Application of the Unified Architecture Framework for the Definition of a Generic System Architecture of a Combat System

Lucio Tirone, Claudia Agostinelli, Paolo Petrinca, Emanuele Guidolotti
BU Systems Engineering
ASTER S.p.A.
Rome, Italy

Lorenzo Fornaro, Manuela Nardini, Simeone Solazzi
Combat Systems Architecture & Validation
LEONARDO COMPANY
Rome, Italy

***Abstract* — The application of the Model Based Systems Engineering approach has become an increasingly necessary tool for an efficient engineering of modern complex systems. However, experience has shown that without a sound model development strategy, shared at corporate level, some of the powerful benefits promised by the MBSE approach can be largely missed, such as for example the reuse of existing model artifacts, or the easier interaction with the system's stakeholders. The work described in the present paper aims at establishing a framework which allows the full exploitation of such benefits, by developing a Generic Systems Architecture of a Combat System, through the application of the Unified Architecture Framework.**

Keywords—architecture framework; system architecture; model based systems engineering;

I. INTRODUCTION

The work described in the present paper is the result of a joint effort with Leonardo and Aster, aimed at the implementation of a Generic System Architecture for the Combat Systems developed by Leonardo for several types of platforms, both land and sea based. Leonardo started working with a MBSE approach for Systems design and development more than 10 years ago [10] but the real challenge of the last years is the necessity to deliver new and complex sytems "faster and better". That means being more efficient in SE activities. The main driver of the work is the need to better exploit some of the powerful benefits promised by the MBSE approach, such as the reuse of existing model artifacts, or the easier interaction with the system's stakeholders. The work is focused on the development of a Generic Systems Architecture of a Combat System, through the application of the Unified Architecture Framework, which establishes the basis for the full exploitation of such benefits.

II. ARCHITECTURE FRAMEWORK APPROACH

A. Architecture Frameworks for Defense Systems

Since the introduction of the concept of Architecture Framework, originated by the work of Zachman in 1987 [1], a rather large number of AFs have been developed by public and private organizations across the world, to better suit the specific modeling needs of each organization. From single companies, up to coalitions of governments (e.g. the NATO), AFs have spread widely, and not only in the defense domain (though it remains the main domain for their application), but also for other types of governmental agencies (such as the Federal Enterprise Architectural Framework, FEAF [2]), or entirely commercial entities (such as the The Open Group Architectural Framework, TOGAF [3]).

The following picture shows the evolution of some of the most known AFs over time, highlighting the many interactions and dependencies between them, as the usage of this methodology is refined through continuous usage by many parties across the engineering community.

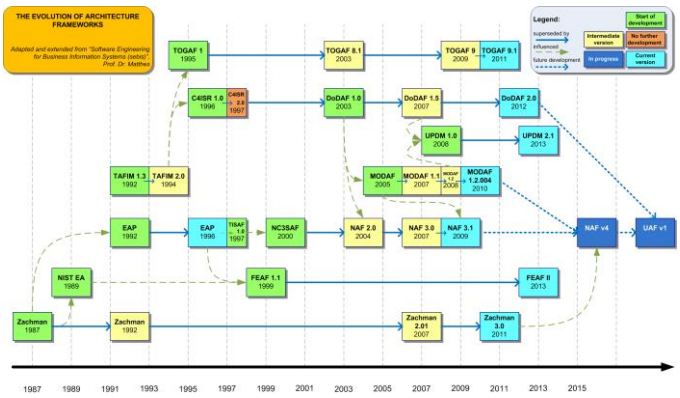

Fig. 1. The evolution of Architecture Frameworks

B. The Unified Architecture Framework

Recently, as can be clearly seen in the previous Fig. 1, a tendency has arisen to merge different Frameworks, rather than creating new ones for the specific needs of an organization, which is mostly due to an increasing need of interoperability between architectures developed by different entities, and thus the derived need of harmonizing the way architectures are developed and described. For example version 4 of the NATO Architecture Framework (NAF [4]) was originally meant to

merge the previous v3 with MODAF [5] and the MODEM [6] methodology. This trend of evolution, has led the Object Management Group, the standardizing body for modeling languages (UML, SysML, BPMN to name the most relevant), to develop the Unified Architecture Framework, or UAF.

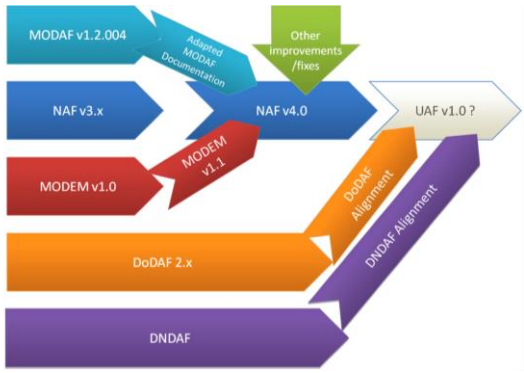

Fig. 2. Merging process of Architecture Frameworks

The Unified Architecture Framework has been created to support a standard representation also for non-defense organizations' architecture descriptions as part of their Systems Engineering (SE) technical processes. UAF supports a standard profile that can be used to implement the UAF in UML/SysML tools.

The Unified Architecture Framework Profile (UAFP [7]) enables the extraction of specified and custom models from an integrated architecture description. The models describe a system from a set of stakeholders' concerns such as security or information through a set of predefined viewpoints and associated views.

