Master Thesis

Aeronautical Engineering

A preliminary study of Models for Manufacturing to define aeronautical assembly lines in 3DExperience

Author: Guillermo Álvarez Murcia

Tutor: Domingo Morales Palma

Dpto. Ingeniería Mecánica y Fabricación
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2021

Master Thesis
Aeronautical Engineering

A preliminary study of Models for Manufacturing to define aeronautical assembly lines in 3DExperience

Author:
Guillermo Álvarez Murcia

Tutor:
Domingo Morales Palma
Profesor Contratado Doctor

Dpto. Ingeniería Mecánica y Fabricación
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2021

Master Thesis: A preliminary study of Models for Manufacturing to define aeronautical assembly lines in 3DExperience

Autor: Guillermo Álvarez Murcia

Tutor: Domingo Morales Palma

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2021

El Secretario del Tribunal

For my friends and family,

Acknowledgements

I want to thank to everyone who have made this project possible, directly or not.

First, I would like to thank Domingo Morales Palma, for its mentoring, supervision, patience, and knowledge, for trusting me and giving me the chance of doing this, and his big help even in these extraordinary and difficult circumstances.

I want to thank my family too, my parents and sister, for cheering me up during the whole project development, encourage me so as to never give up, and hugely contribute to being myself nowadays.

Finally, I want to sincerely thank my friends, for their company and support whenever was needed, even during difficult times.

Thanks to all.

Guillermo Álvarez Murcia

Sevilla, 2021

Abstract

Models for Manufacturing (MfM) is a preliminary approach to a methodology that aims to provide a set of processes, methods and associated tools to help the engineers to support the discipline of manufacturing in a model-based context. It is a proposal from a multidisciplinary team from the University of Seville in collaboration with professionals from the aeronautical sector. MfM is currently in its early stages of development.

The MfM methodology relays on the development of a reference framework, the 3LM (3-Layers Model: Data, Ontology and Service layers), based on the definition of a manufacturing ontology and enabling simulation, behaviors and analytical capabilities, capitalizing the company knowledge. The Ontology layer is the core of the model. It holds all the company processes and scope, data and semantic models, and the associated simulation or behavior requirements. At the beginning of this work there were proposals for the Scope and Data models but not for the Behavior and Semantic models.

This project aims to collaborate in the development of the MfM methodology. The main contributions of this work are: (1) a knowledge representation scheme is proposed to model the behaviour of the system under study; (2) the application of the methodology to the aeronautical assembly line design process is analysed; and (3) a model of the assembly process of an aircraft wing box is built in 3DExperience, in order to be used in future performance tests of the MfM methodology.

Resumen

Models for Manufacturing (MfM) es una aproximación preliminar hacia una metodología que pretende suministrar una serie de procesos, métodos y herramientas asociadas para ayudar a los ingenieros a cimentar la disciplina de la fabricación en un contexto basado en modelos. Es una propuesta de un equipo multidisciplinar de la Universidad de Sevila, en colaboración con profesionales del sector aeronáutico. El MfM está actualmente en fases tempranas de desarrollo.

La metodología MfM recae en el desarrollo de un marco de referencia, el 3LM (3-Layers Model: capas Data, Ontology y Service), basado en la definición de una ontología de fabricación y habilitando las capacidades de simulación, comportamiento y análisis, priorizando el conocimiento de la empresa. La capa de Ontología es el núcleo del modelo. Contiene todos los procesos de la compañía y su alcance, los modelos de datos y semántica, y las simulaciones asociadas y requisitos de comportamiento. Antes de la realización de este trabajo existían propuestas para los modelos Scope y Data, pero no para los Behavior y Semantic.

Este proyecto pretende colaborar en el desarrollo de la metodología MfM. Las principales contribuciones en este trabajo son: (1) se propone un esquema de representación de conocimiento para modelar el comportamiento del sistema bajo estudio; (2) se analiza la aplicación de la metodología al proceso de diseño de una línea de ensamblaje aeronáutico, y (3) se construye un modelo del proceso de ensamblaje de un cajón de ala de una aeronave en 3DExperience, para que pueda ser usado en futuras pruebas de rendimiento de la metodología MfM.

MOTIVATION

Scientists discover the world that exists; engineers create the world that never was.

- Theodore von Karman -

The present project is motivated due to my personal interest in computer modelling techniques focused on industrial applications, specifically aiming to the manufacturing side of a typical aeronautical product lifecycle. My previous work on computer aided technologies (CAx), collected in a bachelor's degree Final Project [1], first introduced me to this field of study, and contributed to enlarge my interest in such tools and methodologies. Thus, the next logical step was to dig into how models are made, from a much more abstract approach, leaving behind software-dependent aspects. This leap, together with an increasing interest among the Companies in having modelling methodologies not limited by software limitations, have greatly encouraged this work.

Computer aided tools first changed workflow inside industrial plants worldwide. New functional design methods allow engineers create, modify, check, and approve concepts on-the-fly, anywhere at anytime, reaching collaboration levels as never before. Similarly, new simulation and process planning tools have managed to foresee different possible industrial scenarios, allowing engineers to be ready for any potential issues as well as optimize plants layouts, workload and line balancing easily, with little or no economic impact on the Company.

