HUMAN FACTORS - ON THE RIGHT TRAK?

Chris Lowe

Liv Systems Ltd.

www.liv-systems.com

Nic Plum

Eclectica Systems Ltd.

www.eclectica-systems.co.uk

HUMAN FACTORS - ON THE RIGHT TRAK?

HUMAN FACTORS - ON THE RIGHT TRAK?

Background

UK rail is characterised by a complex industry organisation with many private sector companies (TOCs, ROSCOs, and other duty holders) with economic and safety regulation by the government. Projects are increasingly more complex and interdependent.

Engineering in the UK rail sector is characterised by strong traditional engineering disciplines. This makes it hard to maintain cross-discipline links, for example, collaboration. Systems Engineering as the "new boy on the block" is not established as a discipline and has to continue to fight its corner and justify its existence in a way that would be unthinkable in domains such as defence and aerospace where it is taken as read. Systems Engineering often doesn't therefore enjoy the influence that the traditional disciplines do.

There is an emphasis on interfaces and system integration which are important aspects but not "the whole" . Systems Engineering is heard to be good because it is "systematic" rather than "systemic" which is worrying as all scientific and engineering disciplines ought to be systematic but only systems engineering looks at structure, interfaces and ways of managing emergent behaviour of the whole. There isn't a common and accurate understanding of what systems engineering is within the industry.

Engineering does have a long and mature pedigree within rail but this makes it hard to add new practice or ideas as we're often faced by the "we've always done it this way" or "we've managed for XX years without systems engineering, so ..."

Architecture description as a relatively newish part of Systems Engineering is therefore right on the fringes of acceptability and understanding, even though some have clearly been engaged in it for many years albeit not labelled as such. It isn't at all well understood or accepted.

Meanwhile the UK Department for Transport, the ultimate customer, are getting fed up with picking up the pieces where systems fail to integrate and therefore acting as the defacto System Design Authority..

A SE-oriented architectural framework if it is be usable has to address these challenges. The framework needs to unite or integrate the different stakeholder viewpoints on the same underlying system or problem. The framework also needs to be usable by these different stakeholders themselves - both in the creation of architecture models, and in the communication of the content and ideas within the models and the application to everyday SE activities.

Therefore, TRAK ....

Against these challenges TRAK:

- needs to bring disciplines together - the engineering of the product system cannot be achieved by any one discipline in isolation.
- must be simple, understandable - not everyone who contributes to or who needs to read from the architecture description will be UML experts. Stakeholders are often project managers or they might have a commercial rather than engineering background.
- must be usable and relevant to typical problems faced by SE practitioners/stakeholders - it must reflect common development process needs and applications and be able to represent typical problems faced therein. If it is difficult to use or not focussed on what is needed there is little point. If it doesn't fit it won't be used.
- must provide a means to illustrate, describe and facilitate discussion. Much of this is about communication and providing understandable and practicable ways to highlight and solve problems. We therefore need to think about how we provide a common language and how it is presented. We also need to think about how people navigate through the architecture description.
- must augment or complement existing methods, tools, techniques. No one tool does everything or everything well. We need to make the best use of existing specialist tools and their products whilst adding products and capabilities to explore relationships that only architecture description provides.

All of these are very human-centric factors which affect the definition of TRAK as well as the development process and the application of TRAK architecture products. People do matter a lot! But then again as systems engineers we all knew this, didn't we...?

The Application of HF in the Design of TRAK

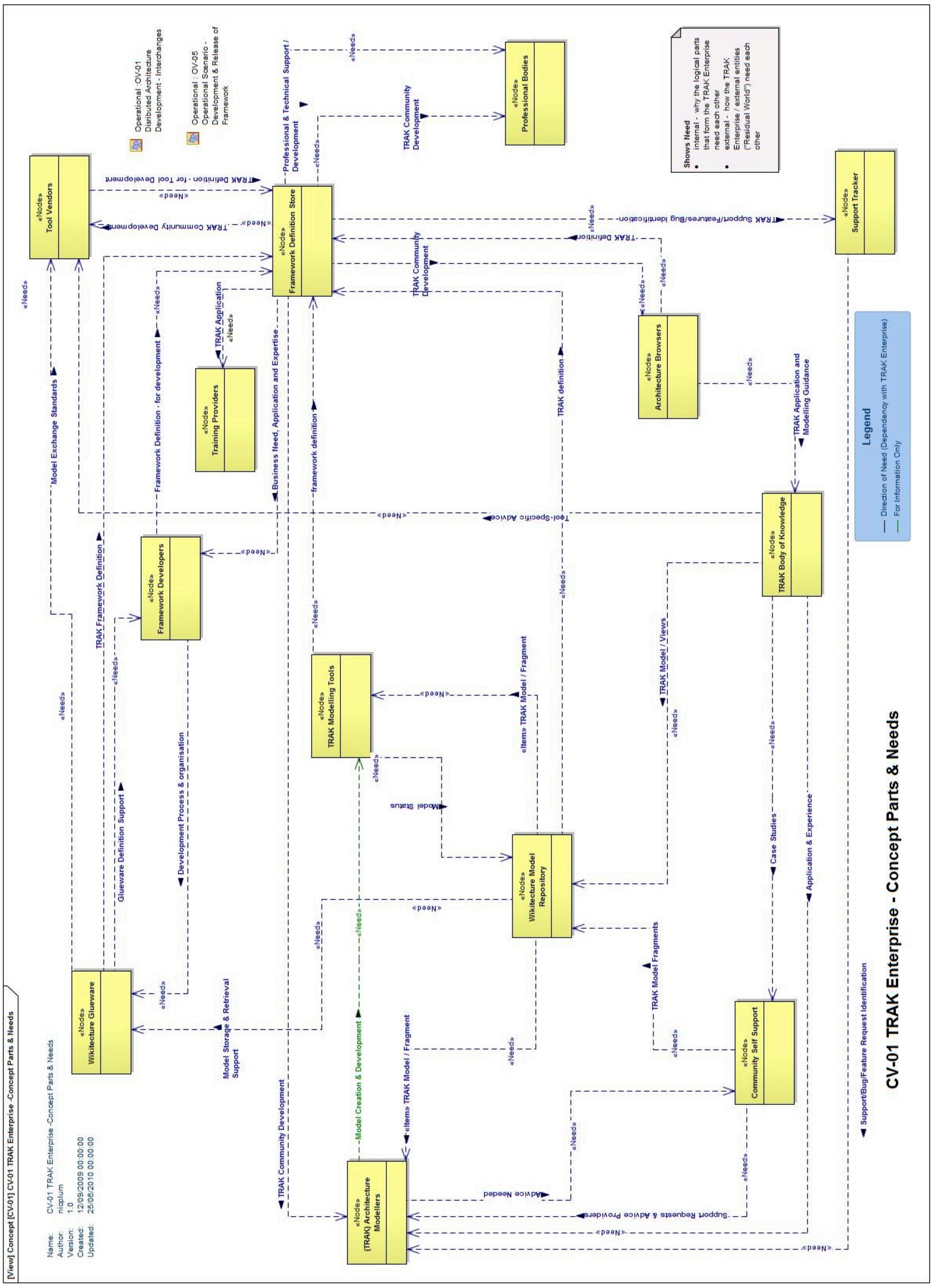
Does an Architecture Framework Have a User Interface? ...The TRAK Enterprise

An enterprise architecture framework on its own probably doesn't have a user interface (ignoring the expression in documentation) so what is the user interface and how can we be concerned with the qualities and interactions that any user interface might have?

