**Date:** December 2021

OBJECT MANAGEMENT GROUP®

# Tactical Decision Aids Interface (TDAI)

## V1.0 – beta 1

---

**OMG Document Number:** dtc/22-01-02

**Normative reference:** https://www.omg.org/spec/TDAI/1.0/

**Machine readable file(s):** https://www.omg.org/spec/TDAI/20211101

**Associated Normative Machine Consumable Files:**

https://www.omg.org/spec/tdai/20211101/tdai.xmi/
https://www.omg.org/spec/TDAI/1.0/tdai.graphqls/
https://www.omg.org/spec/TDAI/1.0/tex.graphqls/
https://www.omg.org/spec/TDAI/20211101/IDL/DataSink.idl
https://www.omg.org/spec/TDAI/20211101/IDL/PlanExecution.idl
https://www.omg.org/spec/TDAI/20211101/IDL/PlanExecutionRecommendations.idl
https://www.omg.org/spec/TDAI/20211101/IDL/Recommendation.idl
https://www.omg.org/spec/TDAI/20211101/IDL/Recommendations.idl
https://www.omg.org/spec/TDAI/20211101/IDL/TacticalPicture.idl
https://www.omg.org/spec/TDAI/20211101/IDL/TacticalPictureRecommendations.idl
https://www.omg.org/spec/TDAI/20211101/IDL/Utils.idl

---

This OMG document replaces the submission document (c4i/21-11-06). It is an OMG Adopted Beta Specification and is currently in the finalization phase. Comments on the content of this document are welcome and should be directed to issues@omg.org by July 31, 2022.

You may view the pending issues for this specification from the OMG revision issues web page https://issues.omg.org/issues/lists.

The FTF Recommendation and Report for this specification will be published in December 2022. If you are reading this after that date, please download the available specification from the OMG Specifications Catalog.

## USE OF SPECIFICATION - TERMS, CONDITIONS & NOTICES

The material in this document details an Object Management Group specification in accordance with the terms, conditions and notices set forth below. This document does not represent a commitment to implement any portion of this specification in any company's products. The information contained in this document is subject to change without notice.

## LICENSES

The companies listed above have granted to the Object Management Group, Inc. (OMG) a nonexclusive, royalty-free, paid up, worldwide license to copy and distribute this document and to modify this document and distribute copies of the modified version. Each of the copyright holders listed above has agreed that no person shall be deemed to have infringed the copyright in the included material of any such copyright holder by reason of having used the specification set forth herein or having conformed any computer software to the specification.

Subject to all of the terms and conditions below, the owners of the copyright in this specification hereby grant you a fully-paid up, non-exclusive, nontransferable, perpetual, worldwide license (without the right to sublicense), to use this specification to create and distribute software and special purpose specifications that are based upon this specification, and to use, copy, and distribute this specification as provided under the Copyright Act; provided that: (1) both the copyright notice identified above and this permission notice appear on any copies of this specification; (2) the use of the specifications is for informational purposes and will not be copied or posted on any network computer or broadcast in any media and will not be otherwise resold or transferred for commercial purposes; and (3) no modifications are made to this specification. This limited permission automatically terminates without notice if you breach any of these terms or conditions. Upon termination, you will destroy immediately any copies of the specifications in your possession or control.

## PATENTS

The attention of adopters is directed to the possibility that compliance with or adoption of OMG specifications may require use of an invention covered by patent rights. OMG shall not be responsible for identifying patents for which a license may be required by any OMG specification, or for conducting legal inquiries into the legal validity or scope of those patents that are brought to its attention. OMG specifications are prospective and advisory only. Prospective users are responsible for protecting themselves against liability for infringement of patents.

## GENERAL USE RESTRICTIONS

Any unauthorized use of this specification may violate copyright laws, trademark laws, and communications regulations and statutes. This document contains information which is protected by copyright. All Rights Reserved. No part of this work covered by copyright herein may be reproduced or used in any form or by any means--graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems--without permission of the copyright owner.

## DISCLAIMER OF WARRANTY

WHILE THIS PUBLICATION IS BELIEVED TO BE ACCURATE, IT IS PROVIDED "AS IS" AND MAY CONTAIN ERRORS OR MISPRINTS. THE OBJECT MANAGEMENT GROUP AND THE COMPANIES LISTED ABOVE MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS PUBLICATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF TITLE OR OWNERSHIP, IMPLIED WARRANTY OF MERCHANTABILITY OR WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE OR USE. IN NO EVENT SHALL THE OBJECT MANAGEMENT GROUP OR ANY OF THE COMPANIES LISTED ABOVE BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, RELIANCE OR COVER DAMAGES, INCLUDING LOSS OF PROFITS, REVENUE, DATA OR USE, INCURRED BY ANY USER OR ANY THIRD PARTY IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The entire risk as to the quality and performance of software developed using this specification is borne by you. This disclaimer of warranty constitutes an essential part of the license granted to you to use this specification.

## RESTRICTED RIGHTS LEGEND

Use, duplication or disclosure by the U.S. Government is subject to the restrictions set forth in subparagraph (c) (1) (ii) of The Rights in Technical Data and Computer Software Clause at DFARS 252.227-7013 or in subparagraph (c)(1) and (2) of the Commercial Computer Software - Restricted Rights clauses at 48 C.F.R. 52.227-19 or as specified in 48 C.F.R. 227-7202-2 of the DoD F.A.R. Supplement and its successors, or as specified in 48 C.F.R. 12.212 of the Federal Acquisition Regulations and its successors, as applicable. The specification copyright owners are as indicated above and may be contacted through the Object Management Group,9C Medway Road, PMB 274, Milford, MA 01757, U.S.A.

## TRADEMARKS

CORBA®, CORBA logos®, FIBO®, Financial Industry Business Ontology®, FINANCIAL INSTRUMENT GLOBAL IDENTIFIER®, IIOP®, IMM®, Model Driven Architecture®, MDA®, Object Management Group®, OMG®, OMG Logo®, SoaML®, SOAML®, SysML®, UAF®, Unified Modeling Language®, UML®, UML Cube Logo®, VSIPL®, and XMI® are registered trademarks of the Object Management Group, Inc.

For a complete list of trademarks, see: http://www.omg.org/legal/tm_list.htm. All other products or company names mentioned are used for identification purposes only, and may be trademarks of their respective owners.

## COMPLIANCE

The copyright holders listed above acknowledge that the Object Management Group (acting itself or through its designees) is and shall at all times be the sole entity that may authorize developers, suppliers and sellers of computer software to use certification marks, trademarks or other special designations to indicate compliance with these materials.

Software developed under the terms of this license may claim compliance or conformance with this specification if and only if the software compliance is of a nature fully matching the applicable compliance points as stated in the specification. Software developed only partially matching the applicable compliance points may claim only that the software was based on this specification, but may not claim compliance or conformance with this specification. In the event that testing suites are implemented or approved by Object Management Group, Inc., software developed using this specification may claim compliance or conformance with the specification only if the software satisfactorily completes the testing suites.

## OMG's Issue Reporting Procedure

All OMG specifications are subject to continuous review and improvement. As part of this process we encourage readers to report any ambiguities, inconsistencies, or inaccuracies they may find by completing the Issue Reporting Form listed on the main web page http://www.omg.org, under Documents, Report a Bug/Issue.

# Table of Contents

# Preface

## OMG

Founded in 1989, the Object Management Group, Inc. (OMG) is an open membership, not-for-profit computer industry standards consortium that produces and maintains computer industry specifications for interoperable, portable, and reusable enterprise applications in distributed, heterogeneous environments. Membership includes Information Technology vendors, end users, government agencies, and academia.

OMG member companies write, adopt, and maintain its specifications following a mature, open process. OMG's specifications implement the Model Driven Architecture® (MDA®), maximizing ROI through a full-lifecycle approach to enterprise integration that covers multiple operating systems, programming languages, middleware and networking infrastructures, and software development environments. OMG's specifications include: UML® (Unified Modeling Language™); CORBA® (Common Object Request Broker Architecture); CWM™ (Common Warehouse Metamodel™); and industry-specific standards for dozens of vertical markets.

More information on the OMG is available at http://www.omg.org/.

# OMG Specifications

As noted, OMG specifications address middleware, modeling and vertical domain frameworks. All OMG As noted, OMG specifications address middleware, modeling and vertical domain frameworks. All OMG Specifications are available from the OMG website at:

*http://www.omg.org/spec*

All of OMG's formal specifications may be downloaded without charge from our website. (Products implementing OMG specifications are available from individual suppliers.) Copies of specifications, available in PostScript and PDF format, may be obtained from the Specifications Catalog cited above or by contacting the Object Management Group, Inc. at:

OMG Headquarters
9C Medway Road, PMB 274
Milford, MA 01757
USA
Tel: +1-781-444-0404
Fax: +1-781-444-0320
Email: *pubs@omg.org*

Certain OMG specifications are also available as ISO standards. Please consult http://www.iso.org

# 1 Scope

This specification defines the interface between components of a C2 (Command and Control) system concerned with the dissemination of tactical picture information, recommendations for changes, refinements and enhancements to that picture and recommendations for courses of action that relate to the picture with resources for which the C2 system's user have responsibility. As such it is a specification for an interface between tactical picture management components and tactical decision aids, supporting the development of open modular C2 systems.

# 2 Conformance

This specification defines conformance points to promote both applicability and interoperability. The conformance points recognize the decomposition of the specification into services relating to Tactical Picture and services relating to Plan Execution. Services within the specification relating specifically to either Tactical Picture or Plan Execution are optional. The mandatory services within the interface are those in the AbstractRecommendations package. Conformance Points define a set of services to be implemented by a Tactical Picture or Plan Execution component and a dependency for a Tactical Decision Aid. Conformation points are defined for functional subsets, PSM technologies and PSM external standards. A

**Table 2.1 - Conformance Points for TDA**

| **Conformance Point** | **Service Interfaces** | **Rationale** |
|---|---|---|
| **Functional** | | |
| Basic Tactical Picture | Configuration, Response, Categorization, PictureManagement | These interfaces include the types of tactical picture recommendation most likely to be made by decision aids. |
| Basic Plan Execution | Configuration, Response, PlanDataSink, ResourceDataSink, PlanExecutionAction, PlanExecutionControl | These interfaces include the types of plan execution recommendation most likely to be made by decision aids. |
| Extended Tactical Picture | As per Basic Tactical Picture plus ExtendedCategorization, ExtendedPictureManagement | These are the additional interfaces for tactical picture recommendations |
| Extended Plan Execution | As per Basic Plan Execution plus ExtendedPlanExecutionAction, ExtendedPlanExecutionControl | These are the additional interfaces for plan execution recommendations |
| **PSM Technologies** | | |
| DDS | As defined by functional conformance points | A PSM technology for near real time operation |
| GraphQL | As defined by functional conformance points | A PSM for flexible data access |
| **External Standards** | **Schema Prefix** | |
| STANAG 5516 | s5516.* | Naval applications |
| JC3IEDM | jc3iedm | Applications for joint operations |
| APP6 | app6b, app6c | General C2 applications |

| | | |
|---|---|---|
| SOPES | sopes | General C2 applications |

# 3 Normative References

The following normative documents contain provisions, which, through reference in this text, constitute provisions of this specification. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply.

**Table 3.1 – Normative References**

| Title (Acronym) | Version / Date | Organization | Reference / URL |
|---|---|---|---|
| TACSIT Data Exchange (TEX) | 1.0 / February 2021 | OMG | formal/2019-12-02 www.omg.org/spec/TEX |
| Data Distribution Service (DDS) | 1.4 / March 2015 | OMG | formal/2015-04-10 www.omg.org/spec/DDS |
| Interface Definition Language (IDL) | 4.2 / January 2018 | OMG | formal/2018-01-05 www.omg.org/spec/IDL |
| Extended View of Time (EVOT) | 2.0 August 2008 | OMG | formal/2008-08-01 www.omg.org/spec/EVOT |
| DDS Security | 1.1 July 2018 | OMG | formal/18-04-01 https://www.omg.org/spec/DDS-SECURITY/ |
| Shared Operational Picture Exchange Services (SOPES) | 1.0 May 2011 | OMG | formal/11-05-04 www.omg.org/spec/SOPES |
| Graph Query Language (GraphQL) | June 2018 | Facebook | www.spec.graphql.org/June2018 |
| Quantities and units | November 2011 | ISO | ISO 80000-1 :2009 https://www.iso.org/standard/30669.html |
| NATO Tactical Data Exchange – Link 16 | Edition 6 | NATO | STANAG 5516 |
| Joint C3 Information Exchange Data Model | Rev D CN 1 | NATO | STANAG 5525 |
| Joint C3 Information Exchange Data Model (JC3IEDM) | v3.1.4 | NATO | |
| NATO Joint Military Symbology (APP-6(B)) | June 2008 | NATO | |
| NATO Joint Military Symbology (APP-6(C)) | May 2011 | NATO | |

| | | | |
|---|---|---|---|
| World Meteorological Organization (Sea State Code) | latest | WMO | www.public.wmo.int/en/resources/standards-technical-regulations |

# 4 Terms and Definitions

For the purposes of this specification, the following terms and definitions apply.

- API (Application Programming Interface)
- APP (Allied Procedural Publication)
- C2 (Command and Control)
- CMS (Combat Management System)
- CORBA (Common Object Request Broker Architecture)
- CWM (Common Warehouse Metamodel)
- DDS (Data Distribution Service)
- EVOT (Enhanced View of Time)
- GraphQL (Graph Query Language)
- IDL (Interface Definition Language)
- IIOP (Internet Inter-Orb Protocol)
- IPR (Intellectual Property Right)
- ICAO (International Civil Aviation Organization)
- ISO (International Organization for Standardization)
- IMO (International Maritime Organization)
- JC3IEDM (Joint Consultation, Command and Control Information Exchange Data Model)
- LOI (Letter of Intent)
- MDA (Model Driven Architecture)
- MOF (Meta Object Facility)
- NS (Naming Service)
- NATO (North Atlantic Treaty Organization)
- OARIS (Open Architecture Radar Interface Standard)
- ODF (Open Document Format)
- OMG (Object Management Group)
- PIM (Platform Independent Model)
- PSM (Platform Specific Model)

- SOA (Service Oriented Architecture)
- SoaML (Service oriented architecture Modeling Language)
- STANAG (NATO Standardization Agreement)
- TF (Task Force)
- UML (Unified Modeling Language)
- XMI (XML Metadata Interchange)
- XML (eXtensible Markup Language)

# 5 Symbols

No special symbols are introduced in this specification.

# 6 Additional Information

## 6.1 Acknowledgements

The following companies submitted this specification:

- BAE Systems

# 7 Tactical Decision Aids Interface Overview

The goal of the Tactical Decision Aids Interface specification is to support the on-going need to extend and upgrade C2 Systems, particularly to meet growing demand for automation and the exploitation of machine intelligence. This specification addresses the need to insert functionality into command and control systems that supports the user's decision-making process.

To insert such functionality efficiently and affordably, C2 system integrators need the freedom to source such functionality from multiple potential providers in the form of modular applications. This is especially so for systems meeting complex requirements in the military domain. This specification enables such an approach by standardizing the interface for such functionality.

This specification defines a set of services and a supporting data model as C2 Systems typically have demanding assurance requirements and are qualified for use by an overall design authority based on evidence built up from the constituent components. The constituent components need to know, *a priori*, the standardized data-model and services the system uses, so that they are able to provide qualification evidence for overall system assurance.

Consistency of semantics across the definitions of services and the data model is important for the interoperation of different component implementations that meet the standard, enabling the modular solution sought. To this end, the specification provides meta-models for the services and data-model, which define super-classes for the data-model and common patterns for the information and recommendation services.

The specification is organized as follows: -

- Section 8 contains the Meta-Model with packages for super-classes and other classifiers with broad applicability across the Data Models
- Section 9 contains the Data-Models describing a Data Model for Plan Execution and an Extension of the TACSIT Data Exchange Data Model to support the Tactical Picture Data Model requirements
- Section 10 the Service Models describing the Information and Recommendation Service Models for Tactical Picture and Pan Execution
- Section 11 describes the Domain Model Platform Specific Models for DDS and GraphQL
- Section 12 describes the Service Model Platform Specific Models for DDS and GraphQL
- Section 13 describes the Platform Specific Models for Extensible Enumerations; this section standardizes the representation of categories defined in a set of external specifications, whilst providing an extension mechanism that supports additional specifications and system-specific concepts

The specification is captured as an Enterprise Architect (EA) UML version 2.1 model; sections 8, 9 & 10 are automatically generated into the specification from the model.

## 7.1 Use of the Tactical Decision Aids Interface

This subsection provides non-normative outline usage of the interface specification for two simple use cases: creating a 'Classification App' that recommends classification categories of TACSIT Entities (e.g. Frigate, Helicopter, Truck or Submarine); creating a 'Plan Monitoring App' that recommends when to start and stop different elements of an overall plan. These simple use cases use the Tactical Picture and Plan Execution services separately, other use cases may use these services in combination.

### 7.1.1 Classification App use of the Tactical Decision Aids Interface

(Non-Normative)

An outline design of the Classification App to use the Tactical Decision Aids Interface is as follows: -

- Connect to the TEX and TDAI interfaces of the Tactical Picture component using the appropriate PSM method and system-specific configuration.
- Use a PSM method to register implementation of the recommendationProcessed operation on the Response interface.
- Use the PSM mapping of the isSupported operation on the Configuration interface for recommendClassification to verify that the TacticalPicture implementation supports the relevant recommendation operation.
- Use the getSupportMapping operation on the Configuration interface to access the extensible enumeration mapping-file. Verify that there are classification categories in the mapping file corresponding to the categories in the App's business rules. E.g. if the App has a rule for distinguishing helicopters find a helicopter category in the mapping file. Note that it is an option to design the App with known categories in mind.
- Create and add an Entity Listener to the TEX DataSink interface.
- Use the TEX getSet operation on the DataSink interface to get an initial view of the TEX Entities in the Tactical Picture.
- Process the initial view of Entities and any changes (including new Entities) from the dataChanged callback on the listener with the App's business logic.
- For any Entities for which the App's business logic can offer an improved classification category, use the recommendClassification operation on the Categorization interface to recommend the category to the TacticalPicture component. Internal to the App, note that a recommendation for this Entity is in progress (do not send further recommendations whilst in-progress).
- Implement the recommendationProcessed operation to clear the in-progress indicator if accepted, clear and log explanation (error code) if rejected and optionally log if deferred.
- Continue until the App is stopped (system specific mechanism outside of the scope of this specification).

### 7.1.2 Plan Monitoring App use of the Tactical Decision Aids Interface

An outline design of the Plan Monitoring App to use the Tactical Decision Aids Interface is as follows: -

- Connect to the TDAI interface of the Plan Execution component using the appropriate PSM method and system-specific configuration.
- Use a PSM method to register implementation of the recommendationProcessed operation on the Response interface.
- Use the PSM mapping of the isSupported operation on the Configuration interface for start and terminate to verify that the Plan Eecution implementation supports the relevant recommendation operation.
- Use the getSupportMapping operation on the Configuration interface to access the extensible enumeration mapping-file. Verify that there are plan constituent categories in the mapping file corresponding to the categories in the App's business rules. E.g. if the App has a rule for 'search

and rescue' plans find a 'search and rescue' category in the mapping file. Note that it is an option to design the App with known categories in mind.

- Create and add an Plan Listener to the TDAI PlanDataSink interface.
- Use the TDAI getSet operation on the PlanDataSink interface to get an initial view of the TDAI PlanConstituents known to the system.
- Process the initial view of PlanConstituents and any changes (including new PlanConstituents) from the dataChanged callback on the listener with the App's business logic.
- For any PlanConstituents which the App's business logic suggests should be started, use the start operation on the PlanExecutionControl interface to recommend that the PlanConstiuent is started. Internal to the App, note that a recommendation for this PlanConstiuent is in progress (do not send further recommendations whilst in-progress).
- Similarly, for any PlanConstituents which the App's business logic suggests should be terminated, use the terminate operation on the PlanExecutionControl interface to recommend that the PlanConstiuent is terminated.
- Implement the recommendationProcessed operation to clear the in-progress indicator if accepted, clear and log explanation (error code) if rejected and optionally log if deferred.
- Continue until the App is stopped (system specific mechanism outside of the scope of this specification).

# 8 MetaModel

**Parent Package:** tactical decision aids

Meta-Model containing super-classes, data-types, patterns and the generic forms of interfaces that have applicability across the Domain Model and Service Model requirements for the Tactical Decision Aids specification. This is an extension mechanism supporting the requirements of similar Domain Models and Service Models that could apply in the future or within a system specific context.


**Figure 8.1 MetaModel (Package diagram)**

## 8.1 DataModel

**Parent Package:** MetaModel

Aspects of the Meta-Model supporting the Domain Models

### 8.1.1 Recommendation

**Parent Package:** DataModel

The Recommendation package of the Data Model Meta-Model defines generic concepts to support system agnostic recommendations.

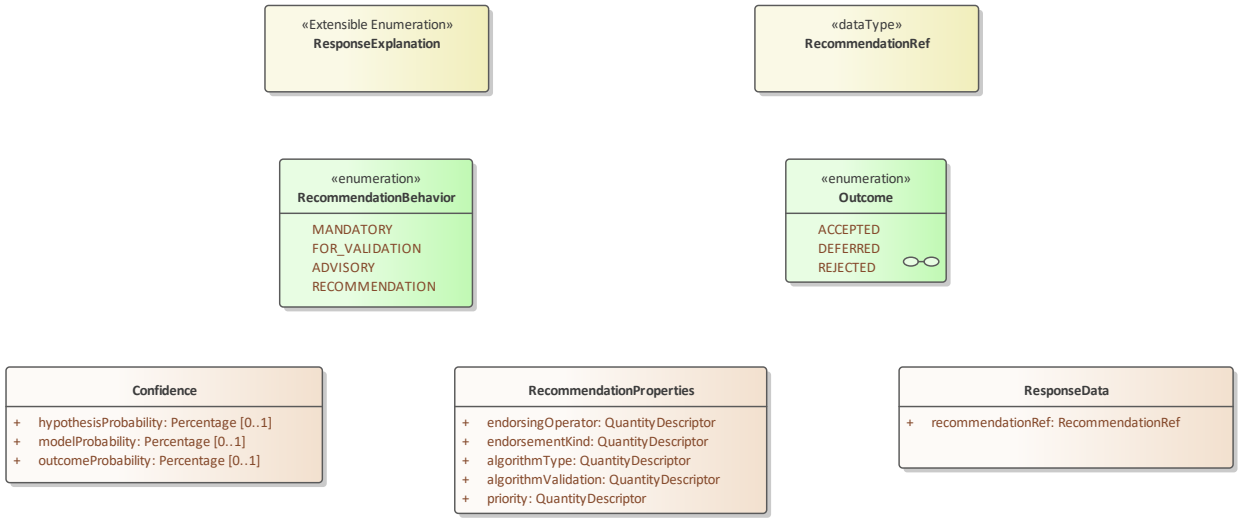


**Figure 8.2 Recommendation and Response (Class diagram)**

#### 8.1.1.1 Confidence

**Type:** Class
**Package:** Recommendation
The statistical confidence placed in a recommendation

**Table 8.1 - Attributes of Class Confidence**

| Attribute | Notes |
|---|---|
| **hypothesisProbability** Percentage [0..1] | The probability of the hypothesis associated with the Recommendation being true, given the model used as the Recommendation basis. This is the confidence that the Recommendation is correct or optimal according to the model. |
| **modelProbability** Percentage [0..1] | The probability of the model on which the Recommendation is based being applicable. This is the confidence in model given the particular model inputs. |
| **outcomeProbability** Percentage [0..1] | The probability that there will be a successful outcome from following a Recommendation according to the model used by the Recommendation. This attribute is only applicable to Recommendations that lead to actions on the external environment. |

#### 8.1.1.2 RecommendationProperties

**Type:** Class
**Package:** Recommendation
Additional extensible metadata relating to the recommendation process

**Table 8.2 - Attributes of Class RecommendationProperties**

| Attribute | Notes |
|---|---|
| **endorsingOperator** QuantityDescriptor | An extensible categorization of a system user who has endorsed the recommendation. |
| **endorsementKind** QuantityDescriptor | An extensible categorization of the type endorsement made by the system user |
| **algorithmType** QuantityDescriptor | An extensible categorization of type of algorithm used to make the recommendation |
| **algorithmValidation** QuantityDescriptor | An extensible categorization of validation process that the algorithm has been subjected to |
| **priority** QuantityDescriptor | An extensible categorization of the priority that a recipient should assign to a Recommendation |

#### 8.1.1.3 ResponseData

**Type:** Class
**Package:** Recommendation
Additional information to describe the action actually performed. Specializations of this call allow the Decision Aid to find tactical picture updates and plan execution update corresponding to the recommendations. Binding this information to the response also means the Decision Aid doesn't need to store recommendation identifiers locally in order to perform post response processing.

