diagrams with lower-level Activities. An Activity that is not decomposed further is a 'leaf-level' Activity.
Work Flow OV-05b to OV-02

OV-05b Operational Activity Model

OV-05b Assign Performers to Activities

OV-02z Create Operational Exchanges

OV-02 Create Need Lines

ARO = ActivityResourceOverlap
APBP = ActivityPerformedByPerformer
OE = Operational Exchange
NL = Need Line
**OV-05a Operational Activity Decomposition diagram**

You use the OV-05a Operational Activity Decomposition diagram to view the hierarchical relationship between Operational Activities (see figure below). Activities and their sub-Activities are joined via the ActivityPartofActivity relationship.

![Diagram of OV-05a Operational Activity Decomposition](image)

The hierarchy can also reflect the nesting of parent-child OV-5 diagrams of a model.

**Creating an Activity Hierarchy on the Diagram**

With System Architect 11.4.1 and later, you use auto-connect functionality to relate an Activity to a parent Activity. Simply move an Activity that you wish to be a child of another Activity underneath it until a line automatically appears. The relationship is formed.

You may open the definition of the child Activity and in its Part of Activity tab, view the name of the parent Activity, automatically added based on the relationship you formed on the diagram.

![Diagram showing auto-connect functionality](image)

**Creating an Activity Hierarchy in the Definitions**

You can relate an Activity to its parent Activity in its definition, without having any diagram open, and then auto-draw an OV-5a diagram.

1. Open the definition of an Activity and in its Part of Activity tab, you can specify the name of its parent Activity. You can also specify its sub-Activities by populating its Parent of Activities tab. All Activities must exist already -- you drag them in by clicking the Choices button.
2. Create an OV-05a Operational Activity Decomposition diagram and drag-and-drop Activities from the Explorer tree onto the diagram workspace; the diagram draws itself, with appropriate relationships.

**LEAF ACTIVITIES** -- Leaf Level Activities are Activities that are not decomposed further by a Sub Activity. In the diagram pictured above, Evaluate Threats, Clear Area of Prospective Threats, Communicate with Downed Airman, Pickup Survivor, and Get Out of Dodge are 'leaf-level' Activities.
Specifying Hierarchical Numbering

Hierarchical numbering of Activities allows you to easily see the decomposition level of an Activity.

To turn on hierarchical numbering in SA 11.4.2.1 and later, perform the following steps:

1. In an open OV-05a Operational Decomposition diagram, edit its Diagram properties -- right-mouse click on an empty area of the diagram workspace and choose Diagram Properties, or select Edit, Diagram Properties.
2. In the Diagram Properties dialog of the OV-05a Operational Decomposition, make sure the Hierarchical Numbering property is toggled on; the property is located in the Hierarchical Numbering tab of the dialog. You may also set the First Node Number -- this will set the number of the highest-parent symbol on the diagram; if you leave it blank, the default is 0.
3. Select (or draw and then select) an Activity symbol on the diagram, and select Format, Symbol Format, Symbol Style.
4. Make sure the options Auto number and Level numbers in the Allow group are toggled on. Note the value of the Next number property -- this number will automatically be updated as you draw.
5. Select Tools, Preferences, and in the Preferences dialog, make sure the Number property in the Auto group is toggled on.

Note that Prefix code is superceded in DoDAF 2 by Alternate Depiction functionality. In SA 11.4.2 and later, you may right-mouse click on any symbol and choose Display Selected Symbols with Adornment. This provides similar functionality the prefix code, which annotated a symbol as a Process (prefix code P), Activity (prefix code A), etc.
**OV-05b Operational Activity Model**

The OV-05b Operational Activity Model describes the Resources exchanged between Activities in the architecture, and Resources exchanged with Activities that are outside the scope of the model, considered External Activities.

A best practice is to create a high-level 'context' OV-5b diagram, which shows the highest-level Activity in the architecture, and how it interchanges with External Activities. You then decompose that highest-level Activity with a child OV-5b diagram that shows its Sub Activities and how Resources flow between those Activities.

**Modeling Flow Between Activities**

You use an ActivityResourceOverlap line to model flow of Resources between Activities. Within each ActivityResourceOverlap you may specify the actual Resources that flow from one Activity to the other.

To specify one or more Resources flowing between two Activities, do the following:
1. Relate the two Activities by an ActivityResourceOverlap line.
2. On the Resources tab of the ActivityResourceOverlap definition, click the Choices button to view all Resources already created in the encyclopedia. The list shows all definitions of type Resource and sub types: Information, Data, DomainInformation, ServiceDescription Performer, Person Type, Organization Type, System, Service, and Port.
3. Select one or more definitions in the Choices list and drag-and-drop them into the Resources grid.

**Modeling Decomposition of Activities**

You may model decomposition of an Activity through the use of Parent-Child diagramming for navigation and visualization, and also relating the Activity to its children through their definitions.

To graphically show decomposition of an Activity, you may do the following:
1. Right-mouse click on an Activity on an OV-5b diagram, and select **Create Child Diagram** (note: you may also use the menu choice Edit, Child Diagram, Create Child Diagram)
2. Name the new diagram, accept the default diagram type of OV-05b Operational Activity Model (DM2), and click OK, then select Yes to save changes to the previously current diagram.
3. Model Sub Activities on the new diagram.
You will now be able to navigate between parent and child Activities on two different OV-5b diagrams, however the definitions must still be related to one another via a composition relationship, described in the steps below.

**Creating a Chain of Parent/Child OV-5b’s that Reflect the OV-5a**

Creating a parent/child linkage between OV-5b diagrams provides easy navigation through the model, but it does not relate the Activity definitions that are pictured on the diagram in an ActivityPartofActivity relationship. Once you have created a parent/child linkage between two OV-5b diagrams, and modeled sub Activities on the child OV-5b, you must perform the following steps:

1. Open the definition of an Activity on the child diagram and in its Part of Activity tab, click the Choices button and drag-and-drop in the name of the parent Activity.
2. Repeat step 1 for all Activities on the diagram that you wish to model as sub Activities to the parent. Note that this step can be performed independently of the parent/child diagrams -- they just serve as a navigation and logic aid.

Furthermore, once you have related Activities to their parent Activity in the Part of Activity tab of the Activity definition, you may auto-build an OV-5a diagram by simply dragging the Activities onto the diagram -- as discussed earlier. Data centric behavior will autodraw the relationships on the diagram for you. This provides two ways of working:
A. Build the OV-5a first — within which you visually build the Activity/SubActivity relationships, then build the OV-5b’s for each main Activity level
B. Build the OV-5b leveled set of diagrams first, and at each level make sure you relate the sub Activity to its parent Activity in the definition; then autodraw the OV-5a.

**Specifying the Performer(s) of an Activity**

You may specify the Performer that performs an Activity via the ActivityPerformedbyPerformer relationship, available on the following diagram types:

- OV-05b Operational Activity Model (DM2)
- OV-01 High Level Operational Concept (DM2)
- OV-02 Operational Resource Flow Alternative (DM2)
- CV-01 Vision (DM2)

---

**ActivityPerformedbyPerformer** lines drawn between Performers and Activities on an OV-05b. If you wish to keep your OV-05b clean (only Activities), then use the OV-02 Alternative diagram to model Performers performing Activities.

---

### Best Practice: Naming Convention for ActivityPerformedbyPerformer

A best practice for naming an ActivityPerformedbyPerformer relationship is:

**Performer Name - Activity Name**

For example, **Search Helicopter - Find Victim**

Following such a naming standard comes in handy in various aspects of your modeling:

- Looking through a list of relationships already formed in the repository and represent
them on a diagram.

- Modeling ActivityPerformedbyPerformers as vertical lifelines on the OV-6c Operational Exchanges diagram or SV-10c Performer-Role Event Trace diagram.

Note that a list of Activities is not provided in the Performer (and subtypes Person, Organization, etc) definition; nor is a list of Performers (and subtypes) provided in the Activity definition. The relationship is an explicit join -- Performer has a relationship to ActivityPerformedbyPerformer which relates it to Activity. To get a quick view of what Activity is being performed by what Performer globally in your architecture, you can do the following:

1. Select an Activity in the Explorer (browser) or on any diagram.
2. Select View, References.
3. In the References pane that opens, expand Definitions, and expand ActivityPerformedbyPerformer within that.
4. Right mouse click on the name of a particular ActivityPerformedbyPerformer relationship and choose References -- another window will open that contains all referenced items for the selected relationship.

Select View, References to view, for anything that you select, what diagrams it is on, and what definitions it is related to.

5. Expand Definitions and Performer (see picture below).
Modeling Where and When a Performer Performs an Activity

You may specify where and when a Performer performs an Activity in the ActivityPerformedbyPerformer relationship. You may model more than one ActivityPerformedbyPerformer relationship between a Performer and an Activity to specify different conditions, times, and locations.

Specifying Where a Performer Performs an Activity

You may specify *where* a Performer performs an Activity.

1. On an ActivityPerformedbyPerformer definition, select the Where and When tab.
2. On page 1, click on Choices for the Where property and view a list of Location type definitions (and subtypes of Location definition) that you have created in the architecture; select and drag items into the Where grid.

The following definition types are available to describe location:
- Location (DM2), and the following subtypes
o GeoPoliticalExtent (DM2), and the following subtypes:
  ▪ Installation (DM2)
  ▪ GeoFeature (DM2)
  ▪ RegionOfCountry (DM2)
  ▪ Country (DM2)
  ▪ RegionOfWorld (DM2)
  ▪ RealProperty (DM2), and the following subtypes:
    ▪ Facility (DM2)
    ▪ Site (DM2)

Specifying When a Performer Performs an Activity
You may specify when the Performer performs an Activity.
  1. Select the Where and When tab of the ActivityPerformedbyPerformer definition.
  2. On page 2 you may click on Choices for the When property and view a list of the following definition type that you have created in the architecture:
    o Measure (Temporal)).