As with any other part of systems engineering it's what we define as the system of interest. The system of interest to us, which we call "The TRAK Enterprise" comprises the following parts:

TRAK Enterprise	
Logical Part	**Description**
Community Self-Support	The parts that enable the TRAK wikitecture/ architectural community to help each other and enable interaction with framework definition, products etc.
Framework Definition Store	The definition of TRAK that is exposed and which can be interacted with. This includes the metamodel and specification of views.
Support Tracker	A means of systematically capturing • bugs • support requests • feature requests in relation to TRAK i.e. metamodel, viewpoints, products/templates in a way that allows the response to be open, visible and linked to the original request or problem.
TRAK Body of Knowledge	The store, means of capturing • use of TRAK • advice / problem solving • tailored templates • application of TRAK products • academic / public papers • case studies • links to external sources of information
Wikitecture Glueware	The wikitecture components that are necessary to enable models to integrate, be exchanged and be understood
Wikitecture Model Repository	Public / shareable store(s) of TRAK models / fragments. Distributed rather than centralised.

The TRAK Enterprise explicitly excludes those things that are outside reasonable control such as tools and the various types of user of TRAK or TRAK products although the enterprise has to clearly interact, influence and anticipate their demands and likely use.


[View] Concept [CV-01] CV-01 TRAK Enterprise -Concept Parts & Needs
Name: CV-01 TRAK Enterprise -Concept Parts & Needs
Author: nicplum
Version: 1.0
Created: 12/09/2009 00:00:00
Updated: 25/06/2010 00:00:00
Operational :OV-01 Distributed Architecture Development - Interchanges
Operational : OV-05 Operational Scenario - Development & Release of Framework
«Node» Tool Vendors
«Node» Framework Definition Store
«Node» Professional Bodies
«Node» Training Providers
«Node» Architecture Browsers
«Node» Support Tracker
«Node» Framework Developers
«Node» TRAK Body of Knowledge
«Node» TRAK Modelling Tools
«Node» Wikitecture Glueware
«Node» Wikitecture Model Repository
«Node» Community Self Support
«Node» (TRAK) Architecture Modellers
Shows Need
internal - why the logical parts that form the TRAK Enterprise need each other
external - how the TRAK Enterprise / external entities ("Residual World") need each other
Legend
Direction of Need (Dependency with TRAK Enterprise)
For Information Only
CV-01 TRAK Enterprise - Concept Parts & Needs
«Need»
TRAK Definition - for Tool Development
TRAK Community Development
Professional & Technical Support / Development
TRAK Community Development
TRAK Support/Features/Bug Identification
TRAK Definition
TRAK Community Development
TRAK Application
Model Exchange Standards
Framework Definition - for development
Business Need, Application and Expertise
framework definition
TRAK definition
TRAK Application and Modelling Guidance
Tool-Specific Advice
TRAK Framework Definition
Glueware Definition Support
Development Process & organisation
«Item» TRAK Model / Fragment
TRAK Model / Views
Model Status
Case Studies
Application & Experience
TRAK Model Fragments
Model Storage & Retrieval Support
Model Creation & Development
TRAK Community Development
«Item» TRAK Model / Fragment
Advice Needed
Support Requests & Advice Providers
Support/Bug/Feature Request Identification

The external parts of "the residual world" with which the TRAK Enterprise has to interact are:

External to the TRAK Enterprise (="Residual World")	
Logical Part	**Description**
Architects	The people that create and maintain architecture descriptions/models.
Architecture Browsers	The organisations and people that browse, consume or need to be able to understand the models and other architectural products. Often from other domains, not technical and probably key drivers in the user-interface of TRAK and communication-effectiveness.-interface of TRAK.
Professional Bodies	Often technical, the professional discipline or functional expertise bodies whose members are affected by TRAK
TRAK Modelling Tools	The tools that implement or use TRAK or produce TRAK-compliant architectural products.
Tool Vendors	Companies that provide or develop modelling tools that are relevant to TRAK
Training Providers	Companies that provide or develop training resources/provision for TRAK.

There are clearly many human roles involved and they will interact with the framework definition, in both document form and any implementation through modelling tools. The glueware that aids modelling and exchange of models will impact on the architects. Even the casual browser or reader will interact with the models produced and the definition, the notation language used in implementation and the construction and presentation and layout of the model will all impact.

So, yes, there is a user interface which encompasses many parts of our system of interest and we'd be silly as systems engineers to put blinkers on and ignore this.

