Modelling in a Central Architecture Repository – Lessons Learned

Jens Stavnstrup and Alfred Møller
Danish Acquisition and Logistics Organisation
Lautrupbjerg 1-5, 2700 Ballerup
DENMARK

stavnstrup@mil.dk, avm@mil.dk

ABSTRACT

The paper covers some lessons learned from architecture development in the Danish Defence over the past decade. The paper provides an overview of the challenges of creating a comprehensive view of the architecture for the defence enterprise, and will focus on an approach with modelling in a central architecture repository. The paper discusses benefits and drawbacks of the approach, and provides some of major lessons learned during the actual work.

The emphasis is on Danish Defence with a need to cover architecture in a broad perspective, including relations to NATO, international missions, national, the government outside the defence. The paper also touch the need for cooperative efforts for progress of architecture work, and will provide some conclusions and plans to meet current and future challenges.

1.0 INTRODUCTION

The purpose of this paper is to communicate some lessons learned in architecture work in the Danish Defence over the past decade. The paper will provide an overview of the challenges of creating a comprehensive view of the architecture for the *Enterprise*[1], and the focus will be on how modelling in a central repository can contribute to meet these challenges. The lessons learned are based on both basic studies and practical experience with work on these solutions. Finally, the paper will provide some conclusions and plans for the ways ahead to meet current and future challenges, and will touch the need for cooperative efforts for architecture work in an international forum.

The emphasis of the paper is on architecture work in Danish Defence, but it will cover architecture in a broad perspective with relations to NATO, international missions, national tasks including the North Atlantic area, the government outside the defence, corporation with non-governmental organizations, etc. Therefore architecture elements of interest cover a lot of different aspects, including:

- Capabilities, Systems, Services, Technical and Operational Standards, Organizations, Operations, and Missions.
- Strategies, Business processes, Operational procedures, Operational capabilities, Supporting capabilities, Intelligence capabilities, and Cyber capabilities.
- Administrative and operational information/data
- Military security and Information Assurance in network based operations.
- Multiple stakeholders with different interests and use of different terminology.
- Different levels of abstraction covering the range from high-level and overarching to detailed and specific architectures.

[1] In this paper *Enterprise* means the Danish Defence Enterprise.

- Timeline issues with legacy, baseline, as-planned, and to-be together with long-term visions.
- Collaboration with external parties (both military and non-military), such as NATO, Nationally, Governmental organization outside defence, NGO, suppliers.

The challenge is to show how all these parts fits together in a coherent architecture where all different aspects are included; - this is illustrated in Figure 1.

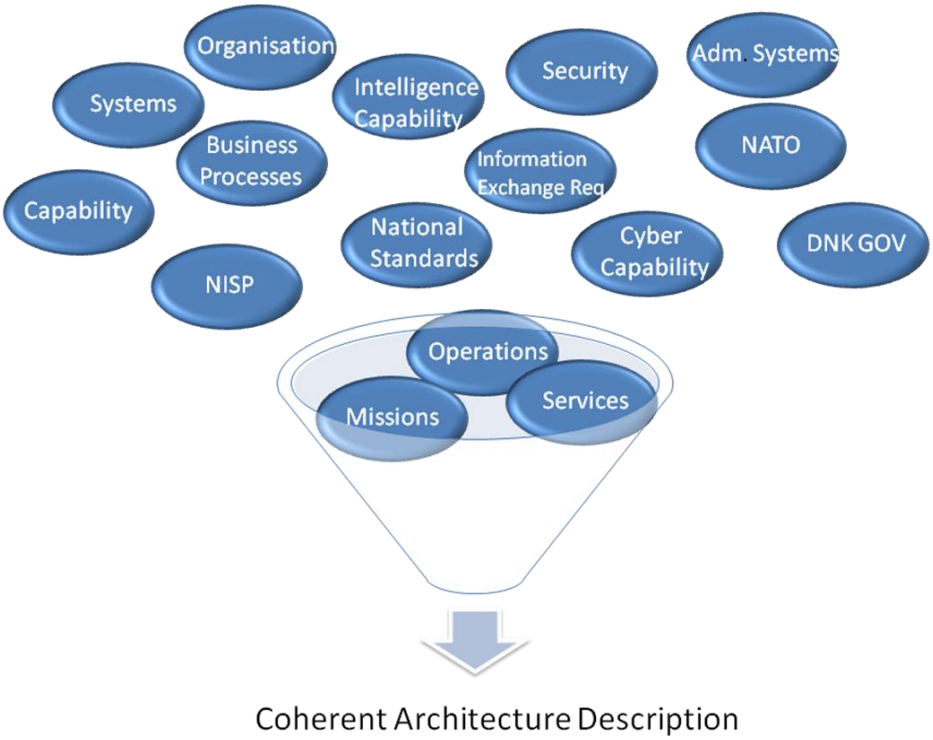

Figure 1: The Challenge of Coherent Architecture.

The challenge of coherent architecture is covering not only the *Enterprise*, but will also cover the *Extended Enterprise*[2] with relations to current and future corporation partners. Figure 2 illustrates the repository with architecture elements and relations between the elements within the enterprise and the boundary to the extended enterprise.

For the purpose of this paper, two different roles are used. The first is a *stakeholder* expressing concerns (in compliance with ISO 42010 [1]) and the second is the *architect*, a role which in this paper is simplified to do modelling in the repository on behalf of the stakeholder. Other architect roles and competences are not considered.

[2] The *Extended Enterprise* means the Danish Defence Enterprise extended with the interface to the environment.