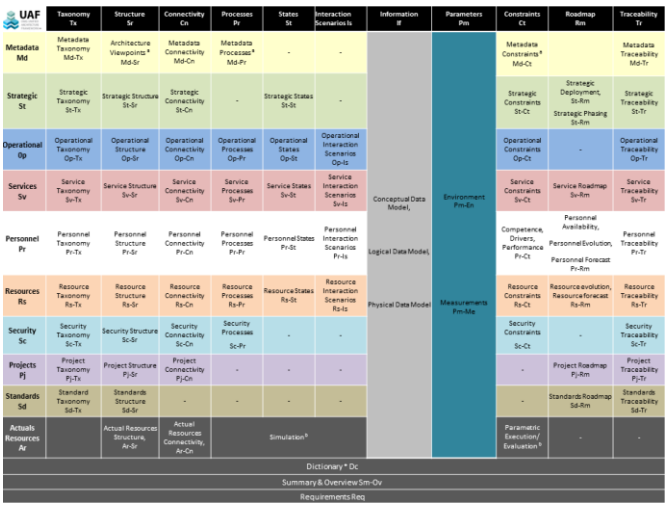

Fig. 3. UAF Matrix View

The UAF metamodel improves the ability to exchange architecture data between related tools which are UML/SysML based and tools that are based on other standards.

The UAF views are classified by types (eg. Taxonomy, Structure, Connectivity etc.) and domains (eg. Metadata, Strategic, Operational etc.); the UAF view matrix is represented in Fig. 3. It specifies the different diagram types across the top and the domains along the side.

C. *Tailoring of the UAF*

The UAF is a very generic framework, suitable for the modeling of any type of entity. As described in the soon to be published ISO/IECIEEE standard 42020 on Architecture Processes [8], the entities which can be subject of an Architecture are several:

- enterprise
- system of systems
- collection of systems
- class of systems
- family of systems
- product line
- individual system
- portion of a system
- product
- service
- individual hardware or software item
- any other entity that is amenable to architectural definition *(eg, data, doctrine, organization, process, method, technique, policy, facilities, etc)*

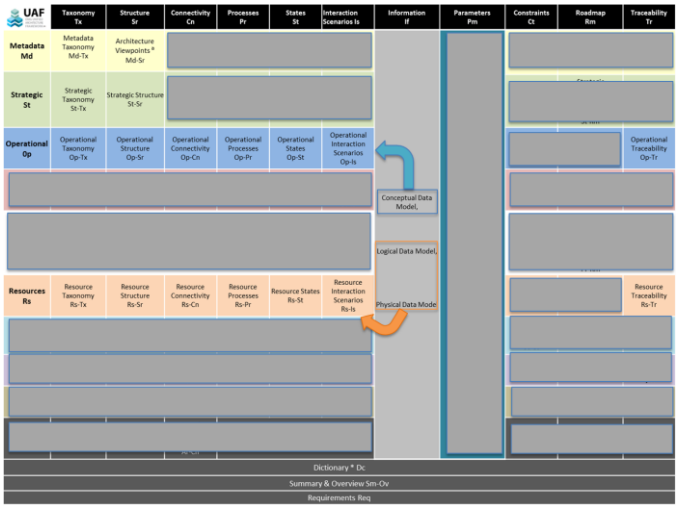

Fig. 4. Tailoring of the UAF Views

For this reason, the choice has been to perform a tailoring on the UAF, in order to render it more fit for the stated purposes of the activity. The tailoring has been twofold: at first in the selection of the relevant views to be employed. The Combat System is definitely a complex system, and a system-of-systems, but still, it is not an enterprise, at least for the purposes of the models to be developed. It is thus an SoS, with human operators considered external to it, and a number of

UAF layers have not been included in this first implementation: the personnel layer, the security layer, the project layer, the actual layer.

Following is the list of Viewpoint domains and the relevant Views that have been employed for the definition of the GSA:

- Metadata, Md: This Viewpoint is related to the Meta-Model upon which the Architecture is based (which results in a tailoring of the full UAF MM, as shown in the next chapter); the Metadata Taxonomy views show the definitions of all elements used within the model, while the Metadata Structure shows the list of applicable Views;
- Strategic, St: The Strategic Taxonomy views describe the hierarchy of Capabilities, while the Strategic Structures show the definition of the System of Interest (SoI), together with its relevant Stakeholders;
- Operational, Op: the next level of abstraction is used to describe the SoI and the external entities, from an Operational viewpoint, which means describing the problem rather than the solution, how the human operators interacting with the SoI perceive it, and interact with it; the Viewpoint is composed of structural views (Op Taxonomy, Op Structure and Op Connectivity), behavioral views (Op Processes, Op States and Op Interaction Scenarios), of the Op Traceability view, showing traceability towards the Strategic layer, and finally of the Conceptual Data Model;
- Resources, Rs: these views show the "solution" to the problem defined above; with the same structure of the Operational views, of which they represent an "implementation": structural views (Res Taxonomy, Res Structure and Res Connectivity), behavioral views (Res Processes, Res States and Res Interaction Scenarios), the Resources Traceability view, showing traceability towards the Operational layer, and finally the Logical and Physical Data Models;
- Dictionary, Dc: this view aims to define all the elements used in an architecture, it contains tables showing the definitions of Terms and Acronyms;
- Requirement, Rq: this view is used to represent requirements, their properties, and relationships between each other and to UAF architectural elements;
- Summary & Overview, Sm-Ov: this view provides executive-level summary information to allow quick reference and comparison among architectural descriptions;

The second level of tailoring of the UAF is related to the Meta-Model, and is fully described in the following Chapter.