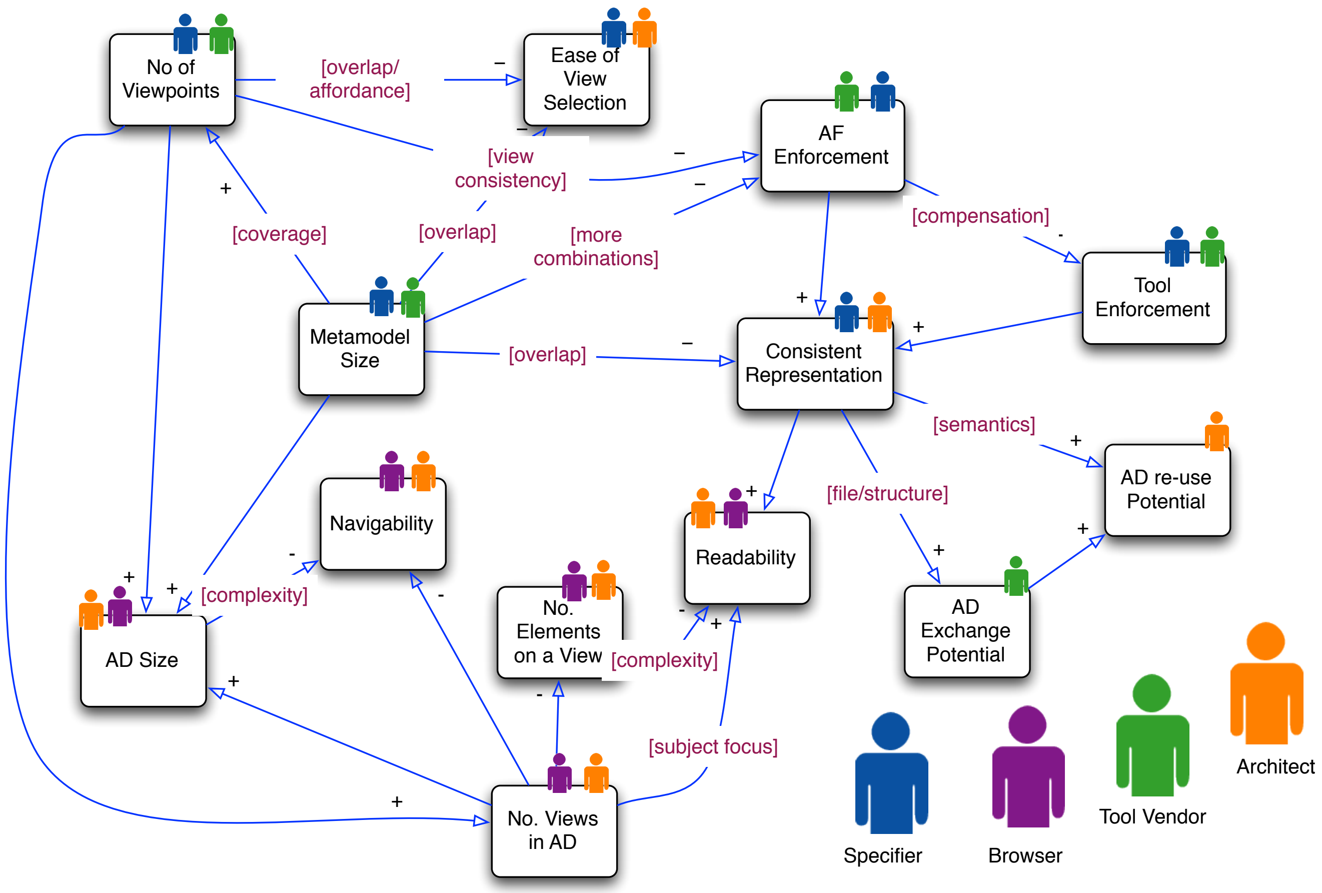


Figure 1 - Interactions With the Architecture Framework and Its Definition

It is a system and we need the parts to work together to be able to construct and successfully exchange architecture descriptions (models). As a system the parts do affect each other. Figure 1 a causal loop diagram - shows impacts surrounding the definition of an architecture framework. The +/- signs indicate whether the quantity or effect in a block increases or decreases as a result of the preceding block. For example,

- as the number of architecture viewpoints (each a specification for an architecture view in ISO 42010 terms) increases the ease of view selection decreases due to increased subject overlap (the choice is greater). As the number of viewpoints increases enforcement for view consistency becomes harder.
- The more enforcement there is by the framework definition the less compensation is needed in
- modelling tools (as has happened through the implementation of the UPDM for DODAF, MODAF and NAF where the lack of explicit rules in the frameworks and the consistency problems that resulted motivated the tool vendors to collaborate to apply rules through a joint UML profile, the UPDM).
- The larger the metamodel size the harder it is to get consistent representation by the architects because because concepts in the metamodel overlap or aren't mutually exclusive and different architects will choose different types of element to represent the same real world thing. A larger metamodel will also generally need more viewpoints to provide coverage of the metamodel and potentially therefore there will be more view types and views in an architecture description. If either the number of views increases or the number of elements on a view increases then this impacts on the readability of the architecture description.
- Consistent representation is aided by better architecture framework and tool enforcement and by a smaller metamodel. Having a more consistent representation aids the ability to exchange a description and therefore its potential for re-use. 'Potential' is the important word as it also depends on the architect and central glueware to maintain consistent meanings of model elements (semantics) as well as decent file exchange mechanisms to preserve this sense.

It is central to the design of TRAK that

- tools, definition, people and central glueware interact and have to be considered both within and outside our system of interest

- architecture description for TRAK is primarily a communication method to facilitate different groups or speciality groups discussing, defining, describing real world things and therefore to be fit for purpose it has to be easy to use and easy to understand.

Framework definition cannot therefore be undertaken in isolation. The overriding principle is that if we look after users the rest will fall into place.

Simplicity

Architecture views provide a visual means of communicating ideas and concerns. If we want to maximise their usefulness and value then the language of communication needs to be a simple and straightforward as possible. The architecture views ought to be able to be used and shared by many disciplines, not just technical ones and therefore to make them easy to read the presentation and constructs need to be simple.

An increasing metamodel size and architecture viewpoint set size increase complexity (decrease simplicity) through providing choice (and with choice comes the the potential for inconsistency) and increase the cost of creating and maintaining an architecture description. A smaller framework is simpler and has to have fewer consistency rules. We should also not forget the cost/ ease of learning to use the framework which becomes harder as the framework size increases.

TRAK has 22 architecture viewpoints, each a specification for a view type. The TRAK metamodel easily fits onto a sheet of paper and uses arrows and text labels and not any technical notation - it is aimed at users and designed to be read as a series of sentences. In comparison the metamodels for the MODAF and NATO Architecture Framework are abstract UML profiles for tool vendors to produce tools. The humanly-understandable TRAK metamodel is *the* TRAK metamodel - it is the master version and anything else is an implementation and possibly therefore an approximation i.e. people first, tools second.

The language / terminology used is another potential source of complexity. Existing frameworks use a variety of terms to mean the same thing and these have changed over time. For example the DODAF 2 now uses 'model' where the DODAF 1.5 it used 'view' . In the DNDAF and the NAF an ISO 'architecture view' is a 'subview' and a DNDAF/NAF 'view' (a collection of related views) is equivalent to DODAF 2 / MODAF 1.2 'viewpoint' which is not equivalent to an ISO 42010 'architecture viewpoint'. TRAK uses the ISO 42010 terminology and meaning for 'architecture description', 'architecture viewpoint' and 'architecture view'.

Consistency, Visibility and Affordance

Consistency in this sense is used in the human factors sense (not the consistent architecture description/modelling sense) and includes:

- organisation of architecture viewpoints into a structure based on their metamodel content
- naming of architecture viewpoints
- presentation

We also need to make sure that the user can more easily see (visibility) and understand what is likely having picked a viewpoint or metamodel stereotype (affordance). In a sense then (HF) consistency, visibility and affordance contribute to simplicity.

In TRAK:

- viewpoint names are deliberately short and keyed into the TRAK perspective
- viewpoint names are mostly centred on a metamodel stereotype name, not a verb or adjective e.g. SVp-04 Solution Function
- mapping viewpoints always map upwards (as you'd expect in a requirement hierarchy) and include both stereotype names and 'mapping'
- we aim for clear separation of content/purpose of each viewpoint so that you cannot use one as a substitute for another e.g. definition of structure is kept separate from definition of behaviour or interchanges
- all '..-01' views are structural - EVp-01 Enterprise Goal, CVp-01 Concept Need, PrVp-01 Procurement Structure, SVp-01 Solution Structure, MVp-01 Architecture Description Dictionary all focus on structure.

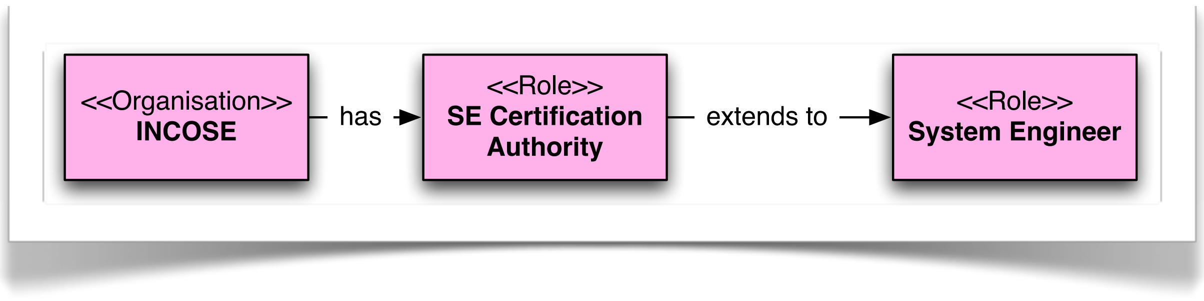
Colour is used in TRAK to identify the TRAK Perspective i.e. whether something belongs to the Enterprise, Concept, Procurement, Solution or Management Perspective. The only exception is for human resources which are pink to distinguish them from 'machines'. Each colour is specified in a platform-neutral way for portability. This provides a number of benefits:

- it stops the multitude of colour schemes that individuals might use where the definition is unique to them - with a learning curve and potential for misunderstanding (if something is needed to denote ownership etc then relationships should be used not colours as relationships will travel and can be queried systematically). There are many more combinations of standardised relationships using TRAK than are colour or shading schemes - this is what provides flexibility.