Specifying Under What Environmental Conditions a Performer Performs an Activity
You may specify under what conditions the Performer performs an Activity. An example of Environmental Condition is in the Desert, or in the Rain, or At Night, etc.
  1. Select the Caveats tab of the ActivityPerformedbyPerformer definition.
  2. On page 1 you may click on Choices for the Conditions property and view a list of the following definition type that you have created in the architecture:
    ▪ Condition (Environmental) (DM2)

Specifying Under What Rules a Performer Performs an Activity
You may specify under what rules the Performer performs an Activity. Rules have subtype Standards -- so an example might be that a Performer rescues a ship under particular Maritime standard.
  1. Select the Caveats tab of the ActivityPerformedbyPerformer definition.
  2. On page 2 you may click on Choices for the Rules property and view a list of the following definition types that you have created in the architecture:
    o Rule (DM2), and the following subtype:
      ▪ Standard (DM2), and the following subtype:
        ▪ FunctionalStandard (DM2)

Specifying the Measures of How a Performer Performs an Activity
You may specify the measures of how a Performer performs an Activity. An example of a measure is that a Performer must rescue a downed airman in a certain amount of time in order to be successful, otherwise the airman may perish or the search may be called off.
Creating Operational Exchanges

An Operational Exchange specifies what Performer performing what Activity sends resources to what other Performer performing what other Activity. Essentially, it is a relationship between two ActivityPerformedbyPerformers. The Operational Exchange points at one (and only one) ActivityResourceOverlap between the two Activities involved.

- An ActivityPerformedbyPerformer is a Role (and is noted as such in the DoDAF 2 specification Alias table). For example, a Drone (performer) performing Surveillance (activity) plays the Role of a Spy; a Drone (performer) injecting anti-fire chemicals on a fire (activity) plays the Role of Firefighter.
- An Operational Exchange is a relationship between two ActivityPerformedbyPerformers, aka Roles.
- The Operational Exchange points at one (and only one) ActivityResourceOverlap between the two Activities involved.

You may manually create Operational Exchanges on the fit-for-purpose OV-02z or on an OV-06c Activities Event Trace diagram. On these diagrams ActivityPerformedbyPerformers are visualized as node symbols, and you may draw an Operational Exchange relationship line between them.

Starting in version 11.4.2.4, System Architect provides capability to automatically create Operational Exchanges.

- Automatically-generated Operational Exchanges have an 'SA-controlled' property set.
- Manually-created Operational Exchanges will not be deleted when you run the automatic Operational Exchange function.
- You can report on Operational Exchanges that were manually created, if you wish to remove them.

Operational Exchange Example

Consider this Search and Rescue example:

- Satellite finds/observes victim, reports victim position to Command & Control, which Manages Position Information
- Search Helicopter finds/observes victim, reports victim position to Command & Control, which Manages Position Information
- Ground Search Team finds/observes victim, reports victim position to Command & Control, which Manages Position Information

In each case, same Activities, same informational resource (victim position) comes across. However, the quality of the data, or the trust you have in it, may vary depending on the weather, the time of day, or other conditions.

- Search Helicopter finds victim, reports victim position to Command & Control -- this OperationalExchange may not provide good data when over enemy territory during clear skies when Search Helicopter can be observed or fired upon.
- Satellite finds victim, reports victim position to Command & Control -- this OperationalExchange may not provide good data in the rain.
- Ground Search Team finds victim, reports victim position to Command & Control -- this OperationalExchange may not provide good data in the dark (at night).
The figure below shows how the Operational Exchanges are created to reflect ActivityPerformedbyPerformer relationships are modeled as rectangular nodes on OV-2z (see 1 and 2 callouts), with OperationalExchanges drawn between them. Each OpExchange points at a particular ActivityResourceOverlap (3).

Automatically Creating Operational Exchanges
System Architect provides a smart-algorithm utility to automatically generate Operational Exchanges based on information you have entered into the architecture.

1. Select **Tools, DoDAF 2 Utilities, Generate Operational Exchanges**. The operational exchanges are automatically generated and a report opens in a grid on the screen.
2. Examine the report. The Description specifies where no Operational Exchange was found, due to an ActivityResourceOverlap between two Activities performed by Performers, it created one -- and states the name of that Operational Exchange in the middle column.

3. Note that in the column headers:
   • (DM2r) = DM2 relationship
   • (DM2rx) = DM2 relationship extension, where extension means the type is not explicitly in the DM2 metamodel, but can be inferred from interpretation of the spec.

**Manually Creating/Visualizing Operational Exchanges on OV-06c**

You may use the OV-02z diagram to create new ActivityPerformedbyPerformer relationships, or visualize existing ones already created in the repository.

1. Create a new OV-06c Operational Exchanges diagram.
2. From the Explorer tree, drag-and-drop ActivityPerformedbyPerformers onto the diagram.
3. Note that on this diagram, you can create new ActivityPerformedbyPerformer relationships (which are represented by the object lifeline symbol) and create new Operational Exchanges between them by drawing an Operational Exchange relationship line, available in the Draw toolbar.

**Manually Creating/Visualizing Operational Exchanges on OV-02z**

You may use the OV-02z diagram to create new ActivityPerformedbyPerformer relationships, or visualize existing ones already created in the repository. The following steps show you how to do the later.

1. Create an OV-02z Operational Exchanges diagram.

   **Add Existing ActivityPerformedbyPerformer Relationships to OV-02z Diagram**

2. In the Explorer (Browser), expand Definitions, and then ActivityPerformedbyPerformer to reveal all such relationships already formed in the repository. Expand each one to see the Performer and Activity involved. Select one or more and drag-and-drop onto the diagram.

   Alternatively, you can right-mouse click on the diagram workspace and click Choices from the popup menu. In the Select and Drag dialog that opens, right-mouse click on it and choose Details to get the details of each ActivityPerformedbyPerformer relationship listed. You may see the Performer and Activity involved in each relationship already created in the repository, and drag-and-drop one or more relationships onto the diagram.
**Explorer Relationship Reports for OpExchange Guidance**

A number of Explorer Relationship reports are available in the Resource Flow Generation Reports.xml file, provided on DeveloperWorks. To use these reports, you can import the xml file into each encyclopedia you are using, as follows:

1. Download the Resource Flow Generation Reports.xml file from DeveloperWorks.
2. In System Architect, right-mouse click on the Definitions selection in the Explorer (Browser), and select Import XML.
3. Find and select the Resource Flow Generation Reports.xml file; use default collision options.

3. After having imported the Resource Flow Generation Reports.xml file into the encyclopedia, run the Explorer Relationship report named OV--02z activityResourceOverlap Generation. This is done in one of two ways:
   a. Expand the Definition type, Explorer Relationship Report in the Explorer (Browser), and drag-and-drop the OV--02z activityResourceOverlap Generation definition onto the OV-02z diagram.
   b. Select View, Heat Map Manager to open the Heat Map Manager pane. Click on the Explorer Relationship Report OV--02z activityResourceOverlap Generation.
Run the Explorer Relationship report OV--02z activityResourceOverlap Generation by dragging-and-dropping it onto the OV-02z diagram.

Run the Explorer Relationship report OV--02z activityResourceOverlap Generation from the Heat Map Manager pane.
The report runs, providing a visual of what Operational Exchanges should be formed, and the Activity Resource Overlap that should be related to the Operational Exchange.

The lines drawn are **not** relationships formed in the model -- they are simply visualizations of a report run of what relationships **should** exist -- based on the logic that says if Performer A performs Activity A, and Performer B performs Activity B, and Activity A has an Activity Resource Overlap to Activity B, then there is a an Operational Exchange between Performer A/Activity A and Performer B/Activity B. The Activity Resource Overlap that causes the inferred relationship is provided as a name of the report line.

![Image of a diagram showing Operational Exchange relationships](image.png)

**Create Operational Exchange Relationships**

4. Use the Operational Exchange symbol on the Draw toolbar of the OV-02z diagram to draw an Operational Exchange between a pair of Activity Performed by Performer symbols on the diagram, using the Explorer Relationship report lines as a guide.

5. In the Operational Exchange definition, go to the Activity Resource Overlap tab to specify the Activity Resource Overlap that relates the two Activities involved. Again, use the name of the Explorer Relationship report line as a guide.
6. Draw an Operational Exchange matching each of the Explorer Relationship Report lines.
7. Once all Operational Exchanges have been created, delete the Explorer Relationship lines.

OV-02z diagram with Operational Exchanges between ActivityPerformedbyPerformer joins.
Building the OV-2 Viewpoint

The OV-02 Viewpoint enables you to understand the high-level, operational view of what performers need to communicate with one another to exchange resources -- where a resource flow can be information, funding, personnel, or materiel. The need for resource flow -- or Needline -- between performers is established without prescribing the way that the Resources handled or without prescribing solutions.

You can draw the following symbols on an OV-02 Operational Resource Flow Description diagram:

- Performer, and any of its subtypes, such as Interface (Port), Organization, Performer, Person, Service Interface, or System
- Needline relationship lines between Performers

The OV-02 can also show the location of Operational facilities (Related Locations is a displayable property of Performer), and can optionally be annotated to show flows of information, funding, people, or materiel between Operational Activities.

Creating the OV-02

On the OV-02 you model Performers found at the Operational level -- typically Organizations, Persons, or the generic Performer definition type.