Currently, the 3D definition of the product using PLM, CAx tools and MBSE models is a mainly focused on the Functional Design processes. However, in the manufacturing side of the lifecycle, despite the use of ERP, PLM, MES and CAx tools, the achieved improvement is far from what has been accomplished in the previous field. This situation motivates the appearance of a methodology capable of modelling scenarios from an industrial and manufacturing-centered point of view.

Models for Manufacturing (MfM) is a new approach proposed by the tutor of this project and his collaborators to apply Model-based Systems Engineering concepts to Manufacturing. The methodology under development is supported by a 3-layer framework (3LM) and simple and user-friendly software tools.

The motivation for this work is to further extend the so called 3LM framework, focusing on the Ontology layer. A preliminary methodology for this layer will be presented, as well as the different issues found during its development. Interaction between all three different layers will be shown, including instancing specific engineering scenarios from the first, abstract models. Finally, a manufacturing use case will be presented, applying such methodology to a simple example involving a wing box assembly, using both open source and top tier PLM software (ARAS and 3DExperience, respectively). A future development in Data layer would translate the Ontology knowledge into full developed CAx models, independently of the commercial software used.

Ontology is the term used for naming shared understanding of some domain of interest. Ontology modelling, also known as Ontology building, is a really popular subject of study nowadays, and tries to define a general framework in which Ontology models can be built (known as a meta-model).

This work aims to expand the manufacturing-focused approach of MBSE, known as MfM, more specifically making use of the 3LM method, shown later. This will provide a methodology able to be applied during the several phases along the aerospace product lifecycle different from Functional Design: Industrial Design, Serial Production/Manufacturing, and In-Service Support. On a typical lifecycle of a commercial aircraft, it can be clearly seen that, despite the immense economic and human effort that Functional Design implies, it involves only around 10 years of the whole lifecycle, which is less than a quarter of the total length. Production and In-Service support take a much longer period, over 40 years, covering both Functional and Services Design, manufacturing, assembly, and management of the supply chain, MRO (Maintenance, Repair and Overhaul) and product services activities.

Considering the aerospace lifecycle phases, four main software systems are used to generate, manage, and exploit the aircraft related data or information: Computer Aided applications (CAx), Product Lifecycle Management (PLM), Enterprise Resource Planning (ERP) and Manufacturing Execution System (MES). This diverse information is dispersed through several databases from different software systems, which are operated along the lifecycle. Every software system considered has a traditional structure: Database, Data model and Service. Databases are usually provided by a vendor (Oracle, MySQL, and others). Data model, the core of the system is defined and developed by the provider with little or no user influence. Service is the mathematical, simulations, behaviors, or business functions to apply. Even though each system ensures the consistency of its data, the approach fails to ensure a data model consistency between systems.

Manufacturing is a large and wide part of the lifecycle and covers several different stages with similar models. Nowadays many different software applications are running with interfaces between them, without a full common model. Data continuity cannot be ensured and is partially devoted to interfaces between the applications, simulation is done under far from desirable circumstances and consistency with the Company processes is achieved via customization, legacy software add-ons or Excel spreadsheets.

The proposed solution by several authors is the 3LM framework and the MfM methodology. Creating a common ontology is the way to define, manage and maintain the Company knowledge. It has already been applied with moderated success to a few specific industrial scenarios and has produced promising results despite being on a preliminary phase of development. These will be seen more in depth in the following lines.

OBJECTIVES

This work aims to collaborate in the development of the MfM methodology and to carry out an application focused on the design of assembly lines for aeronautical products. It tries to be an ultimate demonstration of the viability and robustness of the methodology in terms of Ontology building and its practical applicability to complex real-world scenarios.

In order to do so, the following objectives have been established:

- Develop a solid methodology to create Ontologies from scratch, as part of the Ontology layer inside 3LM method. In particular, it is intended to develop a knowledge representation scheme for the Behavior model, not yet implemented in the MfM methodology.
- Develop an application of the MfM methodology for the aeronautical assembly line design process. The application will be implemented through simple models, easy to understand and use, made with open-source and user-friendly software.
- Prove the model practical application instancing a specific aeronautical use case. This would be done in two complexity levels, a simple case instanced directly on the model structure, and a more complex one using the commercial collaborative platform 3DExperience, as a result of a hypothetical application of the fully developed methodology.

Table of contents

LIST OF FIGURES

1 INTRODUCTION

Manufacturing is more than just putting parts together. It's coming up with ideas, testing principles and perfecting the engineering as well as final assembly.

- James Dyson -

This chapter introduces the Model-Based Systems Engineering (MBSE) concept, and its implications in the industry. Besides, the Ontology concept and a brief state of art are developed, in order to provide some context in which Models for Manufacturing (MfM) is originated. This methodology is then developed, focusing on the three-layer model (3LM) and its different components. Finally, prior 3LM approaches are acknowkedged.