- it provides a simple top-level error-checking method - if an element of the wrong colour is present it is obvious

TRAK allows graphics to be used to construct views that look like pictures, but if not dressed in this way the base colour must be visible.

Any architecture framework metamodel provides the technical language (verbs and nouns) to describe architecture so why not make it accessible to all? This is exactly what we've done in TRAK through the simplified metamodel where all relationships are designed to read like a sentence - for all readers and not just architects.



We've found that all sorts of people need to be able to read and understand architecture views so it helps if the language is explicit and simple. To this end any implementation of TRAK must keep the text labels on relationships.

Although we don't expand much on this, there is the problem of navigability of the architecture description produced by the architect. Whilst this is dominated by how the architect organises views (and his/her thoughts!) and therefore whether this matches the reader's internal model, the framework definition does play a part. TRAK deliberately provides clear separation of view content but also provides for joining-up content in the optional tuples for any one viewpoint as these provide the context and rationale (often the "why") and the overlap between views that is helpful for understanding how the views fit together and therefore navigation between them.

Adequacy

In keeping with the need for simplicity and ease of understanding a framework needs to be good enough for the task i.e. fit for purpose. TRAK takes a slightly different emphasis from established architecture frameworks in that the primary aim is to be able to produce views that address the architecture task sponsor's concerns in a way that is straightforwards and understandable - a basis for discussion. Whilst is has constructs that allow the entire enterprise to be described and for ADs to be exchanged, it does not set this as the intent. It just has to be good enough to articulate the problem and solution.

It supports other tools such as planning and requirement management tools by providing just enough touching points to keep the description consistent with these other forms of description. Architecture description isn't always the best or most appropriate technique to use. Some things will always be difficult to represent, for example, visualisation of accurate geographic position and therefore the user needs to be able to choose the most appropriate tool for the task and to suit them.

TRAK therefore is rich in detail for solutions and it allows human resources (organisations, roles, jobs) to be part of systems and for their behaviour and interactions to be described. It is deliberately skimpy in procurement where we only worry about when systems are introduced or removed from service as it is assumed that specialist planning tools rather than modelling tools are used to manage programmes. The richness comes from the combination of stereotypes and relationships and as a generic framework is can be used to represent both the product and the business that produces the product.

The viewpoints address typical systems engineers' needs:

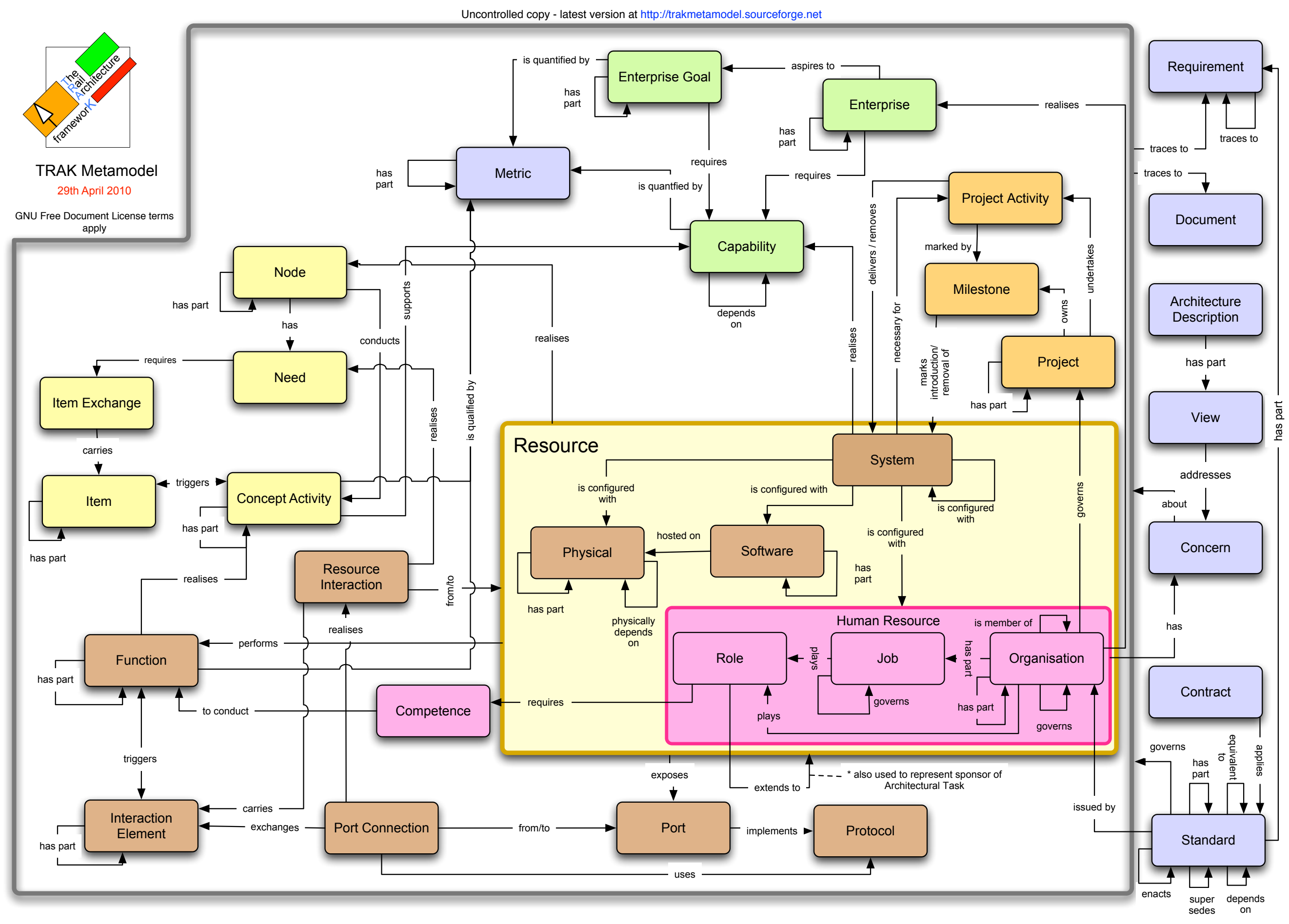
- define structure
- define function
- define boundaries of responsibility
- define requirements
- define interface exchanges
- map interface exchanges to function (to justify the need for the exchanges)
- define the standards and the precedence between them that are applied through contract to a system

There is a minimal process based on ISO 42010 that simply says find out what the sponsor's concerns are and use these to select the viewpoints/views required. Beyond this there are only requirements to keep the AD consistent and to make it easy for the reader. TRAK also provides a view to allow the task itself to be described and tied to the AD and the analysis and findings.

The final check mechanism is ensuring that the TRAK definition addresses user's needs (not the specifier's pet hang-ups by including them in the management loop and making it explicit and responsive (see 'Designing for Whole Life).

The TRAK Metamodel & Viewpoints

The core of the framework is the metamodel and the viewpoint set. The following is an uncontrolled copy taken from Sourceforge.



