SEMANTIC MODERNISATION: LAYERING, HARVESTING AND INTEROPERABILITY

Chris Partridge[1], Michael Lambert[2], Michael Loneragan[2], Andrew Mitchell[1]

[1]BORO Solutions Limited
25 Hart Street, Henley on Thames,
Oxfordshire, RG9 2AR,
United Kingdom

[2]QinetiQ,
Portsdown Technology Park, Southwick Road,
Cosham, Portsmouth,
Hants, PO6 3RU,
United Kingdom

partridgec@borogroup.co.uk,
mjlambert@QinetiQ.com,
mjloneragan@QinetiQ.com,
mitchella@borogroup.co.uk

ABSTRACT

There is a well understood requirement for semantic interoperability within NATO and an emerging strategy to address it. One of the strategy's key components – the 'semantic description' – requires further clarification. What is less well recognised is that this 'semantic description' can also be viewed as a component of a wider strategic requirement for semantic modernisation. This paper describes how the semantic modernisation techniques of layering and harvesting provide a strong foundation for the production of semantic descriptions. It describes two projects that illustrate how these techniques are being used to do this. Finally, it reflects upon how this could help to refine the current NATO NEC (NNEC) semantic interoperability strategy.

1.0 INTRODUCTION

It is well understood that the semantic interoperability requirement identified for Network Enabled Capability (NEC) by NATO and its member nations is an example of the general issue of lack of semantic interoperability currently affecting information systems: "This is entirely aligned with the lately recognized fact that semantic understanding and interoperability is a key challenge for organizations and their systems to successfully and competitively provide their services."[1]
NATO has been developing a strategy to provide semantic interoperability: "The proposed solution to this challenge … is utilising formal representations of semantics (meaning) through ontologies in order to exchange not only the information, but also its meaning and intent."[2] A key element in the strategy – the semantic description – has been identified, but the details of what this is and how it will be produced requires further work: "… semantic descriptions of systems can be obtained or created in some way."[3]

[1] Section 2.2 - Ontology Utilization – in [1]

[2] Section 1 - Executive Summary – in [1]

[3] Section 3.4.1 - Assumptions and Preconditions – in [1]

What is not so well recognised is that this key requirement for a semantic description can also be viewed as a component of a wider strategic requirement for application and semantic modernisation. And that the techniques of semantic modernisation – including layering and harvesting – provide a mechanism for the production of semantic descriptions. Recognising this will provide NATO with an opportunity to refine its strategy for semantic interoperability and aligning it with nations' emerging semantic modernisation strategies.

This paper aims to provide a picture of the strategic global requirement for semantic modernisation, and describe how this is based upon semantic layering and can involve semantic harvesting. It explains how this relates to semantic interoperability, in particular how layering and harvesting provide a foundation for the production of a system's 'semantic description'. It illustrates this with examples from a couple of current projects. Finally it reflects upon how this could help to refine the current NATO NEC (NNEC) semantic interoperability strategy.

2.0 BACKGROUND

NATO Semantic Interoperability Logical Framework (SILF) Background

NATO adopts a reasonably standard notion of semantic interoperability – "the ability of two or more computerized systems to exchange information for a specific task and have the meaning of that information accurately and automatically interpreted by the receiving system, in light of the task to be performed." It has developed the SILF as part of its strategy to enable semantic interoperability. The SILF architecture clearly identifies a 'Semantic Description' and distinguishes it from the 'Existing System' – which is described as:

> "… we describe them briefly nonetheless since they are central to any application of SILF. A semantic description specifies, among other things, the permitted syntactic message structure as well as the intended message meaning (via references to ontologies). However, we note that SILF will derive the intended meaning not only using the semantic descriptions, but also using common concepts in the CG [Common Ground]. In addition, so called "Terms of interest" are used by SILF when deriving the intended meaning."[4]

The SILF architecture for a single message is shown in Figure 1.

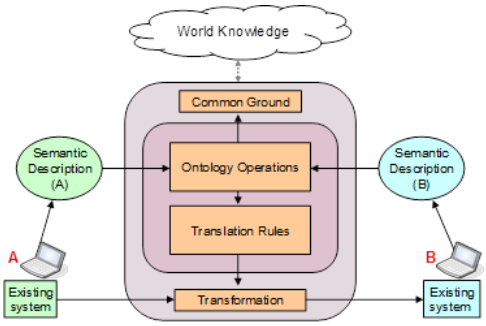

Figure 1: An overall view of SILF

The overall SILF architecture, where multiple systems exchange multiple messages, looks more like Figure 2. Both figures show the structure at a point in time, they do not show systems joining and leaving the SILF over time.