III. A Simplified Meta-Model for the Modeling of Combat Systems

As described in the previous chapter, a tailoring of the UAF full Meta-Model (UAF-MM) has been performed, in order to better fit the needs of a SysML model which describes a System (even if a System-of-Systems), rather than an Enterprise. The tailoring can actually be considered a "simplification" of the UAF-MM, and a key reason for adopting it lies in the fact that currently none of the SysML tools available on the market provides a profile implementing the UAF-MM; the full implementation of the UAF-MM has been considered non appropriate for the timeframe allocated to the project, and therefore a "simplification" has been performed. However, this simplification has been realized in a way to minimize as much as possible any impacts on the definitions of the UAF elements, making sure that once a profile becomes available for the tool used to generate the GSA (IBM Rational Rhapsody), a minimal effort will be necessary to the correctly use the model in compliance with both profiles.

For the purposes of the present paper, three packages will be introduced in the following sections, the Strategic, Operational and Resources components of the Simplified UAF Meta-Model.

A. *Strategic Package Meta-Model*

The key element of the Strategic Package is the Capability. As shown in the following Fig. 5, a Capability is defined as "An expression of a system, product, function, or process ability to achieve a specific objective under stated conditions", definition directly derived from the INCOSE SE Handbook [9].

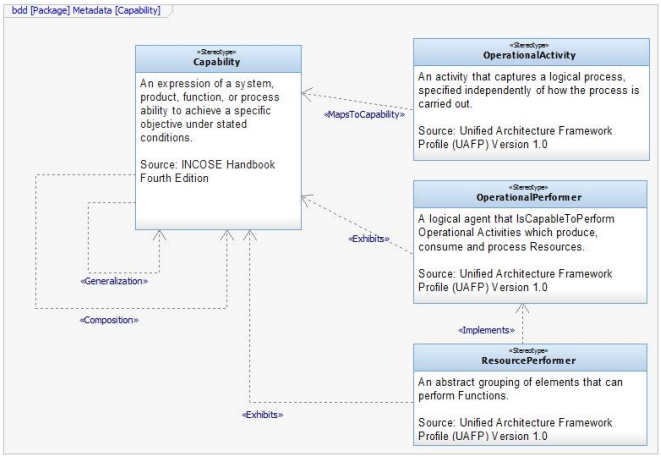

Fig. 5. Definition of Capability

Capabilities represent the highest level functionalities performed by Systems, and are described in the Strategic package in a way that is independent of any specific technology, or solution. In fact, the elements which are shown as "Exhibiting" Capabilities are the Operational Performer, and Resource Performer, which represent respectively the "abstract node" and the "implemented solution" performing the Operational Activities necessary to realize the Capability's objective. The strict separation between the operational and resource layers is essential for the definition of a SysML model with the ambition of being "reusable". Any specific, technology bound implementation of the Capability, would result in model artifacts which are tightly connected with that

specific implementation, and would not be reusable for different scenarios.

The second element included in the Strategic Package is the "Stakeholder". Again following the INCOSE Handbook, a Stakeholder is "A party having a right, share, or claim in a system or in its possession of characteristics that meet that party's needs and expectations":

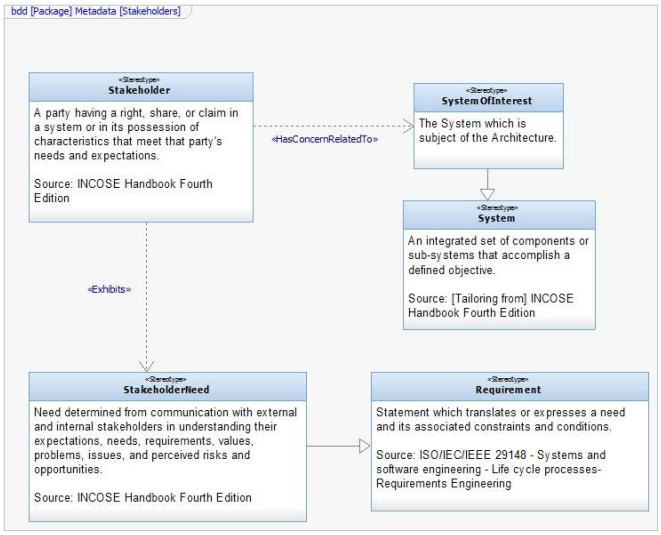

Fig. 6. Definition of Stakeholder

Closely related to the definition of Stakeholder, are those of Stakeholder Need (a specialization of Requirement), and of System of Interest (SoI), which is actually the subject of the SysML model itself. The definition of a specific SoI is exactly what differentiates an Enterprise Model from a System Model, even though they can both be represented using UAF views and Meta-Model.

B. Operational Package Meta-Model

The Meta-Model for the entities included in the Operational Package is less simple than that for the Strategic Package, because behavioral characteristic come in play between the Operational Entities.

The main element of the Operational Package is the Operational Performer. Called "Operational Node" in most of the previous AFs, it is defined in the UAF as "A logical agent that is capable to perform Operational Activities which produce, consume and process Resources". The Operational Performer is thus an abstract entity, considered from a purely operational point of view, able to perform Operational Activities. Different systems, or even humans, can be part of an Op Performer, and on the other hand, a single system might "implement" several different Op Performers.