1.1 Model-Based Systems Engineering (MBSE)

According to Ramos et al. [2], Model-Based Systems Engineering "is an emerging approach in the Systems Engineering (SE) field and can be described as the formalized application of modeling principles, methods, languages, and tools to the entire lifecycle of large, complex, interdisciplinary, sociotechnical systems." A simpler definition is provided by Mellor et al. [3] as "...is simply the notion that we can construct a model of a system that we can transform into the real thing.". This model-centered approach, which main asset is (usually) a 3D model of the system being developed, contrasts with the traditional document-based methodology. This paradigm shift in how modelling is being made was possible thanks to the emergence and quick improvement of computers. MBSE is currently being applied to several engineering disciplines, from mechanical to electrical, and specially to complex, multidisciplinary projects, such as those accomplished in automotive or aeronautical industries.

Ramos et al. [2] also assures that "In the next decade, it is expected that MBSE will play an increasing role in the practice of SE and that will extend its application modeling domains beyond hardware and software systems, including social, economical, environmental, and human-performance components."

MBSE is a methodology that has got more and more important over the past decade and continues to be improved nowadays. One of the main goals of MBSE is to substitute the classic 3D-centric approach and document-oriented information in favor of a simulated model-oriented definition that has several advantages:

• The model is the core of the development, in terms of requisites, design, and manufacturing.

• Ability to manage complexity and to capture knowledge.

• Analysis and trade-off and early detection of issues.

• Keep consistency between requisites along the lifecycle.

• Allow flexibility when changes appear.

All in all, it could be said that this model-based approach, unlike the document-based one, allows a much higher level of communication and collaboration between stakeholders and team members, improves design precision and integrity avoiding potential data loss, grants better information traceability, and greatly reduces development risks.

Hence, MBSE is an attempt to store the Company knowledge about a specific project into a model, rather than documents, with the cited benefits that this implies. The main disadvantages of this concept are that models are usually developed within a specific software framework, and thus are limited and dependent to the software provider, and that they are also usually made individually for a specific project, not being easily applied to other projects of similar areas. These issues have caused the apparition of the three-layer model (3LM) concept among Models for Manufacturing, which will be detailed later.

1.2 Ontologies

In computer science and information science, an ontology encompasses a representation, formal naming and definition of the categories, properties and relations between the concepts, data and entities that substantiate one, many or all domains of discourse. More simply, an ontology is a way of showing the properties of a subject area and how they are related, by defining a set of concepts and categories that represent the subject.

Ontologies are, thus, in the core of MBSE methodology. According to Uschold and Gruninger [4], ontology is the term used to refer to the shared understanding of some domain of interest which embodies some sort of world view with respect to the given domain.

Every academic discipline or field creates ontologies to limit its complexity and organize data into information and knowledge. New ontologies help to improve problem solving within that domain.

Ontology model development is today a global research topic and ontology engineering (also known as ontology building) refers to the set of tasks related to the ontology development process and the ontology lifecycle, the methods and methodologies for building ontologies, and the tool suites and languages that support them.

It aims to make explicit the knowledge contained in software applications, and organizational procedures for a particular domain. Ontology engineering offers a direction for overcoming semantic obstacles, such as those related to the definitions of business terms and software classes.

In order to assist in the creation, modification or manipulation of ontologies, specific applications, known as ontology editors, have been developed. They commonly use one or several ontology languages, such as OWL (Ontology Web Language).

Cited languages are not always intuitive and easy to work with form scratch. So as to be able to fulfil this project, other applications have been used to make the modelling, such as RAMUS, for IDEF0 diagrams, or CMAP Tools for concept maps. This model will then be converted into an ontology language, such as the previously stated OWL. Cited halfway software tools will be treated in depth in the following pages, looking into both functioning and worthiness points of view.

1.3 MBSE Initiatives for manufacturing

MBSE has been globally accepted by the aerospace and automotive industry during the last few years, with lots of development and deployment in the Functional Design processes, specially emphasized in the area of systems design.

Several research, developments, deployments, and projects has been conducted using MBSE, but only recently the interest is also being redirected to manufacturing. Industrial Design of the product, manufacturing and assembly, balancing lines, resources, configuration and change management, and many other tasks performed during the serial production phase of the lifecycle are now taking the attention of the researchers. The following lines collect some of the first initiatives for such application.

Bergenthal [5] defines MBE (Model Based Engineering) in the Model Based Engineering final report for US NDIA (National Defense Industrial Association): "an approach to engineering that uses models as an integral part of the technical baseline that includes the requirements, analysis, design, implementation, and verification of a capability, system, and product throughout the lifecycle ", already including the manufacturing side of the lifecycle into the MBSE concept.

Friedenthal et al. [6] proposed a 2010 status and a 2020 vision on MBSE. Some topics selected for the 2020 vision are applied to manufacturing:

• Extends to domains beyond engineering to support complex areas.

• Enable the engineer to focus on abstract modeling of the user domain.

• Modeling standards supporting high fidelity simulation and real representations.

• Extensive reuse of model libraries, taxonomies, and design patterns.