The basic steps of creating an OV-02 are as follows:

1. If you have started with modeling of the OV-05 views, use System Architect’s Needline generator to automatically generate Needlines.
2. Create an OV-02 Operational Resource Description diagram.
3. Create or drag-and-drop existing Performers on the diagram. If Needlines have been autogenerated, they will be automatically displayed.
4. If you have not modeled the OV-05 views yet, or wish to add Needlines to the architecture, relate the Performers with Needlines by drawing them from the Draw toolbar.
5. Within each Needline, specify the Operational Exchanges that cause the Needline to exist, in the Exchanges tab.
Automatically Generating Needlines

1. Select **Tools, DoDAF 2 Utilities, Generate Needlines**. A dialog opens to show you what is generated.

<table>
<thead>
<tr>
<th>Operational</th>
<th>Need Line</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>_1</td>
<td>TC1: No Need Line (DM2a) found for _1&quot;, so one will be created.</td>
</tr>
<tr>
<td>10</td>
<td>_2</td>
<td>TC1: No Need Line (DM2a) found for _10&quot;, so one will be created.</td>
</tr>
<tr>
<td>2</td>
<td>_3</td>
<td>TC1: No Need Line (DM2a) found for _2&quot;, so one will be created.</td>
</tr>
<tr>
<td>3</td>
<td>_4</td>
<td>TC1: No Need Line (DM2a) found for _3&quot;, so one will be created.</td>
</tr>
<tr>
<td>4</td>
<td>_5</td>
<td>TC1: No Need Line (DM2a) found for _4&quot;, so one will be created.</td>
</tr>
<tr>
<td>5</td>
<td>_6</td>
<td>TC1: No Need Line (DM2a) found for _5&quot;, so one will be created.</td>
</tr>
<tr>
<td>6</td>
<td>_7</td>
<td>TC1: No Need Line (DM2a) found for _6&quot;, so one will be created.</td>
</tr>
<tr>
<td>7</td>
<td>_8</td>
<td>TC1: No Need Line (DM2a) found for _7&quot;, so one will be created.</td>
</tr>
<tr>
<td>8</td>
<td>_9</td>
<td>TC1: No Need Line (DM2a) found for _8&quot;, so one will be created.</td>
</tr>
<tr>
<td>9</td>
<td>_10</td>
<td>TC1: No Need Line (DM2a) found for _9&quot;, so one will be created.</td>
</tr>
</tbody>
</table>

2. Examine the report. The Description specifies where no Needline was found, due to a Operational Exchange existing, it created one -- and states the name of that Needline in the middle column.

Remember that the automatic generation algorithm may encounter Needlines already created -- either by an earlier automatic generation, or ones that a user has manually added to the architecture. If a Needline is encountered that has been manually created, a new Needline will not be created -- the original one will remain in place.


**Optionally Set Recommended Style Settings**

4. Select **Format, Symbol Format, Line**, and choose a style of **Straight - Any Orientation**.

5. Select **Format, Center-to-Center Routing**.

6. Select **Format, Diagram Auto-Routing, Assign Connections**.

**Visualize Performers on the OV-2**

7. From the Explorer, drag Performers onto the diagram.
Optionally Set Display Settings

1. Right-mouse click on one of the lines and choose **Display Mode**.
2. In the Display Mode dialog, select the **Symbol Type on This Diagram** tab, select **Enabled**, and select **Operational Exchanges** – then click **Save All** and **Close**.
Optional: Best Practice on Manually Creating Needlines on OV-02

System Architect comes with several Explorer Relationship reports that can tell you where (between what Performers) Needlines should be drawn on an OV-2. This functionality predated automatic generation of Needlines, so is not really needed in building an OV-2, but you might want to use it when manually creating needlines.

1. Select View, Heat Map Manager to open the Heat Map Manager pane.

- Explorer Relationship Report run to report on Needlines that should be formed.

7. Draw a Needline to correspond to each report line on the diagram.
8. In each Needline, add the Operational Exchange as specified on the report line.

5. Once all Needlines have been created, delete the Explorer Relationship lines. They are simply visual reports – they are not real relationships.
OV-02 shows Needlines between Performers.
Systems Views

How to Build SV’s in System Architect -- Workflow Summary

Overview

Systems modeling involves the following steps that you may do in the following order although it is not mandatory:

1. **Model the hierarchy of System Functions** and their sub Functions using an SV-04a Systems Functionality Decomposition diagram.

2. **Build a leveled set of SV-04 Systems Resource Flow Description diagrams**, typically starting with a context diagram and then a series of child diagrams that are built to examine the sub-Functions of each System Function, until you get down to a level wherein all System Functions do not need to be decomposed further. A System Function that is not decomposed by Sub Functions is considered a 'leaf-level' function.

3. **Relate the Activities from the Operational View to System Functions** that perform them via the SV-05a Activity to System Function matrix.

4. **Specify what System performs each System Function**, using the ActivityPerformedbyPerformer relationship. You can do this on the SV-04 Systems Functionality Description diagram.

5. **Create System Exchanges** using the fit-for-purpose SV-01z System Exchanges diagram. On this diagram you model ActivityPerformedbyPerformer relationships as nodes, and draw System Exchanges between them. A System Exchange is the exchange between a particular System performing a particular System Function, and another particular System performing a particular System Function -- in other words, a relationship between two ActivityPerformedbyPerformers. An Explorer Relationship Report provides automatic guidance on what System Exchanges you should create.

6. **On each System Exchange**, point at the System Data Flow line that carries the Resources between the two Systems involved. A System Exchange has a reference to one and only one System Data Flow. The same Explorer Relationship Report mentioned in the previous step offers automatic guidance on what System Data Flow you should relate to each System Exchange.


8. **View what Systems are related in the SV-03 System to System (System Resource Flow) matrix**. At the intersection of each join is the System Resource Flow definition. You may also add/remove/edit relationships in the matrix.

9. **Generate the SV-06 report**.

**SV-04a Systems Functionality Decomposition diagram**

You use the SV-04a Systems Functionality Decomposition diagram to view the hierarchical relationship between System Functions (see figure below).
Creating a System Hierarchy on the Diagram

With System Architect 11.4.1 and later, you use auto-connect functionality to relate a System Function to a parent System Function. Simply move a System Function that you wish to be a child of another System Function underneath it until a line automatically appears. The relationship is formed.

You may open the definition of the child System Function and in its **Part of Function** tab, view the name of the parent System Function, automatically added based on the relationship you formed on the diagram.

Creating a System Hierarchy in the Definitions

You can relate a System Function to its parent System Function in its definition, without having any diagram open, and then auto-draw an SV-4a diagram.

1. Open the definition of a System Function and in its **Part of Function** tab, you can specify the name of its parent System Function. You can also specify its sub-Functions by populating its **Parent of Functions** tab. All System Functions must exist already -- you drag them in by clicking the Choices button.

2. Create an SV-04a Systems Functionality Decomposition diagram and drag-and-drop System Functions from the Explorer tree onto the diagram workspace; the diagram draws itself, with appropriate relationships.
**LEAF FUNCTIONS** -- Leaf Level System Functions are System Functions that are not decomposed further by a Sub Function. In the diagram pictured above, Request Coverage, Observe Non-Friendlies, and Observe Weather are ‘leaf-level’ Activities.

### Specifying Hierarchical Numbering

Hierarchical numbering of System Functions allows you to easily see the decomposition level of an Activity.

To turn on hierarchical numbering in SA 11.4.2.1 and later, perform the following steps:

1. In an open SV-04 Systems Functionality Decomposition diagram, edit its Diagram properties -- right-mouse click on an empty area of the diagram workspace and choose **Diagram Properties**, or select **Edit, Diagram Properties**.
2. In the Diagram Properties dialog of the SV-04 Systems Functionality Decomposition, make sure the **Hierarchical Numbering** property is toggled on; the property is located in the **Hierarchical Numbering** tab of the dialog. You may also set the **First Node Number** -- this will set the number of the highest-parent symbol on the diagram; if you leave it blank, the default is 0.
3. Select (or draw and then select) an Activity symbol on the diagram, and select **Format, Symbol Format, Symbol Style**.
4. Make sure the options **Auto number** and **Level numbers** in the **Allow** group are toggled on. Note the value of the **Next number** property -- this number will automatically be updated as you draw.
5. Select **Tools, Preferences**, and in the **Preferences** dialog, make sure the **Number** property in the **Auto** group is toggled on.

Note that Prefix code is superceded in DoDAF 2 by Alternate Depiction functionality. In SA 11.4.2 and later, you may right-mouse click on any symbol and choose Display Selected Symbols with Adornment. This provides similar functionality the prefix code, which annotated a symbol as a Process (prefix code P), Activity (prefix code A), etc.
**SV-04b Systems Functionality Description Diagram**

The SV-04b Systems Functionality Description diagram describes the Resources exchanged between System Functions in the architecture, and Resources exchanged with System Functions that are outside the scope of the model, considered External Functions.

A best practice is to create a high-level 'context' SV-04b diagram, which shows the highest-level System Functions in the architecture, and how it interchanges with External Functions. You then decompose that highest-level System Function with a child SV-04b diagram that shows its Sub Functions and how Resources flow between those System Functions.

**Modeling Flow Between System Functions**

You use an System Data Flow line to model flow of Resources between Activities. Within each System Data Flow you may specify the actual Resources that flow from one System Function to the other.

To specify one or more Resources flowing between two System Functions, do the following:

1. Relate the two System Functions by a System Data Flow line.
2. On the Resources tab of the System Data Flow definition, click the Choices button to view all Resources already created in the encyclopedia. The list shows all definitions of type Resource and sub types: Information, Data, DomainInformation, ServiceDescription Performer, Person Type, Organization Type, System, Service, and Port.
3. Select one or more definitions in the Choices list and drag-and-drop them into the Resources grid.

