TRAK

ENTERPRISE ARCHITECTURE FRAMEWORK

COPYRIGHT

CONTENTS

GNU FREE DOCUMENTATION LICENSE

GNU Free Documentation License Version 1.3, 3 November 2008

The text of the license is at http://www.gnu.org/licenses/fdl-1.3.html ([Ref. 15])

WARRANTY DISCLAIMERS

This Document is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

NETWORK LOCATION

This document is available at http://sf.net/p/trak/files.

HISTORY

Changes to this document are also tracked via a RSS feed . and listed in the trak project version-controlled repository. Changes are also notified via the TRAK_AF Twitter feed [Ref. 16] .

Author(s)	Date	Changes
Nic Plum	14th January 2016	Figure 1-1, Figure 1-2- added document numbers. Figure 3-1 – number of viewpoints changed to 24. Added SVp-11 Solution Event Causes and SVp-13 Solution Risk Viewpoints to 8.5 Solution Perspective.
Nic Plum	5th December 2015	Conversion of original to Open Office document format – some format / layout changes as a result. Added document number - TRAK000004. General – references changed to ISO/IEC/IEEE 42010:2011 rather than FDIS version. Change of network location for file. COPYRIGHT. Removed reference to comparison with MODAF in invariant sections as it isn't in this document. Acknowledgements. Corrected references to comparison of TRAK vs MODAF – in the separate Viewpoints and Metamodel documents. 1 Introduction / Scope. Added reference to Sourceforge project sites as means of release. 1.2 Configuration Management – clarified means to define TRAK baseline by reference to the individual specifications. Glossary. Addition of 'Graph – Directed', addition to Architecture Description Tuple. 2 About TRAK - Important Ideas. Added ability to query / MBSE. 3 Structure of TRAK. Added 'logical' view from ISO/IEC/IEEE 42010 conceptual model. 4.1 ISO/IEC 42010. Added reference to comparison of TRAK vs ISO/IEC/IEEE 42010:2011 to support formal conformance assessment. 5 Glossary. Added Architecture Description Element. Added 'permissible' to Architecture

Author(s)	Date	Changes
		Description Tuple. 8.6 Management Perspective. Added reference to MVp-04 Assurance Viewpoint. 10.1 Introduction. Added 'block element' to master architecture view description. Figures. UML-like notation removed. Figure 1-2 shows boundary between logical definition and implementation. Figure 3-1 – URLs and number of viewpoints corrected. Figure 4-1 adds 'TRAK. Implementation. Architecture Description Elements' as specification that applies standards. Title of Figure 10-1 modified to ...'Architecture Description'. References. Deleted ISO/IEC 42010:2007. New URL for MODAF [Ref. 3] . Document numbers added for Viewpoints, Metamodel documents. New references: conformance of TRAK [Ref. 15] , TRAK Twitter feed [Ref. 16] , Wikipedia - directed graph [Ref. 17] . Index added.
Nic Plum	5th February 2013	#2 . Added requirement to include version identification in a conforming AD (to meet ISO/IEC/IEEE 42010:2011). Figure 3.1 now shows 21 viewpoints (was 22). Corrected Figure numbers 8-1, 8-2 (now 9-1 and 9-2), 9-1 (now 10-1) so that they match the section number.
Nic Plum	2nd January 2012	#3426853 Removed old TRAK logo. Updated ISO 42010 reference to 2011full issue.
Nic Plum	30th Sept 2011	#3415260 Added colour rule CR12. Minor edits for clarification / consistency e.g. to definition of Master Architecture View. Implementation of TRAK now addressed under a separate heading.

Author(s)	Date	Changes
Nic Plum	14th Sept 2011	Added figure 1-2. Modified Figure 2-1 to use standard colour for Standard. Modified Figure 3-1 to include standards the apply to properties of TRAK metamodel. Updated comparison with ISO/IEC 42010 to include FDIS (from FCD) version. Tidied definitions of Architecture Description and Metamodel. Added reference to assessment of UML in section 6. Corrected errors in numbering of figure and table in section 4.
Nic Plum	30th March 2011	Added text to scope and important ideas identifying that TRAK isn't concerned with populating a data model.
Nic Plum	4th February 2011	#3166981 changed 'BLV-7' to 'BLV-12' in TRAK Bye Laws
Nic Plum	25th January 2011	Added section 3 - Structure of TRAK. Added link to comparison of TRAK against ISO/IEC 42010.
Nic Plum	20th January 2011	Added BLV-11 to explicitly require minimum view sets.
Nic Plum	18th January 2011	Original release. Derived from trakmetamodel and trakviewpoints projects at Sourceforge. #3138586 added Bye Law requiring object type shown. ISO 42010: #3142866. Added Bye Law requiring version identification. Added section on standards providing mappings to ISO 42010. #3140866 Added section on conformance with TRAK. Responds to INCOSE UK AWG change requests: #3138686 added missing bye law number. #3138673 separated viewpoints from views in TRAK Architecture Perspectives. #3138669 Separated 'model' from 'architecture description'. #3138671Added definitions for architecture tuple and master architecture viewpoint.

February 2010 Original Release of TRAK - based on MODAF® 1.2 (and hence also DODAF 1.5).

ACKNOWLEDGEMENTS

This work was originally commissioned by London Underground Ltd.

This Document is based on and incorporates aspects of the Ministry of Defence Architecture Framework MODAF Version 1.2.

A summary of the differences between TRAK Meta-model Version 1 and the MODAF® Version 1.2 can be found in the TRAK Metamodel specification [Ref. 4] . A comparison of the set of TRAK viewpoints[1]/views against the MODAF® 1.2 view set is outlined in the TRAK Viewpoints specification [Ref. 5] .

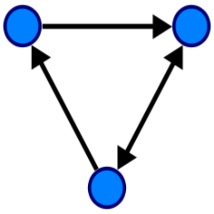
The Document incorporates:

- beta testing and feedback from Joe Silman at the Centre for Railway Research and Education at The of University of Birmingham, UK.
- Human Factors advice and feedback from Christopher Lowe at Liv Systems Ltd.
- advice on viewpoint definition and ISO 42010 from Colin Wood at London Underground Limited
- MODAF® architectural modelling experience, architecture viewpoint definitions & metamodel relationships - Nic Plum at Eclectica Systems Ltd for London Underground Ltd.

[1] TRAK uses 'viewpoint' and 'view' in accordance with ISO/IEC 42010. A MODAF viewpoint is a collection of views.

GLOSSARY

Term	Definition	Source
Architecture	(system) fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution.	ISO/IEC/IEEE 42010 [Ref. 1]
Architecture Description	Work product used to express an architecture.	ISO/IEC/IEEE 42010 [Ref. 1]
Architecture Description Language	An architecture description language is any language for use in an architecture description. Examples include Architecture Analysis & Description Language (AADL) , SysML], and ArchiMate.	ISO/IEC/IEEE 42010 [Ref. 1]
Architecture Description Tuple	Fundamental unit of TRAK architecture description. Comprises of a named architecture description element (block) with a named relationship with itself or another architecture description element. Forms a declarative statement e.g. '(Organisation) Make This PLC has part (Organisation) Engineering Directorate'. This is equivalent to a 'triple' in software engineering terms or a 'graph' in mathematical terms.	TRAK Metamodel.
Architecture Framework	Conventions, principles and practices for the description of architectures established within a specific domain of application and/or community of stakeholders	ISO/IEC/IEEE 42010 [Ref. 1]
Architecture Viewpoint	Work product establishing the conventions for the construction, interpretation and use of architecture views to frame specific	ISO/IEC/IEEE 42010 [Ref. 1]

Term	Definition	Source
	system concerns. Note: 1 Architecture Viewpoint governs 1 Architecture View	
Graph (Directed)	In mathematics, and more specifically in graph theory, a directed graph (or digraph) is a graph, or set of vertices connected by edges, where the edges have a direction associated with them. 	Wikipedia [Ref. 17]

ABBREVIATIONS

AD	Architecture Description
ADL	Architecture Description Language
BPMN	Business Process Modelling Notation
DODAF	Department of Defense Architecture Framework
MODAF	Ministry of Defence Architecture Framework
RDF	Resource Description Format
UML	The Universal Modelling Language

1 INTRODUCTION / SCOPE

This represents part of the logical definition of TRAK, an enterprise architecture framework. It provides a means of describing the architecture of systems and is based on the requirements of ISO/IEC/IEEE 42010:2011 [Ref. 1] .