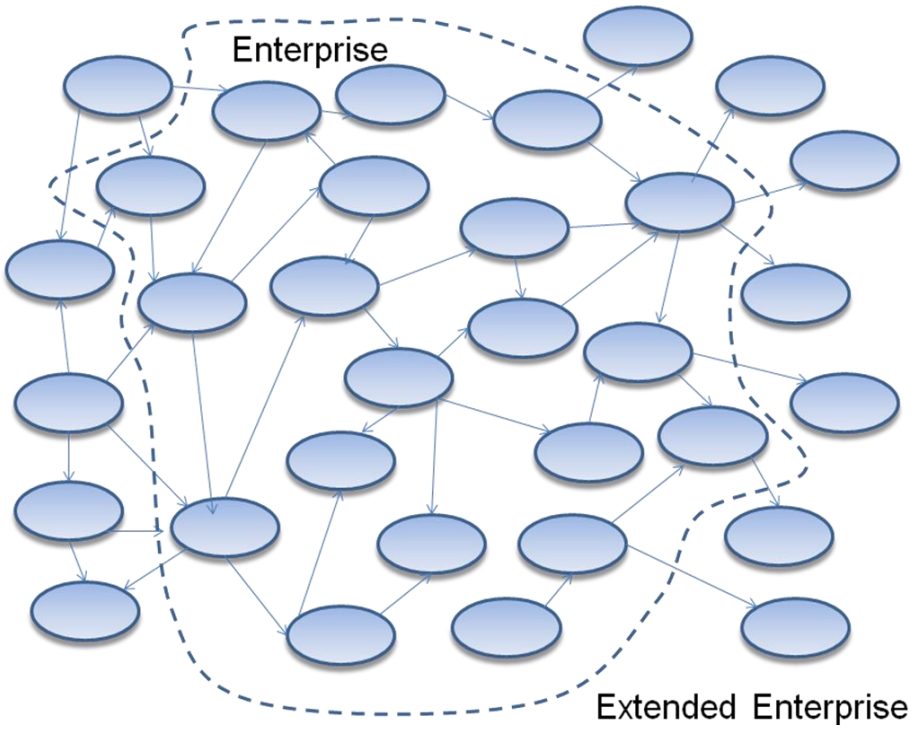

Figure 2: Architecture Repository showing Enterprise and Extended Enterprise.

2.0 APPROACH

One approach taken to meet the challenges of the extended enterprise has focused on four selected topics:

1. Focus on use of a central architecture repository has been selected to provide a good foundation for the "end goal", which we envision as a coherent architecture for the extended enterprise.
   The term central repository means that there is only one enterprise wide architecture repository in the Danish Defence.

2. Focus of modelling of architecture has been selected in order to have a formalised background for reuse, coherency, interoperability, corporation, and future benefits such as simulation. This approach means that modelling is encouraged, but only to the extent where it makes sense, i.e. modelling with elements and relations should only be used if it serves a purpose.

3. Focus on the evolutionary aspects has been selected in order to avoid putting a lot of restrictions on the architect or trying to prepare everything in advance, i.e. the architects may do it the way they want, and the way they understand or are motivated for. The evolutionary approach means that architecture development is not set up as a "big bang with complete setup/guidance", but provides room for development "freedom" for different architects. This approach means that the repository baseline consists of a lot of fragments which at a first glance are not necessarily coherent.

4. Focus on stakeholder driven architecture should ensure that each part of the architecture description identifies a concern of one or more stakeholders.

In summary; the approach has been rather pragmatic in order to get started even though tools, methods, practical experiences, etc. are lacking; – however the target (or end goal) is still to show how everything fits

together. I.e., the pragmatism is mainly a way to get started – well aware of the difficulties that have arisen or will arise in the future.

3.0 HISTORICAL BACKGROUND

The need for addressing the challenges became clear early on, because of the introduction of network based operations (NBO), a wish to optimize business processes, and the increased complexity of large scale software systems.

Architecture development work in Defence spans more than a decade, and a number of activities have been initiated to accommodate the mentioned challenges. Some of these activities include modelling in a central repository, where lessons learned are covered by the rest of this paper.

At an early stage it was decided to use NATO Architecture Framework (NAF) [3] as the standard architecture framework for ICT architecture. NAF2 was considered, but use of NAF was postponed until the availability of NAF3. However, until NAF3 was available some work was based on MODAF [2].

Today the standard for ICT architecture descriptions is NAF3. This implies the use of NAF views and sub-views, and NAF metamodel (NMM) is mandatory for modelling of operational ICT systems. However, there are additional requirements from national authorities [4] where the usage of a military oriented framework might be inappropriate.

It was also obvious that architecture tools were necessary to meet the challenges. Therefore an architecture tool survey was carried out. The survey revealed that the tools at that time were not mature and unable to cover all aspects of the enterprise. The tool ARIS (from Software AG) was selected, mainly because it was already available in Defence for Business Process modelling. It did have NAF2 support, but it was not useful for the purpose of ICT architecture modelling; – however, UML turned out to be a viable solution until NAF3 support was available. The selection of ARIS served the purpose of having one repository for all models.

3.1 Current Status of Repository

The repository is currently exclusively operated by architects.

The repository is populated with architecture elements and relations created by a number of architects working within different domains over a long period of time. As such, the repository contains inhomogeneous and incomplete architecture descriptions. Logically, the contents of the repository can be separated into four parts, which reflects the domains modelled by the different architects and the timeframe, in which the modelling has taken place.