**Modeling Decomposition of System Functions**

You may model decomposition of a System Function through the use of Parent-Child diagramming for navigation and visualization, and also relating the System Function to its children through their definitions.

To graphically show decomposition of an System Function, you may do the following:

1. Right-mouse click on a System Function on an SV-4b diagram, and select Create Child Diagram (note: you may also use the menu choice Edit, Child Diagram, Create Child Diagram)
2. Name the new diagram, accept the default diagram type of SV-04b Systems Functionality Description (DM2), and click OK, then select Yes to save changes to the previously current diagram.
3. Model Sub Functions on the new diagram
You will now be able to navigate between parent and child System Functions on two different SV-4b diagrams, however the definitions must still be related to one another via a composition relationship, described in the steps below.

Creating a Chain of Parent/Child SV-4b's that Reflect the SV-4a

Creating a parent/child linkage between SV-04b diagrams provides easy navigation through the model, but it does not relate the System Function definitions that are pictured on the diagram in an Function-part-of-Function relationship. Once you have created a parent/child linkage between two SV-4b diagrams, and modeled sub Functions on the child SV-4b, you must perform the following steps:

1. Open the definition of a System Function on the child diagram and in its **Part of Function** tab, click the **Choices** button and drag-and-drop in the name of the parent System Function.
2. Repeat step 1 for all System Functions on the diagram that you wish to model as sub Functions to the parent. Note that this step can be performed independently of the parent/child diagrams -- they just serve as a navigation and logic aid.

Furthermore, once you have related System Functions in the **Part of Function** tab of the definition, you may auto-build an SV-04a diagram by simply dragging the System Functions onto the diagram -- as discussed earlier. Data centric behavior will autodraw the relationships on the diagram for you. This provides two ways of working:

A. Build the SV-4a first -- within which you visually build the System Function/Sub-System Function relationships, then build the SV-4b's for each main System Function level.
B. Build the SV-4b leveled set of diagrams first, and at each level make sure you relate the sub Function to its parent System Function in the definition; then autodraw the SV-4a.
Specifying the Systems that Perform a System Function

You may specify the System (a type of Performer) that performs a System Function via the ActivityPerformedbyPerformer relationship line, available on the SV-04b Systems Functionality Description (DM2) diagram type.

SV-04b Systems Functionality Description diagram enables you to model Systems that perform System Functions via ActivityPerformedbyPerformer relationship. Note that in this example, a poor choice is used for the name of ActivityPerformedbyPerformer -- a better choice would be to use "System Name - System Function Name".

Best Practice: Naming Convention for ActivityPerformedbyPerformer

A best practice for naming an ActivityPerformedbyPerformer relationship is:

System Name - System Function Name

For example, JSTAR C-130 - Request Coverage

Following such a naming standard comes in handy in various aspects of your modeling:
- Looking through a list of relationships already formed in the repository and represent them on a diagram.
- Modeling ActivityPerformedbyPerformers as vertical lifelines on the OV-6c Operational Exchanges diagram or SV-10c Performer-Role Event Trace diagram.

Note that In the current version of System Architect, a list of System Functions is not provided in the System definition; nor is a list of Systems provided in the System Function definition. The
relationship is an explicit join -- System has a relationship to ActivityPerformedbyPerformer which relates it to System Function. To get a quick view of what System Function is being performed by what System globally in your architecture, you can do the following:

1. Select a System Function or a System in the Explorer (browser) or on any diagram.
2. Select View, References.
3. In the References pane that opens, expand Definitions, and expand ActivityPerformedbyPerformer within that.

Right mouse click on the name of a particular ActivityPerformedbyPerformer relationship and choose References -- another window will open that contains all referenced items for the selected relationship.

**Modeling Where and When a System Performs a System Function**

You may specify where and when a System performs a System Function in the ActivityPerformedbyPerformer relationship. You may model more than one ActivityPerformedbyPerformer relationship between a System and a System Function to specify different conditions, times, and locations.

**Specifying Where a Performer Performs an Activity**

You may specify *where* a System performs a System Function.

1. On an ActivityPerformedbyPerformer definition, select the Where and When tab.
2. On page 1, click on Choices for the Where property and view a list of Location type definitions (and subtypes of Location definition) that you have created in the architecture; select and drag items into the Where grid.

The following definition types are available to describe location:

- Location (DM2), and the following subtypes
  - GeoPoliticalExtent (DM2), and the following subtypes:
    - Installation (DM2)
    - GeoFeature (DM2)
    - RegionOfCountry (DM2)
    - Country (DM2)
    - RegionOfWorld (DM2)
    - RealProperty (DM2), and the following subtypes:
      - Facility (DM2)
      - Site (DM2)

**Specifying When a System Performs a System Function**

You may specify when the System performs a System Function.

1. Select the Where and When tab of the ActivityPerformedbyPerformer definition.
2. On page 2 you may click on Choices for the When property and view a list of the following definition type that you have created in the architecture:
   - Measure (Temporal).

**Specifying Under What Environmental Conditions a System Performs an System Function**

You may specify under what conditions the System performs a System Function. An example of Environmental Condition is in the Desert, or in the Rain, or At Night, etc.

1. Select the Caveats tab of the ActivityPerformedbyPerformer definition.
2. On page 1 you may click on Choices for the **Conditions** property and view a list of the following definition type that you have created in the architecture:
   - Condition (Environmental) (DM2)

**Specifying Under What Rules a System Performs a System Function**
You may specify under what rules System performs a System Function. Rules have subtype Standards -- so an example might be that a System rescues a ship under particular Maritime standard.
1. Select the *Caveats* tab of the ActivityPerformedbyPerformer definition.
2. On page 2 you may click on Choices for the **Rules** property and view a list of the following definition types that you have created in the architecture:
   - Rule (DM2), and the following subtype:
     - Standard (DM2), and the following subtype:
       - FunctionalStandard (DM2)

**Specifying the Measures of How a System Performs a System Function**
You may specify the measures of how a System Performs a System Function. An example of a measure is that a System must rescue a downed airman in a certain amount of time in order to be successful, otherwise the airman may perish or the search may be called off.

**Creating System Exchanges**
A System Exchange specifies what System performing what System Function sends resources to what other System performing what other System Function. Essentially, it is a relationship between two ActivityPerformedbyPerformers. The System Exchange points at one (and only one) System Data Flow between the two System Functions involved.

**Automatically Creating System Exchanges**
1. Select **Tools, DoDAF2 Utilities, Generate System Exchanges from System Data Flows.**

<table>
<thead>
<tr>
<th>System Data Flow (DM2)</th>
<th>System Ex.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Camera Position Commands</td>
<td>1</td>
<td>TC1: No matching System Exchange (DM2) found for 'Camera Position Commands', so one will be added</td>
</tr>
<tr>
<td>Camera Position Commands</td>
<td>2</td>
<td>TC1: No matching System Exchange (DM2) found for 'Camera Position Commands', so one will be added</td>
</tr>
<tr>
<td>Camera Position Commands</td>
<td>3</td>
<td>TC1: No matching System Exchange (DM2) found for 'Camera Position Commands', so one will be added</td>
</tr>
<tr>
<td>Video</td>
<td>4</td>
<td>TC1: No matching System Exchange (DM2) found for Video, so one will be added</td>
</tr>
<tr>
<td>Video</td>
<td>5</td>
<td>TC1: No matching System Exchange (DM2) found for Video, so one will be added</td>
</tr>
<tr>
<td>Video</td>
<td>6</td>
<td>TC1: No matching System Exchange (DM2) found for Video, so one will be added</td>
</tr>
<tr>
<td>Video and Target Detection</td>
<td>7</td>
<td>TC1: No matching System Exchange (DM2) found for 'Video and Target Determination', so one will be added</td>
</tr>
<tr>
<td>Video of Target Area</td>
<td>8</td>
<td>TC1: No matching System Exchange (DM2) found for 'Video of Target Area', so one will be added</td>
</tr>
<tr>
<td>Video of Target Area</td>
<td>9</td>
<td>TC1: No matching System Exchange (DM2) found for 'Video of Target Area', so one will be added</td>
</tr>
<tr>
<td>Video of Target Area</td>
<td>10</td>
<td>TC1: No matching System Exchange (DM2) found for 'Video of Target Area', so one will be added</td>
</tr>
</tbody>
</table>
Optional: Visualize the System Exchanges on an SV-01z
You can optionally use the **SV-01z System Exchanges** to visualize System Exchanges and the ActivityPerformedbyPerformers (alias Roles) they are relating, using the fit-for-purpose SV-01z diagram, which has been introduced into System Architect for this purpose (it is not prescribed by the DoDAF 2 spec).

1. Create a new diagram of type **SV-01z System Exchanges**

**Optional: Select Draw Styles**
For best visualization, optionally select these choices:

2. Select Straight – Any Orientation line style, and center-to-center routing selected.
3. Select **Format, Symbol Format, Line**, and choose a style of **Straight – Any Orientation**.
4. Select **Format, Center-to-Center Routing**.
5. Select Format, Diagram Auto Routing, Assign Connections.

**Auto-Visualize the System Exchanges**

6. Drag and drop Performers from the Explorer tree onto the diagram.
Double click on System Exchange ' _1' to open its definition. Note that on the Generation tab, its Valid and SA-Controlled properties are toggled on.

Name  

<table>
<thead>
<tr>
<th>Described By</th>
<th>Description</th>
<th>Data Flow</th>
<th>Generation</th>
<th>System Resource Flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>✓ Valid</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>✓ SA-controlled</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Right mouse click in the Explorer (browser) and select Refresh. Note that under the System Exchange definition type, all the System Exchanges are listed, including any manually added ones.