TRAK allows you to describe an enterprise, a concept, a solution (and its procurement) and an architecture task. In ISO/IEC/IEEE 42010 terms each is a 'system of interest' and has stakeholders who have concerns that need to be addressed through the resulting architecture description.

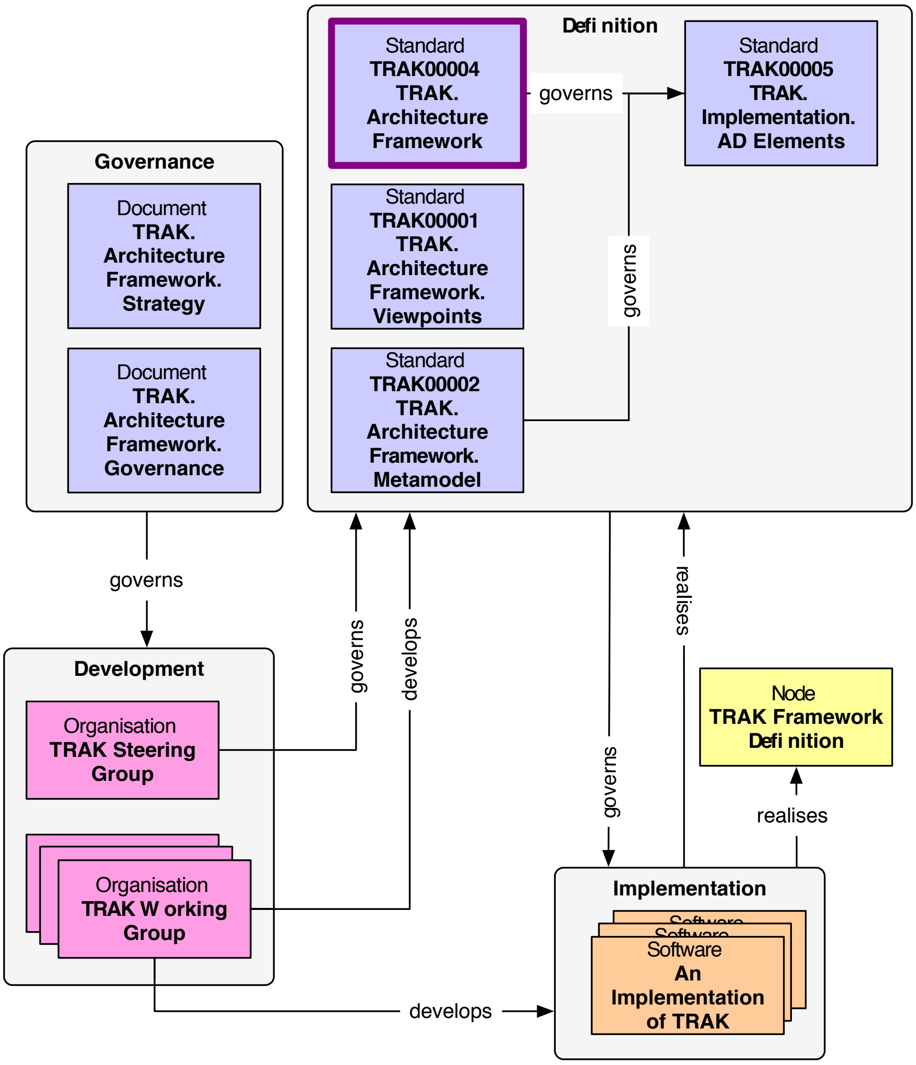

Definition
Standard TRAK00004 TRAK. Architecture Framework
governs
Standard TRAK00005 TRAK. Implementation. AD Elements
Standard TRAK00001 TRAK. Architecture Framework. Viewpoints
governs
Standard TRAK00002 TRAK. Architecture Framework. Metamodel
Governance
Document TRAK. Architecture Framework. Strategy
Document TRAK. Architecture Framework. Governance
governs
Development
Organisation TRAK Steering Group
Organisation TRAK Working Group
governs
develops
governs
realises
Node TRAK Framework Definition
realises
Implementation
Software An Implementation of TRAK
develops

Figure 1-1 Context for the TRAK Architecture Viewpoints Document (This Document)

TRAK is solution or implementation free i.e. any UML profile or template, RDF triple or textual statement is one of many possible solutions to this set of logical requirements in producing TRAK-compliant architecture views and may contain tool or implementation-specific artefacts or constructs. For example the set of attributes that any architecture description element type has is important, the inheritance isn't (to TRAK - it might be for repository management).

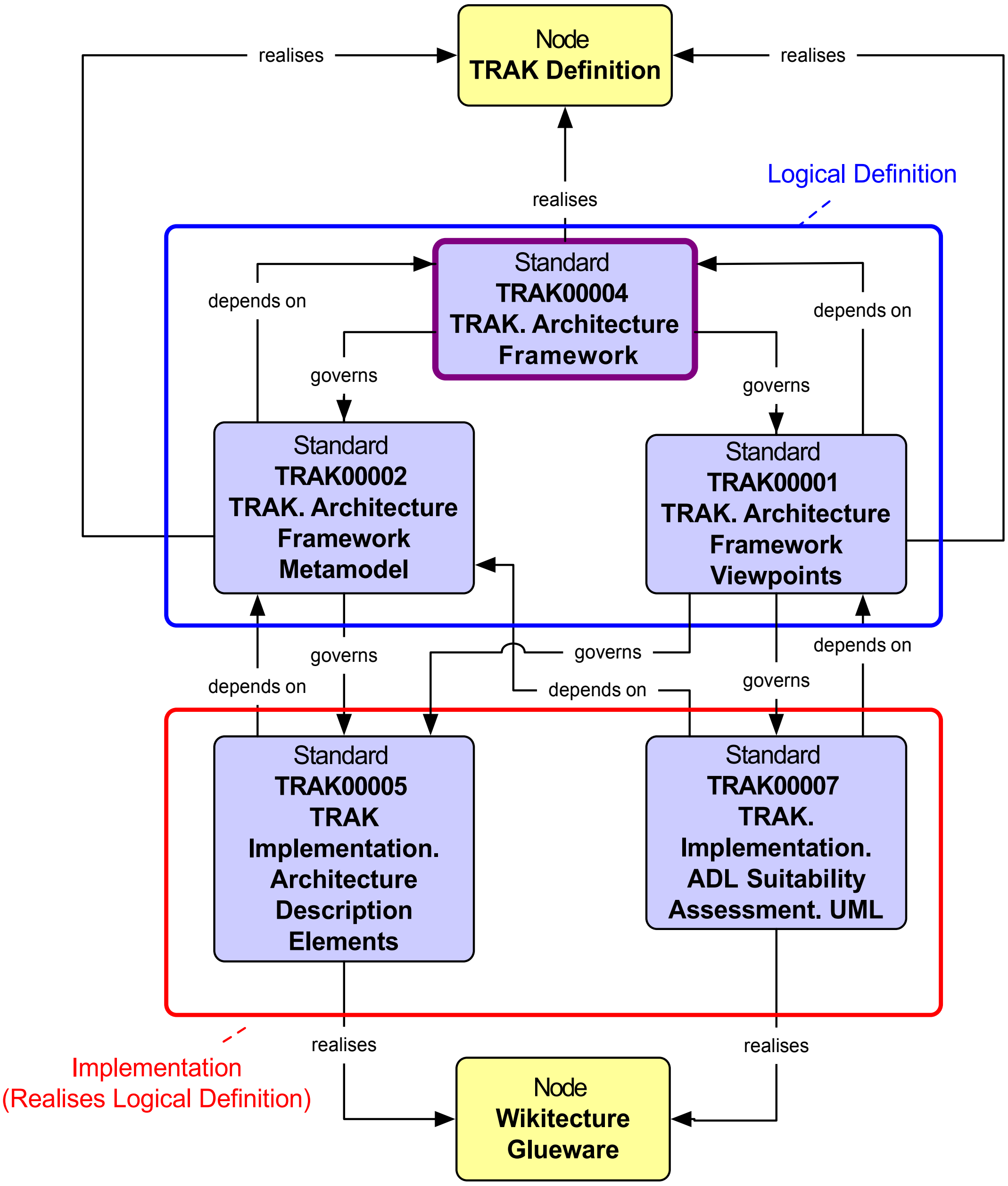


Figure 1-2 Normative TRAK Documents - Logical Definition vs Implementation of TRAK

There are 3 parts to the logical definition of TRAK:-

- **TRAK. Architecture Framework** - this document. Defines and describes TRAK as a whole and invokes the TRAK Metamodel and TRAK Viewpoints documents. It explains important ideas, provides a common glossary, defines rules that apply to colour and presentation. It also provides guidance on choice of a language to represent TRAK. It defines how TRAK aligns with ISO/IEC 42010:2011 and what conformance with TRAK means. It defines a minimum modelling process.