The four parts modelled in ARIS repository are:

- Part 1: This part comprises architecture models developed with the ARIS NAF plug-in., and covers descriptions, both at an overarching level, reference level, and specific system level. ICT architectures (mainly within the operational domain) are described – however no strict methodology has been used, mainly because of the lack hereof in NAF. Some efforts are ongoing for reuse, consistent use of attributes, etc.
- Part 2: This part comprises models based on UML class diagrams. It exists since there was no NAF support early on – a simple home grown metamodel was developed for this purpose. The metamodel is partly compliant with NAF/MODAF and mainly ensures that all models in this part are created on the same foundation. Today this part is primarily used as a draft for “real” NAF compliancy within part 1.

- Part 3: This part comprises Business Process models. Guidelines have been established and education exists for modelling of Business Processes, but only for processes in the administrative domain. Processes are modelled both at high- and low-level, including activities.
- Part 4: This part comprises ICT architecture descriptions incompliant with NAF. It is mainly used for the administrative domain. A TOGAF like approach has been used, but without actually using the TOGAF support in the tool.

The rest of the repository is not related to the four parts described above and consists mainly of models used for specific purposes and/or experimentation.

Attempts to integrate these inhomogeneous pieces are ongoing. One example is the use of enterprise wide architecture principles. The architecture principles are to be seen in a broader perspective within both the administrative and operational domain. Other examples include efforts to identify common attributes for all architecture elements.

3.2 Other Baseline

Although architecture work in Defence does not have a long history, some important steps should be noted.

The first task was to develop the long-term target architecture for NBO communication with the army as the stakeholder. The long-term ICT target architecture today covers the Enterprise with focus on information centricity and a technology forecast founded in the requirements from a set of political selected scenarios.

This has improved the understanding for a number of dimensions in the architecture. This includes Baseline versus To-Be architecture, High-level versus low-level system architecture, and Scope (Overarching or specific) of architecture. This is illustrated in Figure 3.

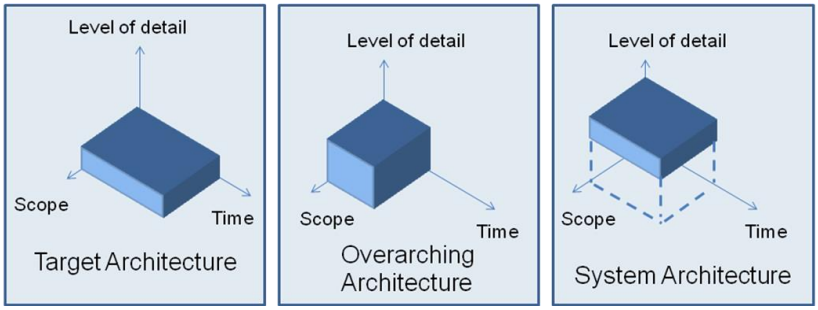

Figure 3: Architecture Types

Finally, it has to be noted that there are a lot of architecture descriptions developed with different tools and not being part of the central repository. This includes models developed in Visio, UML-based models in Enterprise Architect (from SPARX Systems), MODAF based models in Rational System Architect (from IBM), and lots of non-models in text based descriptions. All this is outside the scope of the paper; - yet, there are considerations on how to transfer and model this into the central repository.

4.0 BENEFITS AND DRAWBACKS OF THE APPROACH

In this part of the paper the intention is to describe some benefits and drawbacks for the focus areas of the approach. The benefits and drawbacks are discussed both on a theoretical and practical level. The real practical lessons learned will be in chapter 5 Lessons Learned.

4.1 Central Repository

The purpose of using a centralized architecture repository is to ensure that all architectural artefacts are stored in single place, where they can be discovered and utilized by multiple parties. The central repository is supported by the chosen architecture tool, i.e. ARIS. ARIS provides access to an enterprise wide repository, which can be used for architecture development for different purposes.

The use of a centralized repository also means that architecture descriptions and/or architecture elements should be imported to the central repository, even though the original version is created and maintained externally to the central repository. An example of this is the mandatory standards and profiles defined and maintained in NATO Interoperability Standards and Profiles (NISP)[5].

Being able to cover all aspects of architecture enables us to get a coherent view of the architecture and enables us to trace architectural decisions taken from initial user requirements to system design. At the same time we will be able to justify that architectural decisions taken in the design of a system are based upon user requirement. The use of a central repository also enables us to query the database for the existence of specific objects. E.g. an obvious example would be to search all architecture descriptions for a specific standard, to identify the systems affected by e.g. an upgrade of the standard.

Table 1: Central Repository.

Benefits and Drawbacks: Central Repository	
Benefits	Potential for reuse of architecture elements/patterns, and the ability to discover if solutions are already available.
	The concerns of all stakeholders are expressed in one place, i.e. there is a increased potential for identifying inconsistencies and pointing them out to management.
	Some benefits in use of a common repository terminology.
	The stakeholders and others are able to browse the repository to get a better understanding of the architecture elements and relations in all architecture descriptions.
	Technical means provided by the tool portfolio may support bridging the gap in the different parts of the repository.
Drawbacks	The support is based on one tool, i.e. vendor dependency may limit the use and development. This is a risk.
	Eventually some classification issues may arise, mainly because a lot of information is stored in one place. This may result in requirements for either a single classified repository or multiple unclassified repositories. The latter may introduce inconsistencies between unclassified and classified models.

4.2 Modelling

The approach of creating models using a metamodel was selected at an early stage based on the expected benefits experienced by coalition partners and with knowledge of the upcoming tool support of metamodel. We have experienced the benefits of this approach, but it has not been without a cost. In the four subsections below are given an overview of benefits and drawbacks split up into modelling in general, NAF based modelling, modelling with the architecture tool, and modelling with the tool including the tool vendors NAF support.