Optional: Manually Creating System Exchanges on SV-01z

You may optionally create System Exchanges on the fit-for-purpose SV-01z diagram. On this diagram ActivityPerformedbyPerformers are visualized as node symbols, and you may draw a System Exchange relationship line between them.

1. Create an SV-01z System Exchanges diagram.
You may use the SV-01z diagram to create new ActivityPerformedbyPerformer relationships, or visualize existing ones already created in the repository. The following steps show you how to do the later.

**Add Existing ActivityPerformedbyPerformer Relationships to SV-01z Diagram**

2. In the Explorer (Browser), expand Definitions, and then ActivityPerformedbyPerformer to reveal all such relationships already formed in the repository. Expand each one to see the System and System Function involved. Select one or more and drag-and-drop onto the diagram.

   ![Diagram](image)

   Drag ActivityPerformedbyPerformer definitions onto diagram

   a. Alternatively, right-mouse click on the diagram workspace and click Choices from the popup menu. In the Select and Drag dialog that opens, right-mouse click on it and choose Details to get the details of each ActivityPerformedbyPerformer relationship listed. You may see the System and System Function involved in each relationship already created in the repository, and drag-and-drop one or more relationships onto the diagram.

---

**Explorer Relationship Reports for OpExchange Guidance**

A number of Explorer Relationship reports are available in the Resource Flow Generation Reports.xml file, provided on DeveloperWorks. To use these reports, you can import the xml file into each encyclopedia you are using, as follows:
1. Download the Resource Flow Generation Reports.xml file from DeveloperWorks.
2. In System Architect, right-mouse click on the Definitions selection in the Explorer (Browser), and select Import XML.
3. Find and select the Resource Flow Generation Reports.xml file; use default collision options.

3. After having imported the Resource Flow Generation Reports.xml file into the encyclopedia, run the Explorer Relationship report named SV-01z and SvcV-01 System Data Flow and Service Data Flow Generation. This is done in one of two ways:
   c. Expand the Definition type, Explorer Relationship Report in the Explorer (Browser), and drag-and-drop the SV-01z and SvcV-01 System Data Flow and Service Data Flow Generation definition onto the OV-02z diagram.

Run the Explorer Relationship report SV-01z and SvcV-01 System Data Flow and Service Data Flow Generation by dragging-and-dropping it onto the SV-01z diagram.

The report runs, providing a visual of what System Exchanges should be formed, and the System Data Flow that should be related to the System Exchange.

The lines drawn are not relationships formed in the model -- they are simply visualizations of a report run of what relationships should exist -- based on the logic that says if System A performs System Function A, and System B performs System Function B, and System Function A has a System Data Flow to System Function B, then there is a a System Exchange between System A/System Function A and System B/System Function B. The System Data Flow that causes the inferred relationship is provided as a name of the report line.

**Create Operational Exchange Relationships**

4. Use the System Exchange line symbol on the Draw toolbar of the SV-01z diagram to draw a System Exchange between a pair of ActivityPerformedbyPerformer symbols on the diagram, using the Explorer Relationship report lines as a guide.

5. In the System Exchange definition, go to the System Data Flow tab to specify the System Data Flow that relates the two Systems involved. Again, use the name of the Explorer Relationship report line as a guide.
6. Draw a System Exchange matching each of the Explorer Relationship Report lines.
7. Once all System Exchanges have been created, delete the Explorer Relationship lines.

SV-01z diagram with System Exchanges between ActivityPerformedbyPerformer joins.
**Building the SV-1 Viewpoint**

On the SV-01 you model Systems and their need to interface, using System Resource Flow relationship lines. The SV-01 is modeled as follows:

1. If you have started the architecture by building SV-4’s, then autogenerate System Resource Flows.
2. Create an SV-01 System Interface Description diagram.
3. Create or drag-and-drop existing Systems onto the diagram. If you have autogenerated System Resource Flows, they will automatically be represented.
4. You may manually relate the Systems with System Resource Flow relationship lines.
5. Within each System Resource Flow, specify the System Exchanges that cause the System Resource Flow to exist, in the Exchanges tab.

**Automatically Generating System Resource Flows**

1. Select **Tools, DoDAF2 Utilities, Generate System Exchanges from System Data Flows**.

<table>
<thead>
<tr>
<th>System Data Flow (DMs)</th>
<th>System Exchanges</th>
<th>Description</th>
</tr>
</thead>
</table>
| Camera Position Commands | TC1 | No matching System Exchange (DMs) found for 'Camera Position Commands'
| Camera Position Commands | TI2 | No matching System Exchange (DMs) found for 'Camera Position Commands'
| Camera Position Commands | TI3 | No matching System Exchange (DMs) found for 'Camera Position Commands'
| Videos | TC1 | No matching System Exchange (DMs) found for 'Videos'
| Videos | TI6 | No matching System Exchange (DMs) found for 'Videos'
| Videos and Target Determination | TC7 | No matching System Exchange (DMs) found for 'Videos and Target Determination'
| Videos of Target Area | TC8 | No matching System Exchange (DMs) found for 'Videos of Target Area'
| Videos of Target Area | TI9 | No matching System Exchange (DMs) found for 'Videos of Target Area'
| Videos of Target Area | TI10 | No matching System Exchange (DMs) found for 'Videos of Target Area'

2. Select Systems from the Explorer tree and drag them onto the diagram. The System Resource Flows between them will automatically visualize.
3. Notice that in the diagram above, the System Resource Flow named UAV Position has a note under it that says SA Controlled = F (false).
4. If you open the definition of one of the System Exchanges that are numbered – such as '1' you will notice that Valid and SA-Controlled properties are toggled on.

Optional: Aids on Manually Creating SV-01
1. SA provides guidance reports to help you manually create System Resource Flows on an SV-1 if you wish to.
2. On an SV-1, drag-and-drop Systems onto the diagram.
3. Select View, Heatmap Manager to open the Heatmap Manager.
4. From the Heatmap Manager, click on the Explorer Relationship report named OV-02 and SV-01 and SvcV-01 Operational Exchange and System Exchange Generation.
Explorer Relationship Report run to report on System Resource Flows that should be formed.

5. Draw a System Resource Flow to correspond to each report line on the diagram.
6. In each System Resource Flow, add the System Exchange as specified on the report line.

7. Once all System Resource Flows have been created, delete the Explorer Relationship report lines.
SV-01 shows System Resource Flows between Systems.
Modeling Systems of Systems
On the SV-1 diagram you can simply move one System inside the box of another symbol, and a relationship will automatically be formed (functionality available in SA 11.4.2 and later). In the System definition, there are two properties -- PartOf System (autopopulated to show the parent symbols of a system) and ParentOf System (autopopulated to show the children that the system is parent of. These properties will also auto display on the diagram.

![Diagram showing system relationships](image)

You can move a system inside a Performer and get the same behavior. In the picture below, "New York City Transit Authority" is a Performer.

![Diagram showing performer relationships](image)

Modeling Systems of Systems on the SV-1 Alternative
The SV-1 Alternative diagram enables you to model the "Parent Symbol" relationship as a line drawn between two Systems -- same model behavior as on the SV-1, except that on the SV-1 you are forming the relationship via 'box in box' data centricy. (Note: You could on the SV-1 Alternative put one symbol inside the other, then right-mouse click on the line and hide it if you so choose.)

Location of System
You can specify the Location of a Performer in its definition -- within the Related Locations tab.

Location of System Performing Activity
An ActivityPerformedbyPerformer definition has a Location property; so you can model, for example, that a "Subway Car" (system) 'carries passengers' (activity) in NYC (location) -- you'd model this on the ActivityPerformedbyPerformer relationship between the system and the activity.
You could also model that "Amtrak" (system) ‘carries passengers’ (activity) between NYC and Washington, DC (location) on another ActivityPerformedbyPerformer relationship between that system and activity. You could also model time (temporal measure) on the ActivityPerformedbyPerformer relationship, as well as other measures and conditions. You can form multiple ActivityPerformedbyPerformer relationships between a system and an activity.
**Building the SV-2 Viewpoint**

You use the SV-2 to specify the System Resource Flows between Systems and may also specify physical connection information, such as bandwidth or protocol stacks used in connections, in the Physical Resource Flow connection.

The Physical Resource Flow connection allows you to specify System Exchanges that cause an interface between systems.

Alternatively, you may wish to model ports of a system, and connect the systems through the ports. To do this you use a Port symbol, and place it inside the System; data centric behavior automatically updates the definitions to show containment of the port inside the system. You may draw a Physical Resource Flow line between Ports, or between a Port and another System (into a black box of the System, without needing to specify the other System's Port).
**Modeling Timing of System Exchange Between Systems**

To model the sequential timing of resource exchange between Systems, you can use one of three different types of SV-10c diagrams:

- **SV-10c Systems Event Trace**
- **SV-10c System Functions Event Trace**
- **SV-10c Performer-Role Event Trace**

**SV-10c Systems Event Trace**

The SV-10c Systems Event Trace enables you to model Systems as vertical 'object lifelines', and System Resource Flows as lines drawn between the Systems, in a chronological fashion.

**SV-10c System Functions Event Trace**

The SV-10c System Functions Event Trace diagram enables you to focus on System Functions (represented by vertical 'object lifelines') and the System Data Flows as horizontal lines drawn between them, in chronological order from top to bottom on the diagram.
This view is an alternate to the SV-05b, enabling you to model timing.

SV-10c Performer-Role Event Trace
The SV-10c Performer-Role Event Trace enables you to model System Exchanges between ActivityPerformedbyPerformer pairs. This is a powerful view, and an alternate way to build System Exchanges than on the SV-01z diagram.

Each System performing a particular System Function, as represented by an ActivityPerformedbyPerformer relationship, is represented as a vertical 'object lifeline'. You may specify the particular location or time that the System is performing the System Function within the ActivityPerformedbyPerformer definition. System Exchanges are modeled as horizontal lines between the System/System Function lifelines, modeled in chronological order from top to bottom on the diagram.