**Table 8.3 - Attributes of Class ResponseData**

| Attribute | Notes |
|---|---|
| **recommendationRef** RecommendationRef | A reference to the original recommendation that is unique across all clients in the system. |

#### 8.1.1.4 Outcome

**Type:** Enumeration

**Package:** Recommendation
The categories of outcome supported by the recommendation response interface.

**Table 8.4 - Attributes of Enumeration Outcome**

| Attribute | Notes |
|---|---|
| «enum» **ACCEPTED** | The recommendation has been accepted and applied |
| «enum» **DEFERRED** | The recommendation has been deferred, for instance for operator approval. An additional response will occur once a decision has been made. |
| «enum» **REJECTED** | The recommendation has been rejected. The explanation attribute contains any reason given for the rejection. |

### 8.1.1.5 Recommendation Outcome

**Type:** StateMachine
**Package:** Recommendation

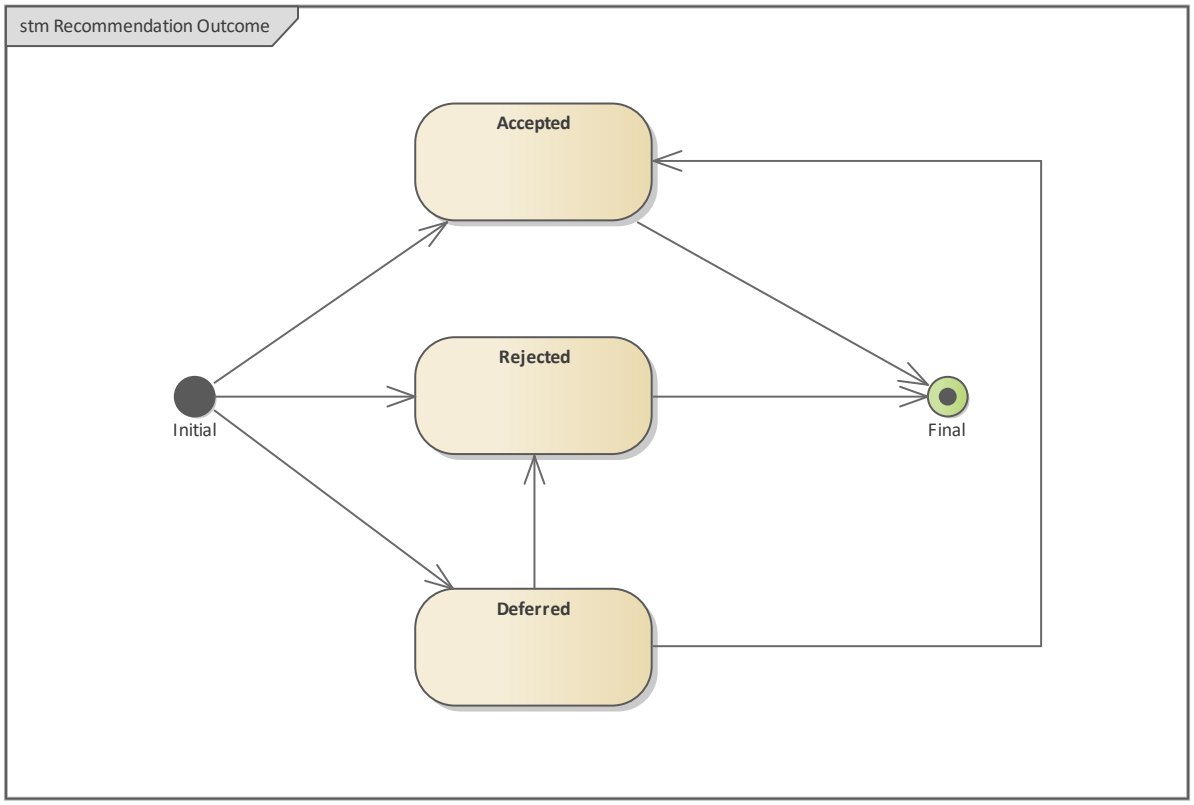


**Figure 8.3 Recommendation Outcome (StateMachine diagram)**
This diagram defines the state transitions of outcomes of recommendations from Tactical Decision Aids.

#### 8.1.1.5.1 Accepted

**Type:** State
**Package:** Recommendation
The outcome of the recommendation is that is accepted and will be applied to the referenced instances.

#### 8.1.1.5.2 Deferred

**Type:** State
**Package:** Recommendation

The initial outcome of the recommendation is that is deferred and will be either accepted or rejected after input from an operator or another system..

#### 8.1.1.5.3 Rejected

**Type:** State
**Package:** Recommendation
The outcome of the recommendation is that is rejected and will not be applied to the referenced instances.

### 8.1.1.6 RecommendationBehavior

**Type:** Enumeration
**Package:** Recommendation
Categorization of Recommendations in terms of the recipient's behavior

**Table 8.5 - Attributes of Enumeration RecommendationBehavior**

| Attribute | Notes |
|---|---|
| «enum» **MANDATORY** | The Recommendation must be followed by the recipient. |
| «enum» **FOR_VALIDATION** | The recipient should enact subject to a confirmation process |
| «enum» **ADVISORY** | The recommendation should be considered alongside alternative advisory sources of information. |
| «enum» **RECOMMENDATION** | No statement with respect to Recommendation categorization |

### 8.1.1.7 RecommendationRef

**Type:** DataType
**Package:** Recommendation
A reference to the recommendation that a Decision Aid has made. This must be unique within a system as a whole and not just within the lifetime of a decision aid or other system component.

### 8.1.1.8 ResponseExplanation

**Type:** DataType
**Package:** Recommendation
An explanation of the response to the recommendation. For example an error code.

## 8.1.2 Utils

**Parent Package:** DataModel
The Utils package in the Data Model Meta-Model defines utility classes required by other Data Model and Service Model packages.

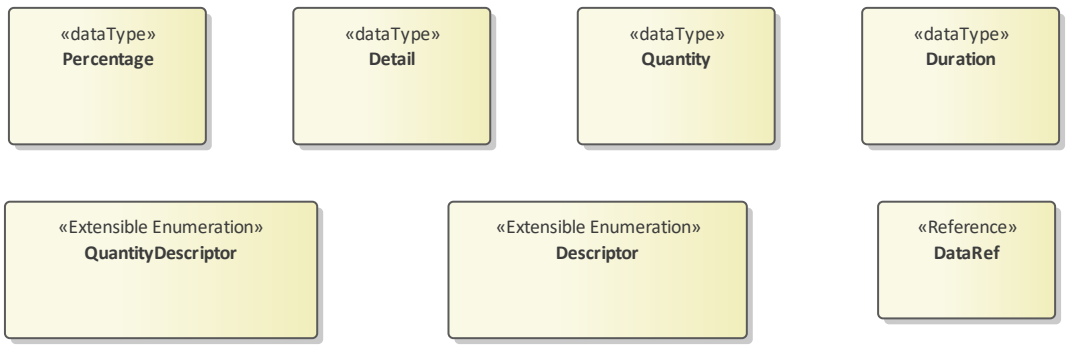

**Figure 8.4 DataTypes (Class diagram)**


**Figure 8.5 Structs (Class diagram)**

#### 8.1.2.1 AdditionalData

**Type:** Class
**Package:** Utils
Standardized encapsulation of qualitative, quantitative and unstructured data extension mechanisms.

**Table 8.6 - Attributes of Class AdditionalData**

| Attribute | Notes |
|---|---|
| **unstructuredData** ExtendedData [0..*] | |
| **qualifier** Qualifier [0..*] | A set of additional qualitative attributes as an extension mechanism. |
| **quantifier** Quantifier [0..*] | A set of additional quantitative attributes as an extension mechanism. |

#### 8.1.2.2 Qualifier

**Type:** Class
**Package:** Utils
A class to represent additional, system-specific qualitative or categorical values as a extension mechanism.

**Table 8.7 - Attributes of Class Qualifier**

| Attribute | Notes |
|---|---|
| **name** Descriptor | The name of quality being described |
| **value** Descriptor | The category value of the quality being described |

#### 8.1.2.3 Quantifier

**Type:** Class
**Package:** Utils
A abstract mechanism to quantify capabilities and dependencies

**Table 8.8 - Attributes of Class Quantifier**

| Attribute | Notes |
|---|---|
| **value** Quantity | The numerical value of the concept being quantified |
| **descriptor** QuantityDescriptor | An extensible categorization of the type of quantity being specified. The descriptor is the determiner of the units (if any) associated with the quantity. |

#### 8.1.2.4 DataRef

**Type:** DataType
**Package:** Utils
A datatype with a platform specific mapping to represent a reference to a data item instance.

#### 8.1.2.5 Descriptor

**Type:** DataType
**Package:** Utils
A general abstraction of categories to qualify an object.

#### 8.1.2.6 Detail

**Type:** DataType
**Package:** Utils
This is a datatype with a platform specific mapping to represent additional information through an extension mechanism

#### 8.1.2.7 Duration

**Type:** DataType
**Package:** Utils
A datatype with a platform specific mapping to represent a relative length of time

#### 8.1.2.8 Percentage

**Type:** DataType
**Package:** Utils
A datatype with a platform specific mapping to represent a percentage value

#### 8.1.2.9 Quantity

**Type:** DataType
**Package:** Utils
A datatype with a platform specific mapping to represent a scalar quantity

#### 8.1.2.10 QuantityDescriptor

**Type:** DataType
**Package:** Utils
An abstraction of the categories of quantity

## 8.2 ServiceModel

**Parent Package:** MetaModel
Aspects of the Meta-Model supporting the Service Models

### 8.2.1 Recommendations

**Parent Package:** ServiceModel
This package defines the elements of the generic recommendation and response pattern used by Tactical Decision Aids to make recommendations

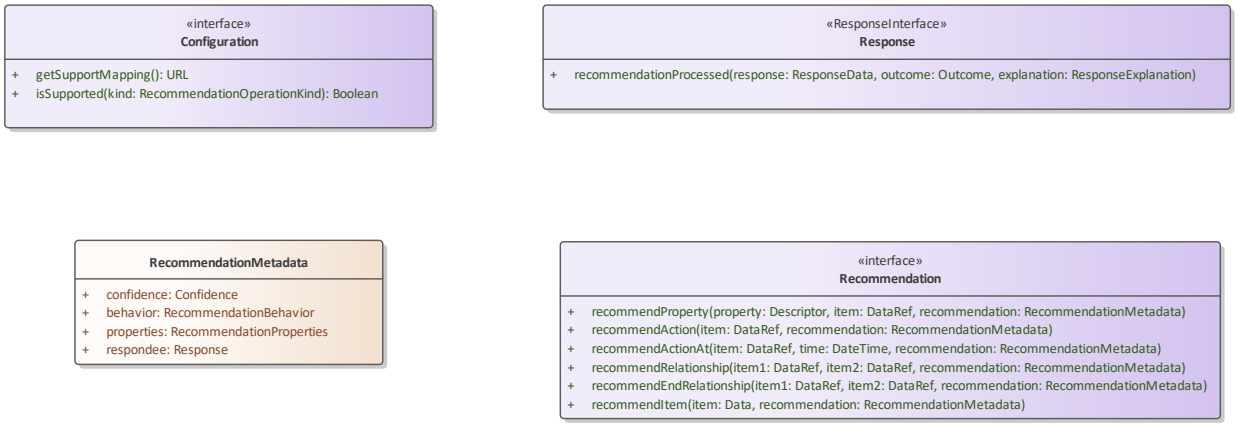


**Figure 8.6 Recommendation (Class diagram)**

#### 8.2.1.1 RecommendationMetadata

**Type:** Class
**Package:** Recommendations
Additional information to describe and qualify all recommendations

**Table 8.9 - Attributes of Class RecommendationMetadata**

| Attribute | Notes |
|---|---|
| **confidence** Confidence | The statistical confidence in the Recommendation |
| **behavior** RecommendationBehavior | The behavior required of the recipient |
| **properties** RecommendationProperties | Additional extensible properties of the Recommendation |
| **respondee** Response | The interface instance to which responses to the recommendation should be directed |

#### 8.2.1.2 Configuration

**Type:** Interface
**Package:** Recommendations
This interface allows clients (tactical decision aids) to determine system support for the interface in terms of extensibility methods and the operations implemented.

**Table 8.10 - Methods of Interface Configuration**

| Method | Notes | Parameters |
|---|---|---|
| **getSupportMapping()** | This operation returns the location of a resource defining the implementing component's support for extensible enumeration values. The resource defines supported extensible enumeration values, maps them to extensible enumeration datatypes defined by this specification and external specifications from which they are derived. | |
| **isSupported()** | This operation defines whether a particular recommendation function is implemented in the system. | RecommendationOperationKind **kind** The operation for which support is being queried |

#### 8.2.1.3 Recommendation

**Type:** Interface
**Package:** Recommendations
The generic form of a recommendation interface with prototype recommendation operations.

**Table 8.11 - Methods of Interface Recommendation**

| Method | Notes | Parameters |
|---|---|---|
| **recommendProperty()** | The prototype form of an operation to recommend a value for the property of an item. | Descriptor **property**<br>DataRef **item**<br>RecommendationMetadata **recommendation** |
| **recommendAction()** | The prototype form of an operation to recommend performing an operation on an item. | DataRef **item**<br>RecommendationMetadata **recommendation** |
| **recommendActionAt()** | The prototype form of an operation to recommend performing an operation on an item at a future time. | DataRef **item**<br>DateTime **time**<br>RecommendationMetadata **recommendation** |
| **recommendRelationship()** | The prototype form of an operation to recommend the creation of a relationship between two data items. | DataRef **item1**<br>DataRef **item2**<br>RecommendationMetadata **recommendation** |
| **recommendEndRelationship()** | The prototype form of an operation to recommend the ending of a relationship between two data items. | DataRef **item1**<br>DataRef **item2**<br>RecommendationMetadata **recommendation** |
| **recommendItem()** | The prototype form of an operation to recommend the creation of an item. | Data **item**<br>RecommendationMetadata **recommendation** |

#### 8.2.1.4 Response

**Type:** Interface
**Package:** Recommendations
This interface is implemented by a tactical decision aid in order to receive responses to its recommendations. Each response operation contains a reference to the information contained in the corresponding recommendation.

**Table 8.12 - Methods of Interface Response**

| Method | Notes | Parameters |
|---|---|---|
| **recommendationProcessed()** | This callback operation is invoked on the Tactical Decision Aid when the recommendation that it has made is accepted, deferred or rejected. This allows the Tactical Decision Aid to understand when a recommendation is in progress and to avoid redundantly repeating recommendations. | ResponseData **response** Additional contextual and qualification data for the response.<br>Outcome **outcome** The outcome of the recommendation process<br>ResponseExplanation **explanation** Where available, an explanation of the recommendation processing, such as a reason for rejection. |

| | | |
|---|---|---|
| | There is one invocation of this callback per recommendation unless the first Outcome is Deferred, in which case there are two. This is shown in the Recommendation Outcome State Machine diagram. | |

## 8.2.2 DataSink

**Parent Package:** ServiceModel

The package defines the pattern for Tactical Decision Aids to receive Information on the instances of a particular class of data item.

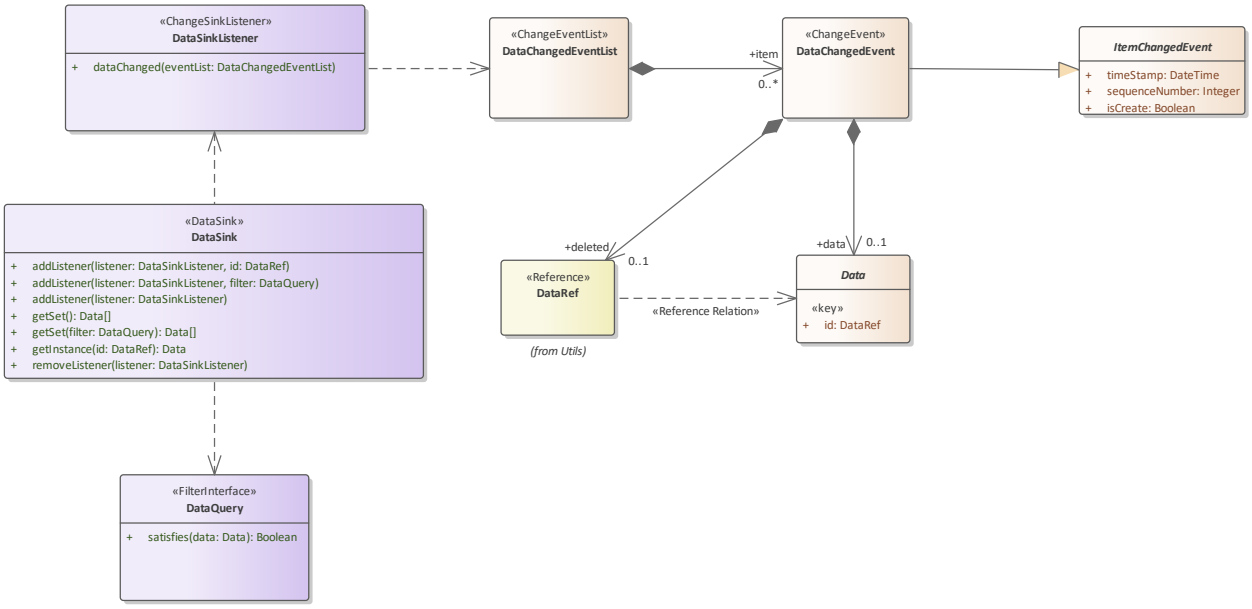


**Figure 8.7 DataSinkPattern (Class diagram)**

### 8.2.2.1 Data

**Type:** Class
**Package:** DataSink

Represents the primary class of data for the Data Sink. Data items are the Data Sink's atomic unit.

**Table 8.13 - Attributes of Class Data**

| Attribute | Notes |
|---|---|
| «key» **id** DataRef | Unique reference for the data item within the scope of the system. |

### 8.2.2.2 DataChangedEvent

**Type:** Class
**Package:** DataSink

Represents information about a change to a <Data Instance>

### 8.2.2.3 DataChangedEventList

**Type:** Class
**Package:** DataSink
Represents the list of changes to <Data> since the last event notified to that instance of the listener. Multiple changes may be consolidated into a single callback to a listener on the interface

#### 8.2.2.4 ItemChangedEvent

**Type:** Class
**Package:** DataSink
An abstraction of data sink change event

**Table 8.14 - Attributes of Class ItemChangedEvent**

| Attribute | Notes |
|---|---|
| **timeStamp** DateTime | The time of the change |
| **sequenceNumber** Integer | The time sequenced position of the event for the listener |
| **isCreate** Boolean | The event created a new instance |

#### 8.2.2.5 DataQuery

**Type:** Interface
**Package:** DataSink
This is an interface through which a client can define Queries on <Data> so as to filter the information returned. Classes implementing the interface provide means to set the query parameters (such as a constructor).

**Table 8.15 - Methods of Interface DataQuery**

| Method | Notes | Parameters |
|---|---|---|
| **satisfies()** | This operation is the client's implementation of a filtering query for <Data> | Data **data** The data being filtered |

#### 8.2.2.6 DataSink

**Type:** Interface
**Package:** DataSink
This interface contains operations that give a Tactical Decision Aid access to information about the execution of <Data>. A Tactical Decision Aid can add and remove listeners as well as reading the information about individual <Data> or all or a filtered subset of <Data>.

**Table 8.16 - Methods of Interface DataSink**

| Method | Notes | Parameters |
|---|---|---|
| **addListener()** | Operation to add a listener for callbacks relating to a single <Data Instance> | DataSinkListener **listener** The listener object to receive the callback DataRef **id** A reference to a specific instance of interest |
| **addListener()** | Operation to add a listener for callbacks relating to all <Data> that satisfy the Query | DataSinkListener **listener** The listener object to receive the callback DataQuery **filter** The filer object to apply to changes |
| **addListener()** | Operation to add a listener for callbacks relating to all <Data> | DataSinkListener **listener** The listener object to receive the callback |
| **getSet()** | Operation to obtain the information relating to all the <Data> | |

| | | |
|---|---|---|
| **getSet()** | Operation to obtain the information relating to all the <Data> satisfying the query | DataQuery **filter** The filter object to apply to instance of the class |
| **getInstance()** | Operation to obtain the information relating to the <Data> reference | DataRef **id** A reference to the speciifc instance of interest |
| **removeListener()** | Operation to remove a listener | DataSinkListener **listener** The listener object to no longer receive callbacks |

#### 8.2.2.7 DataSinkListener

**Type:** Interface
**Package:** DataSink
This is an interface for clients to implement callback to receive information on changes to <Data>.

**Table 8.17 - Methods of Interface DataSinkListener**

| Method | Notes | Parameters |
|---|---|---|
| **dataChanged()** | This operation is implemented by the client to process the data change callback. Multiple changes can be notified through a single invocation. | DataChangedEventList **eventList** The list of data instances that have changed |

# 9 DataModel

**Parent Package:** tactical decision aids

The Tactical Decision Aids Data Model defines the representation of information that is passed between Picture Management Components and Tactical Decision Aid Components.

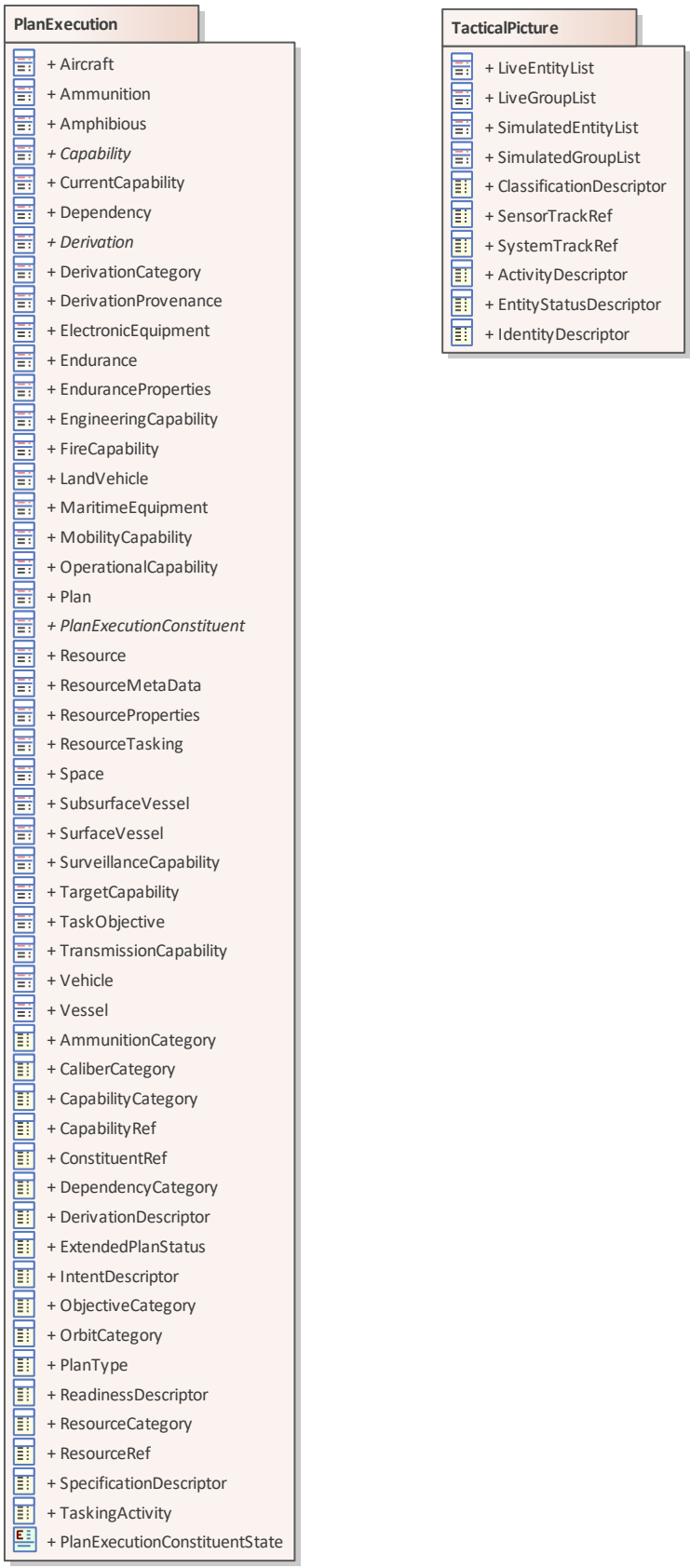
PlanExecution
+ Aircraft
+ Ammunition
+ Amphibious
+ Capability
+ CurrentCapability
+ Dependency
+ Derivation
+ DerivationCategory
+ DerivationProvenance
+ ElectronicEquipment
+ Endurance
+ EnduranceProperties
+ EngineeringCapability
+ FireCapability
+ LandVehicle
+ MaritimeEquipment
+ MobilityCapability
+ OperationalCapability
+ Plan
+ PlanExecutionConstituent
+ Resource
+ ResourceMetaData
+ ResourceProperties
+ ResourceTasking
+ Space
+ SubsurfaceVessel
+ SurfaceVessel
+ SurveillanceCapability
+ TargetCapability
+ TaskObjective
+ TransmissionCapability
+ Vehicle
+ Vessel
+ AmmunitionCategory
+ CaliberCategory
+ CapabilityCategory
+ CapabilityRef
+ ConstituentRef
+ DependencyCategory
+ DerivationDescriptor
+ ExtendedPlanStatus
+ IntentDescriptor
+ ObjectiveCategory
+ OrbitCategory
+ PlanType
+ ReadinessDescriptor
+ ResourceCategory
+ ResourceRef
+ SpecificationDescriptor
+ TaskingActivity
+ PlanExecutionConstituentState
TacticalPicture
+ LiveEntityList
+ LiveGroupList
+ SimulatedEntityList
+ SimulatedGroupList
+ ClassificationDescriptor
+ SensorTrackRef
+ SystemTrackRef
+ ActivityDescriptor
+ EntityStatusDescriptor
+ IdentityDescriptor

**Figure 9.8 DataModel (Package diagram)**

# 9.1 PlanExecution

**Parent Package:** DataModel

The Plan Execution package of the Data Model defines the generic concepts necessary to recommend, execute and amend tactical plans. Domain and system specific concepts are abstracted using the Descriptor extension mechanism.