4.2.1 Modelling versus non-Modelling in general

Modelling in architecture context means an accurate description of relevant architecture elements with relations between them. Modelling as shown in Figure 2 is the most obvious, but it may also be in the form of appropriate tables, spreadsheets, matrices, and drawings with clear indication of architecture elements and relations. Architecture documentation in a form with text and drawings, where additional interpretation is needed, is not regarded as modelling in this context.

If you need to verify the architecture against a metamodel, the modelling approach may be needed to a certain level, but below this level, it may be useful only to describe the architecture in plain text, tables, figures, etc.

It is important to remember that architecture is only the first step of the design. A number of other issues should also be taken into account. To what level of detail should the architecture be modelled? It has to make sense. There should there be an interface to other technical documentation, inventory list, project management, and other existing disciplines and/or applications.

Table 2: Modelling in General.

Benefits and Drawbacks: Modelling in General	
Benefits	You must be clear about the specific stakeholder concern. Modelling will force you to give exact answers where ambiguity is possible, e.g. whether the element is a system or a capability.
	You get relation accuracy compared to ambiguities (or interpretable relations)
	You get a better and more consistent terminology.
Drawbacks	The costs for initial setup are high, - including selection of framework, metamodel, tool, etc.
	The initial costs for gaining the necessary modelling experience are high.
	Not everyone needs a modelled architecture, i.e. they don't accept possible high costs.
	Modelling not useful unless a metamodel and/or taxonomy references are available.
	Modelling is an expert domain.

4.2.2 NAF Based Modelling

The NAF views and sub-views are important for both modelling a non-modelling, because this is where the stakeholders must/should identify themselves. However, when it comes to NAF based modelling, the metamodel is of major importance.

Broad acceptance of NAF is difficult to achieve, because many systems in Defence must interoperate with national systems outside defence. And outside the defence domain, military frameworks are not used at all. This provides a discrepancy in attitude towards civilian and military frameworks, because of the interoperability requirements.

NAF claims conformance to ISO 42010, but in reality interoperability is only realized through a common metamodel. There are e.g. no methods in NAF, which makes it difficult to figure out how to attack a given problem. We need to attack the problem from many different angles and consider a number of factors, including the mentioned dimensions of architecture. All the issues are solved using different methods/processes, and the methods together with the resulting architecture descriptions all contribute to a common understanding and therefore enhanced interoperability.

NAF does a nice job of separating the concerns of the different stakeholders, although the number of viewpoints and architecture elements representing these concerns can be a bit intimidating for the architect at first. A number of architecture elements in the metamodel are not included for the benefit of architects, but are included for UML technical reasons, such as UML meta-classes, abstract classes and stereotype originally defined in SysML. The combination of many viewpoints, many architecture element types and a deviation in the interpretation of the NAF metamodel makes architecture development quite intimidating for architects with little experience. And if NAF3 is intimidating for the architects, consider the reaction of the stakeholders.

Table 3: NAF based Modelling.

Benefits and Drawbacks: NAF Based Modelling	
Benefits	NMM is rather well-defined and has proven its value.
	Alignment with operational partners is relatively straightforward, provided they use NAF (or compliant framework like MODAF, DoDAF, DNDAF, etc.)
Drawbacks	Alignment with civilian architecture descriptions is not obvious.
	No method is described by NAF.
	Attributes to objects and relations are not described in NAF, i.e. national solutions are implemented, which prevent seamless exchange with collaboration partners.
	NMM may be well-defined, but the UML based definition is not always easy to understand, - requires experience and best practise.
	NAF/NMM does not provide direct support of all element types we are currently using; - examples are processes and principles. They must be described by activity models and text based, respectively.

In summary, the usage of multiple military frameworks is confusing, and the connection to civilian frameworks should be more straightforward.

4.2.3 Modelling with Tool (ARIS)

An architecture tool is necessary for support of modelling in a central repository and at the same time supporting the broad enterprise perspective. Also support of frameworks (including metamodel) is essential. Another aspect is exchange of architecture descriptions with partners and industry.

Short about ARIS

The ARIS tool is a tool primarily intended for business process modelling, although it has support other domain, such as e.g. NAF, MODAF, DoDAF, TOGAF, ArchiMate compliant architectures. ARIS supports UML and has an application portfolio for management, simulation, etc. The ARIS tool comes with an extensive and rich metamodel use to express different aspects of process modelling, some of which are specific to the methodology defined by ARIS. The metamodel is a three layer model. On the lowest layer ARIS defines a very generic data-model which is more or less identical to the data structure used in the architecture repository. The next layer constitutes the object-model, where all the elements used in process modelling are defined. And finally there is a presentation layer with presentation objects, each of which is representations of objects in the lover-layer metamodel. It is this model, which enable the tool to reuse objects across models. So when we inspect objects in the presentation layer, what we actually do is through a pointer in the presentation object, do an inspection of the actual object in the object layer. Parts of the defence organization use ARIS for business Process modelling only, i.e. a different metamodel and other parts of the Defence organization are using yet another metamodel for enterprise architecture modelling. The usage of multiple and partly overlapping metamodels do not foster interoperability or reuse, but it supports engagement from multiple parts of the organization.

Table 4: Modelling with Tool (ARIS).

Benefits and Drawbacks: Modelling with Tool (ARIS)	
Benefits	The tool supports the central repository.
	Common foundation for different architecture types in Defence (BPM, NAF, TOGAF). Different types of architects have the same user experience.
	Flexible with support of filters, methods, privileges, access rights for users and groups, etc.
Drawbacks	Education is needed for operate, i.e. it becomes mainly an expert tool.
	It is very expensive, which also limits the number of users.