• Standards supporting integration and management across a distributed repository.

Kulvatunyou et al. [7] present several ontologies for industrial problems that have been a topic of research for several years, most of the projects in the EU Horizon 2020 program have adopted ontology as a component and similarly, in the US NIST (National Institute of Standards and Technology), manufacturing projects also have ontology as a component. Actually, it reinforces the concept of commonality between the ontologies, long term interoperability between the different engineering, manufacturing, and supply chain disciplines.

NIST [8] organized a workshop to explore the idea of a framework for curating ontologies, an IOF (Industrial Ontologies Foundry). The goal for the workshop was to identify industry needs, to develop consensus and to identify the issues that need to be addressed to move forward. Workshop participants reported the main reason in seeing an industrial ontology foundry is interoperability, information linking, and formalization of requirements through information constraints, incorporation of business process aspects, and quality and traceability.

Several authors are researching on the development and deployment of MBSE methodologies and tools in manufacturing. Aspects like process planning, human resources, robotics, IoT (Internet of Things) among others are recently research topics.

1.4 Models for Manufacturing (MfM) Methodology

1.4.1 3-Layers Model (3LM)

Model-Based for Manufacturing methodology proposed is based on a 3-Layers Model, as shown in Figure 1. The 3LM ensures the independence between layers, maintaining both Data Layer and Ontology Layer isolated. This ensures the definition of the Ontology, the knowledge of the Company, is being made without interacting with Data and Services layer. Therefore, the 3LM decouples the traditional system developed by the software vendors allowing users to change software providers inside the Service Layer easily. All the Company knowledge can be safely stored and used no matter which software is being used, granting a huge flexibility and interoperability to the whole model.

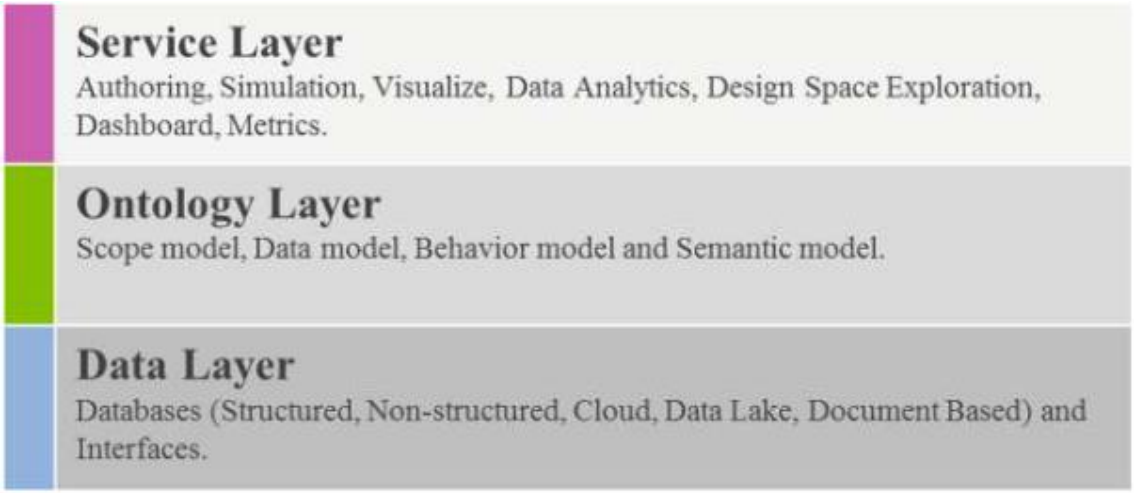


Figure 1. 3-Layers Model (3LM)

The bottom layer, Data layer, collect all the databases and interfaces: legacy databases from the legacy software, databases from the commercial software applications, clouds, and many others. Included in the Data layer are those databases to hold the information instanced using Ontology layer.

The central layer, the Ontology layer, is the core of the model. It holds all the Company processes and scope, data and semantic models, and the associated simulation or behavior requirements. Given its crucial importance, this work will be focused on the development of such layer, digging into the definition of every of its components, and leaving both Data and Service layer to future research. These Ontology building tasks will be carried out using the Ontology editor apps introduced before in Section 1.2.

The top layer, Service layer, holds the software services, such as authoring and simulation tools, visualizers, data analytics and dashboard and space design exploration tools. Services are used thanks to information stored in the Data layer, instanced through the Ontology layer.

1.4.2 Ontology layer

Modelling what a Company knows about any subject is not an easy task, due to its enormous complexity and degree of abstraction. In order to properly store knowledge, four main components are going to be created within the Ontology model: Scope model, which aims to define the Ontology framework and its rules; Data model, which collects all the different concepts known by the Company, as well as how these concepts are connected between each other; Behavior model, giving concepts and relations its dynamic character and evolution in time, and Semantic model, so as to ensure a common glossary of technical definitions avoiding misunderstandings caused by polysemic words and different interpretations of language.