[4] Section 3.4.2 – Brief description of SILF and its main components – in [1]

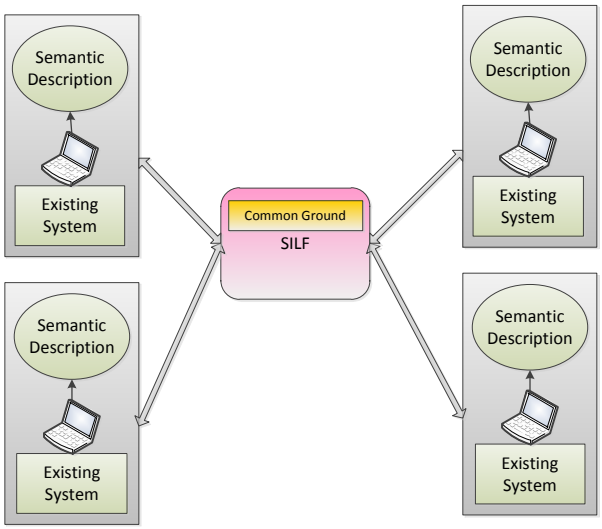

Figure 2: Overall SILF Architecture

As both figures make clear, the semantic descriptions are a key component of the SILF as they, along with the Common Ground (which provides "common references for the semantic descriptions supplied by independent systems"), are the basis for ensuring semantic interoperability. However, there are typically no descriptions for the 'existing systems' that need to interoperate and so they will need to be created. NATO's strategy is "[The basic idea of SILF is] to insist on having a semantic description of all of the information to be exchanged." And a key assumption (and precondition) for the SILF architecture is:

> "… that semantic descriptions of systems can be obtained or created in some way. These descriptions can more or less automatically be (partly) derived from the systems, but in order to achieve the necessary quality of the descriptions the process normally requires human intervention."[5]

What is in need of clarification or development is a more detailed explanation of what the semantic descriptions are and how they are 'obtained'. This becomes critical if the assumption that these semantic descriptions can be automatically (or even partly automatically) derived from the systems turns out to be optimistic. Unless there is a feasible solution, then the strategy is not implementable. What this paper aims to do is provide one feasible solution and, in the process, identify some of the challenges any such solution will face.

Application Modernisation Background

Many of the domains that can be automated, have been automated, usually more than once. This is reflected in the nature of current work; more developments are brownfield than greenfield, more time is spent on maintenance and less on development. Software is also aging. Unlike 25 years ago, most large organisations, in defence and elsewhere, now have in their software inventories applications that are 15, 25 years old or older; one consequence of this is that more and more programmers work on systems that are older than them. These are all signs of the maturing of the industry.

As noted above, the SILF architecture takes a view at a point in time. However, the systems it aims to integrate will change over time. Once the systems are built, they do not last forever; there eventually comes a time that they need renewing, or retiring. There are various drivers for this. Internally, there is tendency for systems in general to degenerate. Classic examples are scientific theories, where the renewal is called a paradigm shift (Kuhn [5]); applications systems are systems and suffer the same kind of degeneration. Externally, the environment changes; new technologies emerge and business drivers change. Eventually, the system needs to be retired or renewed. There are various forms this renewal can take;

[5] Section 3.4.1 – Assumptions and Preconditions – in [1]

including merging and separation of applications (as shown in Figure 3). These renewals are, from a semantic modernisation perspective, key intervention points.

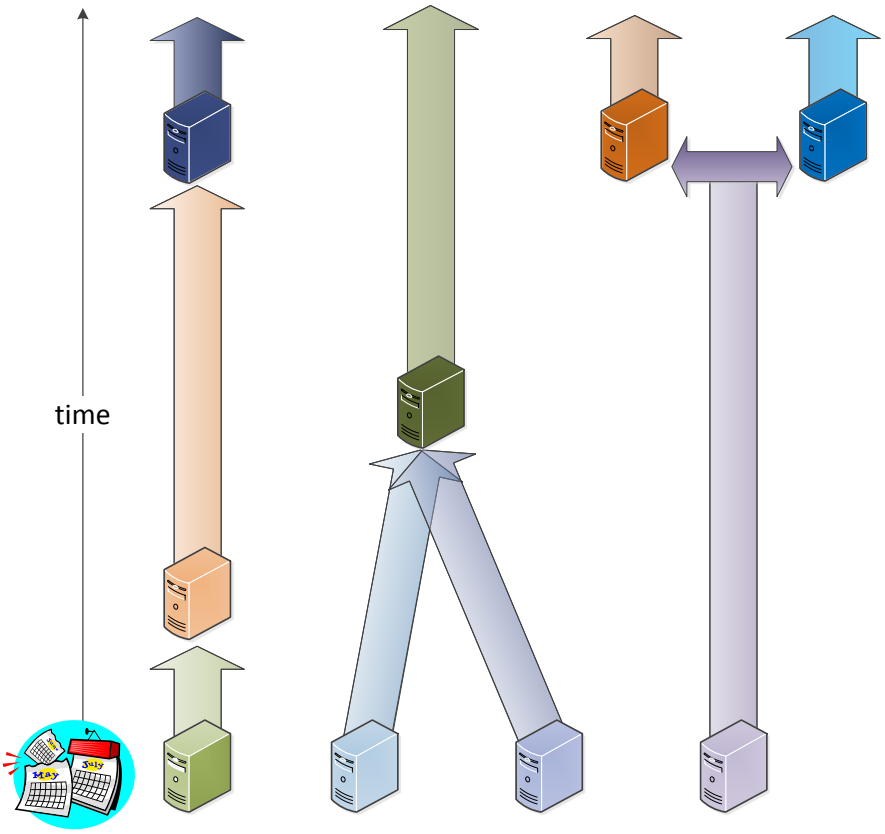

Figure 3: Patterns of Renewal

Managers are learning how they need to adjust their priorities to reflect this maturity. One area of current concern is the risk and cost associated with renewing applications. A belief is growing that these can be mitigated by shifting from a "From Scratch Development" mind set (more suitable for a greenfield situation) towards a "Phased Component Reuse" mind set, which, in this context, is often described as 'Application Modernisation'. What this involves is:

- Replacing a "throwaway" attitude with a "reuse" attitude, and
- Shifting from an "all or nothing / go for broke" approach to a phased, componentised deployment approach.

The aim is to enable lower risks, higher returns, improved quality and faster delivery.

3.0 MODERNISATION, RE-USE AND LAYERS

One important key to a successful modernisation is identifying what can and should be migrated from the old system to the new system; what historic investment can and should be harvested from the old system. It is worth noting that one harvests the whole life investment, including both development and operational investment. Operational use is often seen as an irrecoverable sunk cost not an investment; however it provides by far the best test of a system and when this trustworthiness can be harvested in a suitable modernisation project, it becomes a long term investment.

Application Modernisation Maturity

In the spirit of re-use and phased delivery, application modernisation projects typically face two

challenges:

- clearly separating the areas that require transformation from those that can be re-used, and
- developing a process for modernisation of the separated areas, typically one that reduces cost by maximising its automation.

Where application modernisation techniques are firmly established, these challenges tend to have been met. Examples are the modernisation of hardware and operating systems. These layers tend to be clearly separated both from each other and from other layers. Suppliers into these explicit layers typically ensure that applications that use their product can be ported (modernised) to later versions in the same family. For example, migrating from one Windows version to another is reasonably well supported.

In other areas, modernisation techniques are less firmly established. An example is programming languages. These have been clearly separated but while it is feasible to translate (transform) between languages, the results are not always effective. For example, the automated translation of procedural COBOL to object-oriented JAVA typically results in what is described as JOBOL – a procedural Cobol-looking Java system. If an object-oriented language programming style is required, then, with current technologies, the code needs to be manually re-engineered. It is an open question whether there are algorithms that can adequately automate this process.

Application Development Stage Layers

One area where application modernisation strategies are immature is above the programming language (or database) layer and there are opportunities for improvement. The challenge of working out where the application can be profitably separated into re-use and transformation layers as well as the processes for modernising these layers is being addressed.

An important stratification into layers, one familiar to application developers, is based upon separation of these three concerns:

1. Domain
2. Computation
3. Technology

Here the latter concerns build upon, and so are dependent upon, the prior concerns (as shown in Figure 4).

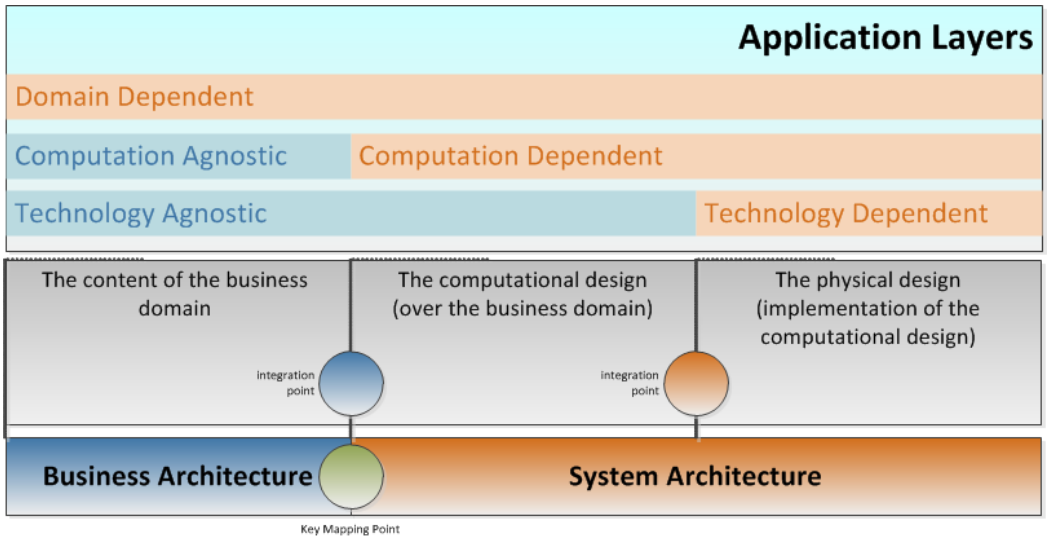

Figure 4: Application Layers – Dependencies

This stratification is often used to separate development into stages – as shown in Figure 5. A well-known example is the Object Management Group's Model Driven Architecture (MDA), where the layers are called the Computationally Independent Model (CIM), the Platform Independent Model (PIM) and the

Platform Specific Model (PSM).