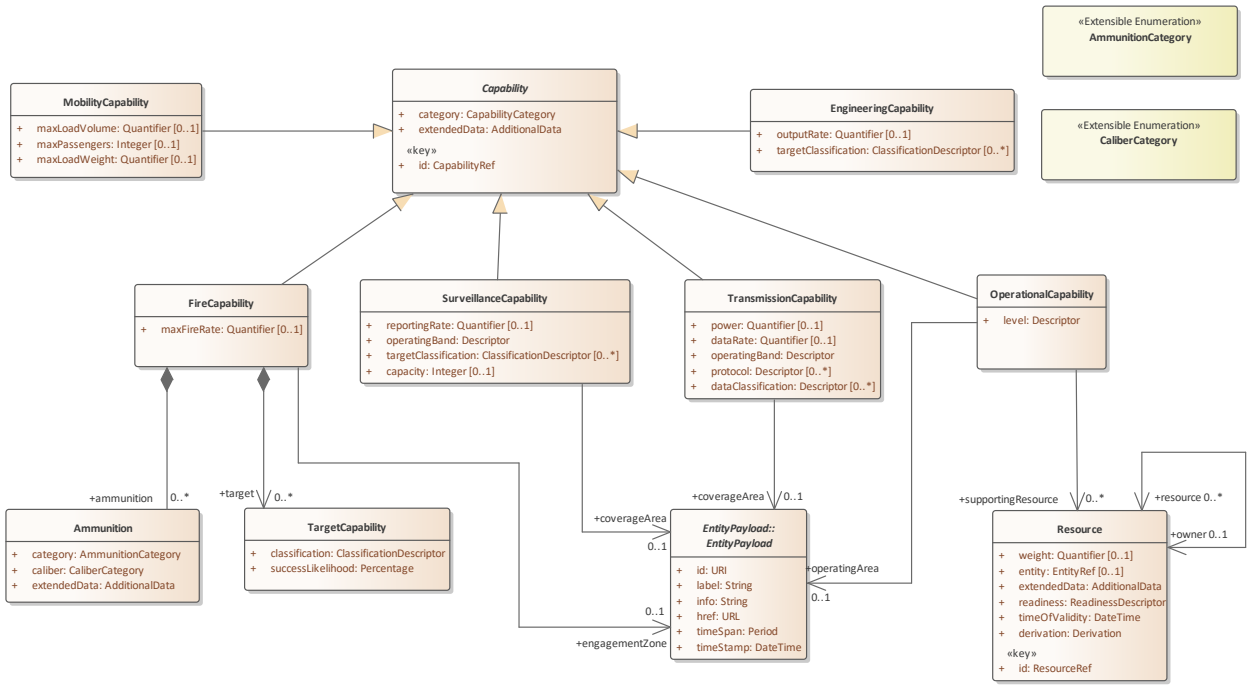


**Figure 9.9 Capability (Class diagram)**

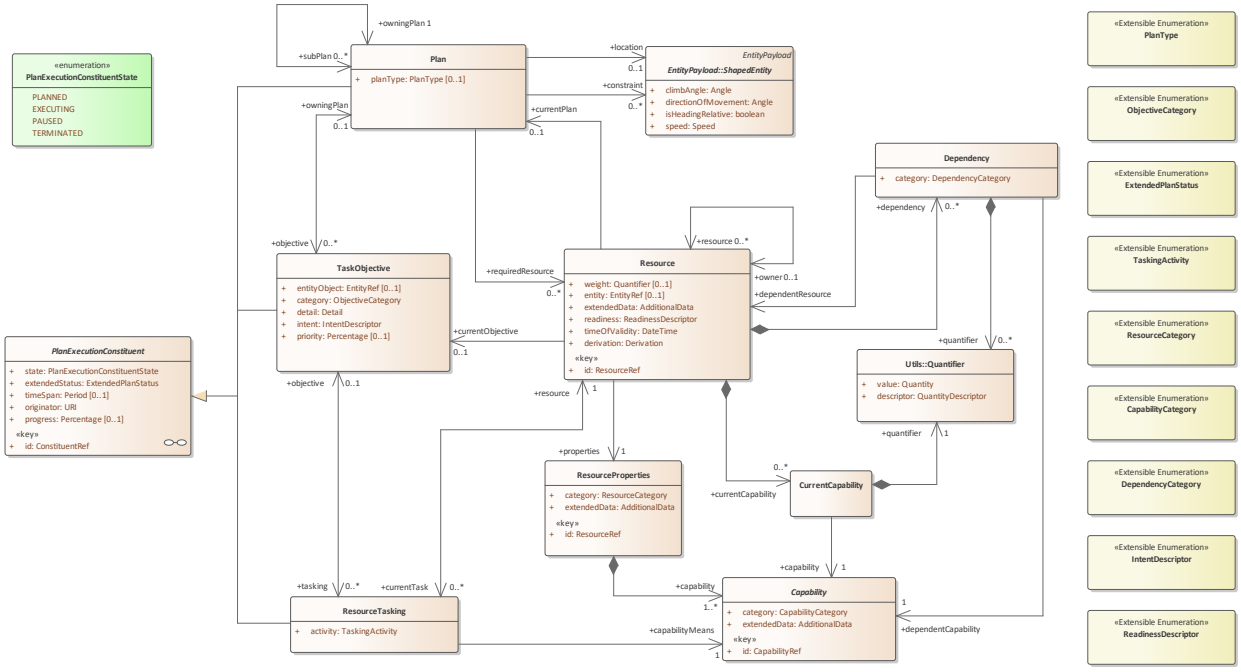


**Figure 9.10 PlanExecution (Class diagram)**

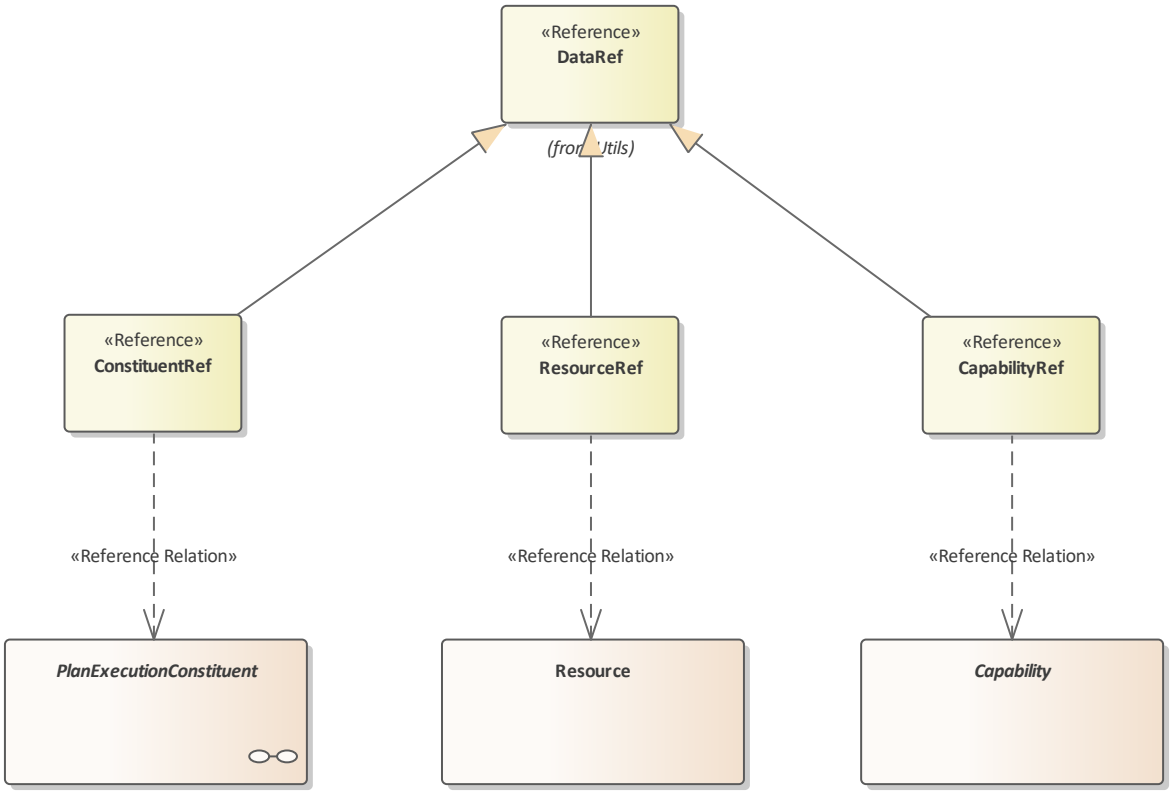


**Figure 9.11 Reference DataTypes (Class diagram)**

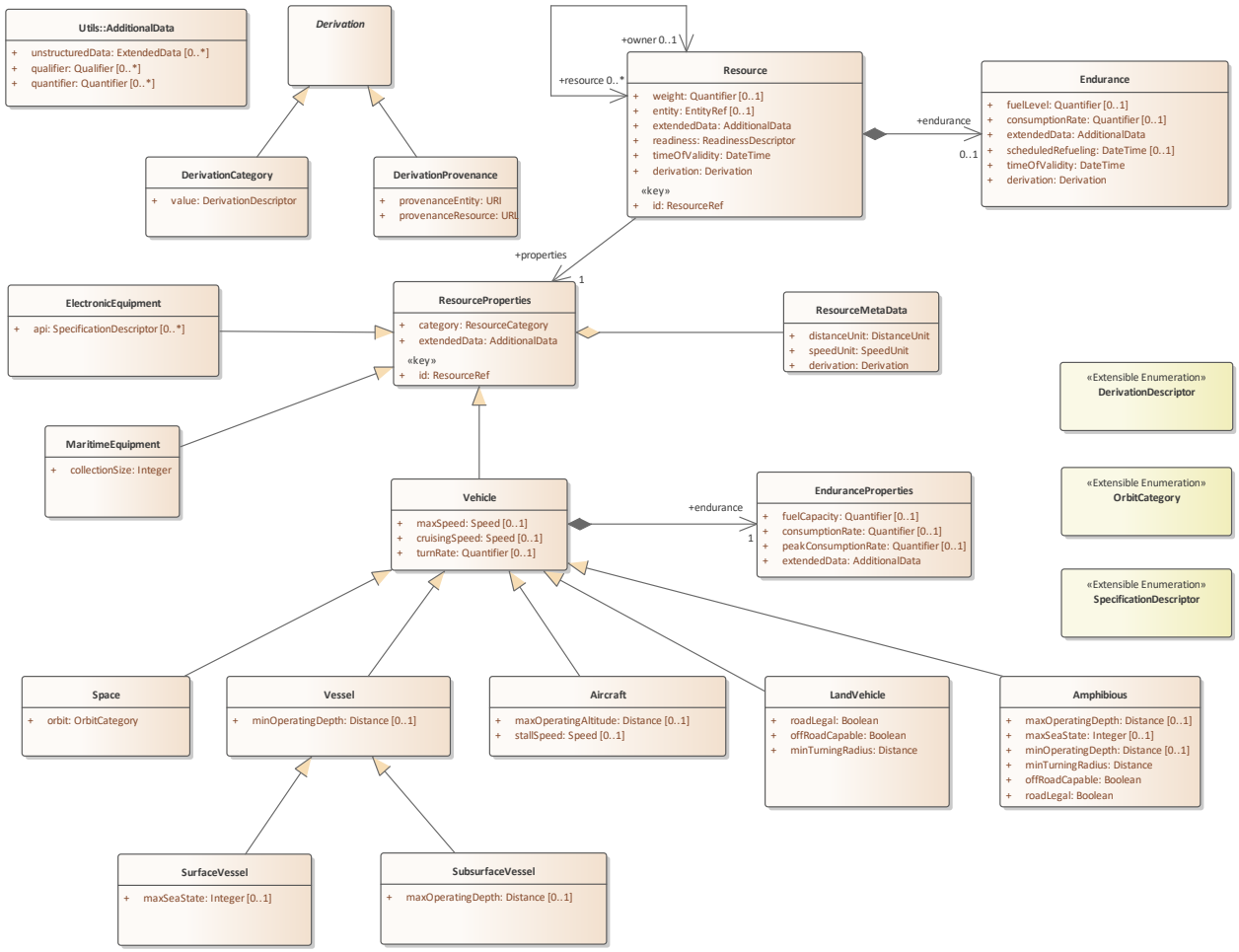


**Figure 9.12 Resource (Class diagram)**

## 9.1.1 Aircraft

**Type:** Class
**Package:** PlanExecution
An airborne resource

**Table 9.1 - Attributes of Class Aircraft**

| Attribute | Notes |
|---|---|
| **maxOperatingAltitude** Distance [0..1] | The maximum altitude (barometric) at which the aircraft is able to operate |
| **stallSpeed** Speed [0..1] | The minimum speed for the aircraft resource to operate below which it risks a stall. |

## 9.1.2 Ammunition

**Type:** Class
**Package:** PlanExecution
Description of the ammunition associated with a fire capability. Here, ammunition generalizes to include bombs, torpedoes, decoys and missiles.

**Table 9.2 - Attributes of Class Ammunition**

| Attribute | Notes |
|---|---|
| **category** AmmunitionCategory | An extensible categorization of the ammunition type |
| **caliber** CaliberCategory | An extensible description of the ammunition caliber (i.e. size). |
| **extendedData** AdditionalData | An extensibility mechanism for system specific ammunition attributes. |

### 9.1.3 Amphibious

**Type:** Class
**Package:** PlanExecution

**Table 9.3 - Attributes of Class Amphibious**

| Attribute | Notes |
|---|---|
| **maxOperatingDepth** Distance [0..1] | The operating limit of the amphibious vehicle with respect to depth in the water |
| **maxSeaState** Integer [0..1] | The World Meteorological Organization (WMO) sea state code - range 0 .. 9. |
| **minOperatingDepth** Distance [0..1] | The amphibious resource's operating limit with respect to shallow water |
| **minTurningRadius** Distance | The radius of the amphibious vehicle's tightest turning circle. |
| **offRoadCapable** Boolean | Whether the amphibious vehicle can be driven off-road. More granular off-road capabilities are specified using the mobility capability specialization. |
| **roadLegal** Boolean | The amphibious vehicle is legal for driving on the roads (in the territory applicable to the current system context). |

### 9.1.4 Capability

**Type:** Class
**Package:** PlanExecution
A Capability is an abstraction of a Resource's fundamental properties with respect to its ability to undertake tasks

**Table 9.4 - Attributes of Class Capability**

| Attribute | Notes |
|---|---|
| **category** CapabilityCategory | An extensible categorization of the kind of capability being described. |
| **extendedData** AdditionalData | An extensibility mechanism for system specific capability attributes. |
| «key» **id** CapabilityRef | The unique identifier for the instance. |

### 9.1.5 CurrentCapability

**Type:** Class
**Package:** PlanExecution

### 9.1.6 Dependency

**Type:** Class
**Package:** PlanExecution

This class represents a dependent linkage between two resources or of a resource on a particular capability.

**Table 9.5 - Attributes of Class Dependency**

| Attribute | Notes |
|---|---|
| **category** DependencyCategory | This is an extensible categorization of the type of dependency. |

### 9.1.7 Derivation

**Type:** Class
**Package:** PlanExecution
An abstract class for derivation metadata

### 9.1.8 DerivationCategory

**Type:** Class
**Package:** PlanExecution
A system specific categorical derivation

**Table 9.6 - Attributes of Class DerivationCategory**

| Attribute | Notes |
|---|---|
| **value** DerivationDescriptor | |

### 9.1.9 DerivationProvenance

**Type:** Class
**Package:** PlanExecution
Derivation conforming to the W3C PROV recommendation

**Table 9.7 - Attributes of Class DerivationProvenance**

| Attribute | Notes |
|---|---|
| **provenanceEntity** URI | The entity pertaining to the derivation in the corresponding provenance resource, conforming to the W3C PROV recommendation. |
| **provenanceResource** URL | The provenance resource conforming to the W3C PROV recommendation, containing the provenance metadata for the derivation |

### 9.1.10 ElectronicEquipment

**Type:** Class
**Package:** PlanExecution
A resource whose primary capabilities relate to its electronic components.

**Table 9.8 - Attributes of Class ElectronicEquipment**

| Attribute | Notes |
|---|---|
| **api** SpecificationDescriptor [0..*] | The set of interfaces standards that the equipment supports through which integrated functionality can be delivered to Plan Execution. |

### 9.1.11 Endurance

**Type:** Class
**Package:** PlanExecution
This class encapsulates the dynamic endurance properties of a resource.

**Table 9.9 - Attributes of Class Endurance**

| Attribute | Notes |
|---|---|
| **fuelLevel** Quantifier [0..1] | The current quantity of fuel available |
| **consumptionRate** Quantifier [0..1] | The current rate at which the fuel is consumed. |
| **extendedData** AdditionalData | Additional dynamic information related to the Resource's Endurance |
| **scheduledRefueling** DateTime [0..1] | The time at which more fuel is scheduled to be available |
| **timeOfValidity** DateTime | The time for which the attributes of the Endurance class are valid. |
| **derivation** Derivation | A description of the means by which the data for the Resource's Endurance attributes were derived. This includes sensing, communication routes and human input. |

### 9.1.12 EnduranceProperties

**Type:** Class
**Package:** PlanExecution
This class encapsulates the static, persistent endurance properties of a resource.

**Table 9.10 - Attributes of Class EnduranceProperties**

| Attribute | Notes |
|---|---|
| **fuelCapacity** Quantifier [0..1] | The maximum quantity of fuel that can be stored |
| **consumptionRate** Quantifier [0..1] | The nominal or mean (as defined by the descriptor) rate at which the fuel is consumed. |
| **peakConsumptionRate** Quantifier [0..1] | The peak rate of fuel consumption |
| **extendedData** AdditionalData | Additional static information related to the Resource's Endurance |

### 9.1.13 EngineeringCapability

**Type:** Class
**Package:** PlanExecution
A capability to build, maintain, breach or demolish structures or infrastructure in the operational environment

**Table 9.11 - Attributes of Class EngineeringCapability**

| Attribute | Notes |
|---|---|
| **outputRate** Quantifier [0..1] | The nominal rate at which the engineering capability can be delivered. |
| **targetClassification** ClassificationDescriptor [0..*] | The categories of object to which the Engineering Capability can be applied. |

### 9.1.14 FireCapability

**Type:** Class
**Package:** PlanExecution
An ability to apply physical effect towards an adversary.

**Table 9.12 - Attributes of Class FireCapability**

| Attribute | Notes |
|---|---|
| **maxFireRate** Quantifier [0..1] | |

### 9.1.15 LandVehicle

**Type:** Class

**Package:** PlanExecution
A vehicle primarily for traveling on land

**Table 9.13 - Attributes of Class LandVehicle**

| Attribute | Notes |
|---|---|
| **roadLegal** Boolean | The land vehicle is legal for driving on the roads (in the territory applicable to the current system context). |
| **offRoadCapable** Boolean | Whether the land vehicle can be driven off-road. More granular off-road capabilities are specified using the mobility capability specialization. |
| **minTurningRadius** Distance | The radius of the land vehicle's tightest turning circle. |

### 9.1.16 MaritimeEquipment

**Type:** Class
**Package:** PlanExecution
Equipment to be deployed in the maritime environment from surface or subsurface vessels. Maritime equipment do not encompass an entire vessel. The inherited owner-resource self-association relation from the parent Resource class applies between the Vessel and MaritimeEquipment classes.

**Table 9.14 - Attributes of Class MaritimeEquipment**

| Attribute | Notes |
|---|---|
| **collectionSize** Integer | Where a resource denotes a set of items this attribute specifies how many there are. A default of 1 is used for single items. |

### 9.1.17 MobilityCapability

**Type:** Class
**Package:** PlanExecution
An ability to transport objects and personnel.

**Table 9.15 - Attributes of Class MobilityCapability**

| Attribute | Notes |
|---|---|
| **maxLoadVolume** Quantifier [0..1] | The maximum load by volume (including passengers) that the Mobility Capability can transport. |
| **maxPassengers** Integer [0..1] | The maximum number of passengers that can be transported |
| **maxLoadWeight** Quantifier [0..1] | The maximum load by weight (including passengers) that the Mobility Capability can transport. |

### 9.1.18 OperationalCapability

**Type:** Class
**Package:** PlanExecution
Operational capability describes the overall capabilities of a resource comprising its associated personnel and equipment. It accounts for training, readiness and equipment status.

**Table 9.16 - Attributes of Class OperationalCapability**

| Attribute | Notes |
|---|---|
| **level** Descriptor | The organizational level at which the operational capability is intended to be performed |

### 9.1.19 Plan

**Type:** Class

**Package:** PlanExecution
A Plan represents an aggregated set of objectives and the resources and tasking to achieve them.

**Table 9.17 - Attributes of Class Plan**

| Attribute | Notes |
|---|---|
| **planType** PlanType [0..1] | The extensible categorization of the type of plan |

## 9.1.20 PlanExecutionConstituent

**Type:** Class
**Package:** PlanExecution
An abstract class for constituent elements of tactical plan execution

**Table 9.18 - Attributes of Class PlanExecutionConstituent**

| Attribute | Notes |
|---|---|
| «key» **id** ConstituentRef | The unique identifier for the instance |
| **state** PlanExecutionConstituentState | The state of the plan constituent according to the PlanExecutionConstituent state machine |
| **extendedStatus** ExtendedPlanStatus | The extensible detailed categorization of the state of the plan execution constituent. |
| **timeSpan** Period [0..1] | The time during which the plan execution constituent is expected to be executed. |
| **originator** URI | The originator of the PlanExecutionConstituent. This is the component that instigated the creation of the instance. |
| **progress** Percentage [0..1] | The proportion of the plan execution constituent's objectives that have been achieved. |

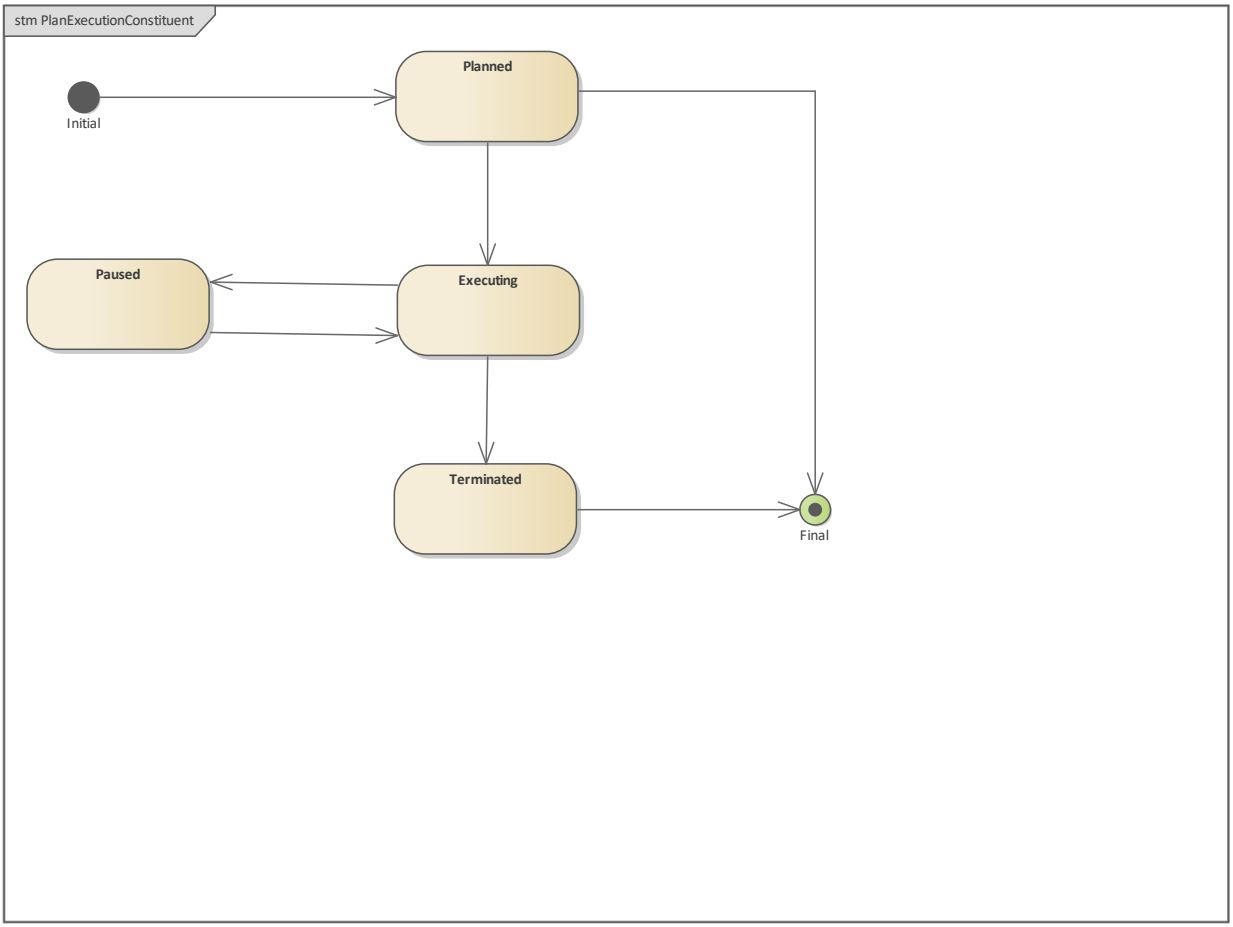


**Figure 9.13 PlanExecutionConstituent (StateMachine diagram)**

This diagram defines the state transitions of a plan execution constituent in response to recommendation actions from Tactical Decision Aids.

#### 9.1.20.1 Executing

**Type:** State
**Package:** PlanExecution

A successful start or resume recommendation transition a plan constituent to this state. A plan constituent also transitions to this state at the time of the start of its time span.

#### 9.1.20.2 Paused

**Type:** State
**Package:** PlanExecution

A successful pause recommendation transitions a plan constituent to this state. The behaviour of the Plan Execution component with respect to the end of a paused plan constituent's Time Span is implementation defined, but observable by Tactical Decision Aids through the relevant Data Sink interface.

#### 9.1.20.3 Planned

**Type:** State
**Package:** PlanExecution

A successful plan recommendation creates a plan constituent in this state.

#### 9.1.20.4 Terminated

**Type:** State
**Package:** PlanExecution
A successful terminate recommendation transitions a plan constituent to this state. Plan Execution should transition plan constituent instances that have been started to this state before they are deleted.

### 9.1.21 Resource

**Type:** Class
**Package:** PlanExecution
A Resource is an abstraction of a physical entity that can be independently tasked to achieve an objective.

**Table 9.19 - Attributes of Class Resource**

| Attribute | Notes |
|---|---|
| «key» **id** ResourceRef | The unique identifier for the instance |
| **weight** Quantifier [0..1] | The current weight of the Resource |
| **entity** EntityRef [0..1] | A reference to the entity representing the resource in the tactical picture |
| **extendedData** AdditionalData | Additional dynamic information related to the Resource |
| **readiness** ReadinessDescriptor | The extensible categorization of the readiness of resource (the extent to which resource is ready and available to be tasked to employ its capabilities). |
| **timeOfValidity** DateTime | The time for which the attributes of the Resource class are valid. |
| **derivation** Derivation | A description of the means by which the data for the Resource's attributes were derived. This includes sensing, communication routes and human input. |

### 9.1.22 ResourceMetaData

**Type:** Class
**Package:** PlanExecution

**Table 9.20 - Attributes of Class ResourceMetaData**

| Attribute | Notes |
|---|---|
| **distanceUnit** DistanceUnit | The unit used to defined the Resources distance properties |
| **speedUnit** SpeedUnit | The unit used to define the Resource's speed properties |
| **derivation** Derivation | A description of the means by which the data for the Resource Property's attributes were derived. This includes sensing, communication routes and human input. |

### 9.1.23 ResourceProperties

**Type:** Class
**Package:** PlanExecution
The static, persistent properties of the resource that are not expected to change as a plan is proposed and executed. Properties are specified in this data model that are expected to be particular pertinent to the planning of operational utilization of resources. For instance those that provide constraints on movement and demand conditions on the operating environment.

