# Tailoring DODAF For *Service-Oriented Architectures*

**Huei-Wan Ang, Fatma Dandashi, Michael McFarren**
The Mitre Corporation
hwang(at)mitre.org, dandashi(at)mitre.org, mmcfarren(at)mitre.org

Abstract[1][2]

This paper contains a brief overview of Service-Oriented Architecture (SOA) and an explanation of how the Department of Defense Architecture Framework (DoDAF) can be used to describe a SOA. DoDAF uses IEEE 1472 definition of an architecture description to define a standard approach to describing, presenting, and integrating a DoD architecture that can be used with a service oriented approach to capability based planning. The principal objective of the Framework is to ensure that architecture descriptions can be compared and related across organizational boundaries, including Joint and multi-national boundaries. SOA is an architectural approach to application integration that enables flexible connectivity of applications or resources implemented as services. Such services have well-defined, platform-independent interfaces that hide the underlying technical complexity of the environment (encapsulation), are self-contained (loosely coupled), and reusable. Capability based planning involves identifying required capabilities, their desired effects, and the ways (operational activities) and means (human functions or system services), as well as the conditions ands standards under which the capability is required. Creating DoDAF architecture descriptions that are capability based and service oriented supports globalization and the integration of geographically dispersed organizations (Net-centricity).

## Introduction

A Service Oriented Architecture (SOA) approach is one where application design and development is based on the concept of services. The principal objective of the DoD Architecture Framework (DoDAF) [DODAF] is to ensure that architecture descriptions can be compared and related across organizational boundaries, by defining a particular set of architectural elements and relationships used for describing architectures. This difference allows DoDAF to effectively describe a SOA. However, some tailoring is required to better support SOA design patterns within DoDAF. In this paper, we describe an approach to Capability based planning through the use of reusable and composable services[3] and describing the resulting SOA with tailored DoDAF products. Our premise is that using DoDAF to describe a SOA enables leveraging the existing body of knowledge and architecture artifacts within DOD. Further, tailoring DoDAF for SOA enables architects to more effectively describe a SOA as an alignment of services to operational activities thus facilitating capability analysis by tracing services through the operational activities to the capabilities they support.

## What is a Service?

The term *service* has been defined by several industry and standards consortia such as IBM, OASIS, etc., with slightly different variations. In general, the current body of knowledge is consistent in applying the term to a certain kind of **software** application with the following characteristics. A service provides well defined, self-contained functionality that is loosely coupled from other functionality/services. The functionality is well encapsulated (i.e., complexity of the implementation is hidden from potential consumers except for the information required by the consumers to determine whether a given service is appropriate for their needs; that information is exposed in the service interface [OASIS]). The semantics of a service should be documented, either directly or indirectly, by its description (also referred to as a service's standard interface(s) or set of messages [W3C]). Services have course granularity, they tend to use a small number of operations with relatively large and complex messages which are exchanged between the provider and consumers. A service is location transparent (i.e., consumers do not need to be aware of the physical location of a hosting server); protocol independent (messages are sent in a platform-neutral, standardized format delivered through the interface); and stateless (i.e., the service remains in the same state after each execution request from a consumer). Services tend to be oriented toward use over a network, though this is not an absolute requirement. [W3C]

## What is SOA?

SOA is a form of distributed systems architecture based on services (as defined above) where a consumer does not

[1] This work was partially supported by the US Space Situation Awareness Integration Office and the AF Warfighting Integration and Chief Information Office

[2] With contributions from David Nicholson of Mitre

[3] While the scope of this paper is software services, the DoDAF tailored products and underlying metamodel are equally applicable to human functions (e.g., as in outsourcing business services)

need to know the internal structure of a provider, including features such as its implementation language, process structure, and even database structure [W3C]. In SOA, the focus is on the sequence of operational activities or business process. The operational process is then mapped to a systems architecture description with specific applications that support the operational process cast as services. Thus, the architecture supports a business process via a set of independent, reusable, but collaborative services. The service integration happens dynamically, via service composition (the execution of several of these independent services in an orchestrated manner). SOA is an "architectural discipline that centers on the notion that IT assets are described and exposed as Services. These Services can then be composed in a loosely-coupled fashion into higher-level business processes,…" [ZapThink]