4.2.4 ARIS Modelling with NAF Support

The original metamodel in NAF 3.0 is defined as an UML profile, and since ARIS is basically not an UML tool, the implementation of NAF in ARIS is an interpretation of the NAF metamodel.

After working with the ARIS implementation for a while, we discovered some irregularities in the way ARIS treat NAF objects compared to the native process oriented metamodel, and it turned out that NAF architecture elements are not implemented at the object layer, but only implemented at the presentation layer basically making NAF3 a second class citizen in the ARIS universe. So what in NAF might be represented by multiple elements is implemented in the native business process metamodel as a single element, which makes mapping between the models impossible.

Table 5: Modelling with Tool (ARIS) and NAF.

Benefits and Drawbacks: Modelling with Tool (ARIS) and NAF	
Benefits	Enforced modelling conventions provided by NAF and tool setup.
Drawbacks	Interpretation of original NAF metamodel with possible inconsistencies. There are observed deficiencies in the metamodel due to flaws in the NAF implementation in ARIS, or lack of understanding or misunderstanding of the metamodel by the architect.
	Lack of interoperability with other NAF implementations, mainly caused by lack of common understanding of framework between vendors of architecture tools.
	It is time consuming to learn a new tool in combination with a new implementation of the framework. In ARIS, NAF has been implemented as a second class citizen.

4.3 Evolution in Architecture Development

With little experience and no methodology support, it was decided to use an ad-hoc approach to modelling and in time use the acquired knowledge to approach modelling in a more systematic way. The evolutionary approach for modelling was chosen to ensure that we were able to produce results quickly, which provided value for the stakeholders. In the future, new and extended frameworks, simulation issues, better vendor support, better exchange of architecture description, etc. are expected to make this approach absolutely necessary. The Enterprise with its diversity of stakeholders will make it necessary to be able to adjust best practises. This is even more important for the extended enterprise because this implies impacts outside your own control.

This makes it impossible to have final versions of conventions and guidance for modelling, because such a guide must be extendable. This is not the same as having no guidance at all, but it must not be prioritized over the need to produce useable artefacts.

An important thing for evolution is also the long term goal: do not prevent architects to get started, but try to insert new parts into the right context. The thought is that getting started with a 20% useable draft is better than waiting for 100% completeness and compliance. Things change over a period of time to provide for evolution/agility.

Some ways to handle the evolution is coordination forums to increase a common understanding; However; - the cultural differences can make it difficult.

Table 6: Evolution.

Benefits and Drawbacks: Evolution	
Benefits	We obtain experience during actual (and useful) work – "no" experience needed. Yet, tool education is mandatory. We can start without complete handbooks.
	We can accommodate new methodologies, new framework versions, etc.
	We are prepared for the diversity of stakeholders within the enterprise, and even changes within the extended enterprise.
Drawbacks	The models/elements are derived from different metamodels (NAF, ARIS BPM and TOGAF).
	The repository may be a mix-up of diverse aspects and elements – i.e. a non-structured total repository (yet chunks may be structured).
	Mix of high-level and low-level elements.

A lesson learned is that the need for coherency only becomes evident for some stakeholders over time, i.e. the motivation may show up eventually.

4.4 Stakeholder Driven Architecture

A recognized way is to address all stakeholder concerns with the goal of establishing a coherent architecture. Under the right conditions this approach may work just fine; - however, it may also fail for a number of reasons in a modelling environment.

With very little experience with architecture frameworks, tools or methodology, the introduction of architecture in the Defence was an experiment on multiple levels. At an early stage the task was to create a comprehensive picture of the baseline architecture of the defence; - not in the form of an Enterprise Architecture, but rather in the form of a high-level picture including all operational and administrative significant ICT systems in the Defence. Furthermore, the evolution of this baseline architecture towards the long-term vision target architecture should be shown. This task involved at lot of different stakeholders.

One important experience from this early task was the necessity of having stakeholders identified and engaged in the process. Although there was a stakeholder to the overall architecture, it was more or less assumed that the architects themselves were able to judge, what was important for the stakeholders of the individual systems. That was not necessarily the case, and being newcomers to modelling, there was a tendency to judge architecture by number of models, which fairly often led to the development of vague architecture descriptions. So identifying and getting stakeholders involved from the beginning is essential in order to realize a successful architecture development. If we can convince the stakeholders of the added value of architecting, they will be much more engaged and willing to provide the necessary information; with which we can express the stakeholders concerns in a meaningful way in the architecture description.

The drawback of the stakeholder driven architecture in a modelling approach is the lack of stakeholder uniformity regarding the concerns. Some stakeholders understand the models, but other stakeholders do not see the usefulness at all. Even if the stakeholder understands the model, objects and relations, it is not the same as taking responsibility for the content of the model and the impacts. It is more like "The model looks right, but I can't judge the implications".

Table 7: Stakeholders in a Modelling Environment.

Benefits and Drawbacks: Stakeholders in a Modelling Environment	
Benefits	All stakeholders are able to include their concerns within the extended enterprise.
Drawbacks	The concerns of the stakeholders are not necessarily compliant with long-term visions, - for some, the tendency is to look at the problems of today - i.e. “It has to work now”.

In summary, the stakeholder driven architecture is a good principle; - however, it does not work in a modelling environment, unless the architect takes responsibility for the model, or provides other means to communicate with the diversity of stakeholders.

5.0 LESSONS LEARNED

In this section, some lessons learned for the approach are provided. The intention is to mention both good and bad experience together with some identified problems.