**Table 9.21 - Attributes of Class ResourceProperties**

| Attribute | Notes |
|---|---|
| **category** ResourceCategory | The extensible categorization of the type of resource |
| **extendedData** AdditionalData | Additional static information related to the Resource |
| «key» **id** ResourceRef | The unique identifier for the instance |

### 9.1.24 ResourceTasking

**Type:** Class
**Package:** PlanExecution
Resource Tasking is a Resource's contribution to a Task Objective

**Table 9.22 - Attributes of Class ResourceTasking**

| Attribute | Notes |
|---|---|
| **activity** TaskingActivity | An extensible categorization of the activity that the resource has been tasked to undertake. |

### 9.1.25 Space

**Type:** Class
**Package:** PlanExecution
A resource beyond the Earth's atmosphere

**Table 9.23 - Attributes of Class Space**

| Attribute | Notes |
|---|---|
| **orbit** OrbitCategory | The kind of orbit or trajectory that the space resource is on |

### 9.1.26 SubsurfaceVessel

**Type:** Class
**Package:** PlanExecution
A resource operating underwater

**Table 9.24 - Attributes of Class SubsurfaceVessel**

| Attribute | Notes |
|---|---|
| **maxOperatingDepth** Distance [0..1] | The operating limit of the subsurface vehicle with respect to depth in the water |

### 9.1.27 SurfaceVessel

**Type:** Class
**Package:** PlanExecution
A resource operating on water.

**Table 9.25 - Attributes of Class SurfaceVessel**

| Attribute | Notes |
|---|---|
| **maxSeaState** Integer [0..1] | The maximum sea state in which the Surface Vessel can operate specified in terms of the World Meteorological Organization (WMO) sea state code - range 0 .. 9. |

### 9.1.28 SurveillanceCapability

**Type:** Class
**Package:** PlanExecution
A capability to sense or observe objects in the operational environment.

**Table 9.26 - Attributes of Class SurveillanceCapability**

| Attribute | Notes |
|---|---|
| **reportingRate** Quantifier [0..1] | |
| **operatingBand** Descriptor | A qualitative description of the band in which the Surveillance Capability operates |
| **targetClassification** ClassificationDescriptor [0..*] | The categories of object which the Surveillance Capability can detect |
| **capacity** Integer [0..1] | The number of objects that the Surveillance Capability can continuously monitor. |

### 9.1.29 TargetCapability

**Type:** Class
**Package:** PlanExecution

**Table 9.27 - Attributes of Class TargetCapability**

| Attribute | Notes |
|---|---|
| **classification** ClassificationDescriptor | A category of target for which the resource has a Fire Capability |
| **successLikelihood** Percentage | The nominal likelihood that an engagement with the specified target will be successful. |

### 9.1.30 TaskObjective

**Type:** Class
**Package:** PlanExecution
A Task Objective represents the discrete intent with respect to a particular Entity from the tactical picture within the context of an overall Plan

**Table 9.28 - Attributes of Class TaskObjective**

| Attribute | Notes |
|---|---|
| **entityObject** EntityRef [0..1] | The Entity from the tactical picture to which the intent of the Task Objective is directed. |
| **category** ObjectiveCategory | An extensible categorization of the kind of tasking objective set. |
| **detail** Detail | Extensible, additional system and/or domain specific description of the tasking objective |
| **intent** IntentDescriptor | Extensible categorization of the kind of effect intended with respect to the object of the tasking |
| **priority** Percentage [0..1] | Optional prioritization of task objectives. HIgher precentages reflect higher priorities. The values for all objectives for a plan are not required to sum to 100%. |

### 9.1.31 TransmissionCapability

**Type:** Class
**Package:** PlanExecution
A capability for the electronic transmission of data (including voice and video).

**Table 9.29 - Attributes of Class TransmissionCapability**

| Attribute | Notes |
|---|---|
| **power** Quantifier [0..1] | The nominal output |
| **dataRate** Quantifier [0..1] | The rate at which the Transmission Capability can transmit data |
| **operatingBand** Descriptor | A qualitative description of the band in which the Transmission Capability operates |

| Attribute | Notes |
|---|---|
| **protocol** Descriptor [0..*] | The transmission protocols supported by the capability. |
| **dataClassification** Descriptor [0..*] | The classification of information supported by the capability |

### 9.1.32 Vehicle

**Type:** Class
**Package:** PlanExecution
A resource with its own movement capabilities

**Table 9.30 - Attributes of Class Vehicle**

| Attribute | Notes |
|---|---|
| **maxSpeed** Speed [0..1] | The maximum speed that at which the vehicle can move |
| **cruisingSpeed** Speed [0..1] | The optimum speed, for planning purposes, at which the vehicle transits between locations. |
| **turnRate** Quantifier [0..1] | The rate at which the vehicle can maneuver to change its heading within the horizontal plane (for planning purposes). |

### 9.1.33 Vessel

**Type:** Class
**Package:** PlanExecution
A waterborne resource

**Table 9.31 - Attributes of Class Vessel**

| Attribute | Notes |
|---|---|
| **minOperatingDepth** Distance [0..1] | The vessel resource's operating limit with respect to shallow water |

### 9.1.34 AmmunitionCategory

**Type:** DataType
**Package:** PlanExecution
An abstraction of the categories of ammunition

### 9.1.35 CaliberCategory

**Type:** DataType
**Package:** PlanExecution
An abstraction of the categories of ammunition caliber.

### 9.1.36 CapabilityCategory

**Type:** DataType
**Package:** PlanExecution
An abstraction of the categories of capabilities

### 9.1.37 CapabilityRef

**Type:** DataType
**Package:** PlanExecution
A datatype with a platform specific mapping to represent a reference to a Capability. A reference is a unique identifier within the scope of the Plan Execution component implementing this specification.

### 9.1.38 ConstituentRef

**Type:** DataType
**Package:** PlanExecution
A reference to a Plan Execution Constituent. A reference is a unique identifier within the scope of the Plan Execution component implementing this specification.

### 9.1.39 DependencyCategory

**Type:** DataType
**Package:** PlanExecution
An abstraction of the categories of dependencies

### 9.1.40 DerivationDescriptor

**Type:** DataType
**Package:** PlanExecution
System specific description of the derivation of the associated data

### 9.1.41 ExtendedPlanStatus

**Type:** DataType
**Package:** PlanExecution
An abstraction of additional sub-categories of plan status; each sub-category logically maps to a specific plan state

### 9.1.42 IntentDescriptor

**Type:** DataType
**Package:** PlanExecution
An abstraction of the categories of intent.

### 9.1.43 ObjectiveCategory

**Type:** DataType
**Package:** PlanExecution
An abstraction of the categories of task objectives.

### 9.1.44 OrbitCategory

**Type:** DataType
**Package:** PlanExecution
An abstraction of the categories of orbits in space

### 9.1.45 PlanExecutionConstituentState

**Type:** Enumeration
**Package:** PlanExecution
Representation of the state machine for plan constituents.

**Table 9.32 - Attributes of Enumeration PlanExecutionConstituentState**

| Attribute | Notes |
|---|---|
| «enum» **PLANNED** | The plan constituent has been created but is not yet being executed |
| «enum» **EXECUTING** | The plan constituent has been started but terminated and has been resumed after any pause. |
| «enum» **PAUSED** | The plan constituent has been paused, but not yet resumed after being executed. |
| «enum» **TERMINATED** | The plan constituent has been terminated after being executed. |

### 9.1.46 PlanType

**Type:** DataType
**Package:** PlanExecution
An abstraction of the categories of plans.

### 9.1.47 ReadinessDescriptor

**Type:** DataType
**Package:** PlanExecution
An abstraction of the categories of readiness

### 9.1.48 ResourceCategory

**Type:** DataType
**Package:** PlanExecution
An abstraction of the categories of resources.

### 9.1.49 ResourceRef

**Type:** DataType
**Package:** PlanExecution
A datatype with a platform specific mapping to represent a reference to a Resource. A reference is a unique identifier within the scope of the Plan Execution component implementing this specification.

### 9.1.50 SpecificationDescriptor

**Type:** DataType
**Package:** PlanExecution
An abstraction of the categories of specifications

### 9.1.51 TaskingActivity

**Type:** DataType
**Package:** PlanExecution
An abstraction of the categories of tasking activity a resource can undertake.

## 9.2 TacticalPicture

**Parent Package:** DataModel
The Tactical Picture package in the Data Model describes the particular usage of the TACSIT Data Exchange (TEX) standard that satisfies this standard's tactical picture requirements.

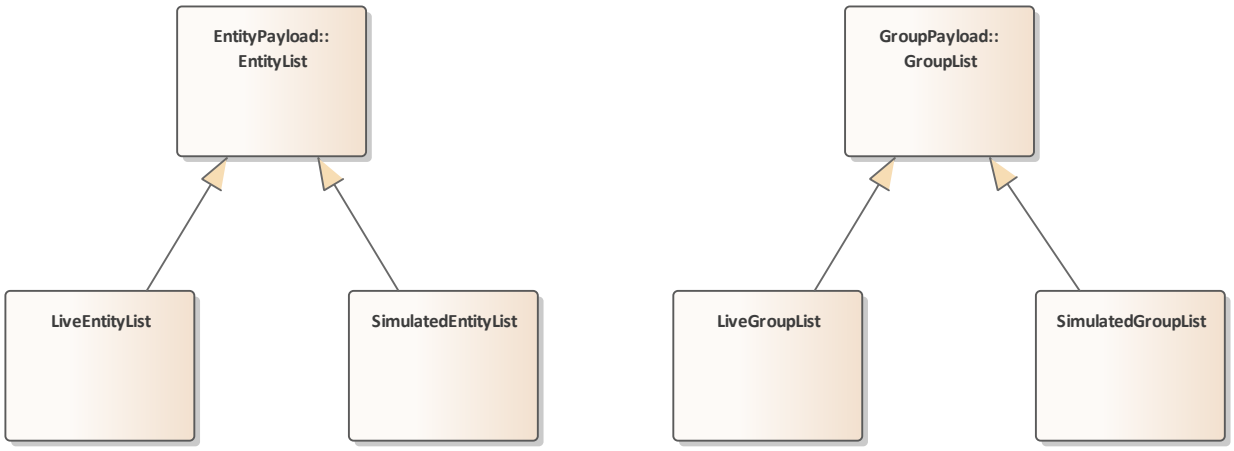

**Figure 9.14 Live Simulated (Class diagram)**

This diagram shows how the live and simulated versions of the tactical picture are represented using classes from the TACSIT Data Exchange (TEX) specification.


**Figure 9.15 Track Categorization (Class diagram)**

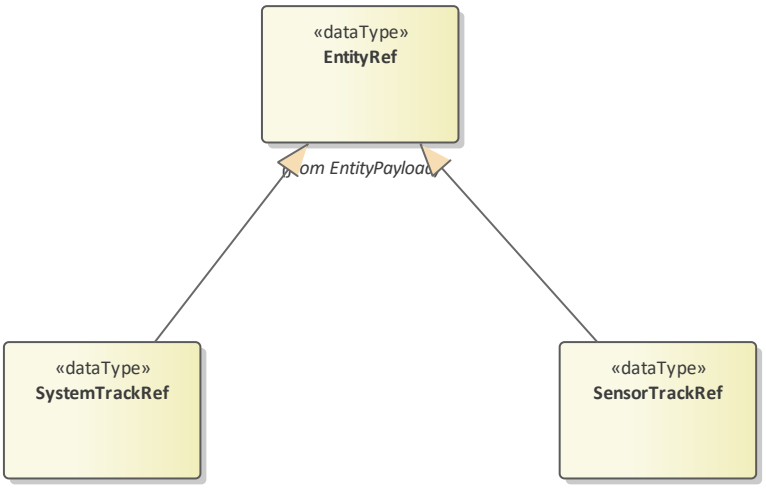


**Figure 9.16 Tracks (Class diagram)**

This diagram shows how system tracks and sensor tracks are represented using a reference to the Entity class from the TACSIT Data Exchange (TEX) specification.

### 9.2.1 LiveEntityList

**Type:** Class
**Package:** TacticalPicture
The list of entities contributing to the live tactical picture (i.e. relating to the real operational environment)

### 9.2.2 LiveGroupList

**Type:** Class
**Package:** TacticalPicture
The list of groups contributing to the live tactical picture (i.e. relating to the real operational environment)

### 9.2.3 SimulatedEntityList

**Type:** Class
**Package:** TacticalPicture
The list of entities contributing to the simulated tactical picture (i.e. relating to a simulation of the operational environment)

### 9.2.4 SimulatedGroupList

**Type:** Class
**Package:** TacticalPicture
The list of groups contributing to the simulated tactical picture (i.e. relating to a simulation of the operational environment)

### 9.2.5 ActivityDescriptor

**Type:** DataType
**Package:** TacticalPicture
Extensible definition of a track's activity.

### 9.2.6 ClassificationDescriptor

**Type:** DataType
**Package:** TacticalPicture
Extensible definition of a track's classification.

### 9.2.7 EntityStatusDescriptor

**Type:** DataType
**Package:** TacticalPicture
Extensible definition of an enitity's status.

### 9.2.8 IdentityDescriptor

**Type:** DataType
**Package:** TacticalPicture
Extensible definition of a track's identity.

### 9.2.9 SensorTrackRef

**Type:** DataType
**Package:** TacticalPicture
A reference to a sensor track - i.e. a track object from the perspective of a sensor subsystem

### 9.2.10 SystemTrackRef

**Type:** DataType
**Package:** TacticalPicture
A reference to a system track - i.e. a track object from the perspective of the compiled tactical picture of a C2 (Command and Control) system.

# 10 ServiceModel

**Parent Package:** tactical decision aids
The Tactical Decision Aids Service Model defines the operations that enable the flow of information from a Picture Management and a Plan Execution component to Tactical Decision Aid Components as well as the receipt of recommendations from Tactical Decision Aid components by the Tactical Picture and Plan Execution components.
The connection between components is initiated by the Tactical Decision Aid components using a PSM method. These components may require security permissions to do, in which case these are authenticated by a PSM protocol.
Use of a Data Sink Listener to subscribe to a series of change events requires a long-lived connection between the Tactical Decision Aid and the Tactical Picture or Plan Execution component providing the Data Sink interface. Other interface operations are self-contained requests initiated by the Tactical Decision Aid component and do not require long-lived connections.
Tactical Decision Aids components make recommendations based on their own internal business logic, information they have access to by other means (including input from system users) and information received from the Tactical Picture and Plan Execution components through the Data Sink interfaces.

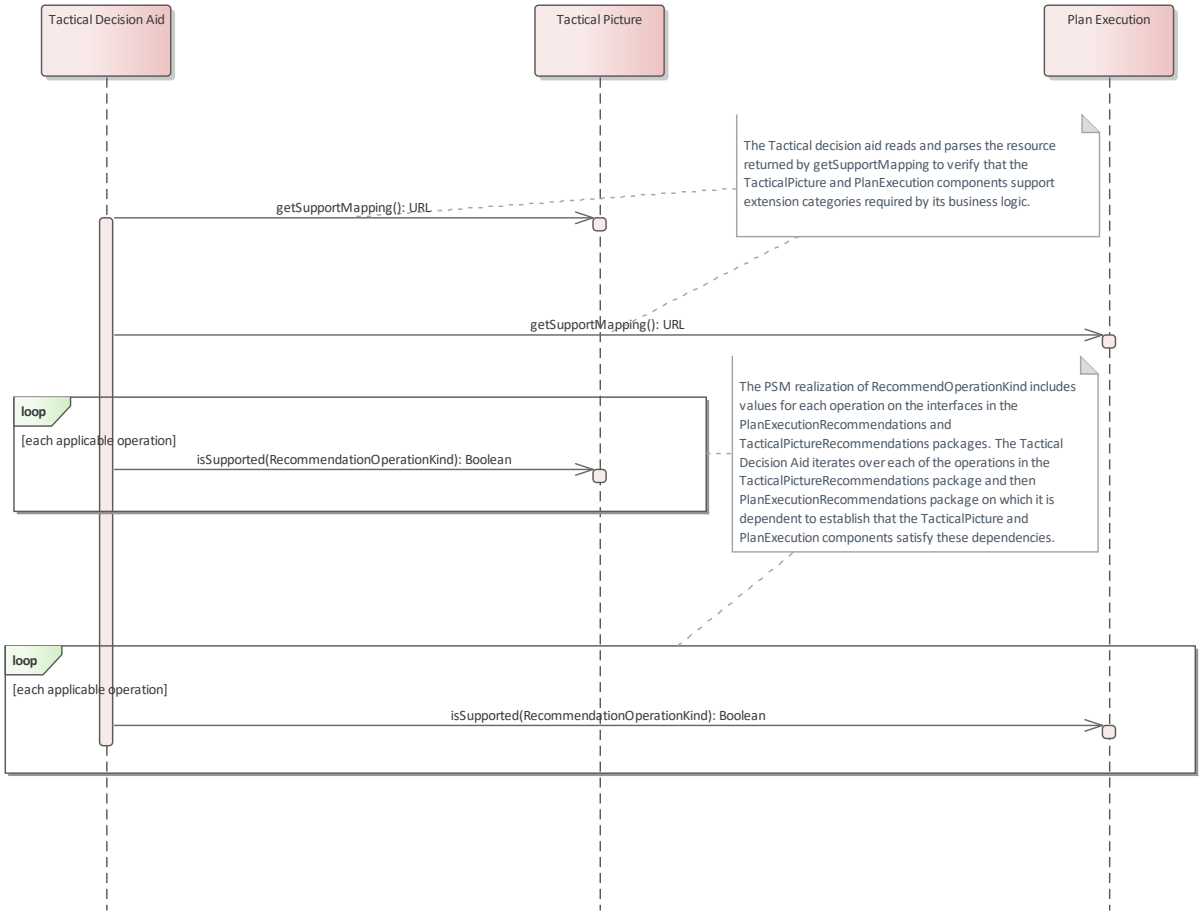


**Figure 10.17 Recommendations (Interaction diagram)**

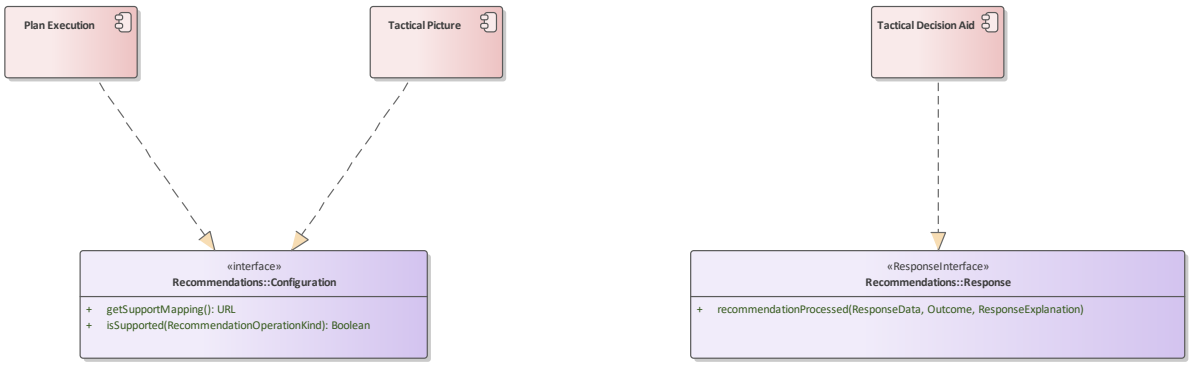


**Figure 10.18 Recommendations Service Mapping (Component diagram)**

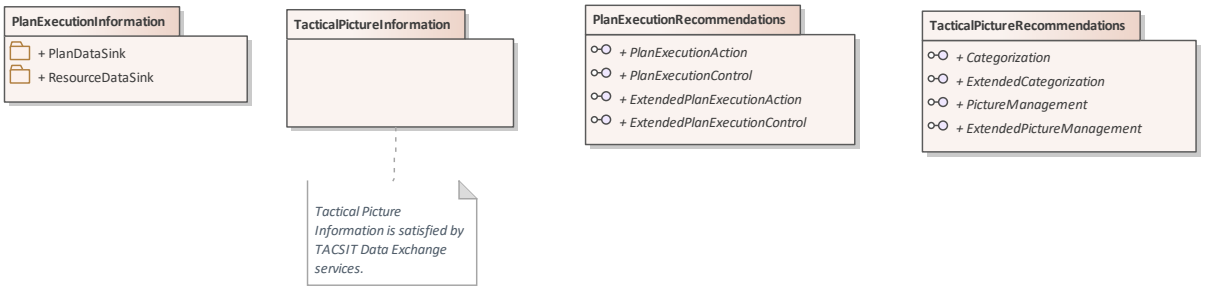


**Figure 10.19 ServiceModel (Package diagram)**

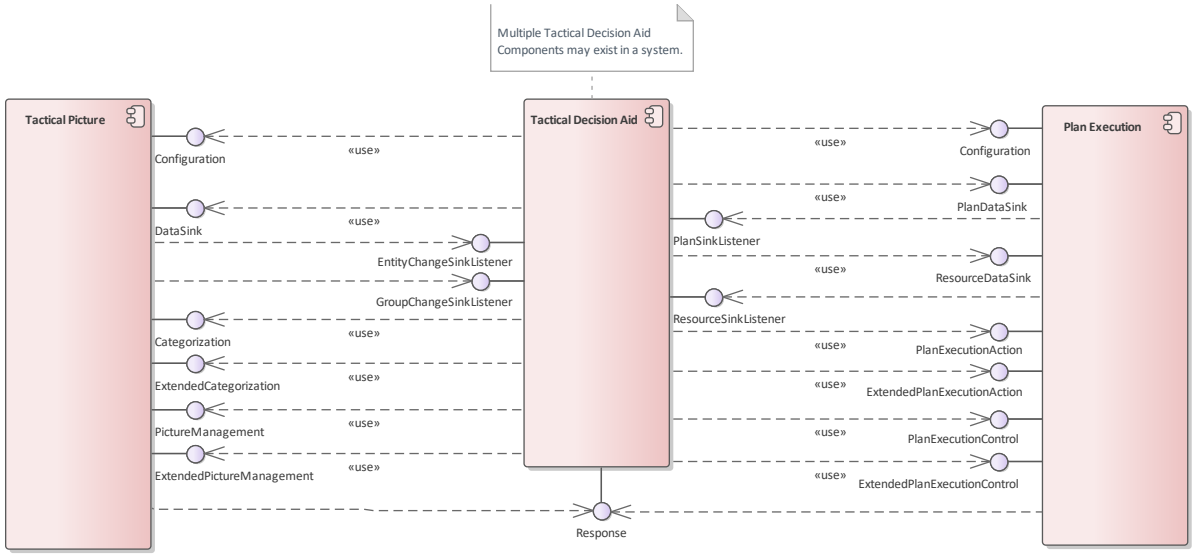


**Figure 10.20 ServiceModel (Component diagram)**

## 10.1 Plan Execution

**Type:** Component
**Package:** ServiceModel
Abstract component representing components with the functionality to manage and monitor the status of plans as they are executed. Plan Execution components receive information from system users, tactical data-links, databases and other components through interfaces outside of the scope of this specification. Tactical Decision Aids receive information about all plans; the plans they have initiated, plans from other Tactical Decision Aids and plans originating from outside the scope of this specification.

## 10.2 Tactical Decision Aid

**Type:** Component
**Package:** ServiceModel
Abstract component representing components that provide the functionality to assist with the making of tactical decisions

## 10.3 Tactical Picture

**Type:** Component
**Package:** ServiceModel
Abstract component representing components that provide the functionality of compiling and managing the tactical picture. Tactical Picture components receive information from sensors, system users and other components through interfaces outside of the scope of this specification.

## 10.4 PlanExecutionInformation

**Parent Package:** ServiceModel
The interfaces to allow Tactical Decision Aids to receive Plan Execution Information. This is achieved through two instances of the Data Sink pattern. One enables Tactical Decision Aids to receive a current view and then changes to Plan Execution Constituents (Plans and their sub-components Task Objectives and Resource Taskings). The other provides an equivalent service for Resources and their composite Capabilities and Dependencies.
Navigability of associations between classes in the Plan Execution data model is facilitated by id attributes with a key stereotype. Navigation of the associations between objects delivered by the Data Sink services is achieved by a PSM specific methods using the id key attributes.

### 10.4.1 PlanDataSink

**Parent Package:** PlanExecutionInformation
The interfaces to allow Tactical Decision Aids to receive Plan Information

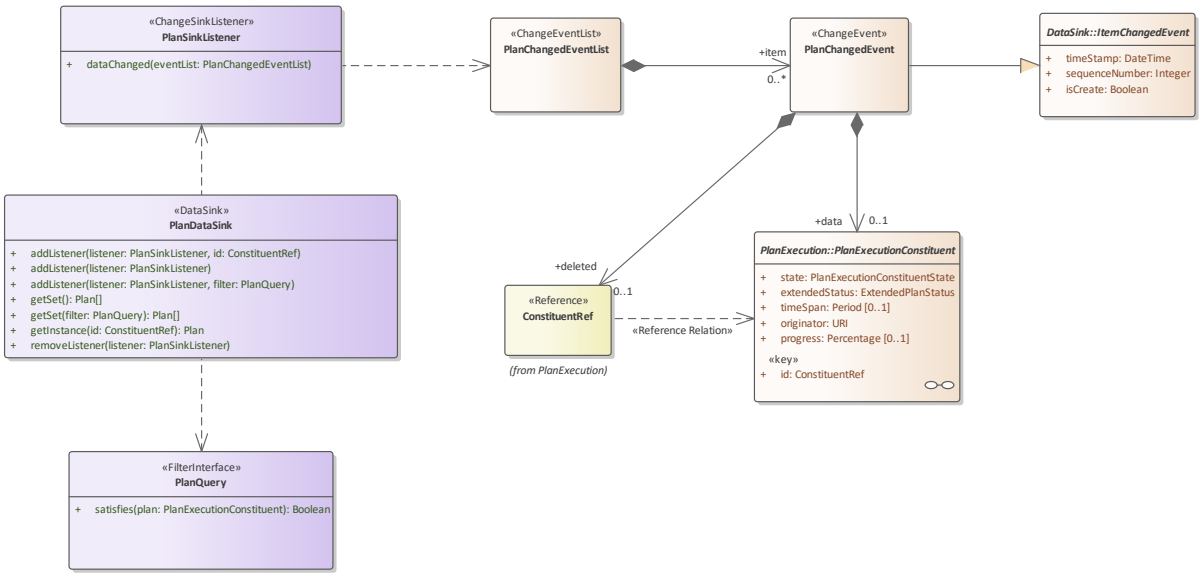


**Figure 10.21 PlanDataSink (Class diagram)**

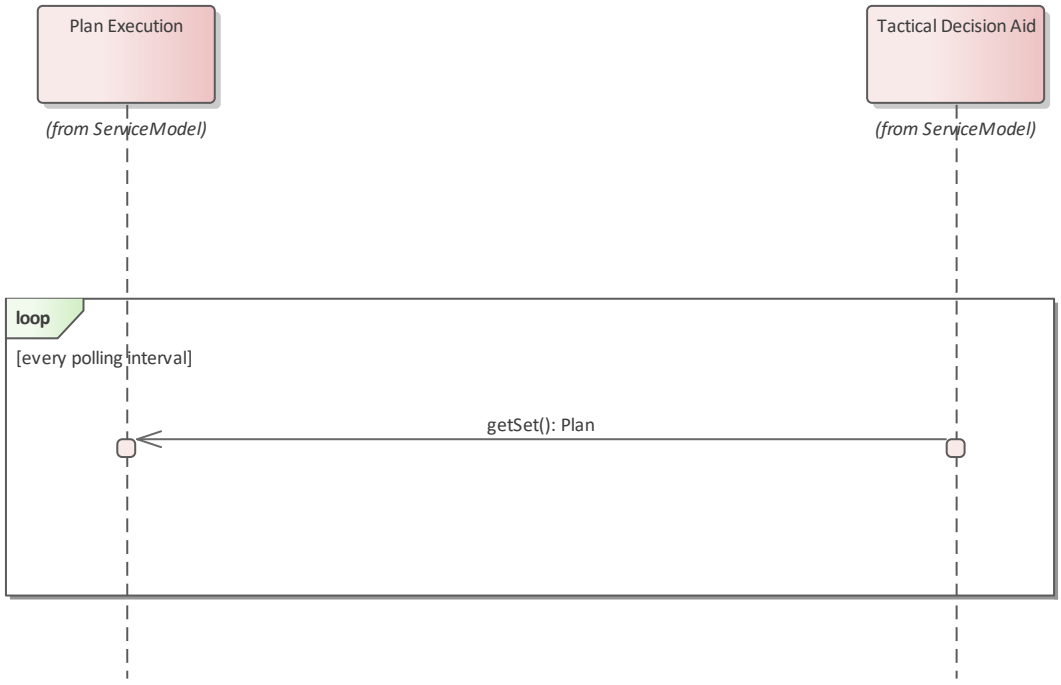


**Figure 10.22 PlanDataSink - All Plan - Polling (Interaction diagram)**

Use of the PlanDataSink interface to get a regular view of all plans. To receive a subset of plans the getSet with a PlanQuery parameter operation is used.