### Advantages of SOA

SOA offers several advantages. 1) An operational process orchestrates simple services (owned by the process) into complex services. 2) Operational performance can be improved through closed feedback loops from runtime behavior back to the operational process. 3) Services allow the exchange of information and data between: a) different computers, from different vendors, b) different programs, from different functional areas, or different members of a Community of Interest (CoI), and 4) SOA supports globalization and the integration of geographically dispersed organizations (net centricity) through service orchestration of distributed services owned and executed across ownership boundaries. [IBM]

### Space Weather Impact Analysis Sample Scenario

In this paper, tailored DoDAF products supporting the following scenario are provided as an example. Space weather information from the Space Weather Service is required to provide Space Situation Awareness (SSA) for Space Command and Control (C2). Space weather information is also required to support theater operation planning and commercial satellite launch planning. A Space C2 Operator uses the Space C2 Application to view a User Defined Operational Picture (UDOP). In providing the UDOP display, the Space C2 Application uses the previously defined UDOP criteria to request the Space UDOP from the Space UDOP Service, which in turn sends a request to the Space Weather Service to provide a space weather impact report for a specified period of time. The Space Weather Service queries the Weather DataBase (DB). The WeatherDB returns the queried space weather impact report present at that moment in the database. If the space weather information requested is not in the database, the Space Weather Service sends a request to the Space Weather Impact Analyzer to make and provide a space weather impact prediction. The Space Weather Service returns a space weather impact report as the response to the Space UDOP Service, which integrates the report with all other SSA information and returns the requested Space UDOP to the Space C2 Application. The Space C2 Application displays the Space UDOP. The example architecture provided here was developed using UML 2.0[4]. Due to paper length limits, only three key products are described to illustrate the tailoring. The example provided does not represent a complete architecture model and each example product does not necessarily include the entire potential content of a product, but includes enough detail to properly demonstrate the tailoring of a product. In addition, the content of the example is based on information provided by the Space Situation Awareness Integration Office (SSAIO) but does not represent actual or planned operational processes or systems from Space Command.

## Tailoring DoDAF

The first step toward reliable, mature practice in any discipline is the definition of the fundamental vocabulary, semantics, and models upon which the practice is built and shared. The vocabulary used in this paper is based on an internal Mitre effort to evolve DoDAF and to define a clear delineation between the requirements submodel: the Operational or Resource View, and a solution submodel: the Systems View which is further divided into two subviews: the Automated System and the Human subviews. We define an *Operational Resource* as an aggregation of *Humans* (i.e., humans) and *Automated Systems* (i.e., machines). That is, we define a system as possibly being an organic or a physical (hardware and software) system.

**All Views (AV):** There are some overarching aspects of an architecture description that relate to all three views. These overarching aspects are captured in the AV products. Two products are defined for AV, Overview and Summary Information (AV-1) and Integrated Dictionary (AV-2). These two products do not need to be tailored.

**Operational View (OV)**: OV describes the capabilities, operational activities, and information exchanges required to conduct operations. A pure OV is materiel independent and is used to describe capability requirements. No tailoring of the OV is required to support SOA. However, the OV is important to SOA because it describes the required capabilities and justification for application

[4] The example was developed using IBM/Rational's Software Architect

investment, and identifies specific operational activities that can be automated via services in a SOA. One of the promises of SOA is that it allows for better alignment of applications with operational processes. In order to accomplish this alignment, SOA requires that the operational processes be both well defined, and defined at a granular enough level to be able to map directly to the services in the Systems View. One of the tenets of DoDAF is to rigorously define the OV products to better enable this alignment. In addition, some tailoring is required to support a capability-based approach. In this paper, a tailored OV-5 matrix for identifying capabilities, its desired *effects*, and *ways* (operational activities) to achieve those effects is provided in Table 1.