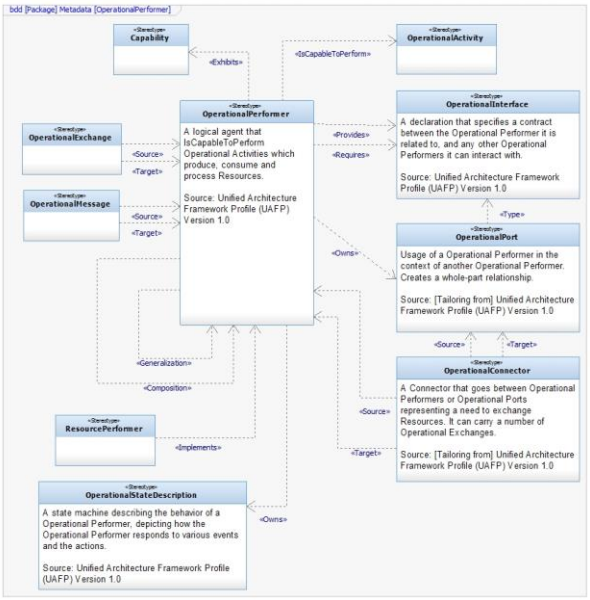

Fig. 7. Definition of Operational Performer

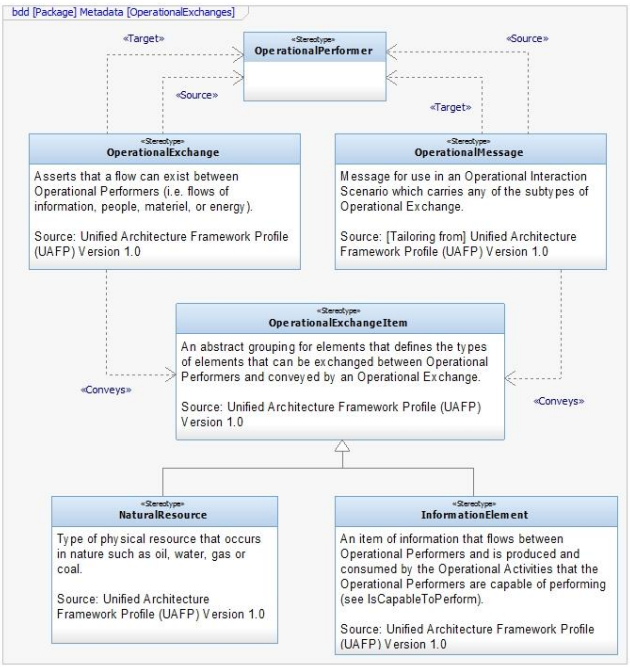

Fig. 8. Definition of Operational Exchanges and Messages

Operational Performers interact among each other exchanging Natural Resources (such as fuel, energy, etc.), or Information Elements (tracks, alarms, video or audio, etc.). These two types of elements are collected under the Operational Exchange Item stereotype, and "conveyed" in two possible ways: through Operational Exchanges (used in SysML Internal Block Diagrams, or Activity Diagrams), or through Operational Messages (used in SysML Sequence Diagrams).

The "system" level approach, not natively included in the UAF, but necessary for the realization of the GSA (as a model representing Combat Systems), requires two new entities to be

added with the tailoring, namely the Actor and Use Case stereotypes. Actor is defined as an Operational Performer which is "external" with respect to the System of Interest, and it is obvious that such an element would not be necessary in an Enterprise approach, where no specific SoI is defined.

On the other hand, a Use Case is defined as a more abstract kind of activity, "realized" by Operational Activities. Actors "participate in" Use Cases, and the typical relations among Use Cases are included, such as Include, Extend and Generalization.

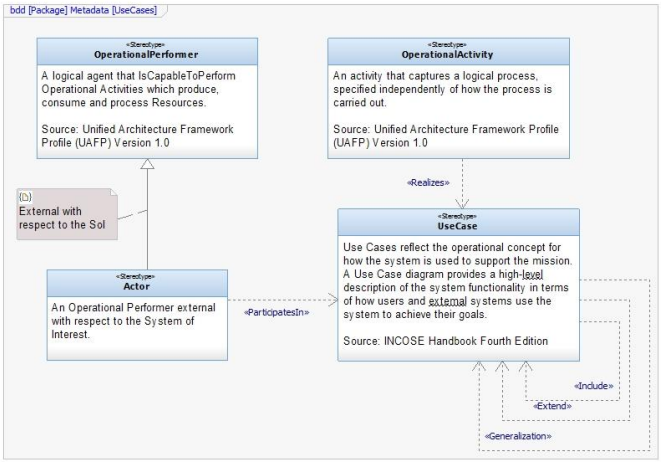

Fig. 9. Definition of Actors and Use Cases

C. Resources Package Meta-Model

The Resources Package contains the "implementation" of what is described in abstract, operational terms, in the Operational Package. It represents the "solution", and no longer the "problem".

The definition of Resource Performer ("An abstract grouping of elements that can perform Functions"), leads to the recognition of Functions as the "implementation" of Operational Activities. So where an Operational Performer executes Operational Activities, the Resource Performer (a System or Component, as will be shown shortly), executes Functions. For usage within the Combat System GSA, the Resource Performer has been specialized in two different entities, namely Systems and Components, as shown in the next figure.

A System is "An integrated set of components or sub-systems that accomplish a defined objective", a definition slightly tailored from the INCOSE Handbook itself (the original definition specifies "elements, subsystems, or assemblies"), and is obviously a composition of other Systems or of Components ("A type of man-made object that contains no human beings", a simple renaming of the UAF "Resource Artifact").