1.4.3 Scope model

As has been stated before, the Ontology layer stores all the Company knowledge in a given field of study, and it is what adds value to the whole model. In order to build an Ontology about any domain, the very first step is to clearly define its scope, i.e., to decide which are going to be the limits and degree of detail for the contents of the Ontology. This ensures that every stakeholder works within a common framework when carrying out the Ontology engineering process and avoids running into a common issue known as *feature creep*, which would be best called in this case *detail creep*.

In this project, the scope model definition has been made using IDEF0 diagrams. This kind of representation allows a simple, clean model, easy to understand at first glance, which has been proved very useful when making changes and powerful enough for a preliminar study in Ontology building techniques. An IDEF0 diagram is shown in Figure 2 for clarification.

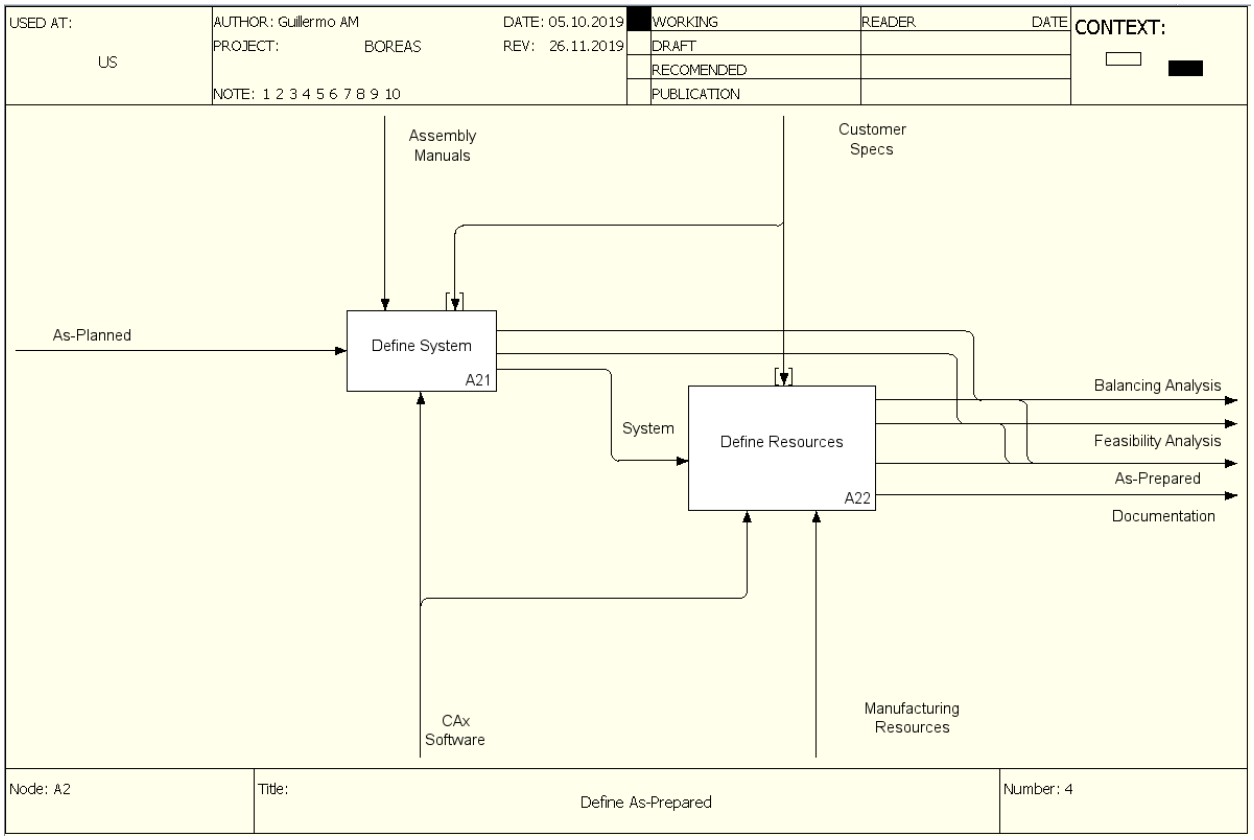


Figure 2. IDEF0 diagram example

The chosen Ontology editor to create such diagrams is RAMUS, in its educational version. The main reasons for this choice are its user-friendly, easy to exploit behavior, the ability to export these diagrams into IDL(Interactive Data Language) code, which is a text format. These text files would then be converted and interpreted by a PLM software, Aras for instance. Once the PLM model is created, it would be the starting point so as to develop Data model, via OWL exportation.

As said before, RAMUS is good enough for a first approach of how to build an Ontology for an academical exercise, being this and next editor tools shown likely to change for more advanced and complex ones if trying to manage a real scenario. IDEF0 diagrams and RAMUS use will be explained in detail in its own chapter.

It is worth remarking the highly iterative character of the scope definition process. The scope model has suffered several major modifications, from its very first conception to its final configuration. It has been involved in a continuous improvement process, always trying to best capture details and issues which are needed to be considered. It has been going through these iterations when having simple and easily understandable models made with a manageable app has shown its huge value.