There is a different emphasis behind TRAK than other frameworks. In the DODAF, for example, the emphasis is on using views to capture data to populate an underlying data model. Similarly there is a different motivation behind the defining metamodels within each framework. In the DODAF, MODAF etc the defining metamodels are abstract UML profiles intended to allow tool vendors to implement the frameworks i.e. they are aimed at tool vendors and are therefore necessarily more complicated. In the MODAF user guides for each viewpoint there are often fragments of 'simplified metamodel' to help the user.. These fragments are an approximation of the defining metamodel - in the same way that we sometimes use 'pseudo-code'.

In TRAK we turn this on it's head. The defining metamodel IS the one the user works with. There is no complicated one and therefore no need for 'simplified'. It is a logical definition which is free of any implementation in a tool. This makes it much, much smaller. We also try to present it in a way we hope people can just read without specialist knowledge. The colours tie the element to the TRAK perspective - the only exception are the human resource elements which are deliberately pink to show that these are tied to humans rather than machines (the man machine interface is therefore pink - brown). The management (blue) elements often can be applied to any element inside the border, for example, Requirement, Concern, Standard or Document.

'System' is deliberately central and you can construct your system from a combination of Physical/Software/System/Organisation/Job/Role. Organisations and jobs have roles, for example System Design Authority, requiring competence to do something and roles have a responsibility extent that can apply to your system or parts of it.

The TRAK viewpoints are a similarly small set.

Viewpoint	Title	Questions / Concerns
EVp-01	Enterprise Goal	What is the Enterprise and what goals does it set out to achieve?
EVp-02	Capability Hierarchy	What are the enduring capabilities the enterprise requires and how is capability measured?
EVp-03	Capability Phasing	How is capability delivered over time? Are there any gaps?
CVp-01	Concept Need	Have the concept needs been identified?
CVp-02	Concept	Has the concept purpose been identified? How is it seen as being used?
CVp-03	Concept Item Exchange	Have the items exchanged by concept nodes been identified? What is required to satisfy the concept needs?
CVp-04	Concept Activity to Capability Mapping	How/are concept activities sufficient to deliver capability?
CVp-05	Concept Activity	What does each concept node need to do?
CVp-06	Concept Sequence	How are concept activities ordered? Is it important?
PrVp-01	Procurement Structure	What is the project structure? How is it governed?
PrVp-02	Procurement Timeline	What other projects is this dependent on? How does their delivery time affect us?
PrVp-03	Procurement Responsibility	What responsibilities do organisations or jobs have in relation to a project or time? Are their boundaries clear?
SVp-01	Solution Structure	What does the solution consist of? Is it structured sensibly? What is the organisation structure / membership? How does responsibility (scope/jurisdiction) apply to the solution components?
SVp-02	Solution Resource Interaction	How are resources connected together? How are the organisations, jobs & roles connected? Have the interactions/ interfaces/exchanges been characterised?
SVp-03	Solution Resource Interaction to Function Mapping	Are there interactions/interfaces that cannot be justified by functional need? Do we have functions that cannot be realised because there isn't an interchange?
SVp-04	Solution Function	Have all solution functions been identified? What does each part do?
SVp-05	Solution Function to Concept Activity Mapping	Do the solution functions meet all of the concept activities? Is there unwanted solution functionality?
SVp-06	Solution Competence	Does the organisation or job through its role have the necessary competence to conduct the function? Is the competence consistent with the solution?
SVp-07	Solution Sequence	In what order do things need to happen?
MVp-01	Architecture Description Dictionary	Is the architecture portable? Can it be understood in the way it was intended to be?
MVp-02	Architecture Description Design Record	Do we understand the scope of the architectural task? What are the issues and findings that resulted?
MVp-03	Requirements & Standards	Have all the constraints been identified? What constraints/ requirements through normative documents/standards apply (or will apply) to the system, project, enterprise?

The TRAK viewpoint set is smaller than existing DODAF-based frameworks (DNDAF 1.6 has 32 views, DODAF 2.0 has 52, MODAF 1.2.004 has 47, NAF 3.1 has 49).

Each TRAK viewpoint is a specification for a TRAK architecture view in accordance with ISO 42010 and includes the following sections:

- version and date
- description
- concerns addressed - used to select the views needed for the task
- mandatory tuples
- optional tuples
- presentation methods
- views needed in order to construct - for consistency and ensure visibility of elements in the right view types e.g. declare structure in SVp-01 before it can be used elsewhere
- consistency rules
- comments

Designing for Whole-Life

How do we maximise maintainability, improve 'mid-life' update capability and responsiveness of framework to users' needs?

We've addressed this in terms of the way in which

- TRAK is defined
- TRAK is published and managed

The definition of TRAK through the viewpoints and the metamodel is a logical one - it includes no implementation or solution. It is purely in (we hope) humanly-understandable terms. Any implementation through a UML profile or anything else is linked but quite separate. In essence we've abstracted or separated "the what" from "the how" which seemed to be good systems engineering practice. This means that TRAK is not tied to implementation in any tool or notation and not hard-wired to any other architecture framework such as DODAF - we have flexibility or "wiggle-room". It also means that any errors or constraints in implementation are not passed upwards into the framework definition.

To improve the quality of the definition and usefulness of TRAK we need to be open and provides ways in which people can interact or engage with it. This is important since the centre of gravity / activity should be with users not specifiers since this is where the expertise is. It also needs to be pragmatic driven by need not pushed from above. For all these reasons 'the TRAK community' needs to be as close to the definition as is possible and to have the means to change it. This one of the reasons TRAK was published as open source (the open source ethic also fits with the public service ethic of London Underground and we didn't want to tax communication).

We also needed a sensible and affordable way of managing the definition and the evolution over time. It was for this reason we've based the management on that used by the Internet engineering Task Force (IETF) where there is a small steering group to set the overall direction, control formal release and co-ordinate but where the proposals, the form of the changes are made by working groups which form themselves according to need and dissolve on delivery.

Having released it as open source through Sourceforge we have now have a platform where 'the TRAK community' can interact through forums, wiki, and notify of bugs or request features in a systematic and managed way. As importantly the whole ethos is to keep the thinking in the open. If we do this it means that other can understand better the intent (or correct it and make it better!). In a sense the design record for TRAK is kept in the open in full public view.

Power to the People!

This seems to make sense because:

- everyone has the opportunity to get involved and improve TRAK
- TRAK benefits from the collective scrutiny and experience brought to bear
- it suits a more incremental rather than seismic event development approach
- it keeps the fixed overhead to a minimum
- it provides a climate more suited to innovation (we don't claim expertise)
- it fits better with the expectations and use of social technology
- the cost is one of individual's time and shared which makes it more acceptable in harsh economic times.

We don't claim perfection (see 'Adequacy') but we hope we've got the right mechanisms in place to make user-generated change easier than has traditionally been the case with closed-door organisations and therefore the responsiveness of TRAK to change should be better.

The Application of TRAK in Rail HF Use

Just as human factors and usability have been considered in the development and management of the TRAK enterprise, the application of TRAK to Human Factors problems has also been attempted.

This part of the paper describes:

- some of the historical background that exists between HF and SE in the area of architectural frameworks (AFs), and how TRAK tries to extend this dialogue;
- how the TRAK AF has been used in Human Factors work activities within the rail industry.