- **TRAK. Metamodel** [Ref. 4] . Defines the element types, their attributes and the relationships between the types. This provides the set of "things" from which a TRAK architecture description is constructed and how they are connected.
- **TRAK. Viewpoints** [Ref. 5] . Defines for each TRAK architecture view, what questions/concerns are addressed by each, what relationships from the TRAK metamodel must and should be used, what is the minimum acceptable content and presentation and what consistency rules apply. This follows the ISO 42010:2011 standard for architecture description.

Each of these specifications is managed and released through a separate project site within Sourceforge under the project names of trak, trakmetamodel and trakviewpoints respectively.

1.1 Implementation of TRAK

TRAK is defined without any notion of how it is implemented in terms of the architecture description language (ADL) or notation used to represent TRAK architecture views or the tool used to do this. The definition of TRAK is concerned with "the what" not "the how". At any time there may be many implementations of TRAK. A list of known implementations of TRAK is maintained on Sourceforge [Ref. 6] ,

An implementation might implement TRAK in full or only partially. Equally an implementation using an ADL or a tool might introduce its own limitations or artefacts. It is hoped that any implementation will identify any limitations or artefacts that it introduces. If this is done it will help users of TRAK understand what is a product of the TRAK definition and what is a product of the implementation of TRAK.

All implementations of TRAK shall comply with TRAK. Implementation. Architecture Description Elements [Ref. 13] . This defines how names and attributes are to be implemented and is an essential part of assuring consistency of implementation of TRAK. As this is a normative document it is represented as a Standard in TRAK in Figure 1-2.

1.2 Configuration Management

The following are identified at separate version or time-stamped releases:

- this document
- TRAK Metamodel document [Ref. 4]
- the TRAK Architecture Viewpoints document [Ref. 5] and each TRAK Architecture Viewpoint..

There is not an overall version number for TRAK as a whole since TRAK is released as open source and there might be many small iterative changes to the parts. A TRAK baseline is therefore identified by the dates of the 3 specification documents (Overall. Viewpoints and Metamodel). See also section 7 Conformance with TRAK .

2 ABOUT TRAK - IMPORTANT IDEAS

This provides essential ideas that help in understanding what TRAK is, and what it isn't:

- TRAK is a standard to promote consistency and therefore interoperability and exchange of architecture descriptions. It is designed to encourage re-use, collaboration and sharing of architecture descriptions. For this you need a consistent set of architecture description elements and relationships and rules that determine what can be shown on an architecture view.
- TRAK is a general purpose architecture framework. Whilst it is system-centric it does not use any domain specific language or constructs. It is oriented towards typical systems engineering/thinking activities and concerns.
- TRAK can represent concepts ranging from high-level business or enterprise goals to the detailed working of a solution, whether an organisation or a physical product. It is important to be able to place the solution in proper context of the enterprise and projects.
- TRAK's purpose is to provide a means to answer task sponsor's concerns - not populate an underlying data model. Once created, however, it may be possible to use the relationships and attributes to answer queries depending on the implementation. TRAK can therefore support Model-Based System Engineering (MBSE).
- TRAK is not UML, SysML, BPMN or any other architecture description language. TRAK is TRAK. It allows you to use any architecture description language to describe the real world and form architecture views. All TRAK mandates is that you stick to the allowed set of architecture description elements and use the relationships specified for the view being created. See Choice of Architecture Description Language (ADL).
- TRAK provides a controlled grammar or language for architectural modelling. TRAK provides a way of describing context, constraints, dependencies and associations using natural language so that views are easy to read. Relationships provide hard-wiring such that in a modelling tool you can analyse, query, navigate between elements to get the information needed. This is very different idea from a 'flat' diagram where you are limited to presentation, such as colour to provide meaning. Getting consistent relationships and therefore meaning is all important. Relationships are portable.
- flexibility and re-use of architecture description elements is achieved by a small set of element types but with many combinations. TRAK provide a rich set of combinations that can be used to describe most situations.
- no one tool or methodology is suited to everything. TRAK does not seek to replace your existing requirement management, project management or other tool sets - it augments them and is very good at showing context (flows, ownership, governance, membership, precedence, responsibility, structure boundaries) using relationships.
- TRAK is not a process, unlike TOGAF. TRAK mandates no process. You can create the views you need in the order you want appropriate to the task.

- architectural descriptions are long term stores of information - they are a significant investment and should be built on rather than start afresh for every task or project.
- an architecture repository is a step towards forming a description of yours or your company's world - a "wikitecture". Architecture description works best if the people contributing reflect the breadth of the TRAK metamodel - in other words not just those with 'architect' in their job title. This helps ensure that the right people own and maintain their respective parts of an architecture description for the collective good.

3 STRUCTURE OF TRAK

TRAK is defined in a logical way - that is to say free of any notion of how TRAK is implemented.

TRAK has 2~~2~~4 viewpoints which are grouped into 5 perspectives. Each viewpoint belongs to a single perspective and specifies a single logical view (which may be split into several physical views for readability and navigability). Each viewpoint specifies what architecture description tuples must and may appear. The architecture description elements are specified by the TRAK metamodel.

This is shown in Figure 3-1 .

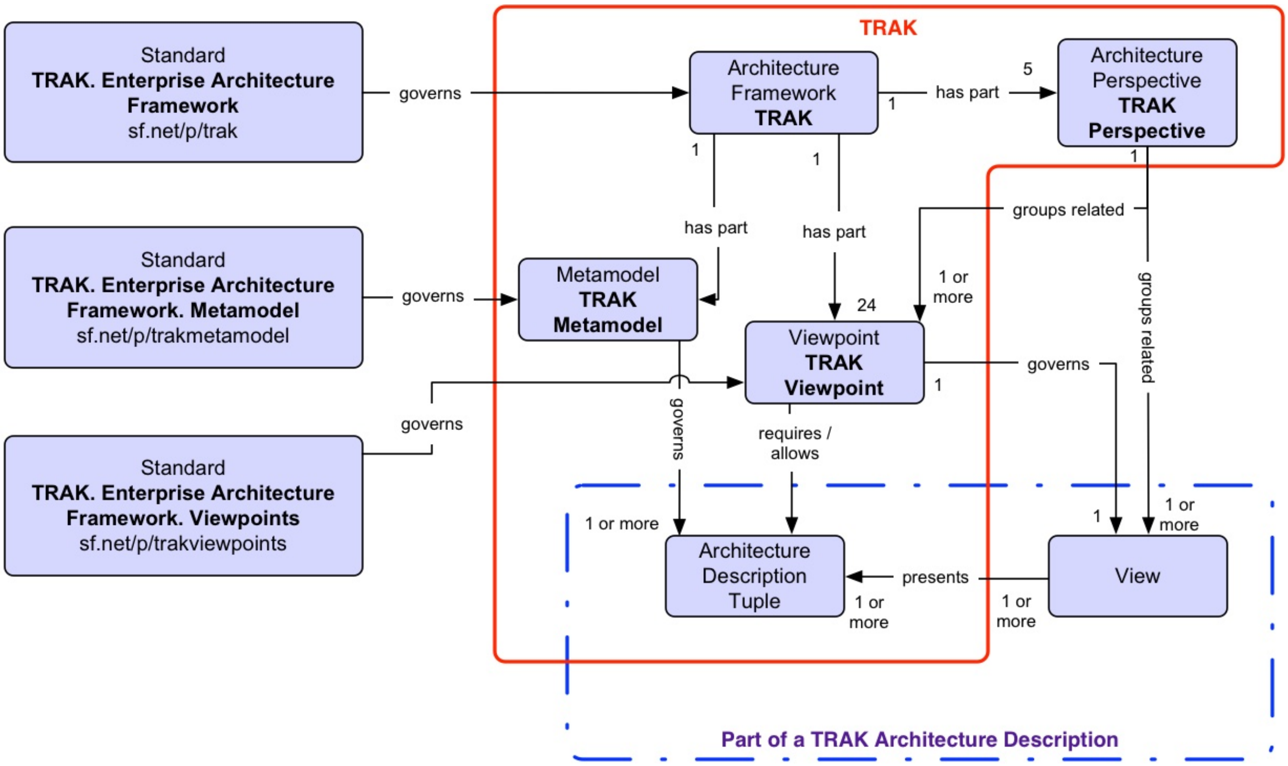


Figure 3-1 Structure of the Framework.