1.4.4 Data model

Next step in building any Ontology is creating what is called the Data model (do not mistake for Data layer). Once the scope of the Ontology is completely defined, the process of storing the information within the Ontology itself can begin. Usually, the most common technique for such task is via graphical representations, i. e., making use of graph theory, specifically using concept maps. This way, different concepts are stored inside shapes, and then relations between them are added using arrows connecting cited shapes. Usually, a connector (commonly a verb) is placed near the arrows to give more information about the relations' nature. For this representation to be effective, a prior meaning code of shapes, colors and arrows must be specified. Further explanation of concept maps and meaning code decisions will be presented on following pages.

This work first used CMap tools as its editor app for carrying out the Data model definition. As explained previously with RAMUS, the reasons behind this choice were its simplicity, ease for interconnection and its export/import capability in text format. However, although proved very useful in presenting final results, making diagrams neater and more visually attractive, CMap tools really lacks an efficient way to make important changes in the Data model. As can easily be foreseen, lots of iteration in scope model came unavoidably with a great amount of changes in Data model.

So as to speed up the iteration process, another editor app was considered. Instead of making use of a visual GUI, based on drag and drop mechanisms, such as CMap tools, DOT is a text-based graph description language . Just by typing some simple scripts, as could happen when programming any auxiliar gizmo, DOT is able to compile concept maps with ease, allowing even some customization for shapes, colors, and arrows behavior. Due to its versatility and quick response for modifications, it is worth a try.

Concept maps using both apps are shown in Figure 3.

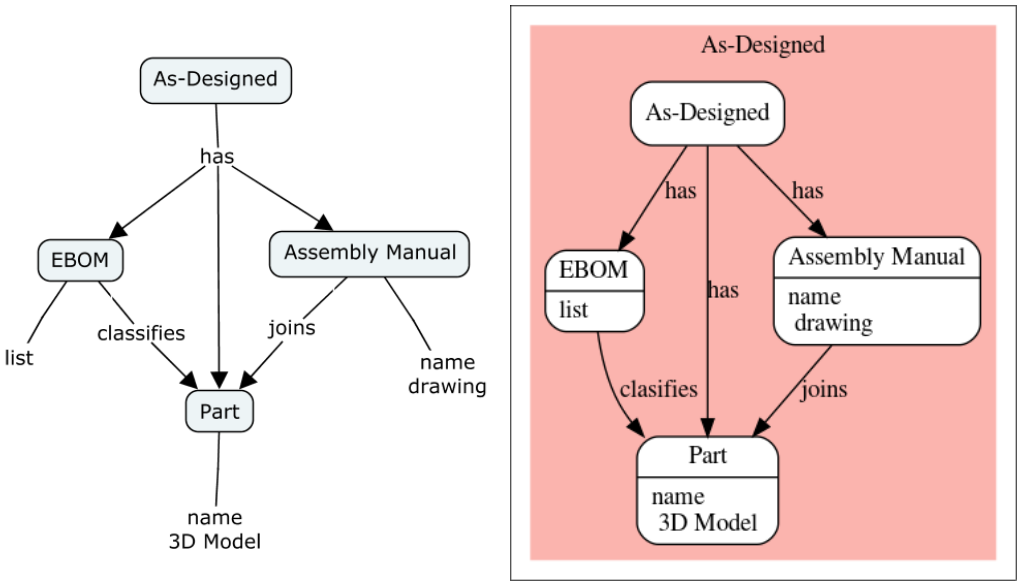


Figure 3. Concept maps example made using CMap tools (left) and DOT programming (right)

In a similar way as what has already been stated in scope model, it can be noted the large number of iterations that have been necessary before reaching the final configuration for the Data model. Again, this has been made to ensure that all possible features and potential issues are being considered, and so the model is as robust as possible.

Once Data model is first created (using some objects from the Scope Model), it can be developed in more detail. It is said that the Data model is being enriched with new information, adding it as much as desired, taking into account that the whole model should not encompass more than what was previously agreed on the scope model.

This enrichment process is based on adding new auxiliary or secondary concepts, which were not so important to define data model itself but can help describing it in depth. Besides, both former and new concepts are given what are called attributes. These are a list of properties unique of each concept and are also used to better

define the latter ones. Attributes have their own code inside de concept map diagrams, and due to software variations, they are also represented differently on CMap tools and DOT (see Figure 3). Data model building and enriching will be shown in more detail in its own chapter, giving a use case as an example of a specific instanciation of the model for better visualization of its practical applicability.

1.4.5 Behavior model

Data model groups lots of concepts and its relations in order to store the Company knowledge. However, those concepts and causal relations need to be sorted via some criteria. For example, when thinking of an assembly procedure, besides knowing which parts need to be assembled and their relativc position, it is mandatory knowing the assembly secuence, this is, the assembly timeline, and exactly how the different assembly operations are carried out. These sort of "whens" and "hows", among other aspects, are collected in the behavior model. It essentially stores how every concept and relation inside Data model behaves, both with itself and others. Thus, behavior model must be an evolved data model, and have the latter as a point of start.

Different approaches were made so as to get a behavior model that while simple and easily understandable, would be also able to fulfill the objectives set for it, explained previously.