Drawing Tip: Use Diagram Auto Routing Draw Option
System Architect comes with many drawing and line routing options. One useful option to turn on when drawing an SV-10c type diagram is Diagram Auto Routing. Select Format, Diagram Auto Routing, Assign Connections (or Retain Connections) to turn it on.
Measures

You may specify measures of how things get done or should get done for every DoDAF 2 definition type. An example of a measure is that a website must be operational 24 hours a day, and 7 days a week, or that communication between Performers must have a Security measure of Top Secret.

Measurement Sets, Measurement Types, Measurement Values

Every DoDAF 2 encyclopedia in SA 11.4.2.1 and later comes with a preloaded set of Measurement Set, Measurement Type, and Measurement Value definitions, listed in Appendix A.

There are several types of Measure definition types:

- **Measurement Set** -- a group Measurement Types (for example, SecurityAttributes)
- **Measurement Type** -- enables you to specify unit of measure, ranking, and list of values (for example, Classification type for the SecurityAttributes measurement set)
- **Measurement Value** -- enables you to specify values measurement types (example, Top Secret for the Classification type of the SecurityAttributes measurement set). You may create a set of Measurement Values to provide a preset list of values for a measurement attribute. Each Measurement Value has a Rank property that you may use for comparison purposes in reports, analytics, heatmaps, etc.
- **Measurement Instance** -- enables you to specify a measurement value to associate with a particular model artifact (Ex. S=Secret with ranking of 4). You may choose from a Measurement Type value list or enter your own value. Each Measurement Instance is an instance of Measurement Type and uses the Unit of Measure from Measurement Type. Each Measurement Instance is keyed by Name and GUID which allows duplicate names to be used, with different meaning.
You may add to, delete, or change the Measurement Sets, Measurement Types and Measurement Values without property set changes (No USRPROPS changes).

**How to Create and Apply Measurements**

To create and apply measures, you do the following:

**View the Measurement Sets and Types Available**

1. Make sure there is a **Measurement Set** you wish to use. In the Explorer (Browser), expand the definition type Measurement Set to see what is available. (Note: you can add a new Measurement Set by creating a new definition.)

2. Make sure there is a **Measurement Type** that you wish to use. Expand Measurement Set to show the Measurement Types available for that Set. (Note: you can add new Measurement Types to a Set by creating new Measurement Type definitions.)

3. Make sure there is a **Measurement Value** that you wish to use. Expand a Measurement Type to see the Measurement Values available for that Type. (Note: you can add new Measurement Values to a Type by creating new Measurement Value definitions.)
4. Create a Measurement Instance that you will apply to a definition.
5. In the Measurement Instance definition, specify the Measurement Set it belongs to, by clicking on the **Choices** button in the **Measurement Set** field and drag-and-dropping a Measurement Set in.
6. Click **Apply**. (Note that you **must** click Apply to see the set of Measurement Types for a Measurement Set in the next step.)

7. Specify the Measurement Type the Measurement Instance belongs to, by clicking on the **Choices** button in the **Measurement Type** field and drag-and-dropping a Measurement Type in.
8. Click **Apply**. (Note that you **must** click Apply to see the set of Measurement Values for a Measurement Type in the next step.)

9. Specify the Measurement Value of the Measurement Instance by clicking on the **Choices** button in the **Measurement Value** field and drag-and-dropping a Measurement Value in.
10. Click **OK** to close and save the new definition.

11. Apply the Measurement Instance to the definition that you wish -- in the example below it is applied to an ActivityPerformedbyPerformer relationship, in this case stating that a *Downed Airman* must communicate with a *JSTAR* through *Secure Communications* that is *Top Secret (TS)* classification.
PV-01 Project Portfolio Relationships

The PV-01 viewpoint describes the breakdown of Projects into Sub Projects, and the dependency relationships between organizations and the Projects they own, and the organizational structures needed to manage a portfolio of projects.

Using the Diagram to Build the Model

The PV-01 Project Portfolio Relationships diagram is a data centric, hierarchically built diagram. It enables the following relationships to be created:

- You may attach a Project to a parent Project by moving it under the parent Project until a connection line appears, auto-connecting the Project to the parent Project. Once you do so, the definition of the parent Project will have its Project Has Subprojects property of its Hierarchy tab automatically filled in appropriately, and the definition of the child Project will have its Project Part of Project property automatically filled in appropriately.
- You may attach a Project (and its hierarchy) to its owning Organization by moving the Project under the owning Organization symbol until a connection line appears, auto-connecting the Project (and its hierarchy) to the owning Organization. Once you do so, the definition of the owning Organization's Organization Owns Projects property in its Projects tab will be automatically filled in appropriately, and the definition of the owned Project will have its Project Owned by Organization property in its Organization tab filled in appropriately.

Creating a hierarchy on the PV-1 diagram: Select ONE symbol and move it and its underlying hierarchy underneath another symbol to form the new hierarchy. In this case, a Project (and all its Sub Projects) is moved underneath an Organizational Unit, and auto-attached. You may also move Projects underneath each other to attach in a hierarchy on the PV-1 diagram. To model Organizations in a hierarchy, use the OV-4 diagram.

Using the Definitions to Build the Model -- Auto Visualizing the Diagram

Further, you may relate the Projects to their owning Projects or sub Projects or owning Organizations within the appropriate definition properties (as stated above), and drag and drop the Project definitions onto the PV-1 diagram -- the diagram and the hierarchical relationships will draw themselves.

PV-01 Project Portfolio Relationships At Time

The PV-01 Project Portfolio Relationships At Time diagram is a non-data centric diagram that allows you to model Projects and their Hierarchy, and the Organizations that own them at a
particular Milestone, or point in time. The Milestone is related to the diagram in the properties of the diagram.

Relationships that you form on this diagram do not effect the properties of the definitions on a global basis throughout the repository. This enables you to sketch out Projects and their owning Organizations without committing the changes to a global view.

**PV-02 Project Timelines**

A Project is a plan or scheme for a task that takes place over a period of time. It contains Milestones. The **PV-2 Project Timelines** diagram enables you to visualize a program or portfolio of Projects and their Milestones based on a timeline. You may also visualize interdependencies of Projects.

You may specify that the Project tracks or does the following:

- **Desired Effect(s) of the Project** -- Desired Effect is a primary definition type in DoDAF 2. You may specify the Desired Effect of a Project on page 2 of the **DoDAF 2** tab of the Project definition. The Desired Effect definition must be created first, then dragged into the Project definition's Desired Effect field using the Choices button.
  - Example: A Desired Effect of the Project to "Renovate Kitchen" is "Modernize Kitchen So House Can Be Rented".
- **Activities** that are performed during the Project.
- **Capabilities that the Project enables at Each Milestone** -- you may specify the Capabilities that the Project enables at each Project Milestone. This is done in the Capability definition, within the **CV-3** tab -- click on Choices and populate the **Capability Increments** property with the Milestone at which it is enabled. (If you right-mouse click..."
anywhere in the Select and Drag dialog and click on Details, you can see the Project that each Milestone is associated with in the Details window.)

- **Hierarchy of Projects** -- You can specify whether the Project has Sub Projects, or is itself a Sub Project reporting to a Parent Project. This is done either in the Hierarchy tab of the definition of a Project or by visualizing/creating/editing the hierarchy on the PV-1 Project Portfolio Relationships diagram.
  - In the Hierarchy tab of the Project definition, you may specify that the Project is part of a higher-level Project in the Part of Project property. The parent Project must be already created in the repository; you click on Choices to drag it into the field.
  - You may specify that a Project has sub Projects in the Project Has Sub Projects field of the Hierarchy tab of the Project definition. You may automatically visualize the hierarchy of projects by dragging them onto the PV-1 Project Portfolio Relationships diagram. You may also use the diagram to place projects in a hierarchy (thereby editing the appropriate project definitions).

- **Who owns the Project** -- you may specify the Organization or Person that own the Project in the Organization tab of the Project definition.

- **Described By** -- this property enables you to provide a description of the Project through a reusable set of Information definitions. You must create the Information definition first, via the Explorer (Browser), and then drag it into the Project definition. The alternative is to use the Description textual property of the Project definition.

- **Measures** -- you may specify Measures for the Project. You must create the Measure definition first (and any of its subtypes) via the Explorer (Browser) and then drag and drop the Measure definition into the Project definition.

Note: You may edit the Project timeline symbol by selecting to edit (or double-clicking on) its horizontal bar symbol. You may edit the Milestone of a Project timeline symbol by selecting to edit (or double-clicking on) its circle symbol.

You may also draw Dependency lines between Milestones of different projects, thereby modeling the interdependencies between the Project stages.

**Project Milestones**

**Milestones** are represented as circles positioned on the project’s timeline bar. Each Milestone circle is divided into sections so that it resembles a pie. Each section of the Milestone ‘pie’ represents a particular **Project Thread** that you are tracking, such as Equipment, Logistics, Infrastructure, Organization, Doctrine, Information, Personal, and Training. The color of the pie section represents the **status** of that particular Project Thread at that Milestone, as specified later on this page.
Default Set of Project Milestones

For DoDAF 2, System Architect provides a standard set of Milestones which you can change (described later on this page). The standard set of Milestones is as follows:

- IG (Initial Gate)
- MG (Main Gate)
- IOC (Initial Operational Capability): The first attainment of the capability to employ effectively a weapon, item of equipment, or system of approved specific characteristics, and which is manned and operated by an adequately trained, equipped, and supported military force or unit.
- FOC (Full Operational Capability): Full capability to employ effectively a weapon, item of equipment, or system of approved specific characteristics, and which is manned and operated by an adequately trained, equipped, and supported military force or unit.
- IS (In Service)
- Out of Service
- Disposal

Creating Your Own Standard Set of Default Milestones for Projects

You may create your own set of Milestones for any Project. You may also set up a standard set of Milestones so that all new Projects that you add to a diagram automatically have the standard set. This is accomplished through use of the "Model Project" specified for an encyclopedia, within the Project definition set. All DoDAF 2 encyclopedias are prepopulated with a Project definition named "Model Project", and this project has the default set of Milestones. If you change the Milestones in this definition, then all new Projects will have your default set of Milestones through inheritance -- the milestones specified in this 'Model Project' become the default milestone set for all new Projects drawn on a PV-2 diagram (if there are no existing Projects on the diagram).