In the sections below, the term 'Human Factors' is generally used to refer to the discipline associated with the application of human sciences to engineering problems - rather than the qualities and capabilities of people.

The Relationship between Architectural Frameworks and Human Factors

There have been a number of attempts to move Human Factors and Architecture Frameworks closer together (Handley and Smillie (2008), Bruseberg (2008)). In general, this has mainly been initiated by a number of different trends in the Human Factors and Systems Engineering communities:

Within the HF community, there has been renewed interest over the past decade in a systems viewpoint in both research and practice (Wilson and Corlett (2005)). This trend re-states the 'socio-technical system' as important unit of analysis for Human Factors analysis.

Some authors (e.g. Reason (2008)) have argued for emphasis to be given to the positive contribution that people can and must make to successful outcomes in joint human-technology systems. In this view, the human must not be seen as a system component to be 'designed out' or 'managed', but as an active contributor with skills and capabilities that must be augmented by the technical system components.

The potential benefits that AFs give to the Systems Engineering discipline also apply to the Human Factors technical discipline. AFs should enable HF Specialists to collaborate with colleagues from other engineering disciplines, reduce the duplication of effort that can be involved in constructing system models, and improve efficiencies and effectiveness through the use of common languages and frameworks.

Taken together, these trends mean that the motivation from the Human Factors community for increased integration with Systems Engineering and Systems Thinking is high.

One theme in the attempts to improve integration between HF and AFs is that of the 'Human View'. Within DODAF-derived AFs (e.g., MODAF) there has been effort over recent years to produce 'Human Views' that exist within the AF (e.g., HFI DTC (2009)). This is an architecture view (with an associated collection of viewpoints) are intended to capture people-related or people-focussed concerns for stakeholders associated with the enterprise of interest. For example, the NATO Human View (Handley and Smillie (2010)) contains the following Human Views:

Viewpoint	Title	Description
HV-A	Concept	High-level representation of the human component.
HV-B	Constraints	A store of constraints that affect people within the system.
HV-C	Tasks	A description of human activities.
HV-D	Roles	An inventory of roles.
HV-E	Human Network	Representation of human-human communication links.
HV-F	Training	This captures the relevant training requirements and strategies.
HV-G	Metrics	A store of human-related performance criteria and values.

Viewpoint	Title	Description
HV-H	Human Dynamics	Represents changes to human-related aspects over time.

The main benefit of the 'Human View' approach is that it provides specific viewpoints to collect human-related concerns - therefore it provides a method by which these issues can be included in the EA modelling undertaken on a particular project. This means that the potential benefits of a common modelling approach can be realised.

The Human Views approach has some theoretical underpinnings in 'cognitive systems engineering'. Cognitive systems engineering (Hollnagel and Woods (1983)) is an approach to Human Factors that stresses the socio-technical systems approach and the role of the relationship between people and artefacts in shaping work situations (especially in 'knowledge work' situations). This means that the information contained within DODAF/MODAF style Human Views can be drawn from 'good practice' HF investigative methods from the cognitive systems engineering approach. Through linking Human Views with cognitive systems engineering, the MODAF/NAF Human Views authors have provided a means by which some rigour can be introduced to the development of people-related system models. Cognitive system engineering methods can help to provide a 'grounding' for system models. This helps with improving the quality of information contained within the people-related system models.

The difficulties with the 'Human View' approach are to some extent related to the perceived advantages. For example, while 'Human Views' provide a focus for human-related concerns, it does separate these issues from the broader system-level modelling activities. Given that (following a socio-technical systems approach) we are interested in the system as a useful unit of analysis, this may be a disadvantage both in technical terms and in terms of encouraging other engineering disciplines to become stakeholders in people-related system topics.

Accordingly, the links with cognitive systems engineering may also be seen as introducing theoretical complexity to the EA undertaking. This constraint may not be necessary, and the outputs of cognitive systems engineering methods have sometimes been criticised as being difficult to understand, especially by non-HF practitioners (Lind (2003), Hale and Schmidt (2008)).

As part of a number of objectives, TRAK is intended to provide a means by which an AF can be used to address or capture human-related systems engineering concerns. The objective has been to extend previous efforts at closer integration between HF and SE, as described above. The aim has been to take account of the advantages and disadvantages of the existing 'Human Views' approach, while emphasising the more general aims of the TRAK enterprise towards ease of communication and simplicity in views and metamodel.

As can be seen (Table x, TRAK Viewpoints), TRAK does not have any viewpoints that can be used to produce models that are explicitly described as 'Human Views'. This is deliberate. The intention is that TRAK accounts for people-related concerns through two main strategies:

A metamodel that is fundamentally based on a socio-technical systems philosophy. To this end, human resource elements and interactions are explicitly included at the metamodel level.

Flexibility in the use of different viewpoints with regard to the involvement of resources. TRAK tries to make it possible to use different viewpoints to capture human-related concerns, just as the viewpoints could be used to capture technical concerns. The philosophy here is that the analyst should be free to choose the most relevant combination of system resources (human, technical, or organisational) in order to communicate about the system of interest. It is a broad group of stakeholders that are interested in people-related system issues - not just Human Factors practitioners. TRAK tries to encourage use by this broad user group.

The potential pitfall of this approach is that the advantages of the 'Human Views' approach are not realised. For instance, the TRAK enterprise could find that human-related issues are not given sufficient emphasis because there is not a specific point of focus, as is given to MODAF and NAF through the use of a 'Human View' approach. In addition, it may be that there are a subset of human-related concerns that need dedicated viewpoints, and cannot be captured through the current TRAK viewpoint set.

The potential benefits of this approach are significant. The benefits are mainly associated with the overall simplicity and effectiveness of the TRAK enterprise to its users, and in particular the usability and readability of the metamodel and viewpoint definitions to different stakeholders. By not introducing any unnecessary complexity, it is hoped that the usability of the TRAK AF can be improved, and therefore it's utility as a communication tool can be increased. The embedding of human resource elements within the TRAK metamodel is also seen as advantage when the AF is used for Human Factors purposes. In other words, TRAK should be suitable for a socio-technical systems modelling approach 'from the ground up'.

The following section presents a short example to show how TRAK has been used to support Human Factors work in the rail industry. The example provides some information on how TRAK has been used to support the traditional 'task analysis' phase of a HF project - the development of the Human Machine Interface (HMI) for an Automatic Train Regulation product produced by Invensys Rail Ltd.