The final representation adopted is what will be called a behavior diagram to embody the requirements of the behavior model. Behavior diagrams are modeled with the same software tool DOT that has been used for the concept maps of the Data Model. A more in depth of these diagrams and how to build them is given on Section 2.3.

1.4.6 Semantic model

Last but not least, a semantic model is needed when building an Ontology. As has been explained before, the main objective in Ontology building is preserving the Company knowledge and being able to use it wherever the place by whomever. In order to ensure that, it is essential to develop a common language, so that every team member using this Ontology has the same concept and definition when talking about any subject. Semantic model aims to achieve a total agreement between stakeholders, preventing future misunderstandings from happening and ensuring that a common semantic framework is being used.

The development of this model is out of the scope of this work and should be considered in future research so as to complete the Ontology modelling.

1.4.7 State of the art of MfM methodology

In the last few years, several authors have applied the 3LM methodology, with varying depth and complexity, to a number of use cases. These preliminary results are being used to refine and deepen 3LM, so as to be able to apply this method to more complex and realistic scenarios.

Even before the 3LM concept was first sketched, Mas et al. [9] pointed the necessity of process-based models, capable of being applied during the whole product lifecycle. These very first steps were made using the Unified Modelling Language (UML), and set the foundations to the actual 3LM method.

Mas et al. [10] first introduced the 3LM methodology, stating the MBSE applicability situation and the objectives presented in this project. The three different layers, their behavior and tightness were first defined, as a solution for the software interoperability issues and aiming to achieve an independent ontology for the Company. As well as this project will do, it focused on the ontology layer, briefly defining its components, and discussing the best way to model them.

Following this current of thought, Rizzi [11] proposed concept maps as the first step in an ontology construction method. Austin [12] explores different mechanisms and diagrams in order to model behavior, and Szejka [13] comes with a preliminary method to develop semantic interoperability in MfM. Some of said ontology construction methods, as well as other self-developed ones will be used in this project to build the ontology model for 3LM.

Finally, some early versions of 3LM have been already applied to simple engineering use cases. Morales-Palma et al. [14] have made use of the methodology to study incremental sheet forming processes, specifically applying it to Single-Point Incremental Forming (SPIF). Mas et al. 2019 [15] applies this same methodology to assembly lines in Airbus, mainly on Final Assembly Lines (FAL). Their results, while very academic and

idealistic, are proof of validity for the methodology concept, and show the great versatility of the method developed.

In order to enlarge this versatility, the present work will apply 3LM to another scenario, concerning a subassembly of a big elemental, such as a wing box, prior to its integration inside the FAL. This will be shown as an example of an instanced model, once the ontology is built, and should be contained in the Data Layer and available for the Service Layer to be exploited regardless which software would be used. Said interaction between layers is yet to be implemented and should be developed within further projects.

2 3LM ONTOLOGY LAYER BUILDING PROCESS

Engineering or technology is the making of things that did not previously exist, whereas science is the discovering of things that have long existed.

- David Billington -

This chapter presents the Ontology building process for the aeronautical assembly process, from the scope definition to the behavior modelling, going through the Data model definition and enrichment. For each model, the selected diagram type is presented along with the specific software used to draw the graphical models. Examples selected to describe the building process of the ontology layer belong to the aeronautical assembly use case and will be presented in the next chapter.

2.1 Scope Model using IDEF0 Diagrams

As has been said previously, the very first step when building an Ontology is defining its scope. This allows to stablish a preliminar framework and clarify the Ontology's boundaries so as to limit the project's degree of detail. Due to its numerous advantages as far as versatility and adaptability are concerned, previously stated, IDEF0 diagrams have been chosen in order to model scope model.

IDEF0('Icam DEFinition for Function Modeling', where ICAM is an acronym for 'Integrated Computer Aided Manufacturing'), is a function modeling methodology for describing manufacturing functions, which offers a functional modeling language for the analysis, development, reengineering, and integration of information systems; business processes; or software engineering analysis.IDEF0 is part of the IDEF family of modeling languages in the field of software engineering, and is built on the functional modeling language Structured Analysis and Design Technique (SADT).

IDEF0 may be used to model a wide variety of automated and non-automated systems. For new systems, it may be used first to define the requirements and specify the functions, and then to design an implementation that meets the requirements and performs the functions. This is the case of this work, in which a new Ontology is being built. Besides, for existing systems, IDEF0 can be used to analyze the functions the system performs and to record the mechanisms (means) by which these are done.

2.1.1 IDEF0 building blocks

The result of applying IDEF0 to a system is a model that consists of a hierarchical series of diagrams which are interrelated. The two primary modeling components are functions (represented on a diagram by boxes) and the data and objects that connect those functions (represented by arrows).