**Table 1: Tailored OV-5 Matrix-Allocation Of Operational Activities To Required Capabilities And Their Desired Effects**

| Capability | Effect | Ways |
|---|---|---|
| Support Global And Theater Commander | Commander Objectives Supported | -Provide Satellite Operations Support<br>-Provide Theater Operations Support<br>-Warn Space Anomaly<br>-C2 Of Space Forces |
| Protect Space Assets | Hostile Space Activities Negated | -C2 Of Space Forces |
| Support Commercial Satellite Operation Planning And Execution | Operation Conducted Successfully | -Provide Satellite Operations Support<br>-Warn Space Anomaly |
| Enable Protection Of Space Assets And Support Of Military And Commercial Operations | Obtained Complete Space Situation Awareness (SSA) | -Monitor Space<br>-Analyze SSA<br>-Integrate SSA |

**Systems View (SV)**: SV consists of a set of graphical and textual products that describe human functions and automated system processes (services) and their interconnections in support of DoD operational activities. The relationship between architecture data elements across the SVs to the OVs can be described as follows: humans and automated systems, or human functions and services, are grouped into nodes and fielded to provide capabilities described as OV requirements and to execute operational activities. Some tailoring of the SV is required to support SOA. In addition, some SV tailoring is needed to document allocation of means, conditions, and standards to capabilities. Means are defined in the SV as humans and human functions, automated systems and machine processes (services). *Conditions, and Standards are defined as* Human and automated system performance requirements. Because of this paper's emphasis on SOA, our example SV products do not all include the human view.

**SV-1 Systems Interface Description**: SV-1 shows where services are desired by describing logical or physical nodes that group services and human functions. Services and human functions may be logically grouped by where they are advertised (e.g., all services published in a particular registry) or by some broad functional category (warfighting, logistics, financial management, HR, medical, etc.); alternatively, services and human functions can be physically grouped, for example all services and/or and human functions that reside on an aircraft carrier. SV-1 also shows logical communication channels between the nodes, systems, or humans. Figure 1 is an SV-1 that shows automated systems nodes (the outermost box, including those denoted as *External*), automated systems, servers and/or applications that provide or consume certain services. Inter-nodal communication channels shown here reflect the fact that the services interact across nodes. A provided service is denoted by a part with a port that binds to a defined interface (the service specification) from SV-4. The provided services shown grouped into nodes are the same ones that are defined in SV-4 diagrams. A consumer port denotes that the consumer (system or application) requires the use of a service through its defined interface. A consumer port is shown on SV-1 applications or services to indicate they consume services provided by others. To reduce SV-1 clutter, not all consumer ports are shown on Figure 1.

**SV-3 Systems-Systems Matrix**

The Systems-Systems Matrix provides detail on the automated system interface characteristics described in SV-1 for the architecture. The product may be tailored to include a matrix that shows the dependency between users of services (as rows) and the used services (columns). Table 2 is an example of such a matrix showing the list of applications as users of services and the services on which they depend.

**Table 2: Tailored SV-3 Showing Application to Service Dependencies**

| Apps | iLocating Service | iOverflight Service | iSpace UDOP Service | iSpace Weather Service |
|---|---|---|---|---|
| Combat Support | X | X | | X |
| Space C2 | X | X | X | X |
| Launch Planning | X | | | X |

The product can also be tailored to include another matrix that shows the dependency from services to other services or system functions. Table 3 is an example of such matrix showing service dependencies. Note the content of both matrices can be generated from information captured in SV-4.

**Table 3: Tailored SV-3 Showing Service Dependencies**

| | iRegistry | iRSO Catalog | iGreat Circle Route | Space Weather DB | Weather Impact Analyzer | Navy Sensor | AF Sensor |
|---|---|---|---|---|---|---|---|
| Overflight Service | X | X | X | | | | |
| Space UDOP Service | X | X | | | | | |
| Space Weather Service | X | X | | X | X | X | X |
| RSO Catalog Service | X | | | | | X | X |

**SV-4 Systems Functionality Description**: The Systems Functionality Description documents specifications of human functions, services, and their known product (data or material) flows, human and machine interactions. While SV-1 shows providers and consumers of services and how they might be structurally designed, SV-4 for SOA can be tailored to show the service interface (or the service description/specification), the service realization, and dependencies as demonstrated in Figure 2. Service interfaces in SV-4 (denoted in UML using the Circle notation) are the same ones shown as ports in SV-1 through which consumers utilize the service (the interface is the exposed part of the service). The service realization is denoted in SV-4 using UML class notation. Operations defined on the service class (also show up as activities in SV-10c) realize the functionality described in the service interface. Service consumers (not always services themselves, e.g., the Space Launch Planning Application) are also shown in both SV-1 and SV-4.