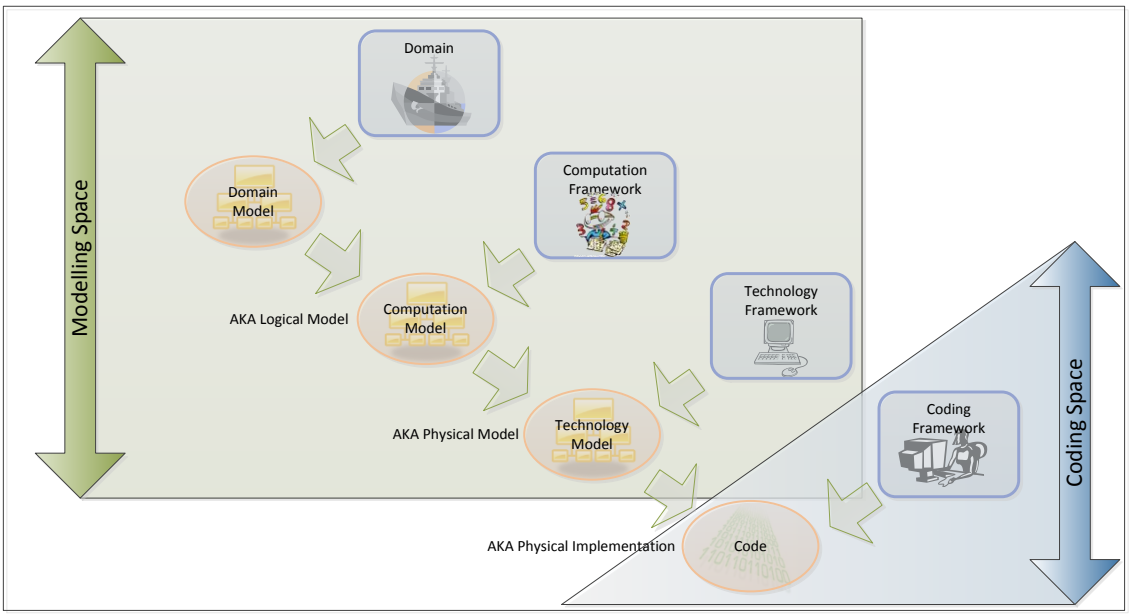

Figure 5: Development Stages

Indeed MDA can be seen to include a kind of application modernisation strategy. The intention is for the models to be independently re-used. For example, a computation model could be implemented using one technology model and then at a later date re-used for an implementation using a different, more modern technology model – as shown graphically in Figure 6.

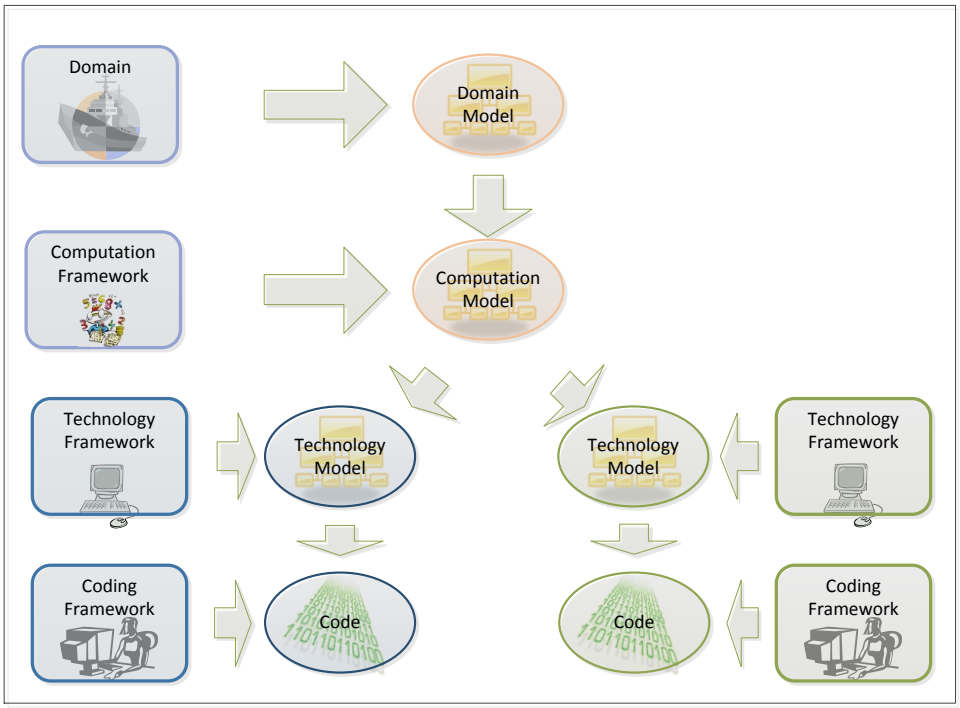

Figure 6: Re-used Models

The Semantic Layer

If one looks at the three layers in terms of whether the subject of the models is the domain or the system, then the three layers collapse into two; a semantic and a system layer[6] – as shown in Figure 7.