5.2 Cultural Barrier

Some stakeholders are conservative in nature and will approach different and new ways of thinking with suspicion or scepticism, and some stakeholders prefer to do things the way they have always been done. Other stakeholders are progressive and are willing to push the envelope. In other word, there is a significant cultural barrier to overcome, when the architect interacts with the stakeholders. It is therefore the job of the architect to interact with the stakeholders in such a way, that architecting becomes a natural part of the process and is something everybody can recognize as a discipline, which provides value to the development process.

The understanding is not only about defining and describing architecture, but also on how to use it for practical purposes with some benefit. An example is the screening process for projects and acquisitions, which also have to include architecture requirements; - in fact you need to have architecture as an integrated part. This is not easy with already well-established procedures, but even small steps may improve the process.

Table 8: Cultural Barrier.

Lessons Learned: Cultural Barrier
Do not underestimate the cultural barriers; - you need to see them as a natural thing.
Handle the barriers in a long-term perspective. You are better off with a few slightly lowered barriers than one barrier being raised.

Summary: Not only is education needed, but you also have to accept that it takes time to reduce the barrier to reach the goal of a coherent architecture for the extended enterprise.

5.3 Experience and Maturity of Architects

In hindsight you can always discuss whether you needed more architects and some with greater experience. The lessons learned are that it is a learning process for architects. This goes for both modelling based on complete guidelines as for business processes, and for modelling based on NAF3 compliant architectures with a new tool, including the interpretation of NAF3 metamodel. Obtaining best practises and having a common understanding is time consuming.

In this context, you can also discuss if the decision to implement MODAF and subsequently NAF as an UML profile is the best solution; - however this is the prerequisite for everyone. At the time, it looked like a very good idea, as it would provide us access to an expressive language, which was supported by an extensive set of tool vendors, and promised seamless exchange of architecture descriptions through a vendor independent exchange format.

For architects, the relationship to stakeholders is a big issue, because they have different interest. The architects need to understand the tool, the repository, the framework, the metamodel, etc.; - however this makes no sense for most stakeholders. The maturity and experience of architects is essential to avoid confusing stakeholders with architecture only aspects.

Table 9: Experience and Maturity of Architects.

Lessons Learned: Experience and Maturity of Architects
Time consuming for architects to understand how to communicate with the different stakeholders.
Architects should avoid using technical terms when communicating with stakeholders and management.

5.4 Experience and Maturity of Stakeholders

The maturity of the stakeholders, when it comes to familiarity with NAF (or other frameworks) varies significantly.

- Some have never heard about it, and could not care less.
- Some are familiar with and use the abbreviation frequently, without actually having a clue what it is really about.
- Some know it is about modelling and have something to do with boxes, lines and arrows, which are supposed to illustrate something - and it is important!
- Some understand the "separation of concern" concept and therefore understand the function of a NAF sub-views – and that is good,
- Some are actually able to read and understand an architecture description.

At the beginning, it was somehow assumed that stakeholders with proper education would not only be able to understand models in their own area of expertise, but some might even be able to do some modelling themselves. For the immediate future, we have come to realize, that we need to take a different approach. We basically need to meet the stakeholders in their own turf - we need to communicate with them using their own language - and then subsequently transform the acquired knowledge into NAF compliant models.

This also means that when we want to verify that the models we have generated is actually what the stakeholder intended, then we again have to present the models in a form the stakeholders understand.

Table 10: Experience and Maturity of Stakeholders.

Lessons Learned: Experience and Maturity of Stakeholders
In general do not expect stakeholders to understand (and even model) elements from their own area of concern.
Do not expect stakeholders to see the need for a coherent architecture as they will and should only focus at their own concern.
Communicate with stakeholders in their own language – both for input and output.
Introduce the models along the way, and be sure not to lose the stakeholder. Hope for improvement over time.

5.4.1 Case 1: Using Programme Architecture

Program views (NPV-1/2) are used to get the relation between projects etc. in a programme.

In one program, the program manager was maintaining multiple inconsistent lists of projects, capability requirements etc. and was not aware, that these aspects was covered by a NAF view providing the relation between the projects, milestones and capability increments (or Out-of-Services gaining's). Furthermore, these capability increments provided a direct relation to improved (new) systems and capabilities.

The stakeholder saw the benefit in having documentation of these relations, and being able to see the coupling back to the projects under the programme, including how the project depended on each other for providing a fielded capability. This had impact on for example the priority of projects.

To convince the primary stakeholder and other project concerned stakeholders, it was necessary to communicate by use of report extracts from the repository. This is because, the NPV-2 view is not well-known and it is not normal to see a model of it. However, after communicating by use of a report extract (spreadsheet), there seem to be a growing understanding of the model.

5.5 Modelling experience

Interoperability in NAF is only enabled at the metamodel level and is therefore only considered a technical issue, but interoperability is much more than that. The lack of a methodology prevents architects to approach problems the same way, and therefore prevents them to instantly recognize the thinking done by other architects. Besides modelling, we also need a process for architecture evaluation to ensure we are able to reach a reasonable quality level.

Table 11: Modelling.

Lessons Learned: Modelling
Using the appropriate level of abstraction when modelling requires experience and understanding of the issues to modelled, in particular the purpose of the architecture.
It provides a good way of identifying ambiguities and inadequacies.
NAF metamodel provides a good way of ensuring the coherency.
The missing methodology and attribute definition in NAF does not make architecture comparison easy.
Multiple inconsistent implementation of the NAF metamodel by different tools.