The logical definition of TRAK consists of 3 documents, each of which is an open source project on Sourceforge:

- TRAK Enterprise Architecture Framework document (this document). http://sf.net/p/trak This controls TRAK as a whole.
- TRAK Enterprise Architecture Framework Viewpoints document [Ref. 5] . http://sf.net/p/trakviewpoints. This defines the TRAK viewpoints.
- TRAK Enterprise Architecture Framework Metamodel document [Ref. 4] http://sf.net/p/trakmetamodel. This defines the architecture description elements that can appear in a viewpoint definition.

4 STANDARDS AFFECTING TRAK

TRAK is derived from the Ministry of Defence Architecture Framework (MODAF) version 1.2. A comparison at launch between TRAK and MODAF 1.2. is provided separately in the TRAK viewpoints [Ref. 5] and metamodel [Ref. 4] documents.

TRAK is designed to be compliant with ISO/IEC 42010:2011. The TRAK metamodel defines properties of every architecture description element, for example geographic information, language, typical document properties and in doing so depends on commonly available standards for these things.

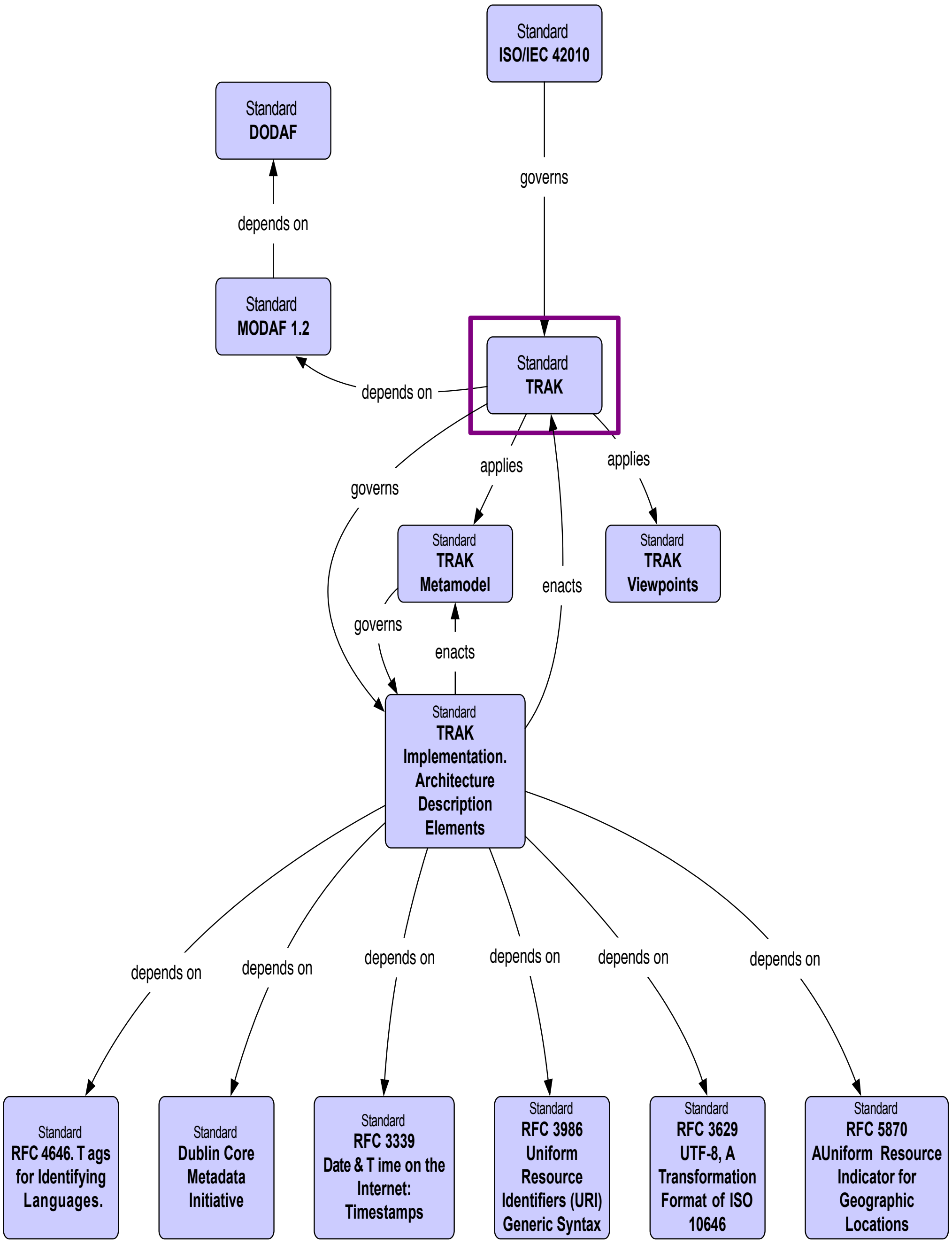

Standard ISO/IEC 42010
Standard DODAF
governs
depends on
Standard MODAF 1.2
depends on
Standard TRAK
governs
applies
applies
Standard TRAK Metamodel
Standard TRAK Viewpoints
enacts
governs
enacts
Standard TRAK Implementation. Architecture Description Elements
depends on
depends on
depends on
depends on
depends on
depends on
Standard RFC 4646. T ags for Identifying Languages.
Standard Dublin Core Metadata Initiative
Standard RFC 3339 Date & T ime on the Internet: Timestamps
Standard RFC 3986 Uniform Resource Identifiers (URI) Generic Syntax
Standard RFC 3629 UTF-8, A Transformation Format of ISO 10646
Standard RFC 5870 AUniform Resource Indicator for Geographic Locations

Figure 4-1 MV-03 - Standards Governing & Used by TRAK

4.1 ISO/IEC 42010

TRAK is governed by ISO / IEC 42010 the international standard for architecture description.

A broad comparison or mapping of the requirements of sections 6 (Architecture Frameworks and Architecture Description Languages) and 7 (Architecture Viewpoints) is provided below. A detailed comparison of TRAK as an architecture framework against the requirements in section 6.1 of the standard is provided on the trak project site on Sourceforge. This also compares a TRAK-conforming architecture description against section 5 (Architecture Descriptions) of the standard.

ISO 42010:2011	TRAK
Definitions	See Glossary
System of Interest'	TRAK has several potential 'systems of interest' in terms of stakeholders. These are:- •the enterprise •the concept •the solution •the architecture task A TRAK architecture description will typically cover at least 2 systems of interest (the task plus at least one other) in ISO 42010 terms.
6.1 Architecture Framework	The TRAK version is identified by issue date. See Configuration Management. TRAK is identified by 'TRAK' and views or viewpoints disambiguated from other frameworks with similarly numbered views using the TRAK:: namespace prefix. TRAK viewpoints are specified (TRAK Viewpoints document). The concerns addressed and the stakeholders having those concerns are identified in each TRAK viewpoint. Correspondence rules, if appropriate, are identified within each viewpoint. TRAK Bye Laws and Master Architecture Views control the allowable sets of views. Additional rules within the TRAK Metamodel control the order in which relationships are made where there are alternative paths within the metamodel.
6.2 Architecture Description Compliance with an Architecture Framework.	Rules for compliance with TRAK and inclusion and marking of non-compliant views are defined. Each TRAK Architecture Viewpoint defines the stakeholder, their concerns addressed and correspondence rules that apply to the architecture description.

ISO 42010:2011	TRAK
6.3 Architecture Description Language.	Advice on the choosing one or more Architecture Description Language (ADL) to represent TRAK views is provided. TRAK does not mandate any ADL nor can it mandate what any ADL is capable of representing - this is why this has to be a choice by the architect in selecting a suitable ADL for the architecture task.
7 Architecture Viewpoints.	Each TRAK viewpoint is specified in terms of concerns, stakeholders, content by reference to the TRAK metamodel and presentation (model kind). Additional conventions for notation and presentation of TRAK views are defined in the TRAK Bye Laws. Each TRAK viewpoint has its own version identification information. A change record for the TRAK Viewpoints document is maintained as part of the subversion repository that is used to managed the TRAK Viewpoints project at http://trakviewpoints.svn.sourceforge.net/viewvc/trakviewpoints/trunk/?view=log

Table 4-1 Mapping of TRAK Against ISO/IEC 42010

There are some slight differences - ISO 42010 uses 'system concern' which is potentially confusing as the concern isn't one that belongs to the system. TRAK uses 'concern' coupled with 'Role has Concern' and 'Concern traces to ... [any architecture description element]' to express both task stakeholder concerns and concerns raised during the architecture description by architects.