Using TRAK in Rail Human Factors

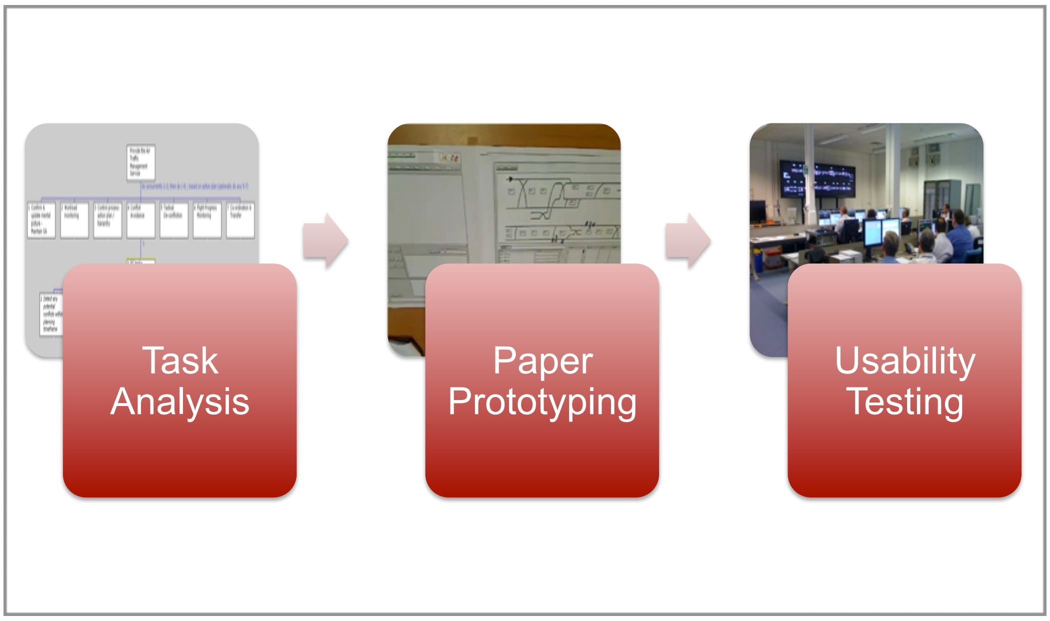
Railways are complex, distributed socio-technical systems (Wilson et al 2007) that comprise many human actors such as train drivers, maintainers, and station staff. Within a modern European Train Control System (ETCS) or Communication-based Train Control (CBTC) system, train driving can be fully or partially automated, often under control from a timetable (via Automatic Routing Setting (ARS)) and signalling interlocking systems that control the movement of trains. Today, in some parts of the world, this is the type of system that is supervised by the traffic controller (sometimes known as a signaller or dispatcher) at a remote control centre.

In a metro or light rail environment, the high throughput of trains means that the effect of unplanned system behaviours (perturbations) can have severely disruptive effects, and consequently affect passenger journey time and satisfaction. The problem is that small deviations from the timetable (such as late departures from stations) tend to propagate through the system and eventually lead to significant service disruptions. With a high throughput railway, even short delays in a train leaving a station can quickly escalate and lead to service degradation and reduced customer service. This is a problem that 'Automatic Train Regulation' (ATR) systems are designed to help solve.

ATR systems monitor and predict the railway service in real-time, checking for actual or predicted service perturbations. If a perturbation is detected or predicted, the ATR system will run a number of 'what if' models, compare them, and come to a conclusion as to the best interventions necessary in order to reduce the impact of the perturbation (Aun and Harris (2005)).

Human Factors is of critical importance to the successful implementation of automation in rail traffic control (Anderssen et al (2009)), and ATR is no different. The way that operators understand, interpret, and interact with ATR must be carefully designed from the outset. If traffic controllers do not accept ATR or lose confidence in it's functioning, it is unlikely to be used, and the desired benefits will not be achieved.

Typically, for this type of project, the Human Factors Engineering lifecycle would follow the steps shown in Figure x:

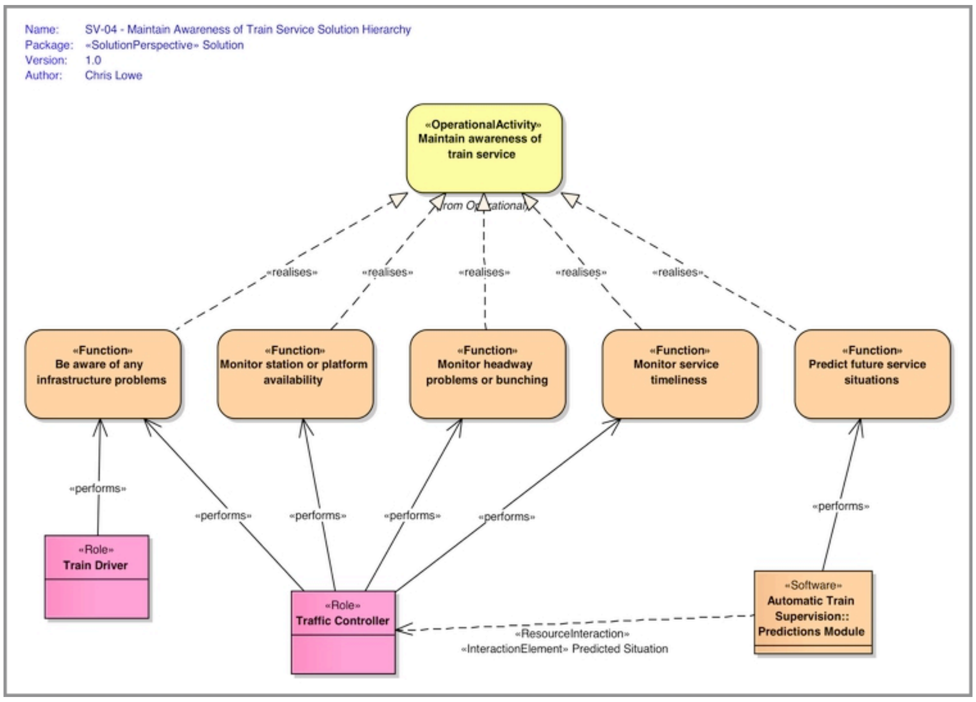


The objective of the series of work packages shown in Figure x would be to understand the underlying work practices (task analysis), work with representative end-users to explore the requirements, constraints, and information needs relevant to the HMI design (paper prototyping), and to then validate the developed HMI through usability testing.

The focus of this case study, and the HF work package within which TRAK was used, is task analysis. Task analysis (Annett et al (1971)) is a means of investigating and representing the tasks and goals associated with human activities. It is a core HF method that is well known, with a number of different varieties, and is presently taught on many undergraduate degrees.

While task analysis is a popular technique, it does have some disadvantages. In particular, as often practiced, the method has difficulty in representing behaviours and communications across teams of people. Also, task analysis (in most of its forms) can only really capture the activities carried out by humans - the behaviour of technological system components in response to human actions is difficult to model.

To address these difficulties within the ATR HMI design project, system modelling using the TRAK AF was used to replace some of the more traditional task analysis activities. A number of different TRAK diagrams were used during this task analysis phase. For example, Capability and Operational Activity models were used in the initial phases in order to link the HMI design to the expected business benefits. The purpose of these diagrams was to show the overall goal of the ATR facility, and the associated operational context and activities. For example, Figure 1 (a SVp-04 view) is intended as a high-level overview to show the relationships between the ATR software and the human system actors, in relation to system functions and operational activities.


Name: SV-04 - Maintain Awareness of Train Service Solution Hierarchy
Package: «SolutionPerspective» Solution
Version: 1.0
Author: Chris Lowe
«OperationalActivity» Maintain awareness of train service
(from Operational)
«realises»
«Function» Be aware of any infrastructure problems
«Function» Monitor station or platform availability
«Function» Monitor headway problems or bunching
«Function» Monitor service timeliness
«Function» Predict future service situations
«performs»
«Role» Train Driver
«Role» Traffic Controller
«Software» Automatic Train Supervision:: Predictions Module
«ResourceInteraction»
«InteractionElement» Predicted Situation

Other work model types were produced (e.g., communication link diagrams (using the SVp-02 view) and role-function matrices (a table form of SVp-04)) using the TRAK framework, as necessary in order to understand the work and support the design of the software.