**Figure 10.23 PlanDataSink - All Plans - On Change (Interaction diagram)**

Use of the PlanDataSink interface to get an initial view of all plans and then receive updates on changes for an on-change style of use of the interface. The listener interface is added first so that events are not missed. In this scenario it is preferable to process no-change events than to miss events.

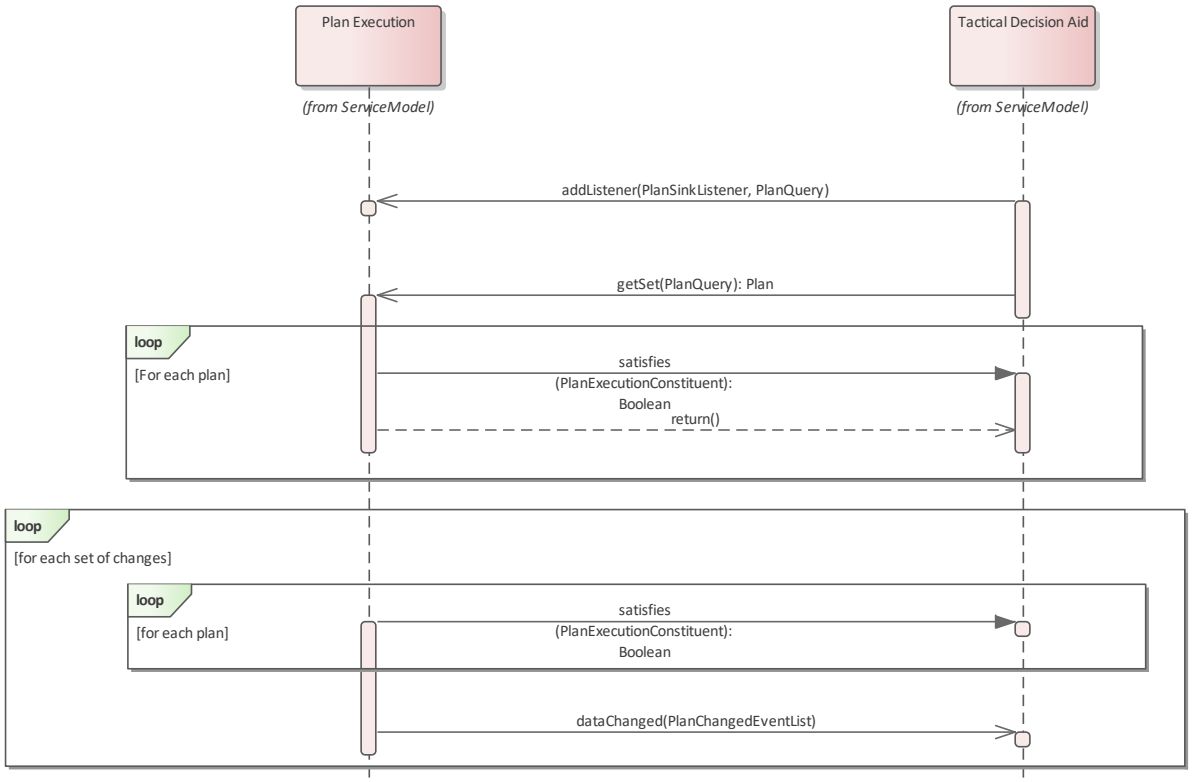


**Figure 10.24 PlanDataSink - Filtered Plans (Interaction diagram)**

Use of the PlanDataSink interface to get an initial view of a subset of plans and then receive updates on changes.


**Figure 10.25 PlanDataSink - Single Plan (Interaction diagram)**

Use of the PlanDataSink interface to get an initial view of a specific plan and then receive updates on changes.

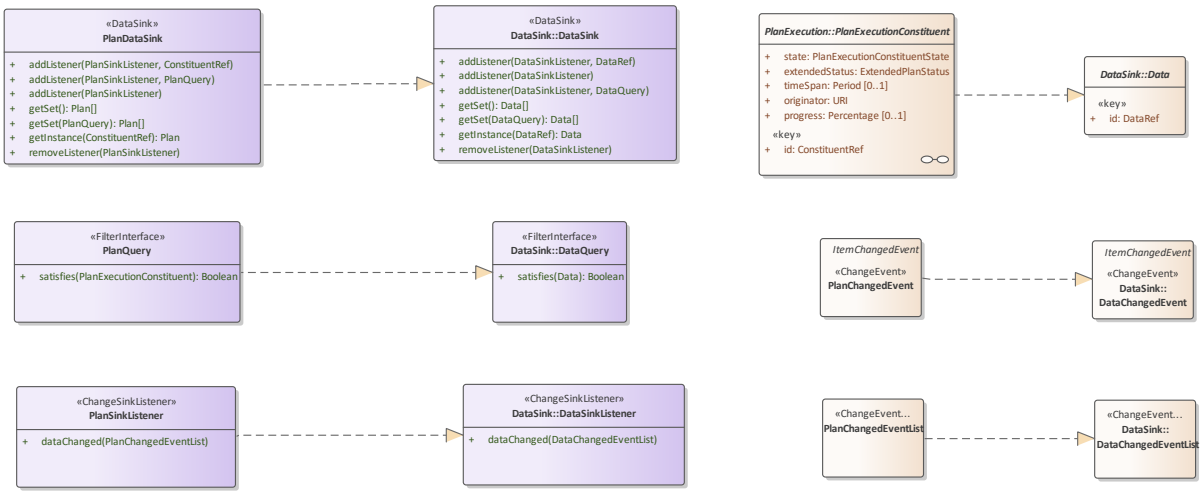


**Figure 10.26 PlanDataSink Realization (Class diagram)**

#### 10.4.1.1 PlanChangedEvent

**Type:** Class
**Package:** PlanDataSink
Represents information about a change to a Plan

#### 10.4.1.2 PlanChangedEventList

**Type:** Class
**Package:** PlanDataSink
Represents the list of changes to Plans since the last event notified to that instance of the listener. Multiple changes may be consolidated into a single callback to a listener on the interface

#### 10.4.1.3 PlanDataSink

**Type:** Interface
**Package:** PlanDataSink
This interface contains operations that give a Tactical Decision Aid access to information about the execution of plan constituents. A Tactical Decision Aid can add and remove listeners as well as reading the information about individual plan constituents or all or a filtered subset of plan constituents.

**Table 10.1 - Methods of Interface PlanDataSink**

| Method | Notes | Parameters |
|---|---|---|
| **addListener()** | Operation to add a listener for callbacks relating to a single plan constituent | PlanSinkListener **listener** The listener object to receive the callback ConstituentRef **id** A reference to the specific plan instance of interest |
| **addListener()** | Operation to add a listener for callbacks relating to all plan constituents that satisfy the Query (including plans created or meeting the query subsequently) | PlanSinkListener **listener** The listener object to receive the callback PlanQuery **filter** The object to filter changes to plans |
| **addListener()** | Operation to add a listener for callbacks relating to all plan constituents, including plan constituents subsequently created. | PlanSinkListener **listener** The listener object to receive the callback |
| **getSet()** | Operation to obtain the information relating to all plan constituents | |
| **getSet()** | Operation to obtain the information relating to all the plan constituents satisfying the query | PlanQuery **filter** The object to filter plan instances |
| **getInstance()** | Operation to obtain the information relating to the plan constituent reference | ConstituentRef **id** A reference to the specific plan instance of interest |
| **removeListener()** | Operation to remove a listener | PlanSinkListener **listener** The listener object to no longer receive callbacks |

#### 10.4.1.4 PlanQuery

**Type:** Interface
**Package:** PlanDataSink
This is an interface through which a client can define Queries on plan constituents so as to filter the information returned. Classes implementing the interface provide means to set the query parameters (such as a constructor).

**Table 10.2 - Methods of Interface PlanQuery**

| Method | Notes | Parameters |
|---|---|---|
| **satisfies()** | This operation is the client's implementation of a filtering query for plan constituents | PlanExecutionConstituent **plan** The plan to which to apply the filter |

#### 10.4.1.5 PlanSinkListener

**Type:** Interface
**Package:** PlanDataSink
This is an interface for clients to implement callback to receive information on changes to plan constituents.

**Table 10.3 - Methods of Interface PlanSinkListener**

| Method | Notes | Parameters |
|---|---|---|
| **dataChanged()** | This operation is implemented by the client to process the data changed callback. Multiple changes can be notified through a single invocation. | PlanChangedEventList **eventList** The list of plan changes recevied by the listener |

## 10.4.2 ResourceDataSink

**Parent Package:** PlanExecutionInformation
The interfaces to allow Tactical Decision Aids to receive Resource Information

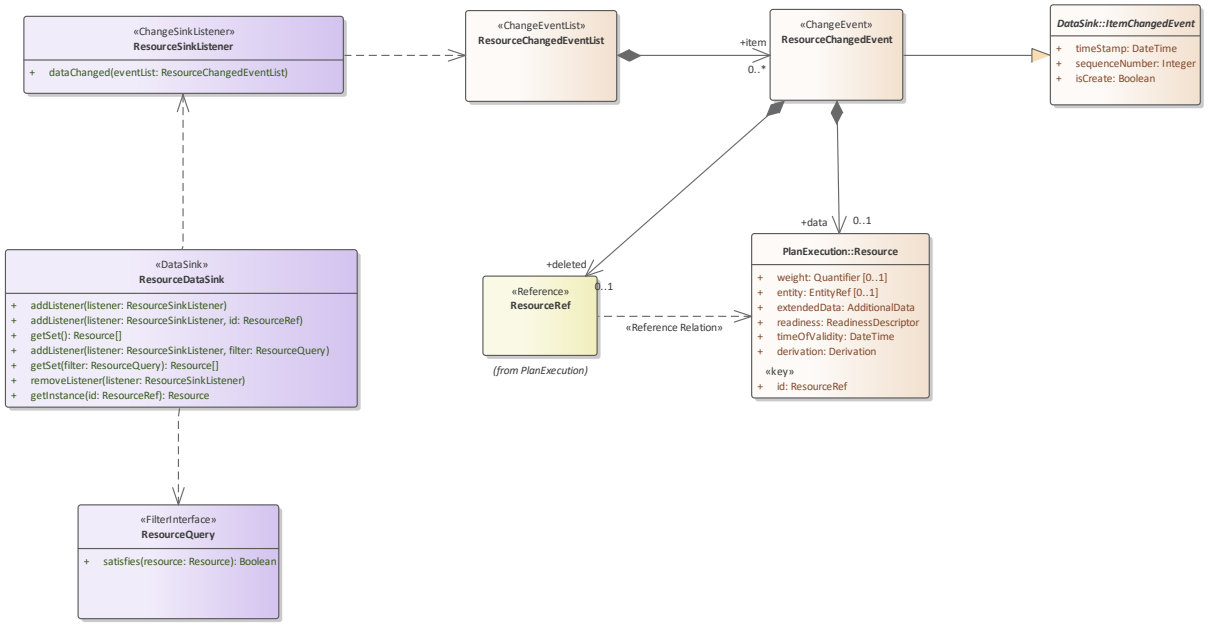


**Figure 10.27 ResourceDataSink (Class diagram)**

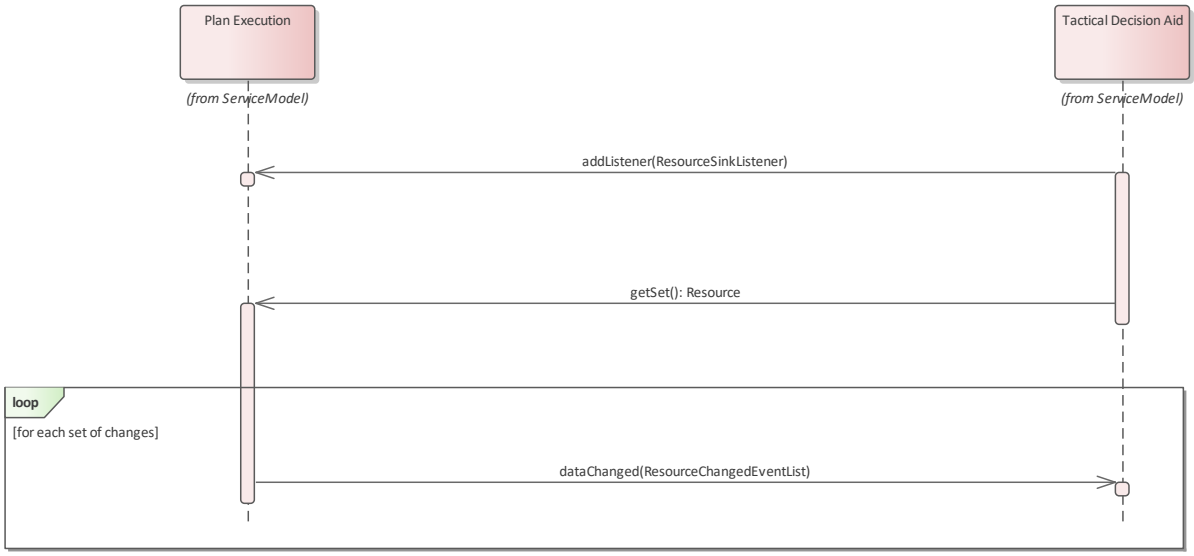


**Figure 10.28 ResourceDataSink - All Resources - On Change (Interaction diagram)**

Use of the ResourceDataSink interface to get an initial view of all resources and then receive updates on changes for an on-change style of use of the interface. The listener interface is added first so that events are not missed. In this scenario it is preferable to process no-change events than to miss events.

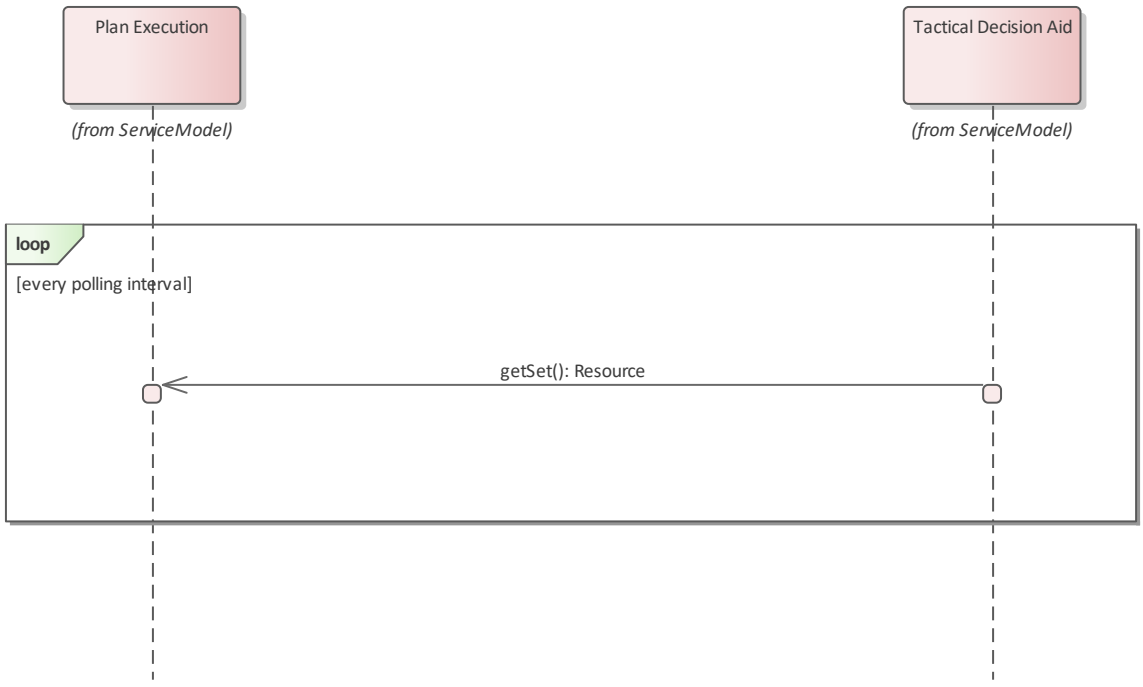


**Figure 10.29 ResourceDataSink - All Resources - Polling (Interaction diagram)**

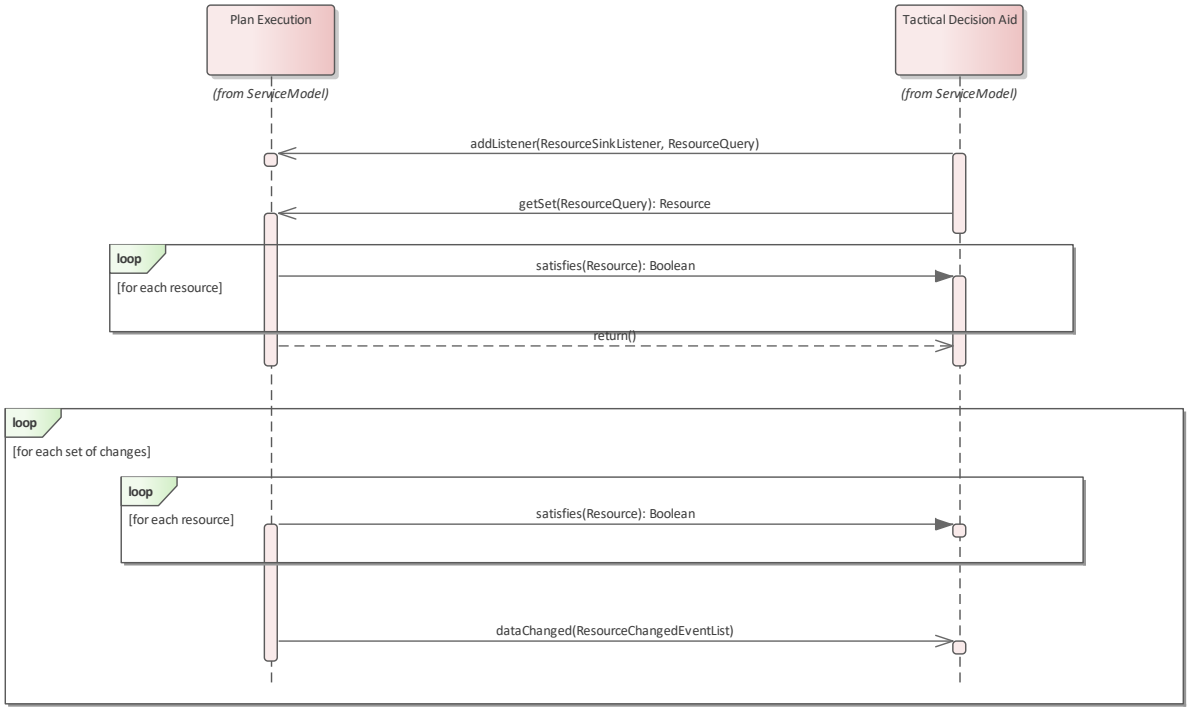


**Figure 10.30 ResourceDataSink - Filtered Resources (Interaction diagram)**

Use of the ResourceDataSink interface to get an initial view of a subset of resources and then receive updates on changes.


**Figure 10.31 ResourceDataSink - Single Resource (Interaction diagram)**

Use of the ResourceDataSink interface to get an initial view of a specific resource and then receive updates on changes.

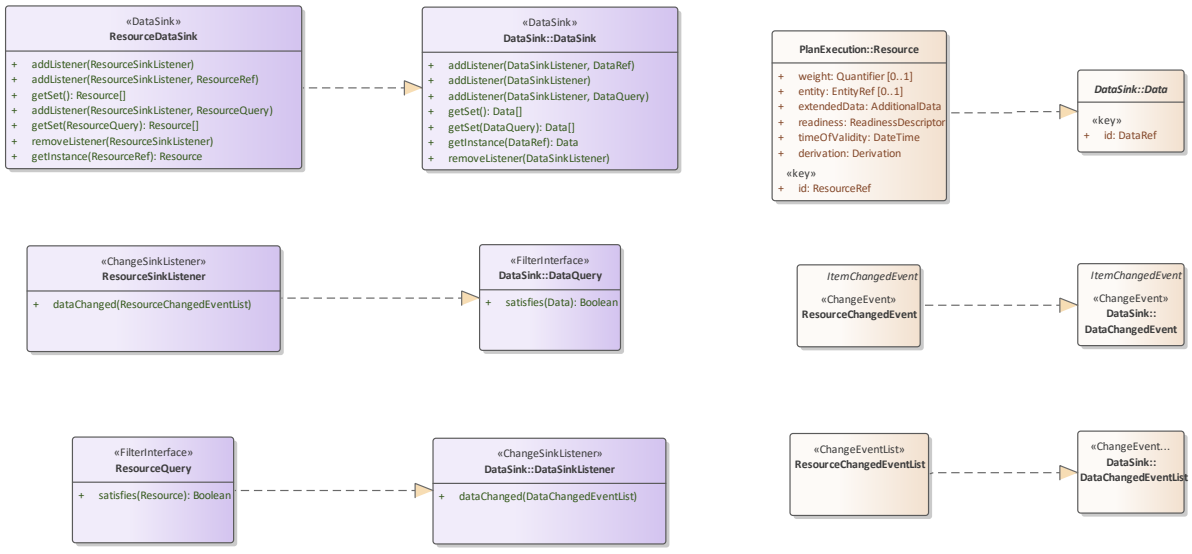


**Figure 10.32 ResourceDataSink Realization (Class diagram)**

#### 10.4.2.1 ResourceChangedEvent

**Type:** Class
**Package:** ResourceDataSink
Represents information about a change to a Resource

#### 10.4.2.2 ResourceChangedEventList

**Type:** Class
**Package:** ResourceDataSink
Represents the list of changes to Resources since the last event notified to that instance of the listener. Multiple changes may be consolidated into a single callback to a listener on the interface

#### 10.4.2.3 ResourceDataSink

**Type:** Interface
**Package:** ResourceDataSink
This interface contains operations that give a Tactical Decision Aid access to information about resources that can execute plans. A Tactical Decision Aid can add and remove listeners as well as reading the information about individual resources or all or a filtered subset of resources.

**Table 10.4 - Methods of Interface ResourceDataSink**

| Method | Notes | Parameters |
|---|---|---|
| **addListener()** | Operation to add a listener for callbacks relating to a single Resource (including Resources created subsequently) | ResourceSinkListener **listener** The listener object to receive the callback |
| **addListener()** | Operation to add a listener for callbacks relating to all Resources | ResourceSinkListener **listener** The listener object to receive the callback ResourceRef **id** A reference to the specific resource instance of interest |

| | | |
|---|---|---|
| **getSet()** | Operation to obtain the information relating to all the Resources | |
| **addListener()** | Operation to add a listener for callbacks relating to all Resources that satisfy the Query (including Resources created or meeting the filter subsequently) | ResourceSinkListener **listener** The listener object to receive the callback ResourceQuery **filter** The object to filter changes to resources |
| **getSet()** | Operation to obtain the information relating to all the Resources satisfying the query | ResourceQuery **filter** The object to filter resource instances |
| **removeListener()** | Operation to remove a listener | ResourceSinkListener **listener** The listener object to no longer receive callbacks |
| **getInstance()** | Operation to obtain the information relating to the Resource reference | ResourceRef **id** A reference to the specific resource instance of interest |

#### 10.4.2.4 ResourceQuery

**Type:** Interface
**Package:** ResourceDataSink
This is an interface through which a client can define Queries on Resources so as to filter the information returned. Classes implementing the interface provide means to set the query parameters (such as a constructor).