5 GLOSSARY

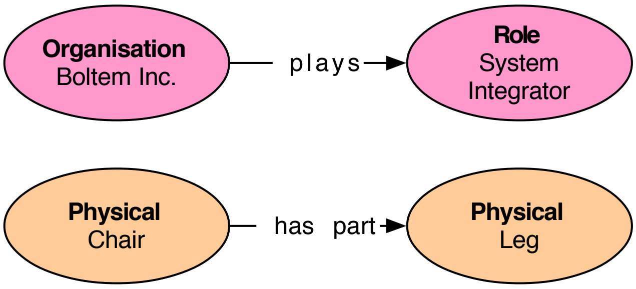
Architecture Description. An architecture product that represents an architecture of a system-of-interest. This is often shortened to 'AD' . As the system-of-interest may itself comprise other systems a AD can exist at many levels and may share architecture views with other ADs if the stakeholder concerns for the different architecture description tasks overlap.

Architecture Description Element. An individual architecture description object that is used to describe or represent an item of real-world architecture. An architecture description element can appear in an architecture description. Note that within the TRAK metamodel not all elements are Architecture Description Elements – only those that may appear in an Architecture View are. The remaining non Architecture Description Elements in the TRAK metamodel are used for communication and the management of TRAK itself.

Architecture Description Language (ADL). ISO/IEC/IEEE 42010 defines an ADL as 'form of expression used for the description of architectures'. Examples include the UML, SysML, BPMN, a directed property graph.

Architecture Perspective. ISO 42010 refers to an Architectural Perspective as 'Sharing of architectural models also facilitates an "aspect-oriented" style of architectural description'. In TRAK it is a grouping of related and overlapping architectural viewpoints. Views produced in response to viewpoints can be similarly grouped by perspective within an architecture description.

Architecture Description Tuple. The smallest permissible unit of architectural description, it consists of a named stereotype/type/label with a named relationship to a named stereotype/type/label. Each of these is an Architecture Description Element.



Architecture Viewpoint. In ISO/IEC/IEEE 42010 terms an architecture viewpoint is something that 'establishing the conventions for the construction, interpretation and use of architecture views'. It is a specification for an architecture view type. Note that this is different to the way in which this term is used for MODAF® and DODAF.

Master Architecture View A TRAK architecture view which acts as the 'master source' for elements of a particular type e.g. Resource, Function, Concept Activity, Project. In an AD this means that if an architectural element of a type is created it has to be shown (declared) first on the appropriate view which is defined by the master architecture viewpoint. For example, before a Resource (System, Physical, Software, Organisation, Job or Role) can be used in any other view within the AD it must be shown first on the SV-01 Solution Structure View (specified by the SVp-01 Solution Structure Viewpoint). The Master Architecture View for each architecture description element type is defined in the TRAK Viewpoints document.

Metamodel. A metamodel contains information about a model. It defines a set of stereotypes and any relationships. The TRAK metamodel defines a generic model of "the real world" according to TRAK in terms of the things that make it up (within the scope of TRAK). TRAK architecture views can only use things taken from the TRAK metamodel - the architecture viewpoint defines what subset of the TRAK metamodel, Architecture Description Tuples, must and may appear in any viewpoint and therefore view.

Stereotype / Type / Label. In the same way that we group a subset of people together and apply characteristics e.g. "All teenagers rebel against their parents" a stereotype in TRAK defines an element within the TRAK metamodel, what real world thing (if applicable) it represents, its relationships and properties. For example, the TRAK stereotype 'Organisation' defines the general characteristics for any organisation element that appears in an architecture description and a Document has properties taken from the Dublin Core Metadata Initiative that apply to every document element. This corresponds to a 'label' in graph notation.

6 CHOICE OF ARCHITECTURE DESCRIPTION LANGUAGE (ADL)

Appendix A of ISO/IEC/IEEE 42010:

> '*The term architecture description language (ADL) has been in use since the 1990s in the software, systems and enterprise architecture communities. Within the conceptual model of this International Standard, an architecture description language is any language for use in an architecture description. Therefore an ADL must be usable in one or more viewpoints within an architecture description to frame some identified system concerns.*'

TRAK does not specify what architecture description language, such as the UML, SysML , BPMN, directed graph or RDF etc. is to be used for architecture description. TRAK is agnostic on this and the choice is a local one.

There are considerations when choosing an architecture description language that will affect the choice:

- how much of the TRAK metamodel is it possible to represent using the ADL? Most ADLs themselves have some form of metamodel and they may or may not have the means to represent parts of the TRAK metamodel. If they cannot then this may not matter depending on what concerns are to be addressed using the architecture description - if the TRAK metamodel elements for the TRAK viewpoints needed for the task are covered then the architecture task can be completed. If there is inadequate coverage either another. ADL has to be used or a combination of ADLs has to be used (although this adds practical complications).
- familiarity and experience. If an ADL is commonly used then the training requirements will be smaller.
- modelling tool support. Some modelling tools support a limited set of ADLs and therefore choosing a new ADL might require either new tools or extension of existing ones.

It is possible that a single ADL can cover the TRAK stereotypes in the viewpoints needed for an architecture task. If it does then there is no need to use multiple ADLs within an architecture description.

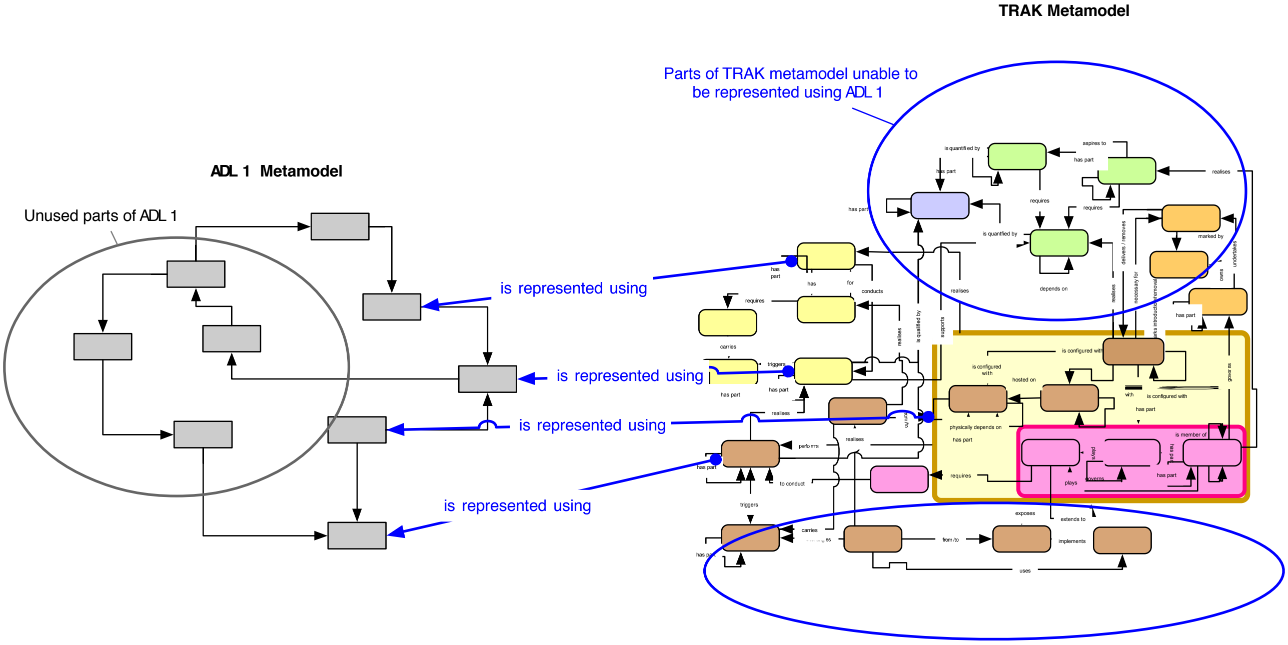


Figure 6-1 A Mapping Between An ADL Metamodel and the TRAK Metamodel Identifies What It Is Possible to Represent of TRAK Using the ADL (Fitness for Task)

The assessment and the decision to use a particular ADL or set of ADLs should be recorded. Usually this decision will be made on behalf of a project and cover many architecture tasks over a long period of time. The decision isn't part of the framework and therefore the best place to record this will be as part of an engineering management pan, for example an Architecture Description Modelling Plan.. A reference to this should be made in the MV-02 Architecture Description Design Record.