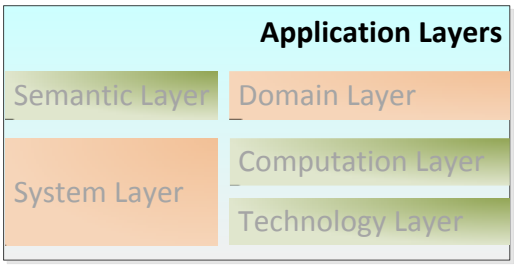

Figure 7: Application Layers – Collapsing the Three Layer View Into Two Layers

This highlights a key feature. The framework provides both a clear foundation for understanding what the semantics is and how to manage it separately from other concerns. To understand this, we need to clarify the meaning of semantics.

While linguists often use semantics in the broad-brush sense of 'meaning', philosophers often prefer the more exact sense of 'the relationship between words and objects'[7]. In the case of application systems, this becomes a relationship between the system and its domain: and in the case of domain models, a relationship between the model and the domain. This makes it clear what an explanation of meaning (in this context) needs to be: it needs to provide an explanation of what the relation is between the domain model and the domain – with a simple reference relation being a favourite candidate; though this may also require a commitment to some kind of top ontology, to make clear what is being referred to. This in turn gives a much clearer explanation of what a 'semantic description' is – it is a model of the domain. And, if one is working within this framework, it is built as the first stage of application development.

The ideas behind this approach are not new within the application development community. One can see their roots as far back as George Mealy's 1967 'Another look at data' [7] and Bill Kent's 1978 'Data and Reality' [4] – and then their subsequent incorporation into standards (for example, Griethuysen [3]). However, to date the implementation of the approach has been more successful at the two system layers than at the semantic layer. The demands of semantic modernisation and interoperability are creating a pressure to change this.

Opportunities for Semantic Modernisation

The opportunities for cost savings and quality improvement come from re-use; the opportunities for improving business performance and agility come from transformation. Semantic modernisation offers both kinds of opportunities.

Many domains are reasonably stable, and so the domain models are not volatile. This makes them prime candidates for re-use. Furthermore, once the domain models are established, they are quality assured and improved through operational use and this investment can be re-used in subsequent modernisation projects. In a "From Scratch Development" approach, the models need to be re-built during each new

[6] NATO [1] talks about a similar divide - Section 2.3 - Levels of Interoperability "Everything below a certain level, *the semantic level,* is about physical connectivity, data exchange, message exchange, common protocols, etc. At the semantic level, the models introduce common reference models based on common ontologies, i.e., the meaning of the exchanged data is unambiguously described."

[7] As Nelson Goodman put it in the Introduction to Quine's lectures published as Roots of Reference – "… an important relation of words to objects – or better – of words to other objects, some of which are not words – or even better, of objects some of which are words to objects some of which are not words."

development and this opportunity for re-use is missed. Hence, over time semantic modernisation can provide significant cost savings and quality improvement.
Where there are changes in the domain, the existence of a semantic layer enables the transformations to be addressed at the right level.

4.0 SEMANTIC HARVESTING

When planning a renewal of an existing system, one hurdle facing most such projects is that the existing system does not have an explicit semantic layer. To enable semantic modernisation downstream, the layer needs to be constructed. As noted earlier, adopting a "From Scratch Development" approach to building this layer is risky and expensive, and also misses the opportunity to salvage investment in the semantics of the existing system. A more appropriate approach is to harvest the implicit semantics from the existing system and re-use this; ontologically based techniques for doing this are emerging [8, 9].

Taking advantage of semantic harvesting

Semantic harvesting also enables non-invasive semantic modernisation approaches in systems with no explicit semantic layer. One example is Enterprise Application Integration (EAI), an architectural approach to integrating systems. Where systems have no explicit semantic layer there is a strong case for mapping between applications at the technology data level – typically the data structures of the implemented interface. However, semantic harvesting exposes the semantics of the existing system making it available for re-use. This allows the integration to be managed at the semantic level. NATO's SILF architecture is an EAI architecture which aims to work at the semantic level. Semantic harvesting can provide the semantic input it requires.

Transformation through the layers

One of the reasons for working at the semantic layer is that this gives a clean picture of the semantics. As the development moves from one layer to the next, additional concerns are taken into account and the structure of the model is transformed. So the structure of the computation and technology layers is not a good reflection of the semantics, and when working at these layers one is often only working indirectly with the underlying semantics.
Much work on MDA focuses on the management of this transformation (see, for example [2]). One often under-appreciated task the semantic harvesting faces is the unwinding of these transformations. Here the ontological concerns play an important role as they help to reveal the underlying implicit semantics.
Consider the simple example in Figure 8.