**Table 10.5 - Methods of Interface ResourceQuery**

| Method | Notes | Parameters |
|---|---|---|
| **satisfies()** | This operation is the client's implementation of a filtering query for Resources | Resource **resource** The resource to which to apply the filter |

#### 10.4.2.5 ResourceSinkListener

**Type:** Interface
**Package:** ResourceDataSink
This is an interface for clients to implement callback to receive information on changes to Resources

**Table 10.6 - Methods of Interface ResourceSinkListener**

| Method | Notes | Parameters |
|---|---|---|
| **dataChanged()** | This operation is implemented by the client to process the dataChanged callback. Multiple changes can be notified through a single invocation. | ResourceChangedEventList **eventList** The list of resource changes recevied by the listener |

## 10.5 PlanExecutionRecommendations

**Parent Package:** ServiceModel

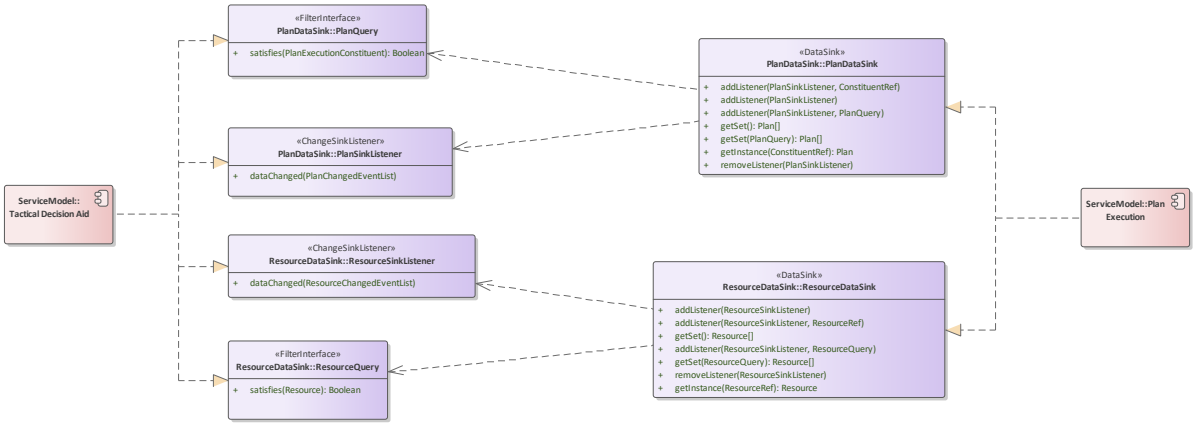


**Figure 10.33 PlanExecutionInformation Service Mapping (Component diagram)**

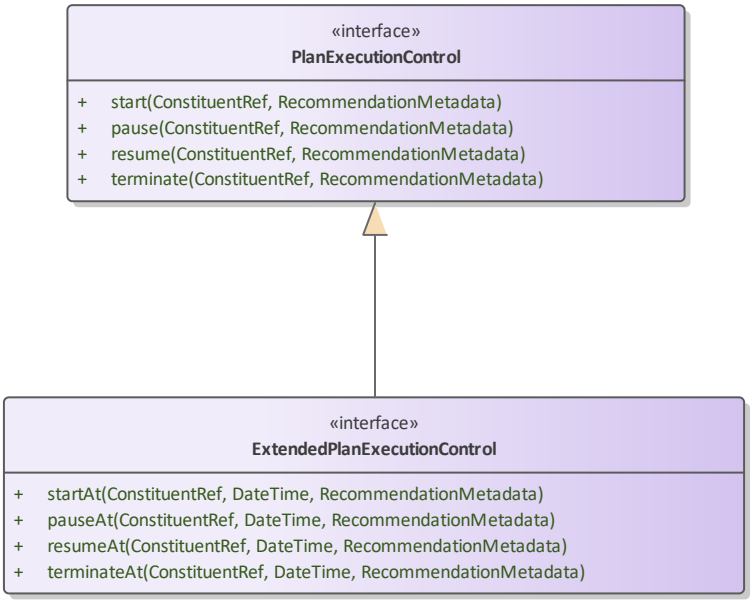


**Figure 10.34 ActionControlRecommendation (Class diagram)**

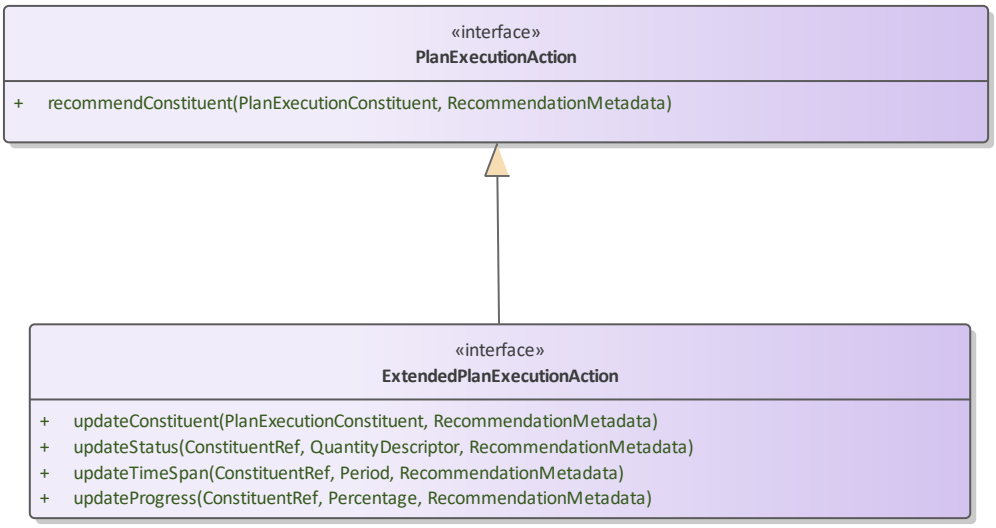


**Figure 10.35 ActionRecommendation (Class diagram)**

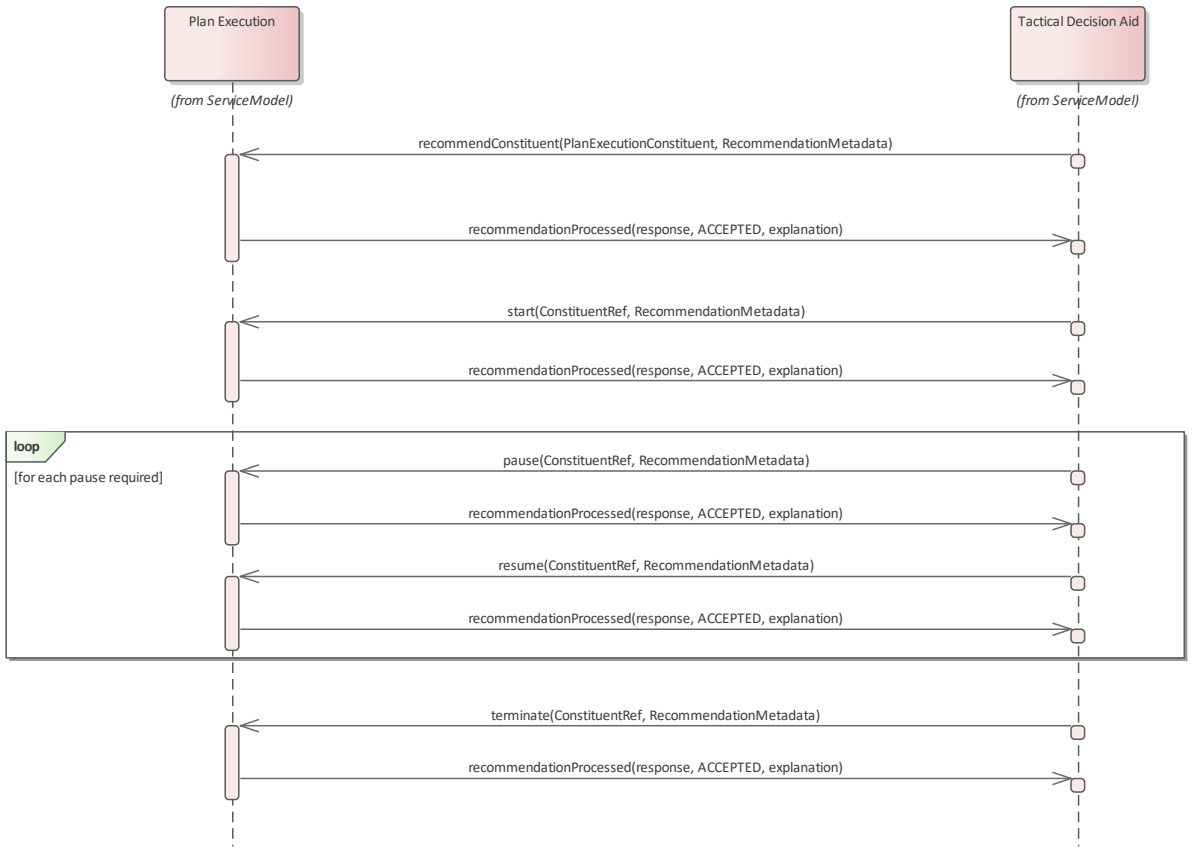


**Figure 10.36 PlanExecutionRecommendations (Interaction diagram)**

Use of the PlanExecutionAction and PlanExecutionControl interfaces to recommend and then control the execution of a plan constituent. The ConstituentRef references the PlanExecutionConstituent. In this

scenario all recommendations are successfully accepted. It is valid for a Tactical Decision Aid to make additional recommendations before a recommendationProcessed response is received.

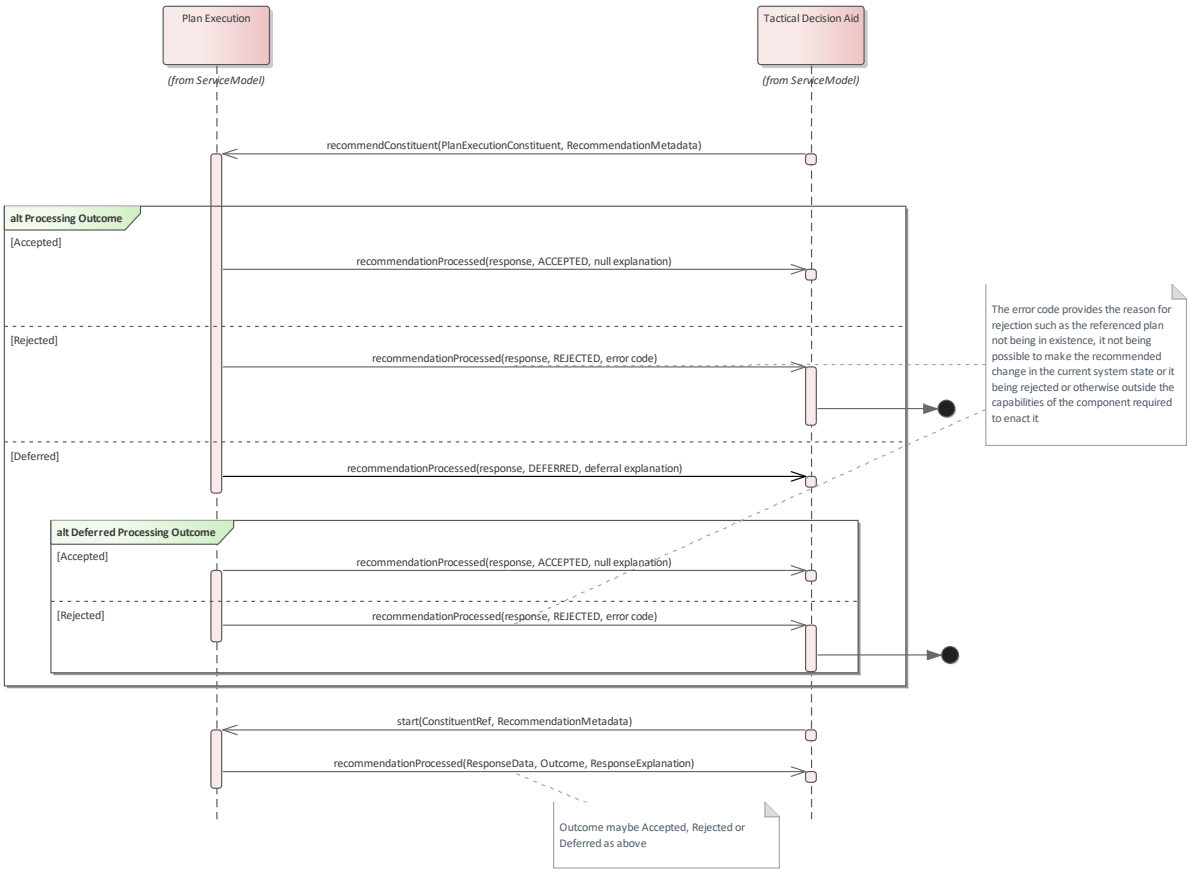


**Figure 10.37 PlanExecutionRecommendations - Alternate Flows (Interaction diagram)**

Use of the PlanExecutionAction and PlanExecutionControl interfaces to recommend and then control the execution of a plan constituent. In this set of scenarios not all recommendations are successfully accepted.

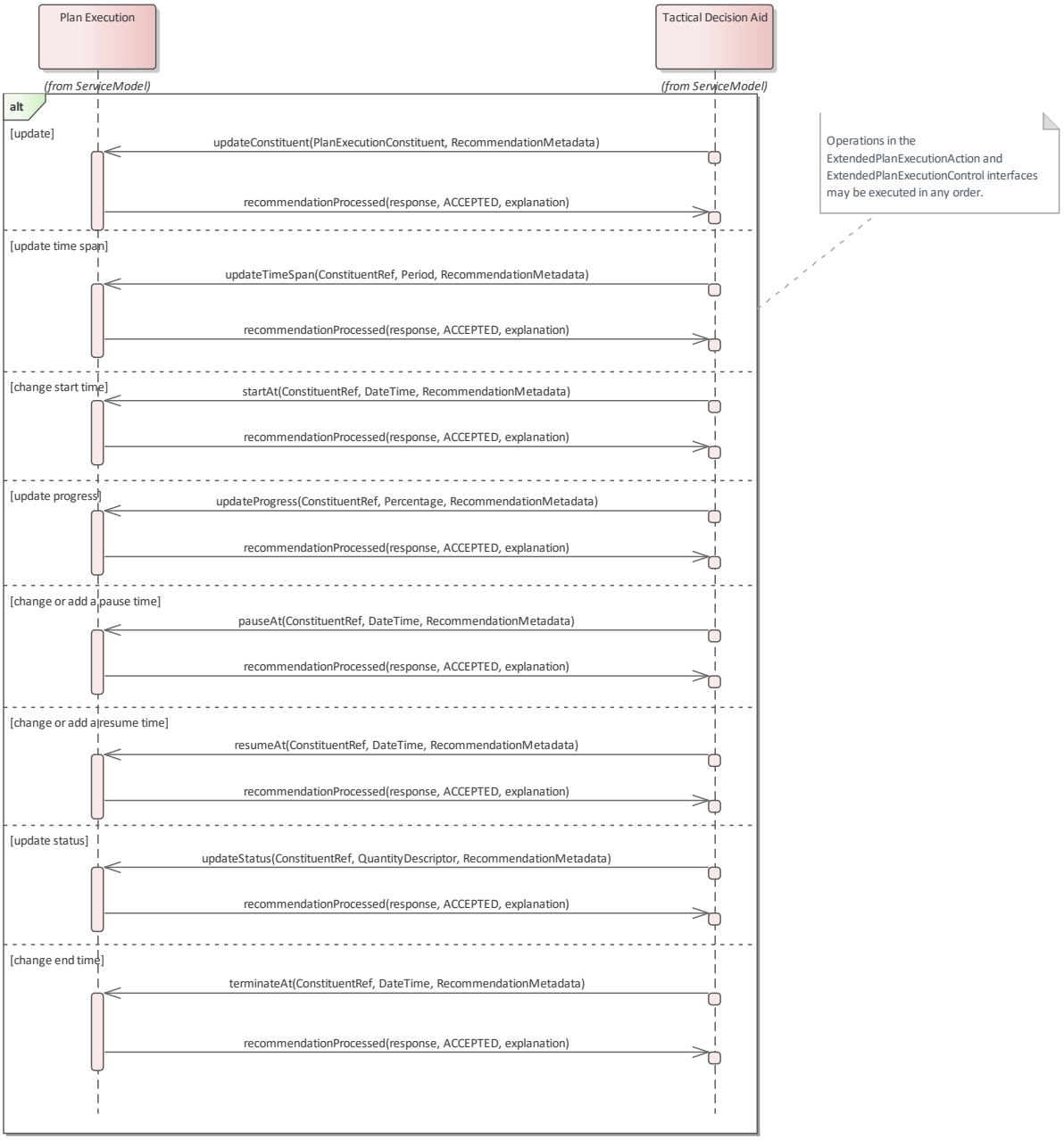


**Figure 10.38 ExtendedPlanExecutionRecommendations (Interaction diagram)**

Use of the ExtendedPlanExecutionAction and ExtendedPlanExecutionControl interfaces to recommend update of the content, timing, status and progress or a plan constituent as well as to control the its future execution. It is valid for the recommendations in this scenario to be made in any order, omitted or superceded with subsequent recommendations. In this scenario all recommendations are successfully accepted. It is valid for a Tactical Decision Aid to make additional recommendations before a recommendationProcessed response is received.

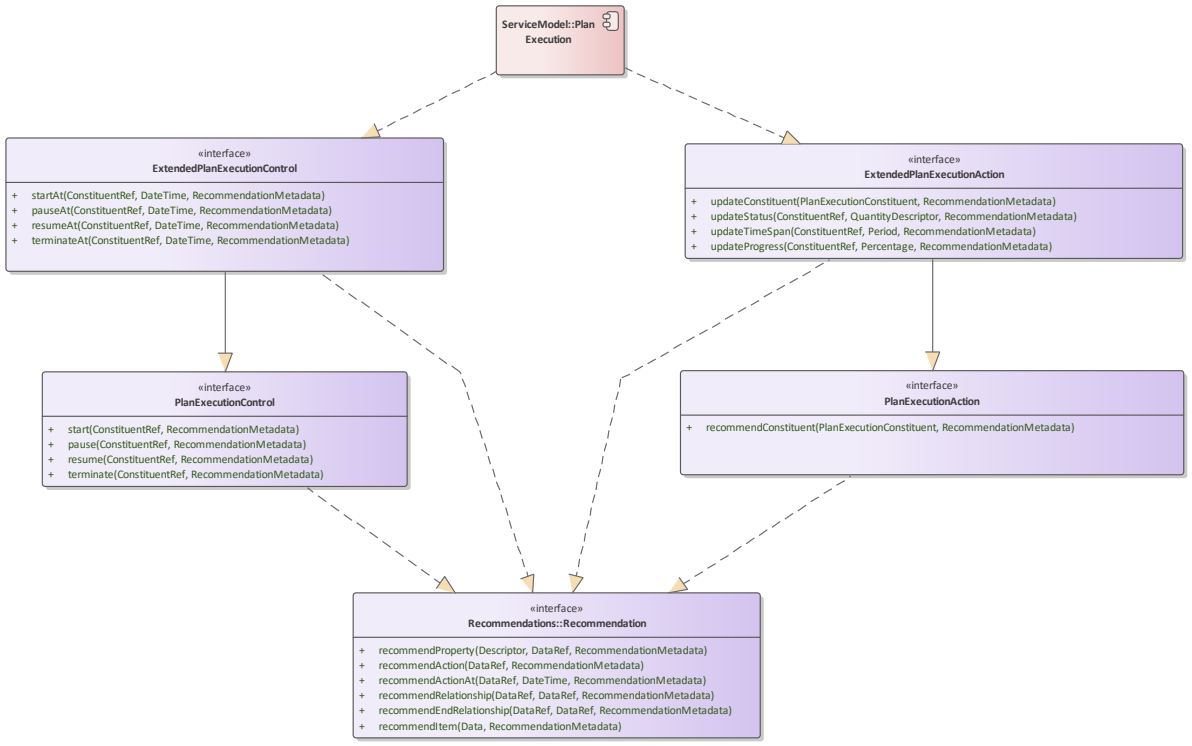


**Figure 10.39 PlanExecutionRecommendations Service Mapping (Component diagram)**

## 10.5.1 PlanExecutionAction

**Type:** Interface
**Package:** PlanExecutionRecommendations
This interface allows client tactical decision aids to make recommendations to enact tactical Plans. Referenced instances must exist. Therefore decision aids should first create any referenced entities, then recommend plan(s), then any sub-plans, then contributing task objectives, then implementing resource tasking recommendations.
All Recommendation operations on the PlanExecutionAction interface receive a PlanExecutionResponse instance in the callback.
It is invalid to recommend a constituent that already exists. That is, a PlanExecutionConstituent is returned for the ConstituentRef through the PlanDataSink interface.

ReferencedClass = Plan

**Table 10.7 - Methods of Interface PlanExecutionAction**

| Method | Notes | Parameters |
|---|---|---|
| **recommendConstituent()** | This is the operation to invoke to recommend a Plan Execution Constituent specialization such as a Plan, TaskObjective or ResourceTasking. | PlanExecutionConstituent **plan** The constituent of plan execution being recommended RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |

#### 10.5.1.1 MessageEnd

**Type:** MessageEnd
**Package:** PlanExecutionRecommendations

#### 10.5.1.2 MessageEnd

**Type:** MessageEnd
**Package:** PlanExecutionRecommendations

#### 10.5.1.3 MessageEnd

**Type:** MessageEnd
**Package:** PlanExecutionRecommendations

#### 10.5.1.4 MessageEnd

**Type:** MessageEnd
**Package:** PlanExecutionRecommendations

### 10.5.2 PlanExecutionControl

**Type:** Interface
**Package:** PlanExecutionRecommendations
This interface allows client tactical decision aids to make recommendations to control the execution of tactical plan-constituents. All Recommendation operations on the PlanExecutionControl interface receive a PlanExecutionResponse instance in the callback.
It is invalid to recommend a change to the execution of a constituent that does not exist. That is, no PlanExecutionConstituent is returned for the ConstituentRef through the PlanDataSink interface.

**Table 10.8 - Methods of Interface PlanExecutionControl**

| Method | Notes | Parameters |
|---|---|---|
| **start()** | This is the operation to invoke to recommend that a Plan Execution Constituent specialization such as a Plan, TaskObjective or ResourceTasking is started immediately. The constituent must not have previously been started. | ConstituentRef **id** A reference to the constituent of plan execution for which the action is being recommended<br>RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |
| **pause()** | This is the operation to invoke to recommend that a Plan Execution Constituent specialization such as a Plan, TaskObjective or ResourceTasking is paused immediately. | ConstituentRef **id** The list of plan changes recevied by the listener<br>RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |

| | | |
|---|---|---|
| **resume()** | This is the operation to invoke to recommend that a Plan Execution Constituent specialization such as a Plan, TaskObjective or ResourceTasking is resumed immediately. The constituent must have previously been paused. | ConstituentRef **id** The list of plan changes recevied by the listener RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |
| **terminate()** | This is the operation to invoke to recommend that a Plan Execution Constituent specialization such as a Plan, TaskObjective or ResourceTasking is terminated immediately. The constituent must have previously been started. | ConstituentRef **id** The list of plan changes recevied by the listener RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |

### 10.5.3 ExtendedPlanExecutionAction

**Type:** Interface
**Package:** PlanExecutionRecommendations
This interface allows client tactical decision aids to make recommendations to update tactical Plans in whole or part.
All Recommendation operations on the ExtendedPlanAction interface receive a PlanResponse instance in the callback.
It is invalid to recommend an update to constituent that does not exist. That is, no PlanExecutionConstituent is returned for the ConstituentRef through the PlanDataSink interface.

**Table 10.9 - Methods of Interface ExtendedPlanExecutionAction**

| Method | Notes | Parameters |
|---|---|---|
| **updateConstituent()** | This is the operation to invoke to recommend the update of a whole Plan Execution Constituent specialization such as a Plan, TaskObjective or ResourceTasking. | PlanExecutionConstituent **planExecutionConstituent** The new values recommended for the plan execution constituent. RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |
| **updateStatus()** | This is the operation to invoke to recommend a change of a status to a Plan Execution Constituent specialization such as a Plan, TaskObjective or ResourceTasking. It is invalid to recommend an update to constituent that does not exist. That is, there is no instance within the Plan Execution component with the specified id. . | ConstituentRef **id** A reference to the plan constituent QuantityDescriptor **status** The status value to update to RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |
| **updateTimeSpan()** | This is the operation to invoke to recommend a change of a time span for a Plan Execution Constituent specialization such as a Plan, TaskObjective or ResourceTasking. It is invalid to recommend an update | ConstituentRef **id** A reference to the plan constituent Period **timeSpan** The time span value to update to RecommendationMetadata **recommendation** Qualifying |

| | | |
|---|---|---|
| | to constituent that does not exist. That is, there is no instance within the Plan Execution component with the specified id.<br>. | information relating to the recommendation |
| **updateProgress()** | This is the operation to invoke to recommend an update to the progress achieved for a Plan Execution Constituent specialization such as a Plan, TaskObjective or ResourceTasking. It is invalid to recommend an update to constituent that does not exist. That is, there is no instance within the Plan Execution component with the specified id. | ConstituentRef **id** A reference to the plan constituent<br>Percentage **progress** The progress value to update to<br>RecommendationMetadata **recommendation** Metadata pertaining to the recommendation |

### 10.5.4 ExtendedPlanExecutionControl

**Type:** Interface
**Package:** PlanExecutionRecommendations
This interface allows client tactical decision aids to make recommendations to control the future execution of tactical plan-constituents. All Recommendation operations on the ExtendedPlanExecutionControl interface receive a PlanExecutionResponse instance in the callback.
It is invalid to recommend a change to the execution of a constituent that does not exist. That is, no PlanExecutionConstituent is returned for the ConstituentRef through the PlanDataSink interface.