**SV-5 Operational Activity to Systems Function Traceability Matrix**

SV-5 is a traceability matrix that identifies the relationships between the set of operational activities applicable to an architecture description and the set of services that apply to that same architecture description. SV-5 shows the traceability of requirements from OVs through their decomposition and allocation to SVs that satisfy required capabilities. It provides a summary of those relationships in one easy-to-examine format. SV-5 may be generated from architecture elements and relationships established in other products. Table 4 is an SV-5 that shows the mapping of capabilities and operational activities to corresponding automated systems and the services they provide.

**Table 4: Tailored SV-5 Showing Operational Activities and Corresponding Human Functions and/or Service**

| *Capability:* | Enable Protection Of Space Assets And Support Of Military And Commercial Operations | | Support Global And Theater Commander |
|---|---|---|---|
| Op Activity: | Monitor Space | Monitor Space Force Operation | Provide Theater Operations Support |
| System1: Space Weather Server | Space Weather Service | | Space Weather Service |
| System2: SSA Server | RSO Catalog Service | Space UDOP Service | Overflight Service |

**SV-6 Systems Data Exchange Matrix**

SV-6 is an automated system Data Exchange Matrix that documents data exchanges between two pairs of functions and can be tailored as a Services Data Exchange Matrix that may be generated from the services and their dependencies as described in SV-4 and SV-10c. A partial example SV-6 is provided below. However, this example does not include all attributes that can be used to describe the characteristics of a service data exchange.

**Table 5: Tailored SV-6 Showing Services Data Exchanges and Their Attributes**

| Service Name | Service User | Data Required | Data Produced | Transaction Type |
|---|---|---|---|---|
| Space Weather Service | Combat Support Application | | Space Weather Report | Broadcast |
| Space Weather Service | Combat Support Application | Date-Time Range<br>Location | Space Weather Impact Report | Request/ Respond |
| Space UDOP Service | Space C2 Application | UDOP criteria | UDOP | Publish/ Subscribe |

**SV-7 Systems Performance Parameters Matrix**

The Systems Performance Parameters Matrix specifies performance characteristics of systems and hardware/software items. SV-7 is important to SOA because it can be tailored to capture the performance requirements of services identified in service level agreements (SLAs), and allows for a comparison between specified performance thresholds and actual performance levels achieved by service implementations during testing. SV-7 is an essential product for describing SOA. A partial example SV-7 is provided below.

**Table 6: Tailored SV-7 Showing A List Of Services And The Quality Of Service Supported By The Provider**

| Service | Quality Metric | SLA | | Service Quality |
|---|---|---|---|---|
| | | Threshold | Objective | |
| Space Great Circle Route Service | Availability | 99% | 100% | 99.5% |
| Space Great Circle Route Service | Response Time | .001 second | .0001 second | .002 second |
| Aviation Great Circle Route Service | Availability | 95% | 100% | 97% |
| Aviation Great Circle Route Service | Response Time | 2 second | .1 second | 1.2 second |

**SV-10 Systems Functionality Sequence and Timing Descriptions**

The SV-10 generally mirrors the OV-6, and describes the dynamic behavior of humans and systems, and of human functions and services. The Systems Event-Trace Description (SV-10c) describes the sequence (control) flows for human functions and services. With SOA, these service sequence flows can be used to specify service interactions, or they can be used to specify the sequence of flows between human functions, application services, and infrastructure services. Figure 3 is a UML Activity diagram (like a flowchart with swimlanes) that details the process flow among interacting services. The swimlanes correspond to the service realizations (classes) that were defined in Figure 2. Activities appearing in the swimlanes correspond to operations defined on these classes.

Figure 4 illustrates a particular scenario of the process flow illustrated by Figure 3. It includes the planner (Space C2 Operator) who sends a request for space UDOP to the SSA server. The messages that are exchanged between the various entities correspond to the service interface operations that were illustrated in Figure 2.

The SV products provided in this paper collectively demonstrate how a service-oriented systems architecture may be described with limited tailoring to DODAF SV products. The example provided here was developed using UML because this modeling language defines constructs that are semantically rich enough to describe a SOA using DODAF.

In addition, some tailoring to facilitate capability based analysis, through the introduction of explicit modeling of the human (skills) and human functions required to support required capabilities (with or without a materiel solution) was also briefly described.

**The Technical View** provides the technical implementation standards upon which engineering specifications are based, and common building blocks are established. The Technical Standards Profile collects the various standards rules that implement and sometimes constrain the choices that can be made in the design and implementation of an architecture description.