1. In the Definitions grouping of the Rational System Architect Explorer (Browser) of a DoDAF 2 encyclopedia, navigate to the Project definition.
2. Notice that there is a Project named Model Project. This definition instance is loaded into the encyclopedia upon creation of any DoDAF 2 encyclopedia.
3. Open the Model Project.
4. Either leave the standard set of Milestones, or add your own.

Milestones filtered by topmost project, phase colors come from settings in topmost project: this setting may be set in the PV-02 diagram properties. When toggled on, it causes the following behavior: If there is at least one Project already drawn on an PV-2 diagram, then any additional new Projects drawn on the diagram will contain Milestones as specified in the top-most Project on the diagram.

Projects dragged onto a diagram from the Explorer (Browser) will display the Milestones contained in their definition – not the ones specified in the Model Project template definition, or the ones on the top-most Project on the diagram. Therefore, you may create a diagram that has a heterogeneous group of Project definitions.
Running Rules Checks for Creation of Milestones

Rules checks can be run to make sure that the Milestones on the diagram are in correct chronologic order based on their Milestone Date property (for example, Stage_2 always comes after Stage_1, and Stage_3 always comes after Stage_2, etc). You may run the Rules Check manually by selecting Reports, Rules Check.

You have the following options available for Rules Checks for the PV-2 diagram, available in the Behavior tab of the diagram's property dialog (either select Edit, Diagram Properties with the PV-2 diagram open, or right-mouse click on the diagram workspace and select Diagram Properties):

- **Base rules check on topmost project** - Rules checks are run to make sure that the Milestones on the diagram are in correct chronologic order based on their Milestone Date property. The Rules Check can be based on the "Model Project" or on the top-most Project on a diagram. To specify that the rules check be based on the top-most Project on a diagram, toggle this property on. Rules checks for new diagrams are based on the Model Project unless you toggle on this option.

- **Run rules check automatically** - As mentioned above, you may run the Rules Check manually by selecting Reports, Rules Check. If you toggle on the Run rules check automatically option, the Rules Check will run automatically when you open a diagram or draw symbols on it.

Milestone Types and Status

Each Milestone of a Project tracks the following threads:

- Equipment
- Logistics
- Infrastructure
- Organization
- Doctrine
- Information
- Personnel
- Training

You may specify the status of each of these threads for each Milestone on the Project Threads tab of the Milestone definition. Each status of a Milestone thread is represented on the Milestone pie-chart symbol by the following colors:

**Milestone Types and Status Colors**

- Not Specified -- White
- No Outstanding Issues -- Green
- Manageable Issues -- Yellow
- Critical Issues -- Red
- Not Known -- White
- Not Required -- Black

You may change these colors, and add new ones, by adjusting the metamodel via usrprops.txt. You may specify up to a maximum of eight statuses. For the out-of-box property set for this
Filtering the Milestones Appearing on a Diagram

You are allowed to place Projects that have a disparate set of Milestones on a diagram. If you want to view these projects in terms of their common Milestones, toggle on the Diagram property named "Milestones filtered by topmost project, phase colors come from settings in topmost project." To do this, perform the following steps:

1. Open the diagram’s property dialog (either select Edit, Diagram Properties with the PV-2 diagram open, or right-mouse click on the diagram workspace and select Diagram Properties).
2. In the Behavior tab, toggle on the following setting: Milestones filtered by topmost project, phase colors come from settings in topmost project.

With this setting turned on, only the Milestones that exist in the top-most Project on the diagram will be displayed within all of the Projects on the diagram. Also, the in-between-Milestone phase colors for all Projects will also abide by what is set in the top-most Project.

Specifically, with this property set, all Milestones take their Following Color value from the common Milestones on the top-most Project. For example, let’s say you have two Projects on a diagram – Project_1 and Project_2, where Project_1 is the top-most project. Each Project has a Milestone named Stage_1. The Following Color property of Stage_1 Milestone on Project_1 is set to red, and the Following Color property of Stage_1 Milestone on Project_2 is set to blue. If you toggle this property on, then on the diagram, the color of the Project line after the Stage_1 Milestone on Acquisition Project_2 will be red (not blue).

Also, with this property set, all Projects take their Pre-First Milestone Color from the top-most project. So if the Pre-First Milestone Color property for Project_1 is green, and the Pre-First Milestone Color property for Project_2 is black, then on the diagram both Project lines will have a green color before their Stage_1 Milestone.

Setting Pink as the Default Color for the Pre-First Milestone

You may set the color of the Project timeline appearing before the first milestone to default to pink.

1. Open the diagram’s property dialog (either select Edit, Diagram Properties with the AcV-2 diagram open, or right-mouse click on the diagram workspace and select Diagram Properties).
2. In the Behavior tab, toggle on the following setting: Use pink as the default only for the pre-first milestone color when not filtering.

Toggling this option on sets pink as the color of the Acquisition Project line before the first milestone, unless you have set a color in the Pre-first Milestone Color property in the Acquisition Project definition. This toggle choice is overridden if you have toggled on the Milestones filtered by topmost project... property.

Adding a Key Symbol (or Legend) to the Diagram

You may select the Key tool from the toolbar and place one on a PV-2 diagram. The Key symbol provides the Legend describing the color coding used for Milestones and Phases. The Key (or
Legend) is automatically based on all milestones in all projects on the diagram. Color codes for Milestone Project Thread Status can only be changed by altering the metamodel via usrprops.txt.

**Milestone Dependency Line**

You can use the Milestone Dependency line to indicate the direction of the flow of data as it moves from one point in the system to another.

**Note:** You may attach a diagram to a Milestone symbol as a 'child' diagram through normal Rational System Architect procedures. The child-diagram indicator is placed over the name of the Milestone (not to the upper left-hand corner as is normally done). This is because the name of a Milestone is forced to be outside the Milestone pie-chart symbol; it cannot be made to be inside the symbol.

**Timeline**

A Timeline symbol is automatically drawn once a PV-2 diagram has been created. The Timeline indicates the period over which the Project development is shown. The following properties apply:

- Start Date
- End Date
- Interval Number
- Interval Units displayed as: Day, Month, Quarter, Week, and Year.

You may edit the Timeline through normal definition editing techniques -- double-clicking on the Timeline symbol on the diagram, right-mouse clicking on the symbol and choosing Edit Timeline, or selecting the symbol and choosing Edit, Edit Timeline from the main menu.

**PV-3**

The PV-3 is a report that generates all Projects, a description of each Project, and the Capabilities that it enables.

The PV-3 report is run by doing the following:

1. Select **Reports, Report Generator**. The Reports dialog will open. You need to be in the **DAF2.rpt** file -- the top of the dialog specifies what report file you are in.
2. If you are not in DAF2.rpt, do the following -- select File, Open Report File, and in the Report File Selection dialog, select DAF2.rpt.

3. In the Reports dialog, select PV-3 By Definitions -- Projects and Capabilities and press Generate (or simply double click on the report).

You may of course click Edit and adjust the out-of-box report, by adding additional properties for to publish for a Project. Below is a custom PV-3 report, which shows Projects, their Milestones, and the Capabilities enabled at each Milestone -- in our example, a new shower is added to a bathroom renovation to enable the capability "Take Shower".

```
<table>
<thead>
<tr>
<th>Project Name</th>
<th>Project Milestones</th>
<th>Capability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acquire New Sink and Faucet</td>
<td>K</td>
<td></td>
</tr>
<tr>
<td></td>
<td>M</td>
<td></td>
</tr>
<tr>
<td></td>
<td>E</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FOC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>OOS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Disposal</td>
<td></td>
</tr>
<tr>
<td>Model Project</td>
<td>K</td>
<td></td>
</tr>
<tr>
<td></td>
<td>M</td>
<td></td>
</tr>
<tr>
<td></td>
<td>E</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FOC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>OOS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Disposal</td>
<td></td>
</tr>
<tr>
<td>Renovate Bathroom</td>
<td>K</td>
<td></td>
</tr>
<tr>
<td></td>
<td>M</td>
<td></td>
</tr>
<tr>
<td></td>
<td>E</td>
<td></td>
</tr>
<tr>
<td></td>
<td>FOC</td>
<td></td>
</tr>
<tr>
<td></td>
<td>OOS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Disposal</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Take Shower</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Wash Hands</td>
<td></td>
</tr>
</tbody>
</table>
```

The report above utilizes the "Partial" command at the upper-most layer -- in the Properties to Print for Project, the Partial command was toggled on, so that all Milestones for every Project are output to the report, and then the Capabilities enabled at each Milestone of a Project.
CV-03 -- Generating the Capability Phasing Report

The CV-03 shows the planned achievement of capability at different points in time or during specific periods of time. The CV-3 shows the capability phasing as activities, conditions, desired effects, rules complied with, resource consumption, production, and measures. It does so without regard to the performer and location solutions.

The CV-3 report can be used to assist in the identification of:
- capability shortfalls (no fielded capability to fulfill a particular function) or
- capability duplication (multiple fielded capabilities for a single function).