5.5.1 Case 2: Evaluation of Information Architecture

In one case we received a document specifying the information exchange requirements to a system. At a first glance the document seemed to be a very well-defined; yet informal example of specific operational requirements. The first task for transforming the document into one or more models was to decipher these requirements and identify all architecture elements that were operational in nature. The result was as set of operational views, describing different concerns of the stakeholders embodied in the views NOV-2 (Operational Information Requirements), NOV-4 (Organization Relationships Chart) and NOV-5 (Operational Activity Model).

Analysis of the generated model and additional discussion with the stakeholders revealed a couple of issues:

- One information element was presented in the paper under different names.
- In the model, a couple of information elements was described, but was actually never used or exchanged with any external participant and was therefore unnecessary to model.

This simple example confirmed our impression, that architecture is very important and also helped convince the stakeholder of the value.

This case turned out to be a success, because the stakeholder focused on information exchange instead of building up a system. Furthermore, the stakeholder was able to understand the loose ends of the information model, and the corporation with the architect worked fine.

5.6 Identifying and Engaging the Right Stakeholders

When acquiring knowledge for the purpose of modelling a specific concern of a stakeholder, it is essential to identify the right stakeholder. Architecture is often seen as an IT-only discipline, so when we e.g. try to identify operational requirement for a system of interest, we will often be referred to an officer, which also happens to have the local ICT competence, i.e. a person who have some knowledge of existing systems. This means that during the collection of concerns, you will not receive operational requirements, but what is essential system requirement. Statement like we want e-mail, web capability, or we want to be able to communicate with these systems, we need a database etc. might all be valid system requirements, but not necessarily covering all the operational requirements. So besides identifying the stakeholder, we also have to convince them of their role in the development process in a NAF perspective.

Table 12: Identify and Engage the Right Stakeholders.

Lessons Learned: Identify and Engage the Right Stakeholders
The right stakeholders are not always easy to identify or get access to. To engage them in expressing their concerns are even more difficult.

5.6.1 Case 3: Capture Operational Requirements

With the increased awareness of the importance of architecture, it was decided, that an operational requirements should be captured as a modelled architecture description. The requirements should be created before a project was formally started and thus before any system development was initiated. As a consequence there was a request for access to operational users. However, it turned out that the request did not end up with the real users, but those involved in the daily operations and maintenance of similar systems. Architecture was seen as a technical discipline.

The result was mostly system requirements, i.e. many System Views and only a few Operational and Capability views. However; - it was useful as well, because the people showing up had some knowledge of the domain. And the captured system requirements do not seem to be in contradiction with the operational requirements. The exercise also improved the awareness and understanding of architecture work in general.

5.7 NAF Experience

During the modelling of architecture descriptions some problems were identified, both for architects and stakeholders.

Table 13: NAF Experience.

Lessons Learned: NAF Experience
The complexity (and size) of NAF is an issue for both architects and stakeholders.
Effective communication is difficult since NAF experience varies significantly.
Many stakeholders have difficulties in identifying themselves in NAF, both for views and in particular sub-views.
The lack of proper modelling methodology for NAF makes it difficult to create architecture descriptions in a consistent way, which is easy recognizable by other architects, including those outside the Defence.

5.8 National versus International

The national and international aspects provide some issues. For instance, the operational and administrative systems are modelled by different stakeholders and organizations, which cause different policies regarding architecture and modelling conventions. One stakeholder group and organization use NAF and use English in all models and for all objects, in order to interoperate with NATO and coalition partners in an operational context. Another stakeholder group and organization focus exclusively on administrative system and the mandatory information exchange requirements mandated by national authorities, where the language is Danish and TOGAF is the preferred framework.

Table 14: National versus International.

Lessons Learned: National versus International
The language issue is a major issue for modelled architecture, - even though the tool (ARIS) have multi language support.
The different requirements for frameworks may be an issue. This will probably make additional artefacts a necessity.

5.9 Reuse

Reuse of architecture components is one of the benefits important to architecture work. The low-hanging fruits are amongst other things to create elements describing organisations, standards and profiles. Denmark is not required to use the standards and profiles mandated by NATO, but it makes sense be to compliant with NATO standards and profiles for participation in operations with other NATO partners. At the same time, we have to follow the standards mandated by national law and regulations.

Another example is standardization by communality (standardization by products). Lots of existing systems use proprietary standards, which has to be in the repository, because they are all an important part (and sometimes the limitation) of almost any technical architecture. Somehow you need to make sure, there is an alternative, based on open standards.

Regarding issues for organisation and personal competences, it is recognised that the actual defence organisation is dynamic with frequent re-organisations. Some models and objects describe detailed processes and business processes – down to individuals. However, you should be careful in reusing such fine grained elements as their relations may change overnight. Similary, it is a problem to use architecture models as inventory list.

Table 15: Reuse.

Lessons Learned: Reuse
Reuse including discovery process across different parts of repository is limited to the features provided by the tool, including filters, reports and checks.
High-level and medium-level elements should be reused as much as possible. Be careful in reusing low-level elements.
Minimize reuse of proprietary standards. If proprietary standards are unavoidable, then investigate for future open standard alternatives.
Be aware of gaps between, NISP, national standards, and other requirements. This is not yet recognised as a major issue, but it is a risk.

5.10 The Coherent Architecture

Modelling in a central repository does not ensure coherency of the architecture; but it is expected to provide a good foundation for the future. However, within the NAF compliant parts of the architecture repository, there is a very good potential for coherency because of the metamodel.

Another way to improve the coherency is to benefit from reuse. If e.g. a system is reused in another context you may discover new relations improving the coherence across the enterprise. Many things can be done technically, but some stakeholders may be the key to improve coherency, if they understand the importance of capturing undocumented relationships.

A simple example is investigation of a comprehensive list of systems already available in the repository. Somehow they normally interoperate without formal documentation available.