**Table 10.10 - Methods of Interface ExtendedPlanExecutionControl**

| Method | Notes | Parameters |
|---|---|---|
| **startAt()** | This is the operation to invoke to recommend that a Plan Execution Constituent specialization such as a Plan, TaskObjective or ResourceTasking is started at a future time. This must be before the end of it's time-span | ConstituentRef **id** A reference to the constituent of plan execution for which the action is being recommended<br>DateTime **time** The time at which it is recommended to start executing the planning constituent<br>RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |
| **pauseAt()** | This is the operation to invoke to recommend that a Plan Execution Constituent specialization such as a Plan, TaskObjective or ResourceTasking is paused at a future time. This must be within it's time-span. | ConstituentRef **id** A reference to the constituent of plan execution for which the action is being recommended<br>DateTime **time** The time at which it is recommended to pause execution of the planning constituent<br>RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |

| | | |
|---|---|---|
| **resumeAt()** | This is the operation to invoke to recommend that a Plan Execution Constituent specialization such as a Plan, TaskObjective or ResourceTasking is resumed at a future time. This must be within it's time-span. | ConstituentRef **id** A reference to the constituent of plan execution for which the action is being recommended<br>DateTime **time** The time at which it is recommended to resume execution of the planning constituent<br>RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |
| **terminateAt()** | This is the operation to invoke to recommend that a Plan Execution Constituent specialization such as a Plan, TaskObjective or ResourceTasking is terminated at a future time. This must be after the start of it's time-span. | ConstituentRef **id** A reference to the constituent of plan execution for which the action is being recommended<br>DateTime **time** The time at which it is recommended to terminate the execution of the planning constituent<br>RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |

## 10.6 TacticalPictureInformation

**Parent Package:** ServiceModel

Interfaces to allow Tactical Decision Aids to receive tactical picture information are contained in the TACSIT Data Exchange specification

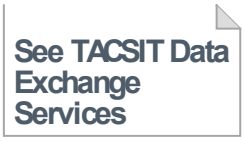


**Figure 10.40 TacticalPictureInformation (Class diagram)**

TacticalPictureInformation requirements are satisfied by TACSIT Data Exchange services

## 10.7 TacticalPictureRecommendations

**Parent Package:** ServiceModel

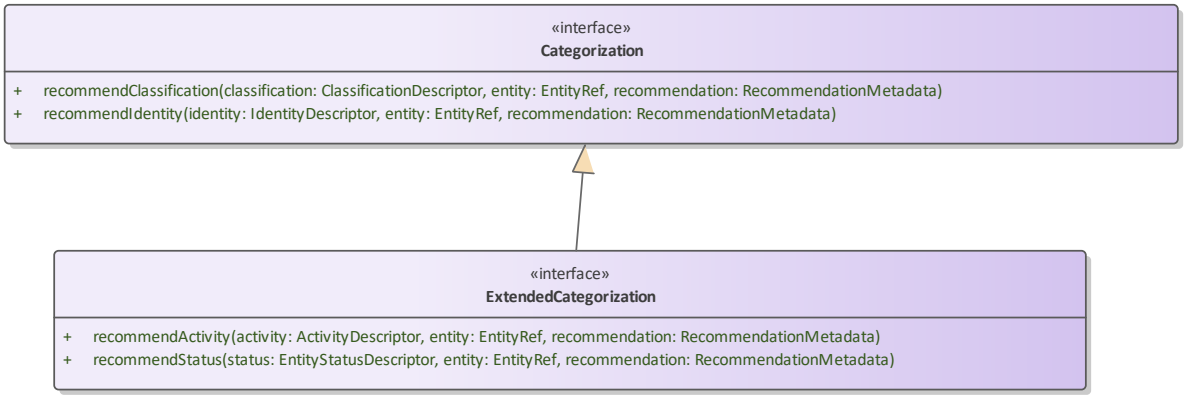


**Figure 10.41 CategorizationRecommendation (Class diagram)**

This is the interface for making Categorization Recommendations for Entities in the tactical picture

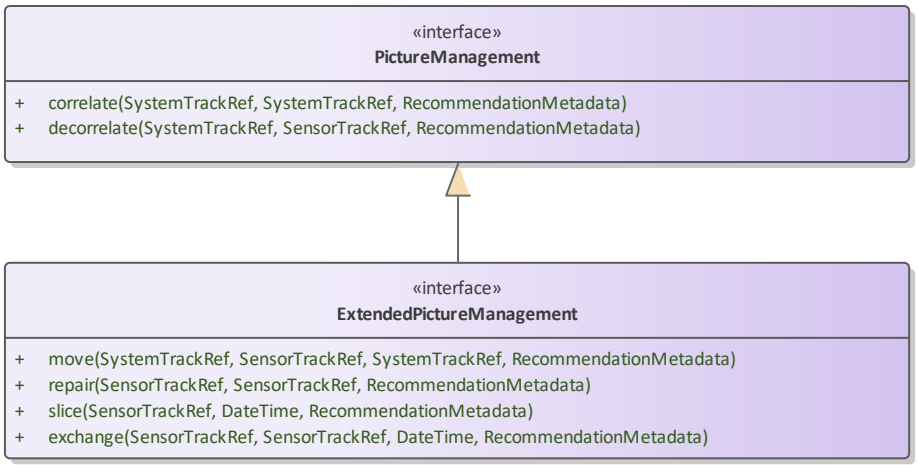


**Figure 10.42 PictureManagementRecommendation (Class diagram)**

**Figure 10.43 TacticalPictureRecommendations (Package diagram)**

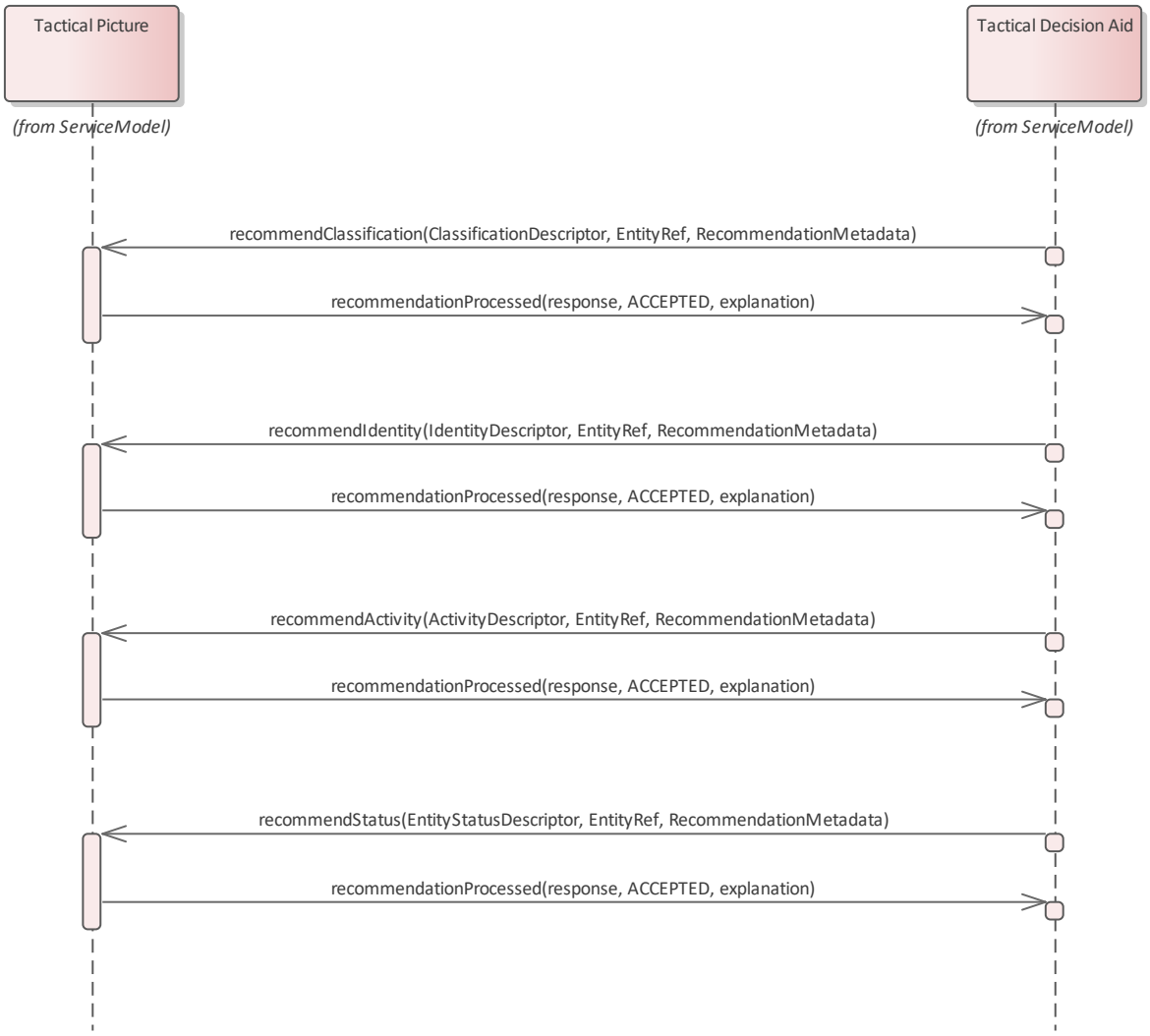


**Figure 10.44 TacticalPictureRecommendations - Categorization (Interaction diagram)**

Use of the Categorization and ExtendedCategorization interfaces to recommend categories for entities. In this scenario all recommendations are successfully accepted. It is valid for a Tactical Decision Aid to make additional recommendations before a recommendationProcessed response is received.

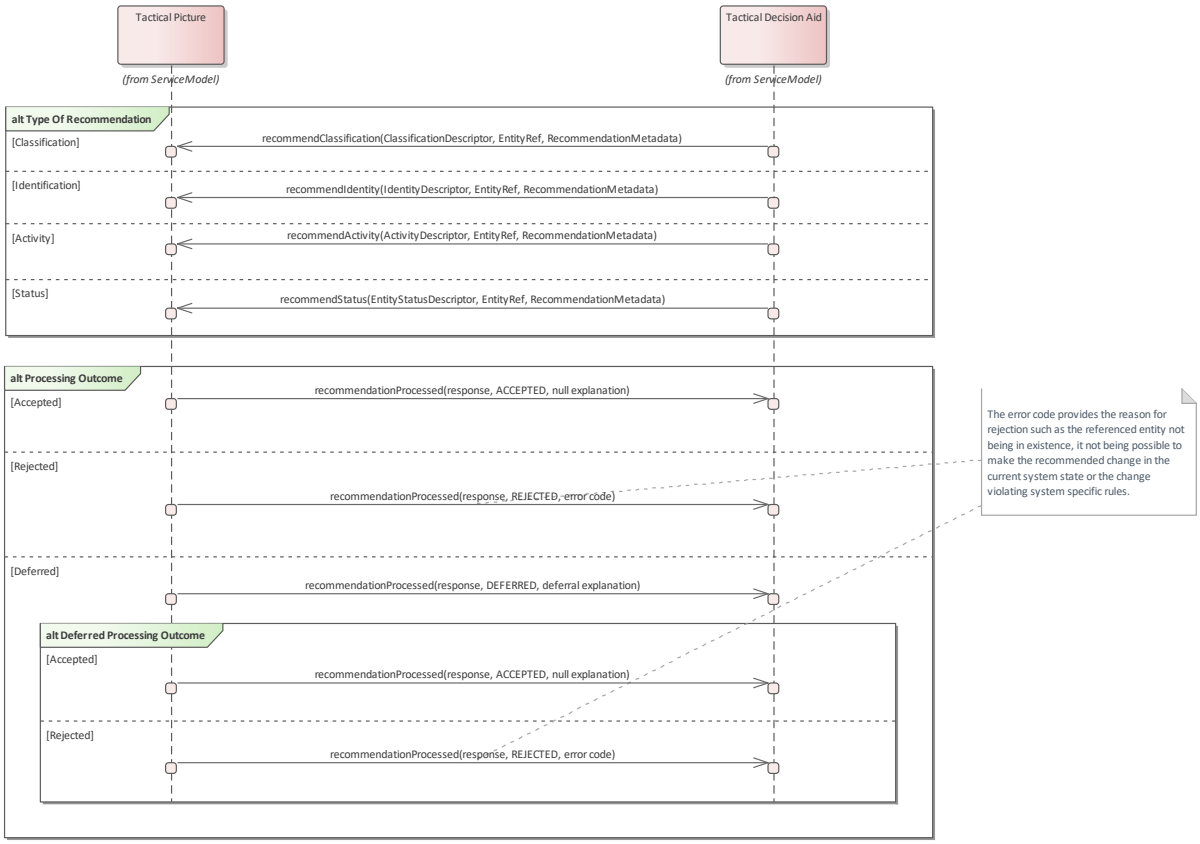


**Figure 10.45 TacticalPictureRecommendations - Categorization - Alternate Flow (Interaction diagram)**

Use of the Categorization and ExtendedCategorization interfaces to recommend categories for entities. In this scenario not all recommendations are successfully accepted. Alternate outcome processing is equivalent for all operations on the Categorization and ExtendedCategorization interfaces.

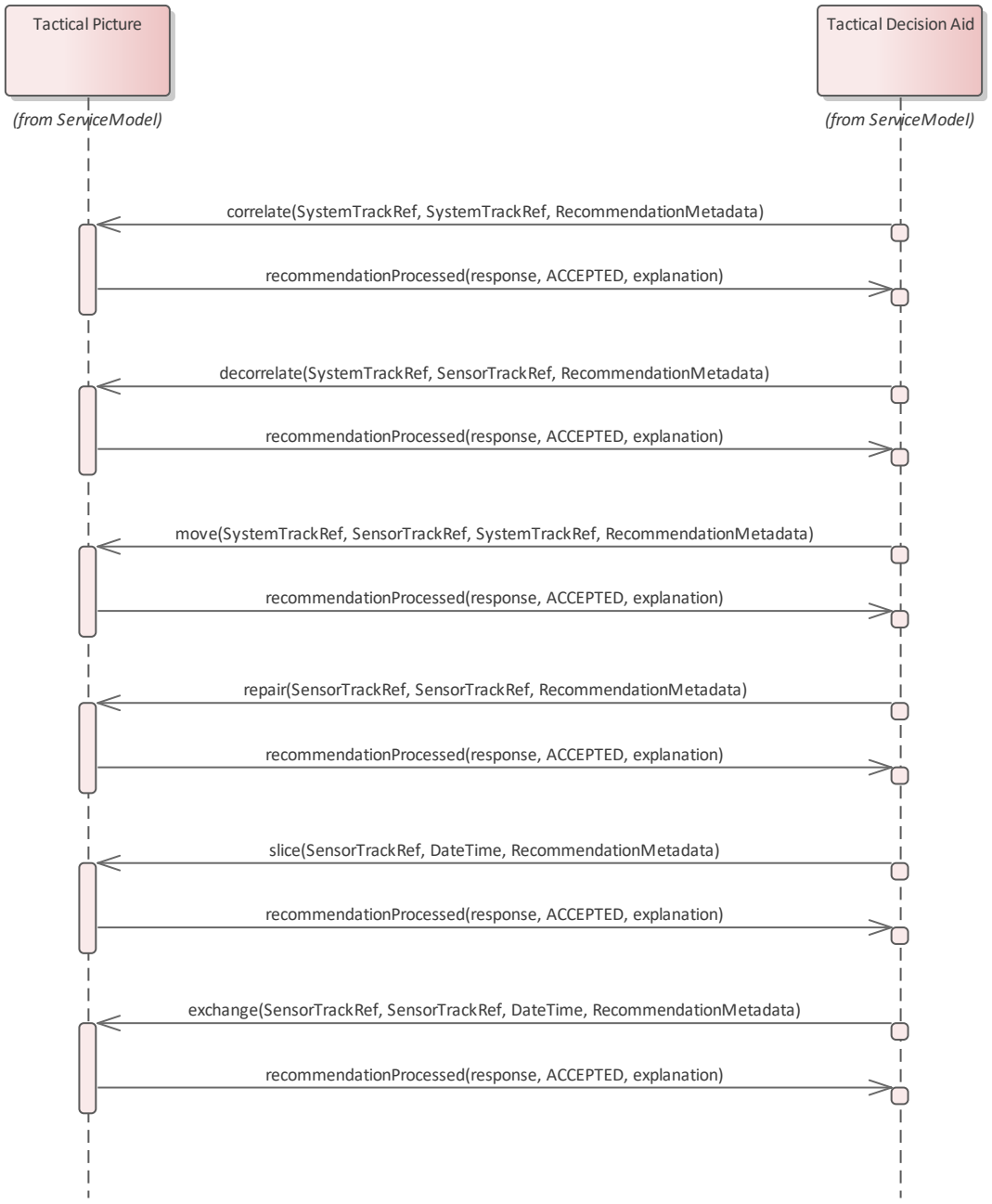


**Figure 10.46 TacticalPictureRecommendations - PictureManagement (Interaction diagram)**

Use of the PictureManagement and ExtendedPictureManagement interfaces to recommend changes to the relationships between entities. In this scenario all recommendations are successfully accepted. It is valid for a Tactical Decision Aid to make additional recommendations before a recommendationProcessed response is received.

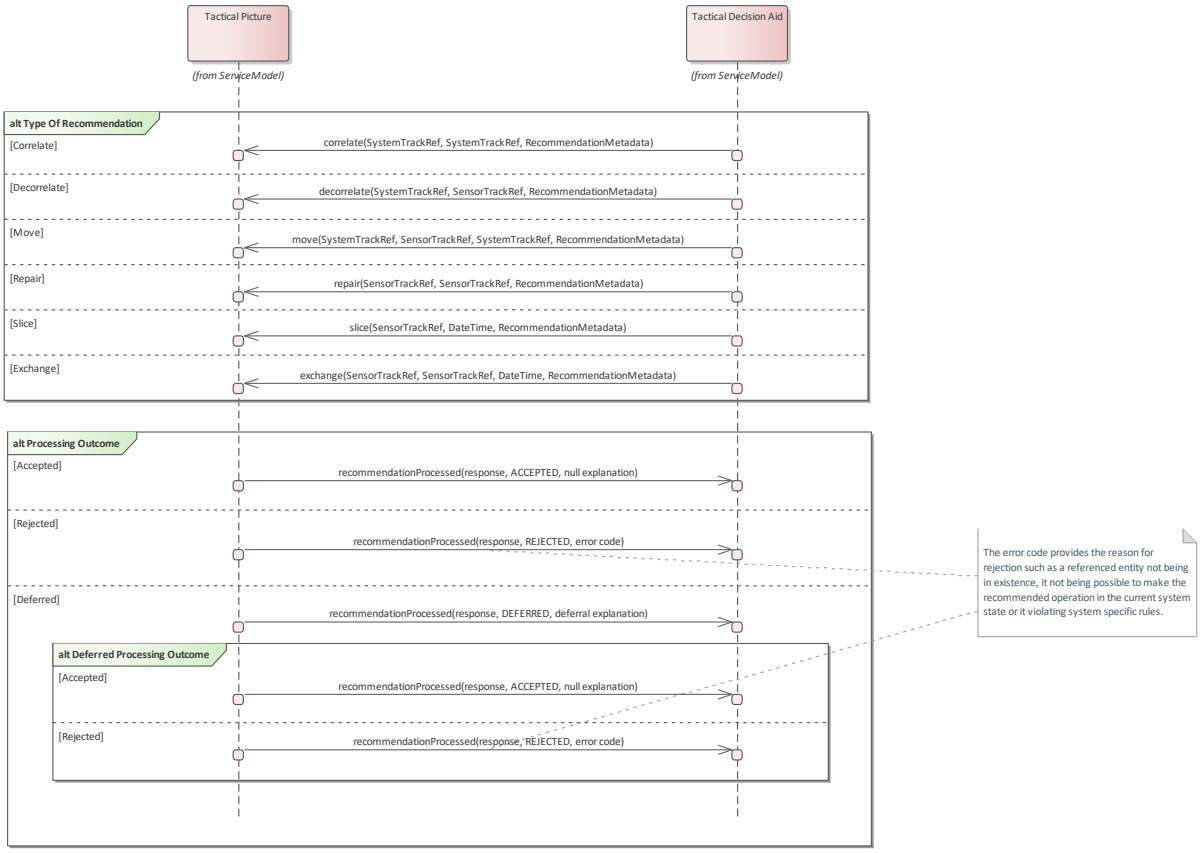


**Figure 10.47 TacticalPictureRecommendations - PictureManagement - Alternate Flow (Interaction diagram)**

Use of the PictureManagement and ExtendedPictureManagement interfaces to recommend categories for entities. In this scenario not all recommendations are successfully accepted. Alternate outcome processing is equivalent for all operations on the PictureManagement and ExtendedPictureManagement interfaces.

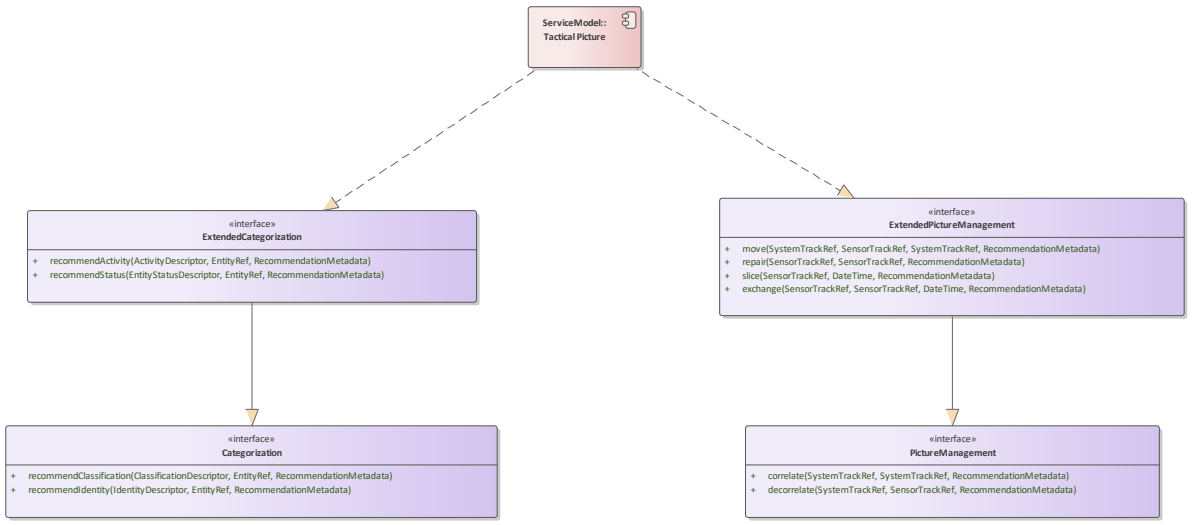


**Figure 10.48 TacticalPictureRecommendations Service Mapping (Component diagram)**

### 10.7.1 Categorization

**Type:** Interface
**Package:** TacticalPictureRecommendations
This interface allows client tactical decision aids to make recommendations to categorize Entities in the tactical picture. I.e. recommendation relating to Entity Categorization data as defined by the TACSIT Data Exchange (TEX) standard. This interface supports recommendations relating to the most common tactical categorization decisions and hence those recommendations most likely to be generated by decision aids.
All Recommendation operations on the Categorization interface receive a CategorizationResponse instance in the callback.
It is invalid to recommend a categorization for an Entity that does not exist. That is, no Entity is returned for the EntityRef through the TEX DataSink interface.

**Table 10.11 - Methods of Interface Categorization**

| Method | Notes | Parameters |
|---|---|---|
| **recommendClassification()** | This is an operation to invoke to make a Classification Recommendation. Classification refers to the kind of platform or vehicle that the Entity represents. Examples include truck, ferry, submarine, helicopter and satellite. | ClassificationDescriptor **classification** The classification being recommended<br>EntityRef **entity** The entity to which the classification applies<br>RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |
| **recommendIdentity()** | This is an operation to invoke to make a Identification Recommendation. Identification refers to the allegiance or ownership of platform or vehicle that the Entity represents. This can be expressed, for example, as a hostility category (also known as standard identity) a nationality, an organization or personal identifier. | IdentityDescriptor **identity** The identity being recommended<br>EntityRef **entity** The entity to which the identity applies<br>RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |

### 10.7.2 ExtendedCategorization

**Type:** Interface
**Package:** TacticalPictureRecommendations
This interface allows client tactical decision aids to make specialized recommendations to categorize Entities in the tactical picture. I.e. recommendation relating to Entity Categorization data as defined by the TACSIT Data Exchange (TEX) standard. This interface supports recommendations relating to more advanced or specialized tactical categorization decisions and hence those recommendations that may not be generated by all decision aids.
All Recommendation operations on the ExtendedCategorization interface receive a CategorizationResponse instance in the callback.
It is invalid to recommend a categorization for an Entity that does not exist. That is, no Entity is returned for the EntityRef through the TEX DataSink interface.

**Table 10.12 - Methods of Interface ExtendedCategorization**

| Method | Notes | Parameters |
|---|---|---|
| **recommendActivity()** | This is an operation to invoke to make an Activity Recommendation. Activity refers to the tasks currently being undertaken by the platform or vehicle that the Entity represents. Examples include Air Defence, Guard, Patrol, Reconnaissance, Refuel and Survey. | ActivityDescriptor **activity** The activity being recommended<br>EntityRef **entity** The entity to which the activity applies<br>RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |
| **recommendStatus()** | This is an operation to invoke to make an Status Recommendation. Status refers to all aspects of the current tactical significance of the platform or vehicle that the Entity represents. Examples of status categories include: the extent to which the Entity poses a threat; the type of offensive action that the Entity is subject to, the outcome of offensive action and other emergencies. | EntityStatusDescriptor **status** The status being recommended<br>RecommendationMetadata **recommendation** Qualifying information relating to the recommendation<br>EntityRef **entity** The entity to which the status applies |

### 10.7.3 PictureManagement

**Type:** Interface
**Package:** TacticalPictureRecommendations

This interface allows client tactical decision aids to make recommendations to manage the relationships between Entities in the tactical picture. I.e. recommendations relating to the constituent Entities of Groups as defined by the TACSIT Data Exchange (TEX) standard. This interface supports recommendations relating to the most common tactical relation decisions and hence those recommendations most likely to be generated by decision aids.

Recommendation operations on the PictureManagement interface receive operation specific ResponseData specialization instances in the callback.

It is invalid to invoke an operation for an Entity that does not exist. That is, no Entity is returned for the EntityRef (SystemTrackRef or SensorTrackRef) through the TEX DataSink interface.

**Table 10.13 - Methods of Interface PictureManagement**

| Method | Notes | Parameters |
|---|---|---|
| **correlate()** | This is the operation to invoke to make a Correlation Recommendation. Correlation refers to the determination that two or more sensor tracks correspond to the same object in the tactical environment. That object is to be represented by a single system track in an unambiguous tactical picture. Correlation relates the sensor tracks to the single system track. Note that it is typically possible for multiple sensors to observe and track the same object. | SystemTrackRef **receiver** The system track to be retained after the operation<br>SystemTrackRef **donor** The system track to discard after the operation<br>RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |

| | | |
|---|---|---|
| | Tactical Decision Aids receive a CorrelationResponse instance in the callback. | |
| **decorrelate()** | This is the operation to invoke to make a Decorrelation Recommendation. Decorrelation is the reverse of Correlation and is used to undo incorrect Correlations or to correct the case when a sensor has started to track a different object with the same sensor track.<br>Tactical Decision Aids receive a DecorrelationResponse instance in the callback. | SystemTrackRef **systemTrack**<br>SensorTrackRef **sensorTrack**<br>RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |

### 10.7.4 ExtendedPictureManagement

**Type:** Interface
**Package:** TacticalPictureRecommendations

This interface allows client tactical decision aids to make specialized recommendations to manage the relationships between Entities in the tactical picture. I.e. recommendations relating to the constituent Entities of Groups as defined by the TACSIT Data Exchange (TEX) standard. This interface supports recommendations relating more advanced or specialized tactical relation decisions and hence those recommendations that may not be generated by all decision aids..
Recommendation operations on the ExtendedPictureManagement interface receive operation specific ResponseData specialization instances in the callback.
It is invalid to invoke an operation for an Entity that does not exist. That is, no Entity is returned for the EntityRef (SystemTrackRef or SensorTrackRef) through the TEX DataSink interface.