The CV-03 report, output to Excel, provides a roadmap of Capabilities that are fielded over time, and the name of the Project Milestone at which the Capability is fielded. It also colors the cell containing the Project Milestone with the color that signifies that phase of the Project -- the color is specified on the Phase tab of the Project Milestone’s definition, in the Following Color property. An empty cell signifies that a Project Milestone is not associated with a Capability.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Take Pictures</td>
<td>Project1</td>
<td>IS</td>
<td>IS</td>
<td>IS</td>
<td>IS</td>
<td>IS</td>
<td>IS</td>
<td>IS</td>
<td>IS</td>
<td>IS</td>
<td>IS</td>
<td>IS</td>
<td>Disposal</td>
</tr>
</tbody>
</table>

The example CV-03 report above was generated from the PV-2 data shown below:

In Project1, each milestone is associated with the Capability Take Pictures, except for the ‘Out of Service’ milestone, which causes that milestone to not be generated to the CV-03 report, causing a gap to appear.

In such a scenario:
- No sorting is applied to milestones, if more than one milestone is placed in an interval cell.
- All milestones which match, or are later than a time interval and before the following time interval, are listed within that interval.
- The cell color is set to the following color of the first milestone that matches the interval (if more than one milestone is placed in a time interval).
- Gaps are shown where a project milestone associated with an interval are not assigned to the capability.

**Before you begin**

To be able to generate the CV-3 report, you must first relate one or more Capabilities to the Project Milestone(s) each is achieved at, as follows:
1. Create one or more Capability (DM2) definitions.

2. Create one or more Projects with Milestones -- this is done by creating one or more Project (DM2) definitions in the Explorer (browser) or by modeling them on PV-01 or PV-02 diagrams.

3. Relate a Capability to the Project Milestone(s) it is achieved at -- on the CV-03 tab of the Capability (DM2) definition, click on the Choices button (see Important Note below) and drag-and-drop the Project Milestone(s) into the **Capability Increments** grid. You are saying that at this point in the Project, this Capability becomes available.

   - **Important Note:** In the Select and Drag dialog for Project Milestones, right-mouse click and select **Details**. You will see a summary of each Milestone that you select in the dialog, and can view its Project at the top of the Details dialog.

4. If you wish to change the Milestone Date while in the Capability definition, select the Capability in the **Capability Increment** grid, and click the Define button (you cannot change the date in the read-only grid).

**Procedure**

In Rational System Architect, you generate the CV-03 as a report, as follows:

1. Click the **Reports** menu and select **DoDAF2 Reports> CV-03 Capability Phasing Report**.

2. Specify the **Start date, End date, Timescale units, and Interval**.

3. Click **OK**. The report is automatically generated to Microsoft Excel.

   **Tip:** If the report is empty ensure that **Project Milestones** are assigned to the capabilities. See Before You Begin section above.

**Results**

The Capability Phasing Report opens in a Microsoft Excel Spread sheet. The report contains the name of each Resource related to a Capability in the appropriate cells of the spreadsheet, per each Milestone Date of its related Acquisition Project. Each cell is colored with the Following Color property of each milestone.
## More Information
For more information on DoDAF 2 and System Architect, please visit the following resources:

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>System Architect Train on Youtube</td>
<td><a href="https://www.youtube.com/user/SystemArchitectTrain">https://www.youtube.com/user/SystemArchitectTrain</a></td>
</tr>
<tr>
<td></td>
<td>System Architect on Twitter</td>
<td><a href="https://twitter.com/unicorn_sa">https://twitter.com/unicorn_sa</a></td>
</tr>
<tr>
<td></td>
<td>System Architect on Pinterest</td>
<td><a href="https://www.pinterest.com/systemarchitect/">https://www.pinterest.com/systemarchitect/</a></td>
</tr>
</tbody>
</table>
## Appendix A -- List of Preloaded Measurement Sets, Types, and Values

Every time a System Architect encyclopedia is created, it is preloaded with a set of Measurement Sets, Types, and Values, as specified in the table below.

In order to adjust the list of preloaded definitions for Measurement Set, Measurement Type, and Measurement Value, you adjust the content in the csv files in the CSV Import\DoDAF2 directory of System Architect’s program directory (usually `<C>:\Program Files (x86)\IBM\Rational\System Architect Suite\System Architect\CSV Import\DoDAF2`, as follows:
- 1457.csv -- contains preloaded Measurement Sets
- 1458.csv -- contains preloaded Measurement Types
- 1459.csv -- contains preloaded Measurement Values

<table>
<thead>
<tr>
<th>Measurement Set</th>
<th>Measurement Type</th>
<th>Measurement Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CommunicationsLinkProps</td>
<td>capacity</td>
<td></td>
</tr>
<tr>
<td></td>
<td>infrastructure</td>
<td>Technology</td>
</tr>
<tr>
<td>DataElementProps</td>
<td>accuracy</td>
<td></td>
</tr>
<tr>
<td></td>
<td>content</td>
<td></td>
</tr>
<tr>
<td></td>
<td>formatType</td>
<td></td>
</tr>
<tr>
<td></td>
<td>mediaType</td>
<td></td>
</tr>
<tr>
<td></td>
<td>scope</td>
<td></td>
</tr>
<tr>
<td></td>
<td>unitOfMeasurement</td>
<td></td>
</tr>
<tr>
<td>ExchangeProps</td>
<td>Classification</td>
<td>C: Confidential</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CTS: Cosmic Top Secret</td>
</tr>
</tbody>
</table>
|                          |                  | CTS-B: Cosmic Top Secret-
<pre><code>                      |                  | Bohemia                |
</code></pre>
<p>|                          |                  | CTS-BALK: Cosmic Top    |
|                          |                  | Secret - Balk           |
|                          |                  | CTSA: Cosmic Top Secret |
|                          |                  | Atomal                  |
|                          |                  | NC: NATO Confidential   |
|                          |                  | NCA: NATO Confidential  |
|                          |                  | Atomal                  |
|                          |                  | NR: NATO Restricted     |
|                          |                  | NS: NATO Secret         |
|                          |                  | NS-A: NATO Secret --    |
|                          |                  | Atomal                  |
|                          |                  | NSAT: NATO Secret Atomal|
|                          |                  | NS-S: NATO Secret       |
|                          |                  | NU: NATO Unclassified   |
|                          |                  | R: Restricted Data (US  |
|                          |                  | Nuclear Information or  |
|                          |                  | For Official Use Only)  |
|                          |                  | S: Secret               |
|                          |                  | TS: Top Secret          |</p>
<table>
<thead>
<tr>
<th>InformationAssuranceProperties</th>
<th>accessControl</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>availability</td>
</tr>
<tr>
<td></td>
<td>confidentiality</td>
</tr>
<tr>
<td></td>
<td>disseminationControl</td>
</tr>
<tr>
<td></td>
<td>integrity</td>
</tr>
<tr>
<td></td>
<td>nonRepudiationProducer</td>
</tr>
<tr>
<td></td>
<td>nonRepudiationConsumer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>InformationElementProperties</th>
<th>accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>content</td>
</tr>
<tr>
<td></td>
<td>language</td>
</tr>
<tr>
<td></td>
<td>scope</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OperationalActivityProperties</th>
<th>cost</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>SecurityAttributes</th>
<th>Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>C: Confidential</td>
</tr>
<tr>
<td></td>
<td>CTS: Cosmic Top Secret</td>
</tr>
<tr>
<td></td>
<td>CTS-B: Cosmic Top Secret - Bohemia</td>
</tr>
<tr>
<td></td>
<td>CTS-BALK: Cosmic Top Secret - Balk</td>
</tr>
<tr>
<td></td>
<td>CTSA: Cosmic Top Secret Atomal</td>
</tr>
<tr>
<td></td>
<td>NC: NATO Confidential</td>
</tr>
<tr>
<td></td>
<td>NCA: NATO Confidential Atomal</td>
</tr>
<tr>
<td></td>
<td>NR: NATO Restricted</td>
</tr>
<tr>
<td></td>
<td>NS: NATO Secret</td>
</tr>
<tr>
<td></td>
<td>NS-A: NATO Secret -- Atomal</td>
</tr>
<tr>
<td></td>
<td>NSAT: NATO Secret Atomal</td>
</tr>
<tr>
<td></td>
<td>NS-S: NATO Secret</td>
</tr>
<tr>
<td></td>
<td>NU: NATO Unclassified</td>
</tr>
<tr>
<td></td>
<td>R: Restricted Data (US Nuclear Information or For Official Use Only)</td>
</tr>
<tr>
<td></td>
<td>S: Secret</td>
</tr>
<tr>
<td></td>
<td>TS: Top Secret</td>
</tr>
<tr>
<td></td>
<td>U: Unclassified</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DeclassManualReview</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>DisseminationControls</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>FGIsourceOpen</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>FGIsourceProtected</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>SARIIdentifier</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>SCIControls</th>
</tr>
</thead>
<tbody>
<tr>
<td>Term</td>
</tr>
<tr>
<td>---------------------------</td>
</tr>
<tr>
<td>classificationReview</td>
</tr>
<tr>
<td>classifiedBy</td>
</tr>
<tr>
<td>dateOfExemptedSource</td>
</tr>
<tr>
<td>declassDate</td>
</tr>
<tr>
<td>declassEvent</td>
</tr>
<tr>
<td>declassException</td>
</tr>
<tr>
<td>derivedFrom</td>
</tr>
<tr>
<td>nonICmarkings</td>
</tr>
<tr>
<td>ownerProducer</td>
</tr>
<tr>
<td>releasableTo</td>
</tr>
<tr>
<td>typeOfExemptedSource</td>
</tr>
</tbody>
</table>