The work modelling highlighted several factors with the ATR design. For example, it emphasized that the train driver needs to comply with platform departure countdowns at all times, noting that they cannot be sure whether ATR is influencing their train or not. We were able to deal with this factor in the broader system design.

Using TRAK in this way was an improvement over traditional task analysis methods. The system models were able to capture human-human and human-technology interactions in a way that is difficult to do using traditional task analysis. We also found that the TRAK models did provide a 'common language' when discussing the system design with System Engineers or end-user representatives.

The disadvantages of using TRAK to support task analysis are probably common to any use of AFs to support this HF activity. Views produced through AFs tend to struggle to capture cognitive behaviour, such as the plans and strategies that govern decision making. Within the ATR project, we relied on other HF methods (such as paper prototyping and scenario walkthroughs) to provide this 'rich picture' input.

In summary we found it useful to use TRAK as part of HF analysis, especially in the initial phases of exploring allocation of functions across human and technological system elements. Using TRAK / AFs in conjunction with other HF techniques (paper prototyping, scenario walkthroughs) is likely to be a strong combination to support automation system design, and this should hopefully be investigated further in the future. Concluding Remarks

Concluding Remarks

Lessons

Systems Engineering doesn't just apply to products but to the 'business' of developing the products. The principles apply equally to an architecture framework, the definition, the maintenance and the processes that surround it and the disciplines that interact with it or use it as to architectural products of it. Applying common sense systems engineering principles, for example, separating 'what' from 'how', help us structure it, identify interfaces, allocate function and stop it becoming a muddle.

People are involved with both the business of defining a framework and it's use and therefore you ignore the HFI at your peril!

Anticipating how users of an architectural framework behave is essential but hard. This is a product of the definition and the tools and any slack in one often gets taken up in the other. It isn't a static situation as it needs to consider the order in which relationships are created and relationships established. Relationships are there in the TRAK metamodel because we think that the information is important. We have, however, to guard against short cuts being taken which might prevent this information being gathered. We also need to make sure the set of views in an AD is consistent. This isn't easy and things you naturally do or assume are often not explicitly written down. It is therefore an empirical process which is another reason why the users need to have the minimum separation from the ability to change the specification.

Everyone thinks their domain/specialism is uniquely tricky and needs it's own terminology or views and are often unhappy to find it isn't/doesn't. People are often unhappy or uncomfortable if they can't see their part of the world presented in a familiar way. In a way this is like the traditional hand-drawn map-makers where you see the sponsor's estate deliberately emphasised at the expense of accuracy. This affects what is on any single view but also the use of colours and symbols. Time is needed to explain the power of combining relationships with object types and the value in providing multi-faceted windows or viewpoints rather than throwing it all on a single diagram.

Architecture description is potentially a disruptive technology within an organisation. We intend TRAK to be used by many disciplines as this is a "whole system" approach within the business. The trouble is that many see architecture description as the sole prerogative of a small specialist "priesthood" which can't help the overall system design. Some specialists might be needed to co-ordinate and provide the infrastructure and glueware but every discipline ought to be involved in developing the repository.

Just because an idea is good doesn't mean that it will get very far. This is particularly true if it is not traditional. You not only need the force of the argument to convince people but also a sponsor to be taken seriously. TRAK has been very fortunate in having the backing of London Underground Limited and also the UK Department for Transport as a sponsor as part of a push for better systems engineering.

Where We Are In Terms of Human Factors and TRAK?

These are still early days for TRAK. Whilst we are confident in the approach and control mechanisms - the architecture - we might not be 100% correct in the detail. It is clear that human factors considerations need to be applied to the "TRAK Enterprise" and the definition of TRAK within this. Thus far the initial feedback has been encouraging.

Identifying how and what architecture views might be useful for (applications) will take a lot longer. This doesn't mean that we need more views necessarily as this depends on the concerns being addressed and therefore the TRAK metamodel elements needed.

Further developments around the human-centred issues in the TRAK enterprise are likely to be driven from the needs and concerns of the TRAK user community. With the open source license we encourage more people to get involved and try things out. to establish what is and what isn't sensible for human factors using architecture description. At the present time, this involves the modelling of aspects of skill and competence, in the context of training needs and safety requirements. It is a new and evolving area, so who knows what might arise?

In summary, by focussing on the people and the 'user interface', it is hoped that in TRAK we have provided a pragmatic enabler for re-use, communication and collaboration.

References

- Andersson, A.W., Isaksson-Lutteman, G., Kauppi, A., & Sandblad, B. (2009). Automated functions in train traffic control: problem and solutions. Third International Conference on Rail Human Factors. Lille, France, 3-5 March.
- Annett, J., Duncan, K.D., Stammers, R.B. & Gray, M.J. (1971). Task analysis. Department of Employment Training Information Paper 6. HMSO, London.
- Aun, C.S., & Harris, M. (2005). Getting the Right Balance in Delivering an Enhanced Automatic Train Supervision Capability for the SMRT Rail Network, IRSE Technical Convention. Singapore.
- Bruseberg, A. (2008). Human views for MODAF as a bridge between human factors integration and systems engineering. Journal of Cognitive Engineering and Decision Making, Vol 2 No 3, 220-248.
- Handley, H.A.H., & Smillie, R.J. (2008). Architecture framework human view: the NATO approach. Systems Engineering, Vol 11 No 2, 156-164.
- Handley, H.A.H., & Smillie, R.J. (2010). Human view dynamics - the NATO approach. Systems Engineering, Vol 13 No 1, 72-79.
- Hollnagel, E., & Woods, D.D. (1983). Cognitive systems engineering: New wine in new bottles. International Journal of Man-Machine Studies, 18, 583-600.
- Human Factors Integration Defence Technology Centre. (2009). The Human View Handbook for MODAF. Draft version 2, second issue.http://www.hfidtc.com/MoDAF/HV-handbook-issue2.pdf
- Hale, C.R., & Schmidt, V. (2008). Four Challenges, and a Proposed Solution, for Cognitive System Engineering – System Development Integration. Industrial Engineering Research Conference. Vancouver, BC.
- IEEE Std 1471–2007, Recommended Practice for Architectural Description of Software-intensive Systems. http://www.iso-architecture.org/ieee-1471/
- Lind, M. (2003). Making sense of the abstraction hierarchy in the power plant domain. Cognition, Technology & Work, Vol 5 No 2, 67-81.
- MODAF http://www.mod.uk/DefenceInternet/AboutDefence/WhatWeDo/InformationManagement/MODAF/
- NATO Architecture Framework. http://nhqc3s.nato.int/HomePage.asp
- Reason, J.T. (2008). The Human Contribution: Unsafe Acts, Accidents, and Heroic Recoveries. Ashgate, Farnham.
- TRAK Enterprise Architecture Viewpoints. http://trakviewpoints.sourceforge.net
- TRAK Enterprise Architecture Metamodel. http://trakmetamodel.sourceforge.net
- The TAO of the IETF. http://www.ietf.org/tao.html
- TRAK Community Site http://trak-community.org
- Wilson, J.R., & Corlett, N. (2005). Evaluation of Human Work, 3rd Edition. CRC Press, FL.
- Wilson, J.R., Farrington-Darby, T., Cox, G., Bye, R., & Hockey, G.R.J. (2007). The railway as a socio-technical system: Human factors at the heart of successful rail engineering. In Proc. IMechE, vol. 221, part F. Journal of Rail and Rapid Transit, 101-115.