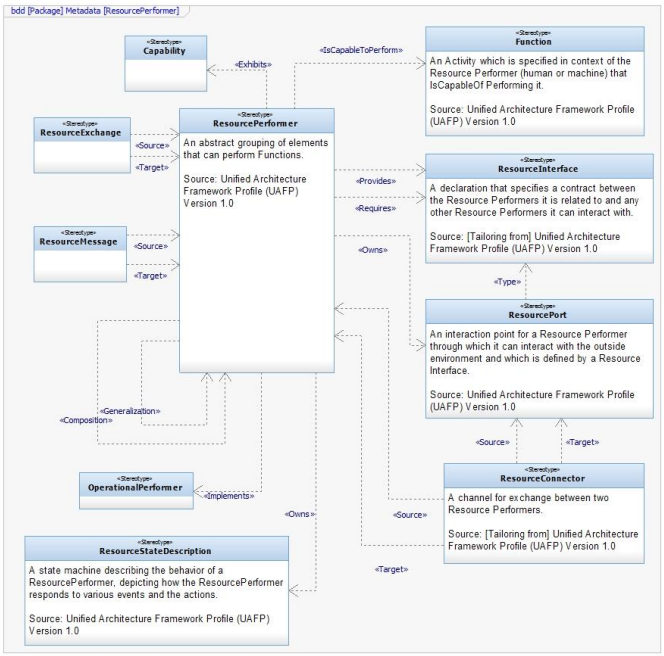

Fig. 10. Definiton of Resource Performer

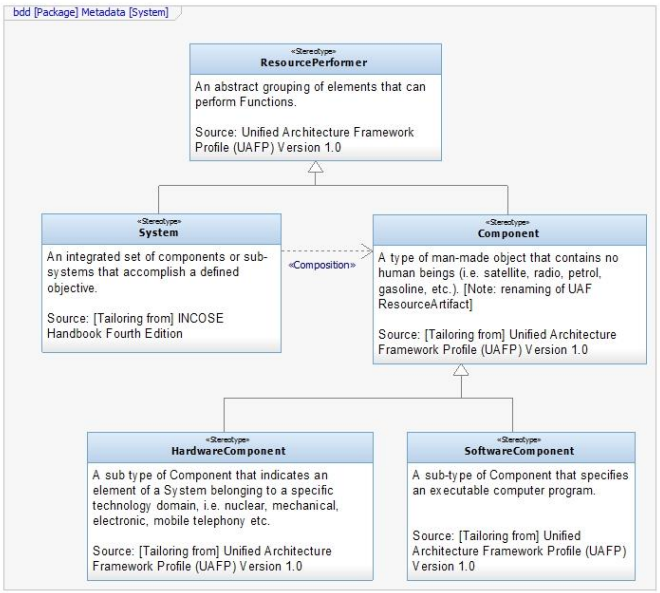

Fig. 11. Definition of System and Component

Systems and Components exchange among themselves Resource Exchange Items (Natural Resources or Data Items), which are "implementations" of the previously defined Operational Exchange Items. Consequently, Data Items represent the "implementation" of the Information Items used in the Operational layer. Resource Exchange Items are themselves "conveyed" by Resource Exchanges (as before, to be used on IBD and Activity Diagrams), and by Resource Messages (to be used on Sequence Diagrams).

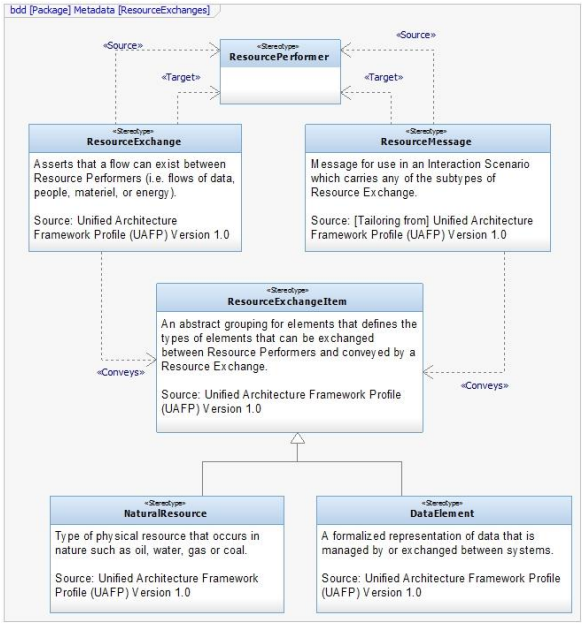

Fig. 12. Definition of Resource Exchanges and Messages

The full list of "implementations", which represent the traceability between the Operational and Resources layers, are shown in the following figure:

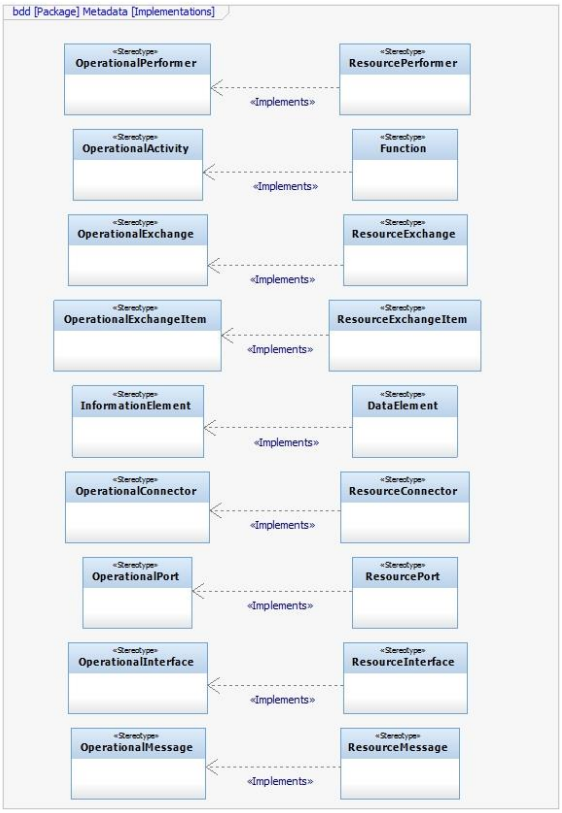

Fig. 13. Implementations between Operational and Resource entities

IV. Generic System Architecture Model Contents

Once the Meta-Model has been defined, it has been applied to generate the whole contents of the various packages of the GSA model. The structure of packages has been arranged as follows:

- Strategic Analysis: includes the system Capabilities, the Stakeholders, and their Needs;
- Operational Analysis: has been split in two separate sub-packages:
  - GSA Operational Analysis: includes the generic Operational Performers, and their behavior
  - Operational Instances: includes the instances of the Operational Performers, and the Use Case analysis
- Resources Analysis: has been split in two separate sub-packages:
  - Logical Architecture: includes the Systems and their behavior
  - Physical Architecture: includes the Components (hardware and software), composing the Product Breakdown Structure (PBS);

A. *Strategic Analysis*

The following highest level Capabilities are "Exhibited" by the System-of-Interest, that is, a generic Combat System:

- Surveillance
- Combat Management
- Threat Management
- Unmanned Vehicles Management
- Environmental Data Management
- Air Traffic Control
- Navigation Support
- Communications
- Own Ship Identification

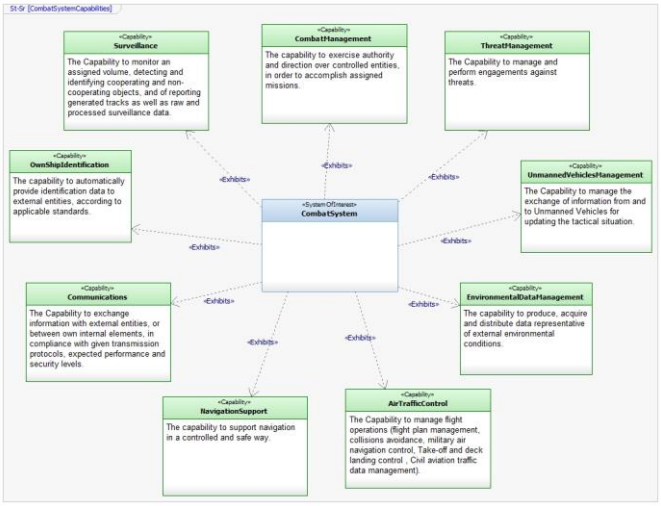

Fig. 14. Highest level Combat System Capabilities

As a representative element of this package, the Combat Management high level Capability has been decomposed in the following way:

- Command and Control
  - Tactical Situation Management
  - Resource Management
  - Threat Evaluation
  - Weapon Assignment
  - Battle Damage Assessment
  - Mission Planning
  - Data Recording
  - Aircraft Control
  - Onboard Training
  - Platform Manoeuvre Recommendation

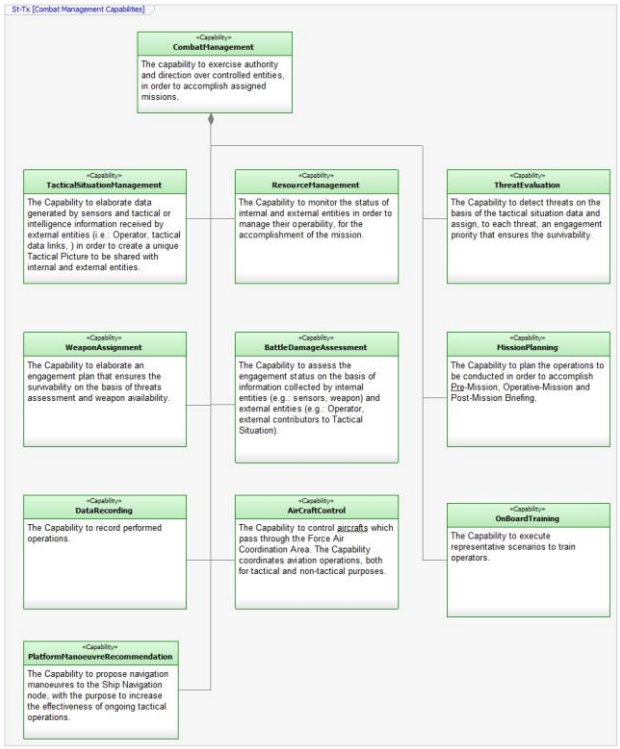

Fig. 15. Combat Management Capabilities

B. *Operational Analysis*

1) GSA Operational Analysis

The Operational Analysis for the Combat System starts with a Generic System Approach, which consist in a definition of the generic Operational Performers that can be considered representative of any instantiation of the system. The GSA definition is meant to be "inclusive", representing elements that belong to all the possible systems developed by Leonardo. Any instantiation of such system shall be derived from the GSA only by removal of not necessary elements, never by adding elements which are not present in the GSA. The behavioral part of the GSA elements will be as well representative of common patterns, generic enough to represent classes of behavior. The instantiation of the Operational Performers and of their behavior, categorized through a Use Case approach, is demanded to the following package "Operational Instances".