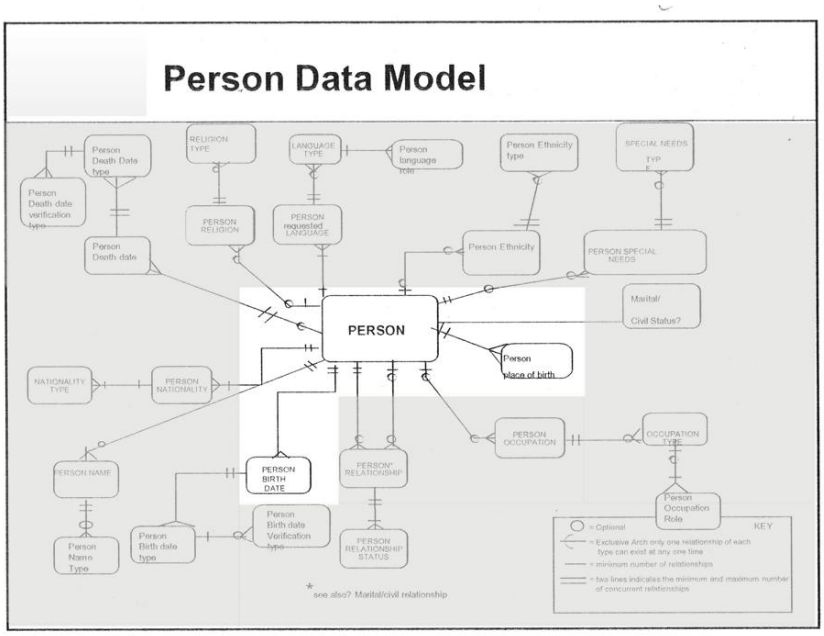

Figure 8: An example 'Person Data Model'

This shows that the 'person' data entity has relationship to a 'person place of birth' data entity and a 'person birth date' data entity. Analysis of the domain quickly reveals that these two entities are implicitly linked by the person's birth event. The birth event occurs to the person at that place at that time – this is shown graphically in Figure 9. This structure explains why the birth place needs to exist on the birth date – so, for example, a birth date before 2006 (but after 1922) and a birth place of the country Montenegro would be inconsistent – as Montenegro did not exist at that time. Algorithmically one could construct a rule that mimics this constraint, but the explicit introduction of a birth event has an explanatory power that the rule does not.

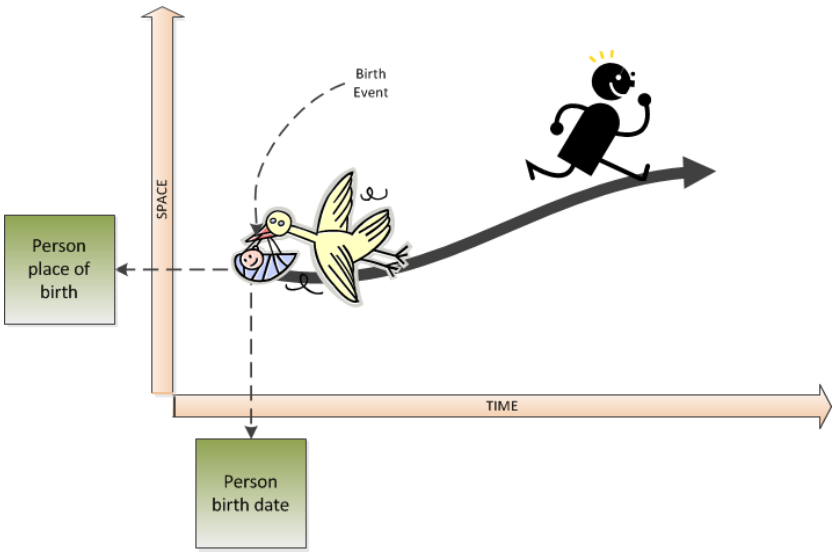

Figure 9: Person Space-Time Map – showing birth event

From this, one can construct a model that shows the birth event and its straightforward connection to the place and time it occurred – the result is shown in the top half 'information model' of Figure 10. If we

compare this with the data model (as Figure 10 does) we can reconstruct the forward transformation from the information model to the final data structure.