An assessment of the UML in representing TRAK Viewpoints as implemented in the UML for TRAK UML profile [Ref. 7] has been provided as a central resource for use by others.

7 CONFORMANCE WITH TRAK

7.1 Introduction

ISO 42010 states: 'An organization desiring to produce an architecture framework for a particular domain can do so by specifying a set of viewpoints and making the selection of those viewpoints normative for any AD claiming conformance to the domain-specific architectural framework.'

In other words TRAK, as an architecture framework, has to specify a set of viewpoints.

7.2 Conformance with TRAK

Any conforming architecture product shall meet the requirements of :-

- this document (sections 8 - 10 inclusive)
- TRAK Metamodel [Ref. 4] (sections 2 - 3 inclusive and Relationship Rules in section 4)
- TRAK Viewpoints [Ref. 5] (sections 8 - 12 inclusive) documents.
- include a means to identify the version of the AD e.g. number, date and time

Any AD that wishes to claim conformance to TRAK shall select from viewpoints within the TRAK Viewpoint document.

Only views that conform to TRAK are allowed to use the TRAK view names or numbering.

The AD can state the version of TRAK to which it conforms by date (see configuration). Alternatively, if no date is stated it shall be deemed by default to comply with the latest version of TRAK.

7.3 Non-Conformance with TRAK

Non-conforming architecture products may be incorporated into a conforming TRAK architecture description. Each product shall, however, be explicitly identified as non-conformant to TRAK.

Non-conforming products shall not use TRAK view names or numbering (in order to maintain a clear separation).

Note that the architecture products of other frameworks can use their own numbering/naming providing that this is qualified using a namespace separator after the framework e.g. MODAF:: (See also TRAK Architecture Viewpoints document [Ref. 5] - Viewpoint Identification).

8 TRAK ARCHITECTURE PERSPECTIVES

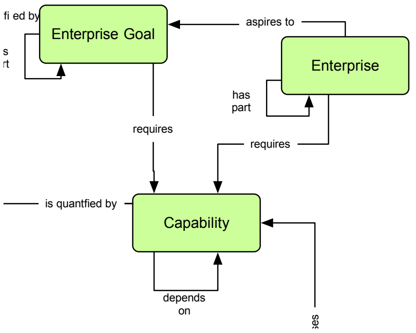
8.1 Introduction

Architectural perspectives provide a useful means of grouping or organising viewpoints (and views in an architecture description) and identifying a top-level theme or focus amongst which viewpoints will overlap. Like any contents list in a document they help organise and bring together what might otherwise be a long and disparate list of viewpoints.

TRAK contains the following architecture perspectives:-

- Enterprise Perspective
- Concept Perspective
- Procurement Perspective
- Solution Perspective
- Management Perspective

8.2 Enterprise Perspective

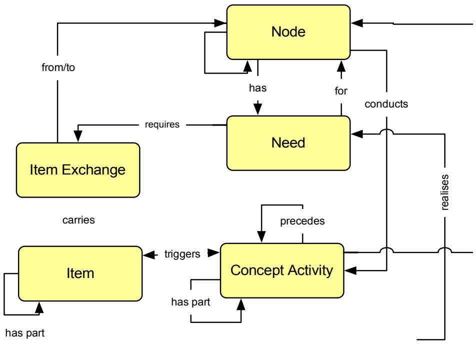


This perspective describes the enterprise in terms of its goals and the enduring capabilities that are required to support the goals. These are high level business needs that everything else contributes to and form part of the long term strategic objectives that need to be managed.

The typical stakeholders are the owner, developer, planner and maintainer of the enterprise.

The Enterprise Perspective is described by the EVp-01 Enterprise Goal, EVp-02 Capability Hierarchy and EVp-03 Capability Phasing TRAK viewpoints.

8.3 Concept Perspective



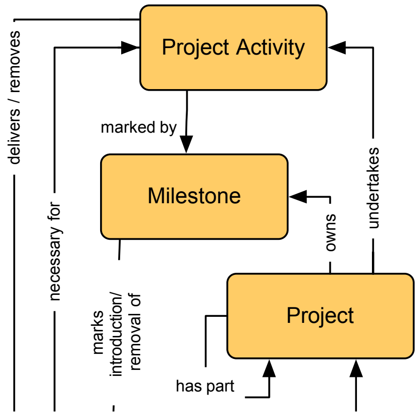
The Concept Perspective describes the solution-free (logical) view of what is needed in response to the capabilities required by the enterprise in the Enterprise Perspective. It describes the logical connection of nodes, for example a service control centre, to other nodes with no recognition of how this might be realised either by organisation or technology. It also implies no particular part of a life cycle – it covers everything from concept to disposal ("lust to dust"!) - time is only introduced deliberately in either the enterprise and / or procurement perspectives.

Any normative documents or standards applied to the concept and described in the Management Perspective are likely to be technology-free – they won't describe "the how".

The typical stakeholders are the user or operator of the concept. Stakeholders for the solution and for the enterprise are also likely to be involved since the concept will impact on the solution and the ability to realise the enduring capabilities.

The Concept Perspective is described by the CVp-01 Concept Need, CVp-02 Concept, CVp-03 Concept Item Exchange, CVp-04 Concept Activity to Capability Mapping, CVp-05 Concept Activity and CVp-06 Concept Sequence TRAK viewpoints.

8.4 Procurement Perspective

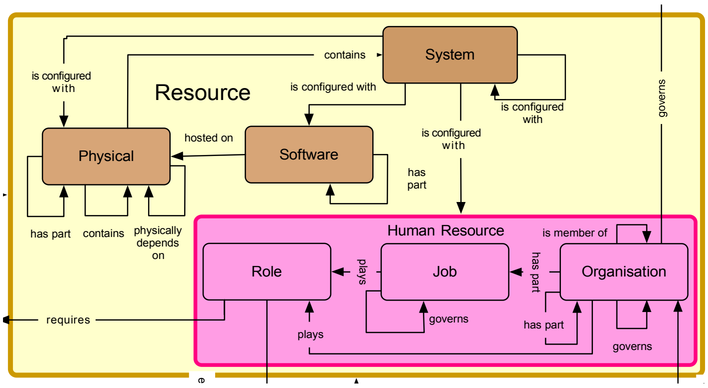


The Procurement Perspective provides a top level view of the procurement of a solution to satisfy the enterprise capability needs outlined in the Enterprise Perspective and developed in the concept perspective. It provides a way of showing how projects deliver the solutions described in the Solution Perspective to provide capability. It provides a way of showing time dependency between projects owing to dependencies on systems being introduced or removed and is an essential for investigating capability gaps. It also provides a way of showing how responsibility boundaries change over time.

The typical stakeholders are the acquirer, developer and builder of the solution. The owner and builder of the enterprise will also have an interest in terms of the effect on enterprise capabilities.

The Procurement Perspective is described by the PrVp-01 Procurement Structure, PrVp-02 Procurement Timeline and PrVp-03 Procurement Responsibility TRAK viewpoints.

8.5 Solution Perspective

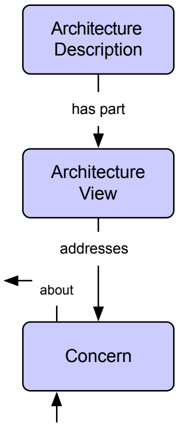


The Solution Perspective describes the solution – whether proposed or realised. It covers the parts of 'systems' whether human or machine, their exchanges and protocols. It describes how organisations and equipments are organised and governed. The Solution Perspective describes how the logical requirements outlined in the Concept Perspective are realised and shows how the solution(s) realise the capabilities needed by the enterprise and described in the Enterprise Perspective.

The typical stakeholders are the owner, acquirer, developer, builder, maintainer and trainer of the solution.