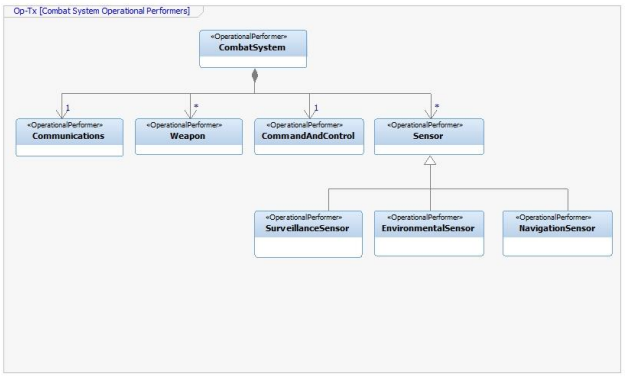

Fig. 16. Combat System GSA Operational Performers

The lower level generic Operational Performers that compose the high level Combat System generic Operational Performer has been identified in the following way:

- Communications
- Weapon
- Command and Control
- Sensor (Surveillance Sensor, Environmental Sensor, Navigation Sensor)

In turn, for each of these nodes an Operational Taxonomy diagram has been realized to represent how they have been decomposed. The following figure shows the Op-Tx diagram for the Surveillance Sensor Operational Performer in which are shown: the Operational Performers that compose the Surveillance Sensor, the defined data type, the Attributes and the activities performed by the Operational Performers. In particular, the Surveillance Sensor is composed of two Operational Performers:

- Acquisition
- Processing

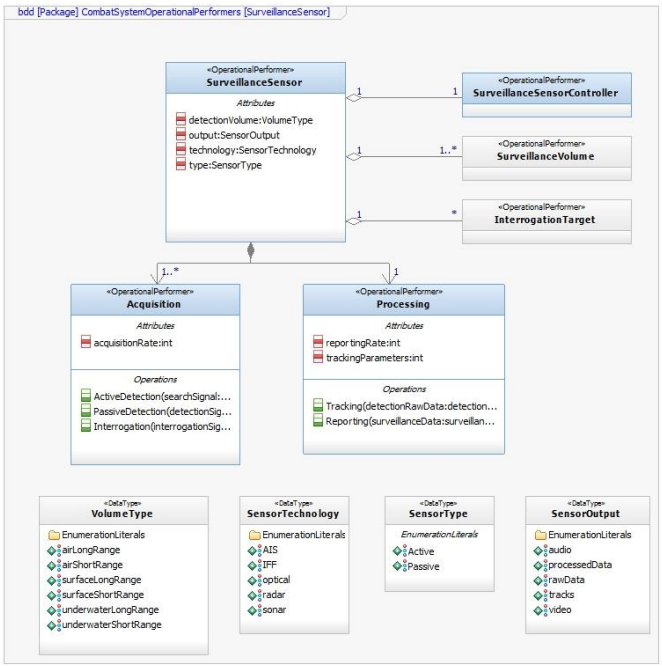

Fig. 17. Definition of Surveillance Sensor

The following figures show the GSA Activity and Sequence Diagrams for the Surveillance Sensor Operational Performer.

The GSA Activity Diagram represents workflows of activities (transformation of inputs to outputs) through a controlled sequence of actions. Each swim lane represents an Operational Performer (internal or external to the System of Interest) and the arrow represents the Operational Exchanges flowing between the Operational Performers.

The GSA Activity Diagram for the Surveillance Sensors represents the scenario in which a generic Surveillance Sensor composed by the two Operational Performers Acquisition and Processing is activated by an interrogation or by an active sensor detection. Depending on the case, the Acquisition and Processing perform several tasks and interact with the external Operational Performer outside the System of Interest to provide the detected data to the Surveillance Sensor Controller.

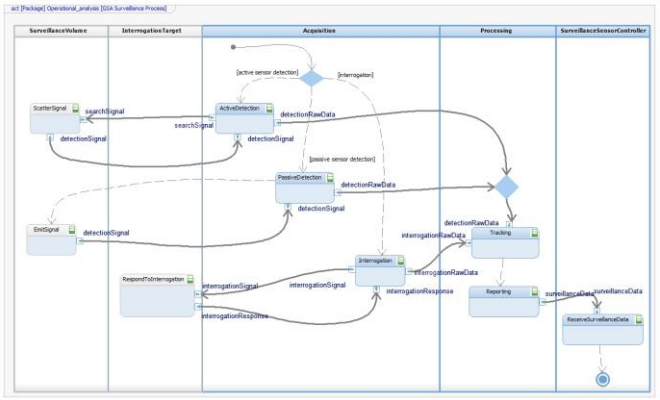

Fig. 18. GSA Activity Diagram for the Surveillance Sensors

The GSA Sequence Diagram allows the tracing of actions in a scenario or critical sequence of operative events. Each lifeline on the top of diagram is associated with an Operational Performer (again, internal or external to the System of Interest). This diagram describes the temporal sequence of Operational Messages between the Operational Performers.