**Table 10.14 - Methods of Interface ExtendedPictureManagement**

| Method | Notes | Parameters |
|---|---|---|
| **move()** | This is the operation to invoke to make a Move Recommendation. Move is a sequence of a Decorrelation followed by a Correlation and is used to correct the case when one sensor has started to track (with the same sensor track) a different object that is already being tracked by another sensor. The sensor track in question is Decorrelated from its original system track and Correlated with the system track that already exists for the new object that the sensor is actually tracking.<br>Tactical Decision Aids receive a MoveResponse instance in the callback. | SystemTrackRef **receiver** The system track to which the sensor track is to be moved<br>SensorTrackRef **sensorTrack** The sensor track to move to a different system track<br>SystemTrackRef **donor** The system track the sensor track is to be moved from. To be a valid recommendation the donor track should be supported by tracks other than the sensor track being moved.<br>RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |
| **repair()** | This is the operation to invoke to make a Repair Recommendation. Repair is an action on a sensor track's track report history that is used to make the track history | SensorTrackRef **newSensorTrack** The track currently supported by the sensor |

| | | |
|---|---|---|
| | continuous when a sensor has declared deletion of track before, later, starting to report the same real world object with a new sensor track. The original sensor track's history is added to the new sensor track's history. The implementation is such that the TEX (TACSIT Data Exchange) Entity History interface returns the complete history for the repaired track and no history for the deleted track as defined by the RepairResponse instance received in the callback. | SensorTrackRef **oldSensorTrack** The track previously supported by the sensor RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |
| **slice()** | This is the operation to invoke to make a Slice Recommendation. Slice is an action on a sensor track's track report history that is used to make the track history discrete when a sensor has started to track a different real world object with the same sensor track. It is the inverse of Repair. The original part of sensor track's history is removed from sensor track's history and placed into a new sensor track. The implementation is such that the TEX (TACSIT Data Exchange) Entity History interface returns the pre-slice history for the original track and the post-slice history for the new track as defined by the SliceResponse instance received in the callback. | SensorTrackRef **sensorTrack** The sensor track whose history is to be sliced into new and old portions DateTime **sliceTime** The time at which to divide the tracks history RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |
| **exchange()** | This is the operation to invoke to make an Exchange Recommendation. Exchange is an action on a pair of sensor tracks' track report histories that is used to make the track histories coherent when a sensor has swapped the real-world objects that a pair of sensor tracks have been tracking. An exchange operation is a composition of two slice operations with a common slice-time followed by two repair operations. The sensor track parameters are semantically commutative: exchanging track-a with track-b is equivalent to exchanging track-b with track-a. Tactical Decision Aids receive an ExchangeResponse instance in the callback. | SensorTrackRef **sensorTrack1** The first sensor track to be exchanged SensorTrackRef **sensorTrack2** The second sensor track to be exchanged DateTime **exchangeTime** The time at which to exchange the tracks history RecommendationMetadata **recommendation** Qualifying information relating to the recommendation |

# 11 Domain Model Platform-Specific Models

## 11.1 DDS PSM

The DDS Data Model PSM defines a set of IDL files for the Data Model packages defined by the PIM. Topic types (i.e. IDL structs with keys) are defined for classes that classify a single parameter on an interface method. This avoids redundant indirection. Comments are added to the IDL files to reflect the mapping rules below.

The detailed rules for the MDA code generation from the Data Model PIM to the DDS PSM IDL are as follows:

- The PIM attributes are mapped to IDL attributes;
- Optional attributes are mapped to a union type with a single member present when the exists case attribute is true;
- Collections in the PIM are mapped to IDL sequences;
- Specialization / Generalization PIM relationships are mapped to IDL unions. Generalization classes that have attributes are mapped to a struct containing a base struct for its common attributes and a variants union for the specialization attributes.
- The Duration datatype is mapped to an unsigned long long with the CORBA time representation (100s of nanoseconds since the start of the Gregorian Calendar).
- Other datatypes for real-valued quantities are mapped to a double
- Navigable, by-reference, association roles are mapped to a datatype stereotyped as 'Reference', which has a 'refers to' relation with the destination class. Reference stereotyped datatypes are mapped to a string to represent an implementation specific unique id
- Extensible Enumeration datatypes are mapped to a struct with a schemaPrefix string attribute and a value string attribute

## 11.2 GraphQL PSM

The GraphQL PSM defines a single schema definition file for a combination of the Data Model and Service Model packages defined by the PIM. Classes from the Domain Model of the PIM are mapped to GraphQL types within the schema.

The detailed rules for the MDA code generation from the Data Model PIM to the DDS PSM IDL are as follows:

- The PIM attributes are mapped to GraphQL attributes;
- PIM attributes with multiplicity 1 are mapped to non-nullable GraphQL attributes
- Collections in the PIM are mapped to GraphQL arrays;
- Specialization / Generalization PIM relationships are mapped to GraphQL unions. Generalization classes that have attributes are mapped to a GraphQL type containing a base GraphQL type for its common attributes and a variants GraphQL union for the specialization attributes.
- The Duration datatype is mapped to a GraphQL Long datatype with the CORBA time representation (100s of nanoseconds since the start of the Gregorian Calendar).
- Other datatypes for real-valued quantities are mapped to a GraphQL Float
- Navigable, by-reference, association roles are mapped to a datatype stereotyped as 'Reference', which has a 'refers to' relation with the destination class. Reference stereotyped datatypes are

mapped to a string to represent an implementation specific unique id and a nullable (by default) attribute for the type of the destination class, so as to enable deep queries over a graph of instances.

- Extensible Enumeration datatypes are mapped to a struct with a schemaPrefix string attribute and a value string attribute

# 12 Service Model Platform Specific Models

## 12.1 DDS PSM

The DDS Services PSM defines IDL files for each package defined in the Services PIM. For each method on each interface class an IDL struct for a DDS topic named for the method is generated; each parameter is mapped to an attribute of the IDL struct. This is unless there is only one attribute (of IDL struct stereotype) in which case the topic type is defined in the Domain Model (i.e. it corresponds to the single parameter's class). Return parameters, where specified, are also mapped to DDS Topics.

The PSM method for connecting to other components is through the creation of DDS Entities (specifically Participants, Data Readers and Data Writers).

Specific rules for the MDA code generation from the Service Model PIM to the DDS PSM IDL are as follows:

- The Response callback interface in the PIM is mapped to a struct with two keyed attributes of type short: clientId and requestId; The clientId identifies the Tactical Decision Aid making the request and the requestId distinguishes the recommendation from others made by the same Tactical Decision Aid.
- The DataSink pattern is mapped to a DDS topic type for the Data class. All interface methods are satisfied by built-in DDS API methods.
- From the Configuration interface, the getSupportMapping method is mapped to a topic for the input parameter and a topic for the return parameter and the isSupported method is mapped implicitly to DDS built-in discovery services.

## 12.2 GraphQL PSM

The GraphQL PSM defines a single schema definition file for a combination of the Data Model and Service Model packages defined by the PIM. The schema supports GraphQL clients for Tactical Decision Aids, Tactical Picture and Plan Execution components. Mutations are used to invoke PIM interface methods; queries and subscriptions are used to process those invocations.

The PSM method for connecting to other components is through the underlying HTTPS web service connection. Web-sockets are used for subscription callbacks.

Specific rules for the MDA code generation from the Service Model PIM to the GraphQL PSM IDL are as follows:

- Each interface method in the Service Model is mapped to a (query) type, an input type and update type; these are for queries, mutations and subscriptions respectively.
- The GraphQL schema Query type support queries for any combination of interface methods in the Service Model.
- The GraphQL schema Mutation type supports invocation of single or multiple instances of any combination of interface methods in the Service Model.
- The GraphQL schema Subscription type supports subscription for any combination of interface methods in the Service Model.

- The Response callback interface in the PIM is mapped to a struct with two keyed attributes of type short: clientId and requestId; The clientId identifies the Tactical Decision AId making the request and the requestId distinguishes the recommendation from others made by the same Tactical Decision Aid.
- The DataSink pattern is mapped to the query, input and update types for the Data class. All interface methods are satisfied by built-in GraphQL features.
- From the Configuration interface, the getSupportMapping method is mapped to the query, input and update types for the input parameters and the query, input and update types for the return parameter and the isSupported method is mapped implicitly to GraphQL built-in discovery services.

# 13 Platform Specific Models for Extensible Enumerations

The Tactical Decision Aids metamodel defines an Extensible Enumeration stereotype for a datatype that takes values from a finite set, where the set of values is not defined by the specification. Implementations define the valid set of values using platform specific mechanisms (see Data Model PSMs). This PSM defines normative alignment with other specifications by mapping Extensible Enumerations defined by this specification to definitions in other specifications.

**Table 13.1 - Extensible Enumeration Mappings**

| Extensible Enumeration | Schema Prefix | Reference Specification | Reference Definition | Notes |
|---|---|---|---|---|
| Utils:: QuantityDescriptor | si | ISO 80000-1 :2009 | N/A | SI units. Values are the unit symbols for base units, special symbols and derived symbols. E.g. "kg", "rad" and "m/s2" |
| TacticalPicture:: ActivityDescriptor | s5516.air | STANAG 5516 Ed 6 | DFI 1798 DUI 001 | Air activities. Values are the string representation of the DI bit code |
| TacticalPicture:: ActivityDescriptor | s5516.surf | STANAG 5516 Ed 6 | DFI 1798 DUI 002 | Surface activities. Values are the string representation of the DI bit code |
| TacticalPicture:: ActivityDescriptor | s5516.sub | STANAG 5516 Ed 6 | DFI 1798 DUI 003 | Subsurface activities. Values are the string representation of the DI bit code |
| TacticalPicture:: ActivityDescriptor | s5516.land | STANAG 5516 Ed 6 | DFI 1798 DUI 004 | Land activities. Values are the string representation of the DI bit code |
| TacticalPicture:: ActivityDescriptor | s5516.sp | STANAG 5516 Ed 6 | DFI 1798 DUI 005 | Space activities. Values are the string representation of the DI bit code |
| TacticalPicture:: ActivityDescriptor | s2525.atac | STANAG 2525 Rev D CN 1 | TABLE A-XLII | Values are the string representation of the Code |
| TacticalPicture:: ActivityDescriptor | jc3iedm.atac | JC3IEDM v3.1.4 | action-task-activity-code | Values are the capitalized abbreviations in the Physical Value column |
| TacticalPicture:: ActivityDescriptor | app6b.task | APP-6(B) June 2008 | task graphics | Values are the specific one or two character code within the symbol id that is associated with the task type |
| TacticalPicture:: ActivityDescriptor | app6c.act | APP-6(C) May 2011 | activity symbol table 6-3 | Values (e.g. "Arrest") are from the function column of table 6-3 |

| Extensible Enumeration | Schema Prefix | Reference Specification | Reference Definition | Notes |
|---|---|---|---|---|
| TacticalPicture:: ActivityDescriptor | app6c.task | APP-6(C) May 2011 | mission tasks table 7-A-1 | Values are the labels (e.g. "Ambush") from the control measure column of table 7-A-1 |
| TacticalPicture:: ClassificationDescriptor | s5516.air.pl | STANAG 5516 Ed 6 | DFI 1797 DUI 001 | Air platforms. Values are the string representation of the DI bit code |
| TacticalPicture:: ClassificationDescriptor | s5516surf.pl | STANAG 5516 Ed 6 | DFI 1797 DUI 002 | Surface platforms. Values are the string representation of the DI bit code |
| TacticalPicture:: ClassificationDescriptor | s5516.sub.pl | STANAG 5516 Ed 6 | DFI 1797 DUI 003 | Subsurface platforms. Values are the string representation of the DI bit code |
| TacticalPicture:: ClassificationDescriptor | s5516.land.pl | STANAG 5516 Ed 6 | DFI 1797 DUI 004 | Land platforms. Values are the string representation of the DI bit code |
| TacticalPicture:: ClassificationDescriptor | s5516.sp.pl | STANAG 5516 Ed 6 | DFI 1797 DUI 005 | Space platforms. Values are the string representation of the DI bit code |
| TacticalPicture:: ClassificationDescriptor | s5516.air.st | STANAG 5516 Ed 6 | DFI 804 DUI 001 | Air specific type. Values are the string representation of the DI bit code |
| TacticalPicture:: ClassificationDescriptor | s5516.surf.st | STANAG 5516 Ed 6 | DFI 808 DUI 001 | Surface specific type. Values are the string representation of the DI bit code |
| TacticalPicture:: ClassificationDescriptor | s5516.sub.st | STANAG 5516 Ed 6 | DFI 809 DUI 001 | Subsurface specific type. Values are the string representation of the DI bit code |
| TacticalPicture:: ClassificationDescriptor | s5516.land.st | STANAG 5516 Ed 6 | DFI 810 DUI 001 | Land specific type. Values are the string representation of the DI bit code |
| TacticalPicture:: ClassificationDescriptor | s5516.sp.st | STANAG 5516 Ed 6 | DFI 749 DUI 002 | Space specific type. Values are the string representation of the DI bit code |
| TacticalPicture:: ClassificationDescriptor | imo.id | IMO | N/A | The value to a vessel assigned by Lloyds Registry |
| TacticalPicture:: ClassificationDescriptor | imo.mmsi | IMO | N/A | The unique Maritime Mobile Service Identity (MMSI) as assigned to AIS equipment |
| TacticalPicture:: ClassificationDescriptor | name | N/A | N/A | The name of the entity (e.g. vessel or aircraft) |
| TacticalPicture:: ClassificationDescriptor | callsign | N/A | N/A | A call-sign used by the entity being classified |
| TacticalPicture:: ClassificationDescriptor | iso.3166 | ISO 3166 | 2 letter code | Values are the 2 letter code for the country associated with the entity |
| TacticalPicture:: ClassificationDescriptor | s5516.nat | STANAG 5516 Ed 6 | DFI 748 DUI 001 | Nationality. Values are the string representation of the DI bit code |
| TacticalPicture:: ClassificationDescriptor | s5516.nat.ex | STANAG 5516 Ed 6 | DFI 748 DUI 003 | Extended Nationality. Values are the string representation of the DI bit code |

| Extensible Enumeration | Schema Prefix | Reference Specification | Reference Definition | Notes |
|---|---|---|---|---|
| TacticalPicture:: ClassificationDescriptor | icao.fi | ICAO | N/A | Flight Id. Values are the string representation of the aircraft flight id. |
| TacticalPicture:: ClassificationDescriptor | icao.id | ICAO | N/A | Values are the string representation of the ICAO unique identifier for the aircraft. |
| TacticalPicture:: ClassificationDescriptor | jc3iedm.air | JC3IEDM v3.1.4 | aircraft-type-category-code | Values are the capitalized abbreviations in the Physical Value column |
| TacticalPicture:: ClassificationDescriptor | jc3iedm.surf | JC3IEDM v3.1.4 | surface-vessel-type-category-code | Values are the capitalized abbreviations in the Physical Value column |
| TacticalPicture:: ClassificationDescriptor | jc3iedm.sub | JC3IEDM v3.1.4 | subsurface-vessel-type-category-code | Values are the capitalized abbreviations in the Physical Value column |
| TacticalPicture:: ClassificationDescriptor | jc3iedm.veh | JC3IEDM v3.1.4 | vehicle-type-category-code | Values are the capitalized abbreviations in the Physical Value column |
| TacticalPicture:: ClassificationDescriptor | app6b.sp | APP-6(B) June 2008 | table B-III | Space Entities. Values are the specific one character code within the function id that is associated with the classification |
| TacticalPicture:: ClassificationDescriptor | app6b.air | APP-6(B) June 2008 | table B-IV | Air Entities. Values are the specific one to four character code within the function id that is associated with the classification |
| TacticalPicture:: ClassificationDescriptor | app6b.ground | APP-6(B) June 2008 | table B-V | Ground Entities. Values are the specific one to six character code within the function id that is associated with the classification |
| TacticalPicture:: ClassificationDescriptor | app6b.surf | APP-6(B) June 2008 | table B-VI | Sea Surface Entities. Values are the specific one to four character code within the function id that is associated with the classification |
| TacticalPicture:: ClassificationDescriptor | app6b.sub | APP-6(B) June 2008 | table B-VII | Sea Subsurface Entities. Values are the specific one to four character code within the function id that is associated with the classification |
| TacticalPicture:: ClassificationDescriptor | app6b.sof | APP-6(B) June 2008 | table B-VIII | Special Operations Force Entities. Values are the specific one to four character code within the function id that is associated with the classification |
| TacticalPicture:: ClassificationDescriptor | app6c.air.icon | APP-6(C) May 2011 | air icon | Values are labels from the function column of table 2-4 |
| TacticalPicture:: ClassificationDescriptor | app6c.air.mod | APP-6(C) May 2011 | air modifier | Values are from the modifier column of tables 2-5 & 2-7 |

| Extensible Enumeration | Schema Prefix | Reference Specification | Reference Definition | Notes |
|---|---|---|---|---|
| TacticalPicture:: ClassificationDescriptor | app6c.mis.mod | APP-6(C) May 2011 | missile modifier | Values are a concatenation of the modifier column of tables 2-9 & 2-10 |
| TacticalPicture:: ClassificationDescriptor | app6c.land.icon | APP-6(C) May 2011 | land icon | Values are labels from the function column of tables 3-3 & 3-4 |
| TacticalPicture:: ClassificationDescriptor | app6c.land.mod | APP-6(C) May 2011 | land modifier | Values labels are from the modifier column of tables 3-5 & 3-6 |
| TacticalPicture:: ClassificationDescriptor | app6c.surf.1 | APP-6(C) May 2011 | sea surface sector 1 modifier | Values are from the modifier column of table 4-2 |
| TacticalPicture:: ClassificationDescriptor | app6c.surf.2 | APP-6(C) May 2011 | sea surface sector 2 modifier | Values are from the modifier column of table 4-3 |
| TacticalPicture:: ClassificationDescriptor | app6c.surf.icon | APP-6(C) May 2011 | sea surface icon | Values are labels from the description column of table 4-5 |
| TacticalPicture:: ClassificationDescriptor | app6c.sub.1 | APP-6(C) May 2011 | sea subsurface sector 1 modifier | Values are from the modifier column of table 4-11 |
| TacticalPicture:: ClassificationDescriptor | app6c.sub.2 | APP-6(C) May 2011 | sea subsurface sector 2 modifier | Values are from the modifier column of table 4-12 |
| TacticalPicture:: ClassificationDescriptor | app6c.sub.icon | APP-6(C) May 2011 | sea subsurface icon | Values are labels from the description column of tables 4-14, 15, 16, 17 & 18 |
| TacticalPicture:: ClassificationDescriptor | app6c.sp.icon | APP-6(C) May 2011 | space icon | Values are from the description column of tables 5-4 & 5-7 |
| TacticalPicture:: ClassificationDescriptor | app6c.sp.mod | APP-6(C) May 2011 | space modifier | Values are from the description column of tables 5-5 & 5-6 |
| TacticalPicture:: EntityStatusDescriptor | s5516.wes | STANAG 5516 Ed 6 | DFI 394 DUI 009 | Weapon/Engagement Status. Values are the string representation of the DI bit code |
| TacticalPicture:: ActivityDescriptor | jc3iedm.org | JC3IEDM v3.1.4 | organisation-status-operational-status-code | Values are the capitalized abbreviations in the Physical Value column |
| TacticalPicture:: ActivityDescriptor | jc3iedm.org.q | JC3IEDM v3.1.4 | organisation-status-operational-status-qualifier-code | Values are the capitalized abbreviations in the Physical Value column |
| TacticalPicture:: ActivityDescriptor | jc3iedm.org.fire | JC3IEDM v3.1.4 | organisation-status-fire-mode-code | Values are the capitalized abbreviations in the Physical Value column |
| TacticalPicture:: IdentityDescriptor | s5516 | STANAG 5516 Ed 6 | DFI 376 DUI 007 | Non-exercise identities. Values are the string representation of the DI bit code |
| TacticalPicture:: IdentityDescriptor | s5516.ex | STANAG 5516 Ed 6 | DFI 376 DUI 001 | Exercise identities. Values are the string representation of the DI bit code |

| Extensible Enumeration | Schema Prefix | Reference Specification | Reference Definition | Notes |
|---|---|---|---|---|
| PlanExecution:: AmmunitionCategory | s5516.mis | STANAG 5516 Ed 6 | DFI 1622 | Non-exercise identities. Values are the string representation of the DI bit code |
| PlanExecution:: AmmunitionCategory | jc3iedm.amm | JC3IEDM v3.1.4 | ammunition-type-category-code | Values are the capitalized abbreviations in the Physical Value column |
| PlanExecution:: CapabilityCategory | jc3iedm.fire | JC3IEDM v3.1.4 | fire-capability-category-code | Values are the capitalized abbreviations in the Physical Value column |
| PlanExecution:: CapabilityCategory | jc3iedm.weap | JC3IEDM v3.1.4 | weapon-type-category-code | Values are the capitalized abbreviations in the Physical Value column |
| PlanExecution:: CapabilityCategory | jc3iedm.w.sc | JC3IEDM v3.1.4 | weapon-type-subcategory-code | Values are the capitalized abbreviations in the Physical Value column |
| PlanExecution:: CapabilityCategory | jc3iedm.mob | JC3IEDM v3.1.4 | mobility-capability-category-code | Values are the capitalized abbreviations in the Physical Value column |
| PlanExecution:: CapabilityCategory | jc3iedm.eng | JC3IEDM v3.1.4 | engineering-capability-category-code | Values are the capitalized abbreviations in the Physical Value column |
| PlanExecution:: CapabilityCategory | jc3iedm.cargo | JC3IEDM v3.1.4 | cargo-category-code | Values are the capitalized abbreviations in the Physical Value column |
| PlanExecution:: CapabilityCategory | jc3iedm.maint | JC3IEDM v3.1.4 | maintenance-capability-category-code | Values are the capitalized abbreviations in the Physical Value column |
| PlanExecution:: CapabilityCategory | jc3iedm.supp | JC3IEDM v3.1.4 | support-capability-category-code | Values are the capitalized abbreviations in the Physical Value column |
| PlanExecution:: CapabilityCategory | jc3iedm.surv | JC3IEDM v3.1.4 | surveillance-capability-category-code | Values are the capitalized abbreviations in the Physical Value column |
| PlanExecution:: CapabilityCategory | jc3iedm.trans | JC3IEDM v3.1.4 | transmission-capability-category-code | Values are the capitalized abbreviations in the Physical Value column |
| PlanExecution:: CapabilityCategory | jc3iedm.op | JC3IEDM v3.1.4 | operational-capability-category-code | Values are the capitalized abbreviations in the Physical Value column |
| PlanExecution:: DependencyCategory | jc3iedm.mob.dc | JC3IEDM v3.1.4 | mobility-capability-descriptor-code | Values are the capitalized abbreviations in the Physical Value column |
| PlanExecution:: ExtendedPlanStatus | jc3iedm.dev | JC3IEDM v3.1.4 | plan-status-development-status-code | Values are the capitalized abbreviations in the Physical Value column |
| PlanExecution:: ExtendedPlanStatus | jc3iedm.state | JC3IEDM v3.1.4 | plan-status-state-code | Values are the capitalized abbreviations in the Physical Value column |

| Extensible Enumeration | Schema Prefix | Reference Specification | Reference Definition | Notes |
|---|---|---|---|---|
| PlanExecution:: ObjectiveCategory | jc3iedm.qual | JC3IEDM v3.1.4 | action-objective-qualifier-code | Values are the capitalized abbreviations in the Physical Value column |
| PlanExecution:: OrbitCategory | app6c.sp.mod | APP-6(C) May 2011 | space modifier | Values are from the description column of table 5-5 |
| PlanExecution:: ReadinessDescriptor | jc3iedm.org | JC3IEDM v3.1.4 | organisation-status-readiness-code | Values are the capitalized abbreviations in the Physical Value column |
| PlanExecution:: SpecificationDescriptor | jc3iedm.elec | JC3IEDM v3.1.4 | electronic-equipment-type-category-code | Values are the capitalized abbreviations in the Physical Value column |
| PlanExecution:: SpecificationDescriptor | jc3iedm.elec.ex | JC3IEDM v3.1.4 | electronic-equipment-type-subcategory-code | Values are the capitalized abbreviations in the Physical Value column |

Note: that the following Extensible Enumerations can (also) use values from the corresponding Extensible Enumerations with mappings defined above.

- PlanExecution::DependencyCategory : PlanExecution::CapabilityCategory
- PlanExecution::IntentDescriptor : TacticalPicture::ActivityDescriptor
- PlanExecution::ObjectiveCategory : TacticalPicture::ActivityDescriptor
- PlanExecution::PlanType : TacticalPicture::ActivityDescriptor
- PlanExecution::ResourceCategory : TacticalPicture::ClassificationDescriptor
- PlanExecution::ResourceCategory : TacticalPicture::CapabilityCategory
- PlanExecution::TaskingActivity: TacticalPicture::ActivityDescriptor

Note: The SOPES specification provides a UML wrapper for the attributes defined by JC3IEDM.

Implementations use the getSupportMapping method to get a URL to a file to determine a components support for specific Extensible Enumeration values. The file is formatted using JSON as per this non-normative example, which shows how values from the external specifications are appended to the schema prefix.

```
{
      “mapping category”: {
            “descriptor”: “TacticalPicture::ClassificationDescriptor”,
            “values”: {
                  {
                        “value”: “s5516.air.pl.4”,
                        “description”: “BOMBER”
                  },
                  {
```

```
                        "value": "s5516.air.pl.13",
                        "description": "MISSILE"
                    },
                    {
                        "value": "s5516.air.pl.22",
                        "description": "CIVIL, AIRLINER"
                    },
                    {
                        "value": "jc3iedm.air.AIRRW",
                        "description": "A machine or device capable of
atmospheric flight and dependent on rotating blades for lift."
                    },
                    {
                        "value": "jc3iedm.air. LGTAIR",
                        "description": "A machine or device capable of
atmospheric flight weighing less than the air it displaces."
                    }
                }
            }
}
```

The ResponseExplanation datatype may take values with meanings from Table 13.2 below.

**Table 13.2 – ResponseExplanation Extensible Enumeration Mapping**

| Value | Meaning |
|---|---|
| tda.error.none | No error |
| tda.deferred.user | Recommendation referred to an operator |
| tda.error.noentity | Entity does not exist |
| tda.error.noplan | Plan constituent does not exist |
| tda.error.noresource | Resource instance does not exist |
| tda.error.noref | Reference to another instance is invalid |
| tda.error.rule | Recommendation violates a system rule |
| tda.error.state | Recommendation is invalid in the current system state |