The Solution Perspective is described by the SVp-01 Solution Structure, SVp-02 Solution Resource Interaction, SVp-03 Solution Resource Interaction to Function Mapping, SVp-04 Solution Function, SVp-05 Solution Function to Concept Activity Mapping, SVp-06 Solution Competence ~~and~~, SVp-07 Solution Sequence, SVp-11 Solution Sequence and SVp-13 Solution Risk TRAK viewpoints.

8.6 Management Perspective



The Management Perspective describes the architectural task and those relationships that are common across other perspectives. It provides ways of defining the scope and findings of the architectural task - structuring the approach and modelling.

The Management Perspective provides ways of describing the requirements and normative standards that apply..

It provides supporting information to aid the portability and understanding of the architecture description produced as a result of the task.

As the Management Perspective underpins all other perspectives all roles are beneficiaries including the lay reader (or external third party) to the architecture description.

The Management Perspective is described by the MVp-01 Architecture Description Dictionary, MVp-02 Architecture Description Design Record , MVp-03 Requirements & Standards and MVp-04 Assurance TRAK viewpoints.

9 USE OF COLOUR IN TRAK

9.1 Introduction

The consistent use of colour within TRAK is essential to the user interface with the framework and the architecture descriptions produced using it. It greatly aids checking and trapping of errors on views. For example, if something brown (Solution Perspective) appears on a EV-01 Enterprise Goal you know there's a problem.

Colours are not used to represent things best served by relationships e.g. ownership or responsibility or other boundaries. This can be achieved using the TRAK metamodel and the appropriate TRAK viewpoint. In most tools relationships can be explored and navigated along and they export well retaining their original meaning. Colours don't and if everyone uses their own scheme the result is clutter and potential confusion.

Web-safe colours have been specified in the hope that this will minimise variation whilst still providing a colour set that is easy to apply using standard tools from word processors upwards. Not perfect, but good enough!

Colours are applied at the Architecture Perspective level in all cases except for Human Resource and Competence. In this case it was felt useful to be able to visually distinguish those elements that are closely aligned with people from those that are likely to be machines.

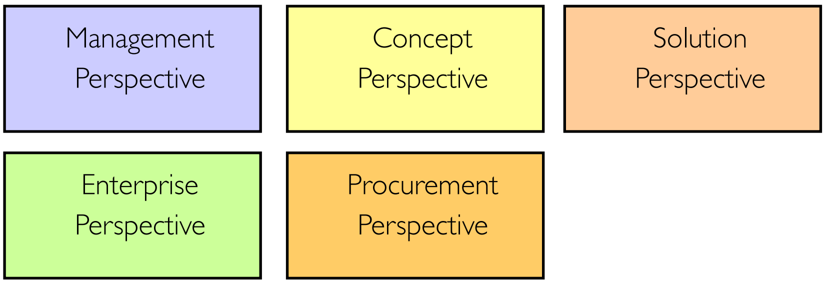


Figure 9-1-TRAK Architecture Perspective Colours

9.2 Colour Rules – General

- CR12 The rules for colour only apply to graphical representation of blocks are displayed on a view. Where a textual notation or similar is used colours must not be applied to the text (legibility).
- CR11 Since colours have a meaning in TRAK the colours defined must not be used for other types of thing in an architecture description.

9.3 Colour Rules – Architecture Perspective and Viewpoints

- CR1. A stereotype takes on the colour of the perspective if it only occurs in that perspective.

- **CR2**. A stereotype takes on the colour of the Management Perspective if it occurs in more than one perspective e.g. Metric.
- **CR8**. The colour for Human Resource (Organisation, Job, Role) is Hexadecimal FF99CC (RGB 255, 153, 204)
- **CR9**. The colour for Competence is Hexadecimal FF99CC (RGB 255, 153, 204)
- **CR10**. A graphic may be used for presentation purposes to dress any architecture element in an architecture view (note that that the TRAK Viewpoints still require an architecture description tuple - TRAK Bye Law BLV-3 and BLV-10). It is essential, however, that the object types and the relationships are preserved and discoverable. See Figure 9-2.

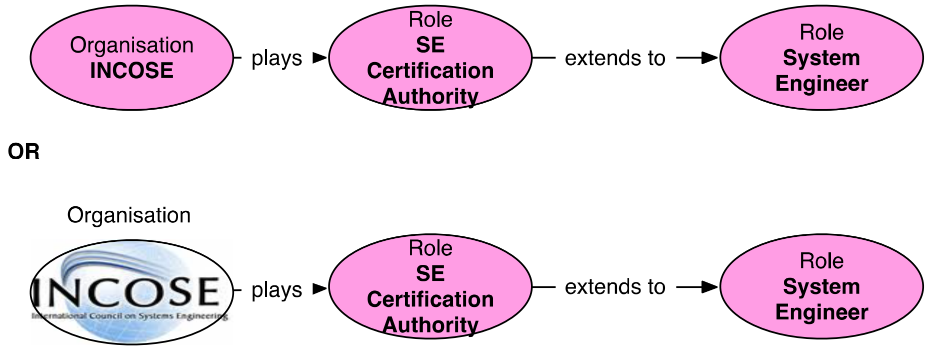


Figure 9-2- Architecture Description Tuple with a Graphic for Presentation Purposes

The following colour rules apply to TRAK architecture perspectives and viewpoints:

- **CR3**. The colour for the Management Perspective is Hexadecimal CCCCFF (RGB 204, 204, 255)
- **CR4**. The colour for the Enterprise Perspective is Hexadecimal CCFF99 (RGB 204, 255, 153)
- **CR5**. The colour for the Concept Perspective is Hexadecimal FFFF99 (RGB 255, 255, 153)
- **CR6**. The colour for the Procurement Perspective is Hexadecimal FFCC66 (RGB 255, 204, 102)
- **CR7**. The colour for the Solution Perspective is Hexadecimal FFCC99 (RGB 255, 204, 153)

10 TRAK BYE LAWS

British Transport Commission
BRITISH RAILWAYS GENERAL BYE-LAWS
are available for inspection
at this booking office

10.1 Introduction

One of the problems faced by any architecture framework is consistency. This affects the meaning or understanding of the models produced and also the organisation and ability to exchange.

Within TRAK there are several mechanisms through which consistency is either enforced or through which the potential for divergence is minimised:

- the consistent use of colour
- completeness of coverage /presentation of the TRAK metamodel
  - specification of mandatory and optional architectural tuples through TRAK architecture viewpoints.
- consistency between architecture views
  - consistency rules are specified where necessary for each TRAK architecture viewpoint
- master architecture views are defined as the prime views in which a TRAK metamodel block element is shown and created ('declared'). This leads to dependencies between TRAK architecture views and minimum allowed architecture view sets (see TRAK viewpoints document [Ref. 5]). This helps provide consistent sets of views and makes it easier for the reader to find elements (it increases the affordance / visibility as the reader knows, for example, that solution structure will be described in a SV-01view and not introduced for the first and only time in another view).

Some rules need to be applied across the architecture description as a whole and apply to the framework definition. This is why the idea of 'TRAK Bye Laws' arose (borrowing from the railway domain).

The following are rules that apply globally in the definition of TRAK and the creation of TRAK-compliant architecture descriptions.

10.2 Metamodel

The following laws apply to the design of the TRAK metamodel:

- **BLM-1**. Every metamodel stereotype must have one viewpoint defined as its Master Architecture Viewpoint i.e. the stereotype cannot appear in any other viewpoint (and view) without appearing in the specified master architectural view type (this ensures visibility to the reader by making sure that the stereotype and therefore tuple appear in the view type expected and are not hidden away where they might not be expected to be found).
- **BLM-2**. Every relationship must have a natural language label. Not everyone is a UML or what-ML expert and it is important that they be able to read a view in a natural way i.e. tuple (subject-predicate-object) as an understandable sentence.
- **BLM-3**. Metamodel stereotypes used in viewpoints, and therefore views, must not be specialised or typed further. The metamodel used in views is not itself the means to create taxonomy hierarchies and in doing so it forces the modeller/architect to make a choice and therefore risks inconsistency of typing. Non-TRAK diagrams can be created for taxonomies and existing constructs can be used for this purpose.
- **BLM-4.** The number of metamodel stereotypes must be kept to an absolute minimum. The corollary is that maximum use must be made of relationships between stereotypes. This reduces the potential for expansion and the complexity that comes with this. There is a role for support and guidance in showing/explaining how things can be represented in order to reduce the pressure to add new stereotypes or relationships where existing tuple sets can do the equivalent job.