Table 16: Coherency.

Lessons Learned: Coherency
NAF compliant architecture has a good potential of coherency.
The different parts of the architecture repository may obtain coherency if reuse potential is identified.
Key stakeholders may provide undocumented relations and thereby significantly contribute to a coherent architecture.

5.11 General Observations

In this section a few general observations having impact on the architecture works as well is mentioned.

Better support for exchanging descriptions between different tools

Exchange of architecture descriptions is difficult when multiple tool vendors are involved. Even for UML based descriptions it is difficult, among other things because of different UML editions. Exchange today requires comprehensive work, because "programming" is needed. Obviously - this should be addressed - possibly in a multi-national and open process.

Even if you use same tool, it is not always straightforward because of different setup. An example is exchange of mission network repository also modelled in ARIS. Attributes are used in different ways, i.e. they could not be merged directly for reuse of architecture elements.

Framework inadequateness

Use of architecture frameworks are not always easy, because the stakeholders have requirements which are not easy to accommodate, i.e. some best fit approaches may be needed.

It is important that Frameworks are not too rigid. I.e. they must be extensible in order to accommodate unexpected concerns from stakeholders. How do we deal with that?

One way: make a model which is not compliant with the metamodel, ex. UML class diagram – or do not model at all! The ARIS tool makes it possible to model UML diagrams as a sub-model to architecture elements.

General Limitations for Architecture

Architecture can't be used for everything – you need to focus on what it can do for you. What does make sense? Some experience is:

- You will have to accept "Black Boxes", leaving details out.
- Show the benefits seen from the stakeholder side, not from a theoretical point of view.
- Use pragmatism instead of enforcement, at least at current stage.
- Architecture is not complete technical documentation, but may be part of it.

- Architecture is not a complete description of all processes from top to bottom – but it should serve a purpose.
- Architecture is not an inventory list or acquisition list.
- You need to limit yourself, because you can't accomplish everything.

6.0 SUGGESTIONS FOR CORPORATION

Nationally, we can create awareness of the architectural efforts and best practices. We can publish architecture descriptions, create guidelines and give lectures on architecture, NAF, the architecture tool, and the way we have organized the work in the enterprise-wide architecture repository. However, the architecture community in the Defence is quite small, i.e. it is limited how much can be done, especially if we want to improve the quality of architecture descriptions and combined architectural knowledge.

Also, a lot of work is directed towards systems, which sooner or later will be used in an international context. Therefore it is recommended to take a broader view on what can be done in order to improve architectural effort. Especially, since we in the future expects more frequently to exchange information with collaboration partners in the form of architecture descriptions.

We therefore propose the following initiatives to be carried out in an international context such as NATO:

- Create a forum, where a collective understanding of the metamodel can be refined - maybe editors (benevolent dictators) are necessary - in order to solve disputes.
- MODEM is based on the IDEAS ontology - how do we extend a framework without introducing inconsistencies in the model. Not everybody have the proper skills to do that. It cannot be done by a committee of politically appointed "experts". The BORO methodology is simple, applying it is not.
- Establish a best practice in modelling
- Examples, examples, examples
- Standard (and possible some mandatory) attributes – do we need them?
- Interchange model (with possible fallback solutions)
- Creation of methodology / methodologies
- Extension of framework - how do we do that?
- We need to accommodate unexpected concerns and new stakeholders
- We need to ensure that models are subsequently understandable
- We need to ensure that extension are interoperable
- Maybe create a mapping between NAF and TOGAF.
- Have capabilities for exchange architecture descriptions, - NATO, nations, etc.

7.0 CONCLUSIONS

Obviously this paper does not describe a completed activity, but rather a start of building a coherent architecture for the extended enterprise. This is as an important contribution to architecture governance.

The central repository is important to obtain the goals; - however the chosen architecture tool has a lot of inadequacies, and it is the hope that the future will bring improvements. The use of NAF is seen as the best solution for operational systems used for international operations, but there may be challenges for national requirements outside Defence. Therefore the development of NAF is followed with huge interest, - in particular regarding backward compatibility and the relationship to civilian architecture frameworks.

The pragmatic evolutionary approach is also seen as a necessity at least for the time being; - however, a stricter regime may be enforced in the future. It has already started, because the architects see some value in doing that. It turns out that some low-hanging fruits can be picked up with limited efforts.

The stakeholder driven approach is very important and stakeholder involvement will continue and be intensified in order to reach a more mature state, where architecture is a natural part of any acquisition process. The definition of architecture and the follow-on screening of projects against the architecture is only a beginning.

The international collaboration both for developing architecture frameworks and tools are seen as a key to the future. Also, exchange of architecture descriptions and verification for coalition operations is expected to increase, though some obstacles have been identified.

Finally, it should be mentioned that the most important aspect is the use architecture for practical purposes, showing the benefit for the stakeholders. It is not without a cost, but can we afford to leave it in the long run?

8.0 REFERENCE LIST

[1] [ISO 42010] Systems and software engineering - Architecture description, ISO/IEC/IEEC 42010:2011.

[2] [MODAF1.2] Ministry Of Defence Architecture Framework Version 1.2.004, May 2010, UK MOD. http://www.mod.uk/modaf/

[3] [NAF 3.0] NATO Architecture Framework, Version 3.0, NATO C3 Board October 2007. http://nhqc3s.nato.int

[4] Danish Agency for Digitisation. http://www.digst.dk/servicemenu/English

[5] NATO Interoperability Standards and Profiles (NISP), AdatP-34(G), (AC/322-N(2013)0024-REV1-AS1 Dated 22. March 2013)