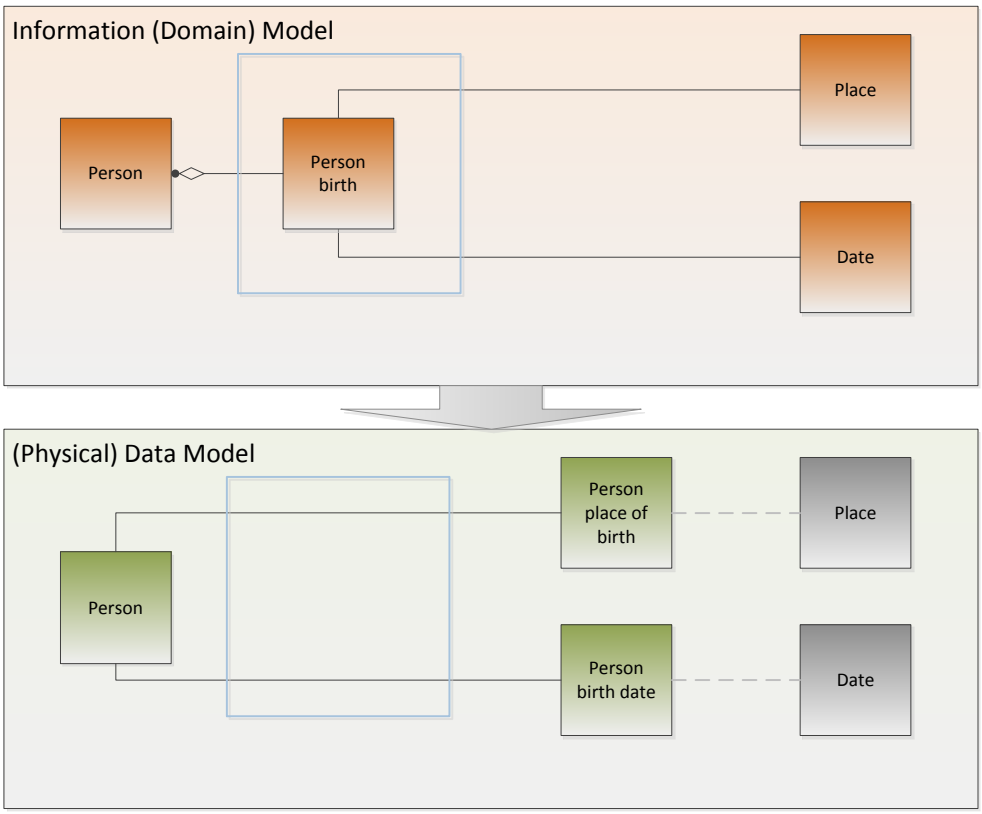

Figure 10: Reconstructed Forward Transformation

This is a reconstruction, as the original production of the data model was not layered. The reconstruction separates the layers and makes it easier to review the concerns (such as performance and memory) that influenced the choice of transformation. It also reveals how the transformation has altered the structure in ways, which, even in this simple case, can be significant.

This example also illustrates how the nature of the analysis makes it difficult to completely automate it. It is unclear what kind of algorithm could unearth the 'birth event' from the data model. However, it is possible to develop support tools for the analysis and we have done this. Or experience leads us to believe that, given current technology, this kind of analysis is at best a semi-automated process.

EXAMPLE PROJECT: SEMANTIC LAYERING

BORO Solutions and QinetiQ are currently engaged in a project instigated by the UK Royal Navy whose goal is to provide a framework for assessing the 'goodness' of a semantic (information) model. This is one of the initial steps towards a long term goal: "…to develop an enduring RN Combat System 'information model'." Where the information model "can be flowed to equipment developers and shared with other services and nations, to avoid misinterpretation and misunderstandings in the meaning of exchanged tactical information."

The project aims to characterise what:

- a model of the semantic layer of a combat system is, and
- a good model needs to contain

Based upon this characterisation, the project is producing a broad-brush, simple method for assessing the

level of 'goodness' of an information model that can be deployed across the candidate RN Combat Systems information models and used to guide their development. A sample of combat system models with varying claims to be information models are being looked at by the project. As well as being used to test run the method, they are being used to identify the areas where improvements are typically required.

The work is currently in progress, but some key features have emerged.

- There seems to be an endemic problem in the modelling community of distinguishing between the represented and the representation – between the domain and the system[8]. Hence, there is a requirement to develop a process that distinguishes between them – and can identify where they have not been distinguished.
- There seems to be a bias in modelling towards only representing what is included in the final implementation. This not only prejudges, to some extent, how the requirements are to be implemented, it also is a bias towards one of the two types of stakeholder (human and machine). This suggests that a requirement to ensure the needs of both types of stakeholder are recognised and balanced is important.

A number of simple straightforward checks are emerging from the analysis. Here are three examples:

1. Check that the model has a clear top ontology with ontological categories and their criterion of identity. This is a stronger requirement that a meta-model, which often only provides the categories of representation – not the represented.
2. Check that model supports general foundation patterns. These should include super-sub-type, type-instance and whole-part patterns. It is important to check that these patterns are not constrained. For example, a common constraint on super-sub-type is restricting the cardinality of super-types to one – also known as single inheritance. A common constraint on type-instance is not allowing types to be instances of other types – also known as non-higher-order types. Many of these constraints reflect the structure of the proposed implementation language. It is, of course, inappropriate to import these constraints to the semantic layer. Domain constraints need to be justified in terms of the structure of the domain.
3. Check that the model has a good, complete, taxonomic structure. One easy review is looking at how many top objects – and how many singleton hierarchies – there are. While the final implementation language might not require the full taxonomic structure, the human stakeholders find it invaluable for common understanding.

These checks have been used to assess a sample of combat systems and also to identify improvements. Initial work has started on actioning the improvements in one of the selected models[9]. To date this has involved stitching a top ontology and foundation patterns into the model and filling in some gaps in the taxonomic structure.