10.3 Viewpoint Definition

The following laws apply to the design of TRAK architecture viewpoints:

- **BLV-1**. Every metamodel tuple must be mandatory in at least one architecture viewpoint (otherwise parts of the metamodel may never get expressed)
- **BLV-2**. Every viewpoint must overlap at least one other viewpoint in terms of content- otherwise the architect cannot easily lead the reader through the architecture description using views alone. This has implications on consistency.
- **BLV-7**. The number of viewpoints must be kept to an absolute minimum - complexity, mistakes in selection and cost of creation and maintenance increases with the number of viewpoints.
- **BLV-8**. Viewpoints must be differentiated/selected based on the metamodel stereotypes/tuples (content) not the application of the viewpoint. There will be many applications for any viewpoint which cannot be anticipated and there is otherwise the danger that the viewpoint set will increase every time

a different community gets involved. The application of viewpoints within domains is best dealt with through support mechanisms e.g. examples and guidance, not framework definition.

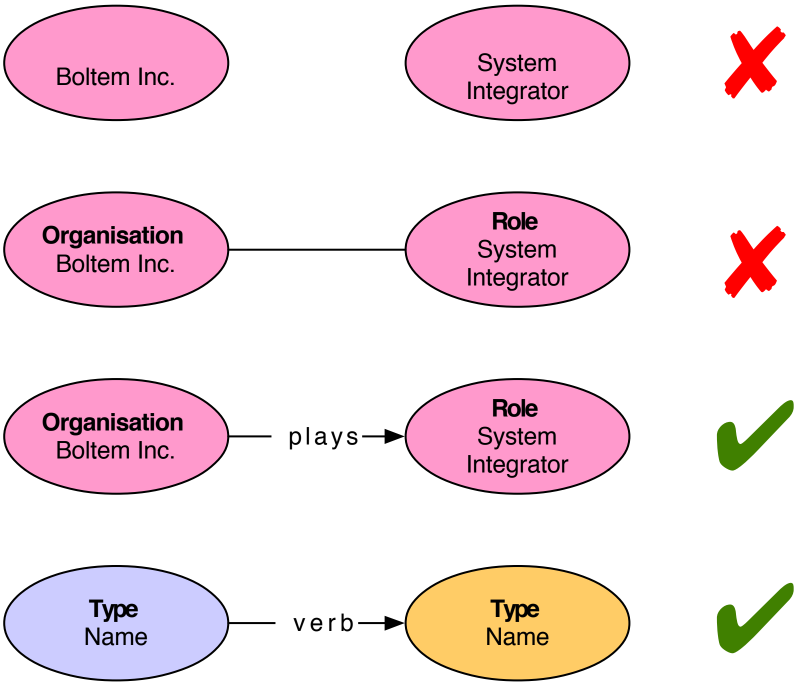


Figure 10-1 - An Architecture Tuple is the Smallest Unit of Architecture Description

10.4 View and Architecture Description Definition

The following laws apply to TRAK-compliant architecture views and architecture descriptions produced:

- **BLV-3**. No orphaned elements. If it's not an ArchitectureTuple it isn't architecture - solitons, orphans have no place in architecture.
- **BLV-4**. Every tuple that applies to the system of interest being described (in a architecture description) must appear on a view within that architecture description.
    - **BLV-4.1** No creating relationships that aren't visible on a view
    - **BLV-4.2**. A tuple involving architecture elements from 2 systems-of-interest must appear in the corresponding view type in both architecture descriptions (e.g. System A has an interface with System B then it must appear in the correct view type - SV-02 Solution resource Interaction View - in the ADs of both System A and System B)
- **BLV-5**. The architecture description must be self-documenting - it needs to be understandable and portable
- **BLV-6**. Every view must overlap at least one other view - otherwise the architect cannot easily lead the reader through the architecture description using views alone.

- **BLV-11**. The set of views forming an AD shall meet the minimum set requirements defined in the TRAK Viewpoints document, 'Minimum Allowed TRAK Architecture View Sets' Table 3-1 [Ref. 5] .
- **BLV-12** The set of views for an AD must have an identifiable start, middle and end. The AD is a 'document' - it should tell a story, have an identifiable structure and reading paths. Following ISO/IEC/IEEE 42010 principles the start point should be the architecture task and its scope. The MV-02 Architecture Description Design Record provides a mechanism to capture this and help top and tail the AD.
- **BLV-9**. Every view and AD must include a version identifier.
- **BLV-10**. Every object must have a natural language label, identifying what type of object it is (taken from the TRAK metamodel) i.e. Organisation, Standard, Node etc. (See also BLM-2)

11 TRAK MINIMAL MODELLING PROCESS

TRAK does not specify a detailed modelling process or the minimum views that must be produced for any architecture task. Elements of process or sequence are introduced because:

- there are dependencies between TRAK views
- the MV-02 Architecture Design Record must always be produced for each architecture description
- being based on ISO/IEC/IEEE 42010 TRAK requires that an architecture description is produced to address the task sponsor's concerns.

The bare-bones process mandated in conforming to TRAK is therefore:

1. Agree architectural task scope with task sponsor and stakeholders and record using the MV-02. The MVp-02 viewpoint includes the constructs needed to capture the task scope.

2. Choose appropriate TRAK architecture views for the task [Ref. 5] . In accordance with ISO 42010 architecture views are selected using the concerns specified in the viewpoints. These will in turn, owing to the concept of Master Architecture Views in TRAK, define the minimum set of architecture views needed for the task.

3. Create the architecture description .

4. Close-out the architecture description with the MV-02 and, if necessary, the MV-01. This will capture the findings from the modelling and provide enough information for others to understand why the architecture description was done and what resulted. The MV-01 is needed for portability.

What TRAK or any other framework won't specify but which are important considerations are:

- how to plan for the models needed or to be developed for the task
- how to organise the structure of the repository
- how to model
- how to organise views for readability or ease of navigation

12 REFERENCES

[Ref. 1] ISO/IEC/IEEE 42010:2011 Systems and Software Engineering — Architecture Description

[Ref. 2] DODAF 1.5[2]. The US Department of Defense Architecture Framework.

[Ref. 3] MODAF 1.2. The UK Ministry of Defence Architecture Framework. https://www.gov.uk/guidance/mod-architecture-framework

[Ref. 4] TRAK00002. TRAK. Enterprise Architecture Framework. Metamodel. sf.net/p/trakmetamodel

[Ref. 5] TRAK00001. TRAK. Enterprise Architecture. Viewpoints. sf.net/p/trakviewpoints

[Ref. 6] Sourceforge. Known implementations of TRAK. trak.sourceforge.net/implementations.html

[Ref. 7] TRAK UML profile. sf.net/p/trakumlprofile

[Ref. 8] MDG Technology for TRAK (Sparx Systems Enterprise Architect). sf.net/p/mdgfortrak

[Ref. 9] TRAK Architecture Framework RSS Feed. http://sourceforge.net/p/trak/activity/feed

[Ref. 10] Architecture Description Language (ADL) vs Architecture Framework. http://trak-community.org/index.php/residualWorld/architecture_description_language_vs_architecture_framework

[Ref. 11] TRAK00010. TRAK, Enterprise Architecture Framework, TRAK vs ISO/IEC 42010. http://sourceforge.net/projects/trak/files/ISO%2042010/

[Ref. 12] TRAK00007. TRAK. Implementation. ADL Suitability Assessment. UML. http://sourceforge.net/projects/trak/files/Suitability_of_Architecture_Description_Languages/UML/

[Ref. 13] TRAK00005. TRAK. Implementation. Architecture Description Elements. http://sourceforge.net/projects/trak/files/Implement%20TRAK/

[Ref. 14] GNU Free Documentation License, version 1.3, November 2008. http://www.gnu.org/licenses/fdl-1.3.html

[Ref. 15] TRAK00013. TRAK. Architecture Description. Conformance Assessment – ISO/IEC/IEEE 42010:2011. http://trak.sourceforge.net/TRAK%20vs%20ISO_42010_AD/index.htm

[Ref. 16] TRAK Twitter Feed - https://twitter.com/TRAK_AF

[Ref. 17] Wikipedia- Directed Graph https://en.wikipedia.org/wiki/Directed_graph [Accessed 5th Dec 2015]

[2] The current version of DODAF is 2.0 at http://cio-nii.defense.gov/sites/dodaf20/

INDEX

BACK COVER