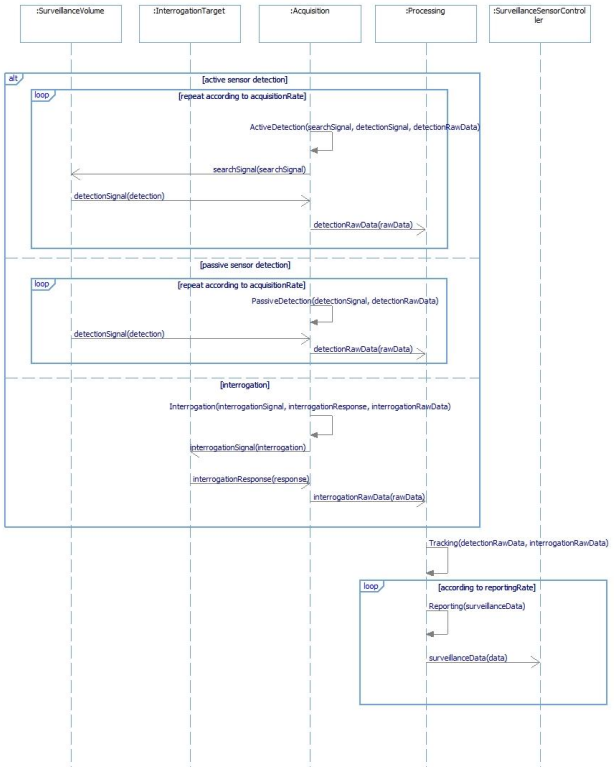

Fig. 19. GSA Sequence Diagram for the Surveillance Sensors

The GSA Sequence Diagram for the Surveillance Sensors represents the same scenario described by the GSA Activity Diagram. The Operational Performer involved are the same, the only difference is the highlight on the sequence of messages exchanged between them.

2) Operational Instantiations

The following Fig. 20 shows the Instantiation of GSA Surveillance Sensor Operational Performers represented by an Operational Taxonomy Diagram.

Each instance in the diagram represents a real sensor, however considered only from the operational point of view. This instance has a relation of Generalization with the GSA Surveillance Sensor generic Operational Performer. As can be seen from the figure, the instance sensors have been grouped in four categories:

- Active Above Water Sensors
- Passive Above Water Sensors
- Active Below Water Sensors
- Passive Below Water Sensors

The purpose of Operational Instances, in this specific case of surveillance sensors, is to represent the "surveillance component" of a real world sensor, abstracted from a purely operational point of view. For example the MFR_Surveillance Operational Performer, represents the surveillance component of a Multi Function Radar. This element derives from the generic Surveillance Sensor, and so inherits its properties, such as the fact of being composed by an Acquisition component (the Antenna) and a Processing component (the Extractor), or the performing of detection in the assigned volume with search signals, and the delivery of extracted radar tracks to the external Sensor Controller.

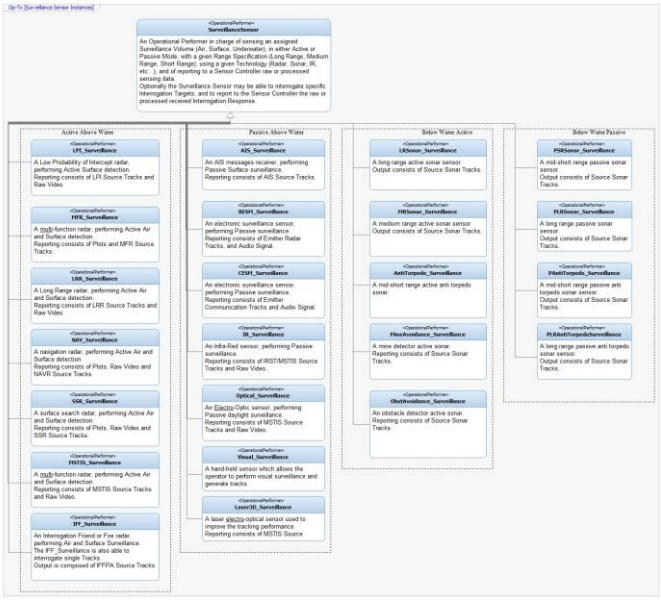

Fig. 20. Surveillance Sensor operational instantiations

The description of the MFR_Surveillance sensor however remains operational, in the sense that no specific technology, or system is specified at this moment. The behavior of this element is represented through a "Conceptual level Data Model", that is, a "radar signal" is exchanged with the Surveillance Volume, "tracks" and "raw video" are exchanged with the MFR_Controller Operational Performer instance.

V. Conclusions

The work described in the present paper has shown the first steps for the implementation of a Generic System Architecture for Combat Systems, which can represent a synthesis of the many architectures produced by the Defense Systems Engineering Unit of Leonardo over time. The adoption of a Generic System Architecture will allow the reuse of components described in different instantiations of the Combat Systems. In particular it will be possible to effectively reuse the capabilities and operational descriptions of previously modeled systems and components, or the message catalog, and a reference template will be available for the definition of new systems.

References

[1] Zachman, J.A. "A Framework for Information Systems Architecture." IBM Systems Journal, Volume 26, Number 3, 1987.

[2] The Chief Information Officers Council A04, "Federal Enterprise Architecture Framework Version 1.1", September 1999.

[3] TOGAF (The Open Group Architecture Framework), www.opengroup.org/togaf

[4] AC/322-D(2007)0048 NATO Architecture Framework V3.

[5] UK Ministry Of Defense, "MOD Architecture Framework (MODAF)", v1.2, 2012

[6] UK Ministry Of Defense, "MODAF Ontological Data Exchange Mechanism, MODEM"

[7] OMG, Unified Architecture Framework Profile (UAFP) – Version 1.0 – FTF Beta 1, 2016.

[8] ISO/IEC/IEEE DIS 42020, Enterprise, systems and software -- Architecture processes, unpublished

[9] INCOSE, "Systems Engineering Handbook, A Guide For System Life Cycle Processes And Activities", Fourth edition, INCOSE-TP-2003-002-04, 2015

[10] Ciambra, F. and Nardini, M. 2004. Naval Combat System Design: System Engineering approach and complexity management. In Proceedings of the Fourteenth Annual International Symposium of the International Council on Systems Engineering (Toulouse). France: INCOSE.