The Technical View does not require tailoring. It is used to capture technical standards that are utilized (or are to be utilized) in specifying service interfaces/descriptions (e.g., WSDL, OWL), service messaging protocols, service implementation languages, etc.

## References

[DODAF] Department of Defense Architecture Framework v1.0

[IBM] “Patterns: Service-Oriented Architectures and Web Services,” http://www.redbooks.ibm.com/abstracts/sg246303.html?Open

[OASIS] Organization for the Advancement of Structured Information Standards, http://www.oasis-open.org/home/index.php

[OMG] Open Management Group, SOA SIG, http://SOA.omg.org

[OpenGroup] The Open Group, http://www.opengroup.org/

[W3C] World Wide Web Consortium, http://www.w3.org/

[ZapThink] http://www.zapthink.com/

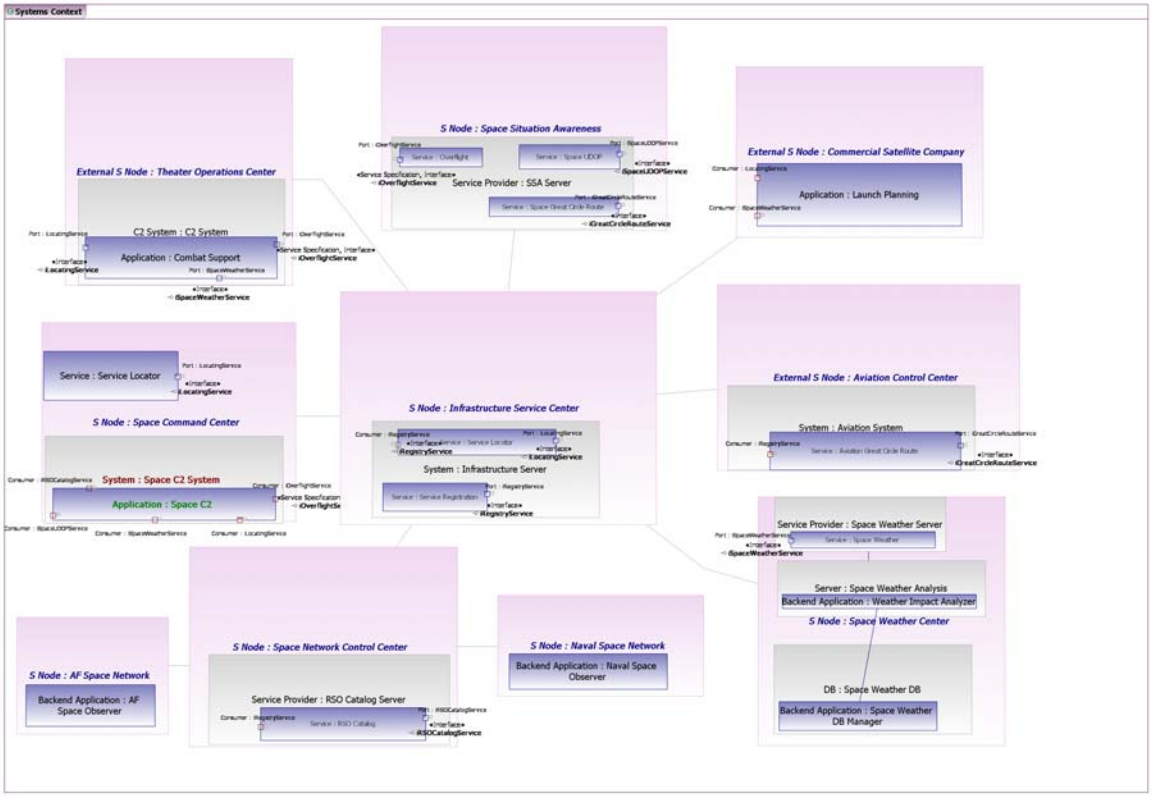


**Figure 1: SV-1 Showing Groupings Of Services By Nodes**

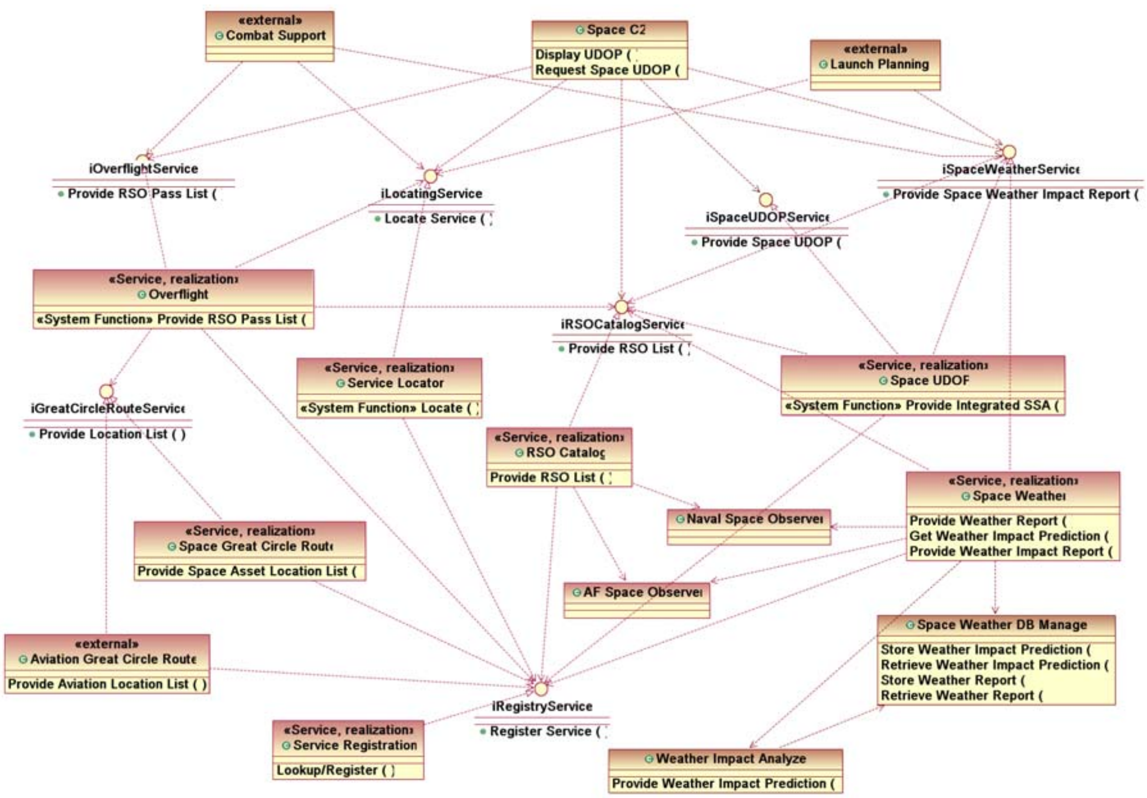


**Figure 2: SV-4 Showing Service Dependencies**

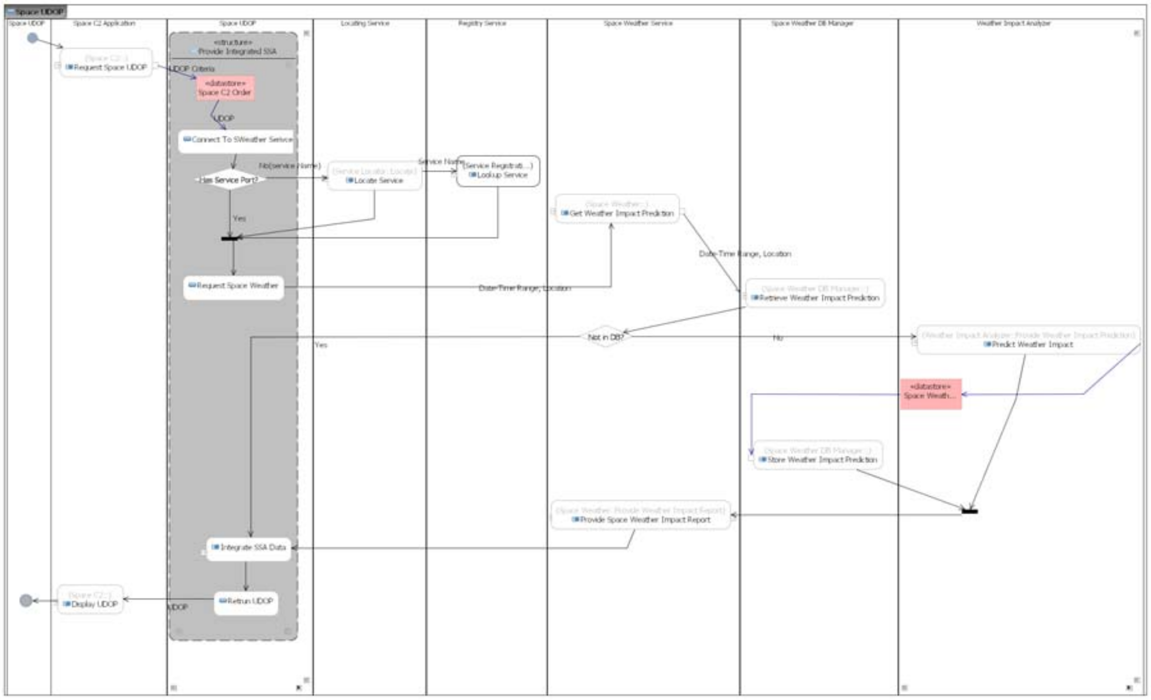

**Figure 3: SV-10c Showing A Process Flow with An Activity Diagram**

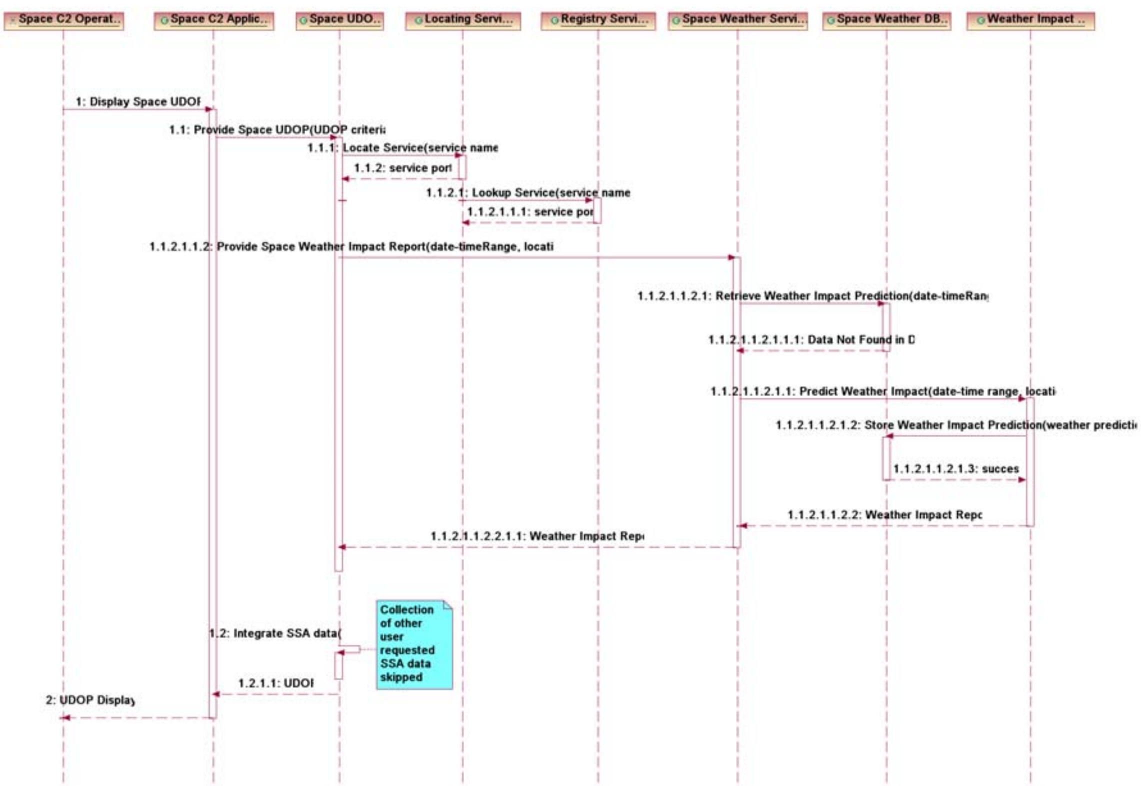


**Figure 4: SV-10c Showing A Scenario with A Sequence Diagram**