EXAMPLE PROJECT: SEMANTIC HARVESTING

Another related project BORO Solutions and QinetiQ are currently engaged in has a goal of demonstrating the feasibility of harvesting the semantics from existing systems where there is no explicit semantic layer into a separate semantic / information model. The project is initially looking at the combat system highway on Type 23 frigates with a view to harvesting the semantics of the interfaces. The harvesting uses the BORO method described in [8, 9] a key feature of which is the use of operational data from the systems along with the specifications. This ensures that actual practice that has not been recorded in the specifications is captured, which helps to raise the quality of the final model.
The intended goals of this stage are to demonstrate clearly:

- what an information model of this domain would look like,

[8] This is a wider problem, for example in philosophy of logic it has been called ‘use-mention confusion’ - see: W.V. Quine (1940) Mathematical Logic, §4 Use versus mention, pp.23-5

[9] The Modular Open System Architecture (MOSA) based Combat System Architecture Model.

- that the information model can be re-used across a number of systems in the domain, and
- that an information model can be harvested from the existing systems, despite their lack of a semantic layer – and indeed, much formal documentation.

The project has produced initial models using selected specifications, which begin to show what an information model of this domain would look like. This will become clearer as additional specifications are included in scope. There is a close relationship between this and the previous project. The harvesting processes have been designed to extract a 'good' semantic / information model. The checks identified in the previous project provide an on-going quality assurance mechanism that the harvesting is functioning well. Furthermore, the extracted model from this project provides a useful example of a 'good' semantic / information model for the previous project.
It is planned that later stages of the project will look at the semantics of other combat systems in the domain to establish the level of overlap. The current belief is that there is significant overlap, but without the explicit semantics, this is difficult to confirm rigorously. If this is confirmed, it will indicate that there is a substantial opportunity for re-using the semantics across both existing and systems in this domain; in other words, that there is a substantial core of the common information model that applies across systems in this domain.
Once the project is complete, it should have clearly established that the semantics can be harvested from existing combat systems where there is currently no explicit semantic layer.

REFINING THE CURRENT NNEC SEMANTIC INTEROPERABILITY STRATEGY

The work done here provides two (related) opportunities to refine the current NNEC strategy:

- an opportunity to provide a more rigorous explanation of, and means of production for, the 'semantic description', and
- an opportunity to align with and influence member nations' long term application / semantic modernisation strategies.

As noted earlier, the NNEC strategy needs to provide a sufficiently detailed description of what a 'semantic description' is, how it is produced and how it guarantees semantic interoperability before it can be implemented completely. The current work on semantic layering and harvesting can be regarded both as a potential solution to this requirement and as raising some useful questions about what the 'semantic description' is – questions such as:

- How does the application modernisation semantic layer model align with the requirements for the SILF's 'semantic description'?
    - Are there significant differences?
- Should a good 'semantic description' have similar characteristics to a good semantic / information model?
    - For example: Should the 'semantic description' reflect a clear distinction between the represented and the representation? And, if so, how should this be checked?
- In the case of semantic modernisation of existing systems without a semantic layer, there is a clear requirement for semantic harvesting. Does the SILF's 'semantic description' architectural component imply a similar requirement?

As also noted earlier, the current SILF architecture reflects an understandable short term, parochial NATO perspective on the systems that need to be integrated. Expanding the perspective to include a long term view of the member nations' systems and their evolution, will allow the current semantic interoperability strategy to be aligned with and influence application and semantic modernisation strategies. The general need for something like this seems to have been already accepted: "This consensus should include how ontologies are to be used in the lifecycle of information systems and what implementation strategy to

follow."[10]
These two opportunities for refinement are closely related, so boundary issues will arise. For example, the framework will need to establish the relationship between semantic harvesting for application modernisation and for semantic interoperability within the SILF.

REFERENCES

[1] Position Paper on Framework for Semantic Interoperability, IST-094 / RTG-044, July 2011

[2] Bézivin, J, (2003). From Object Composition to Model Transformation with the MDA. 39th International Conference and Exhibition on Technology of Object-Oriented Languages and Systems (TOOLS39). Santa Barbara, California. July 29-August 03

[3] Griethuysen, J.v. ISO/TC97/SC5/WG3-N695 - Concepts and Terminology for the Conceptual Schema and the Information Base, ANSI, New York, NY, 1982.

[4] Kent, W. (1978). Data and Reality: basic assumptions in data processing reconsidered. Amsterdam; New York, New York, North-Holland Pub. Co.

[5] Kuhn, T. S. (1970). The structure of scientific revolutions. Chicago,, University of Chicago Press.

[6] Lycett, M. and Partridge, C. (2009). The challenge of epistemic divergence in IS development. Communications of the ACM. Volume 52 Issue 6, June 2009.

[7] Mealy, G. H. (1967). "Another Look at Data." Proceeding of AFIPS 1967 Fall Joint Computer Conference Vol. 31.

[8] Partridge, C. (1996). Business Objects: Re - Engineering for re - use. Oxford, Butterworth Heinemann.

[9] Partridge, C. (2002). LADSEB-CNR - Technical report 05/02 - The Role of Ontology in Integrating Semantically Heterogeneous Databases. Padova, LADSEB CNR, Italy.

[10] Section 2.1 – Introduction – in [1]