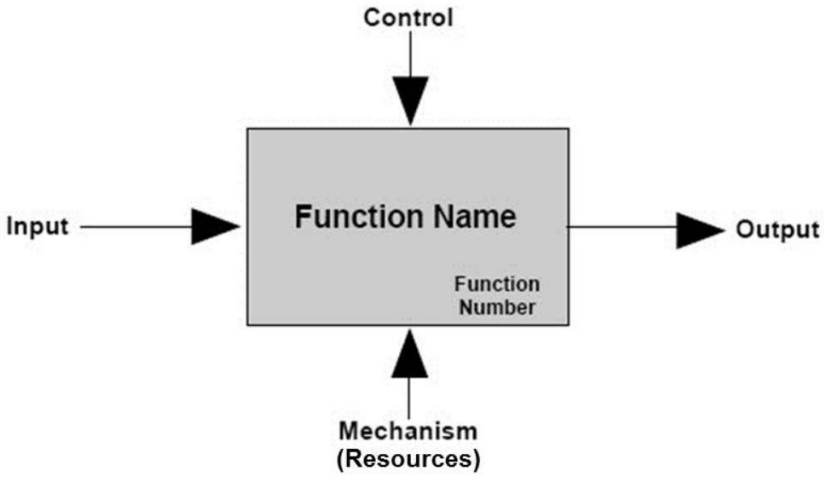


Figure 4. IDEF0 building block.

Figure 4 shows a single block, in which a Function is represented in the middle and one of each type of data connect it. Each activity is described by a verb-based label placed in a box.

Coming from the left, inputs feed the Function. These are data that are used by the function in order to produce the output, being transformed throughout the process.

On the upper section come the control elements. These are objects that help when specifying how the inputs turn into outputs, and they remain unaltered during this transformation.

Coming from the bottom part there is the mechanism arrow. As is clarified in the figure itself, this kind of data are the resources used in order to produce the outputs. These can be industrial, manufacturing, human or software resources, among others.

The logical output is shown as an exiting arrow on the right side, which will then feed the next block or blocks. These are new data generated by the function using all prior objects.

The IDEF0 process starts with the identification of the prime function to be decomposed. This function is identified on a “Top Level Context Diagram,” that defines the scope of the particular IDEF0 analysis. From this diagram lower-level diagrams are generated. So as to be able to quickly identify the function’s hierarchy, a coded number is given to each one.

2.1.2 Software for IDEF0 diagrams: RAMUS

The selected tool to create IDEF0 diagrams is RAMUS. Its educational version offers a simple and open-source software. It is really intuitive and easy to use, elements are inserted with a single command, and so are the arrows between them.

Both Top Level and “child” diagrams are shown in the full IDEF0 anaylisis made on the wingbox assembly case of use, Figure 5-10.

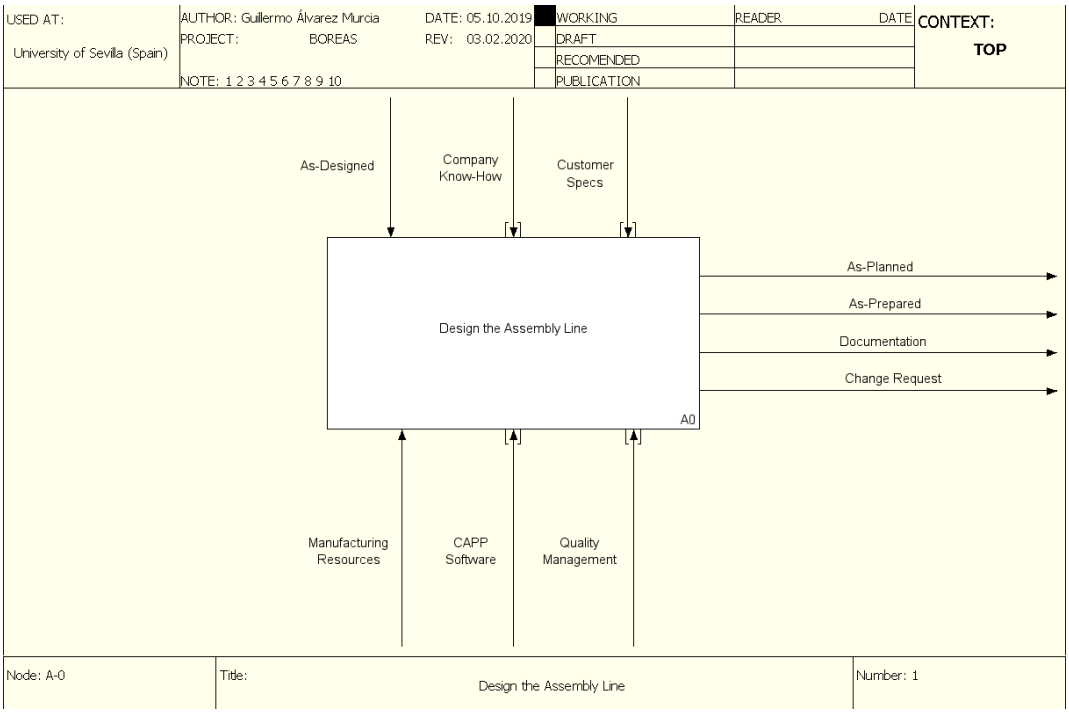


Figure 5. Top Level IDEF0 diagram

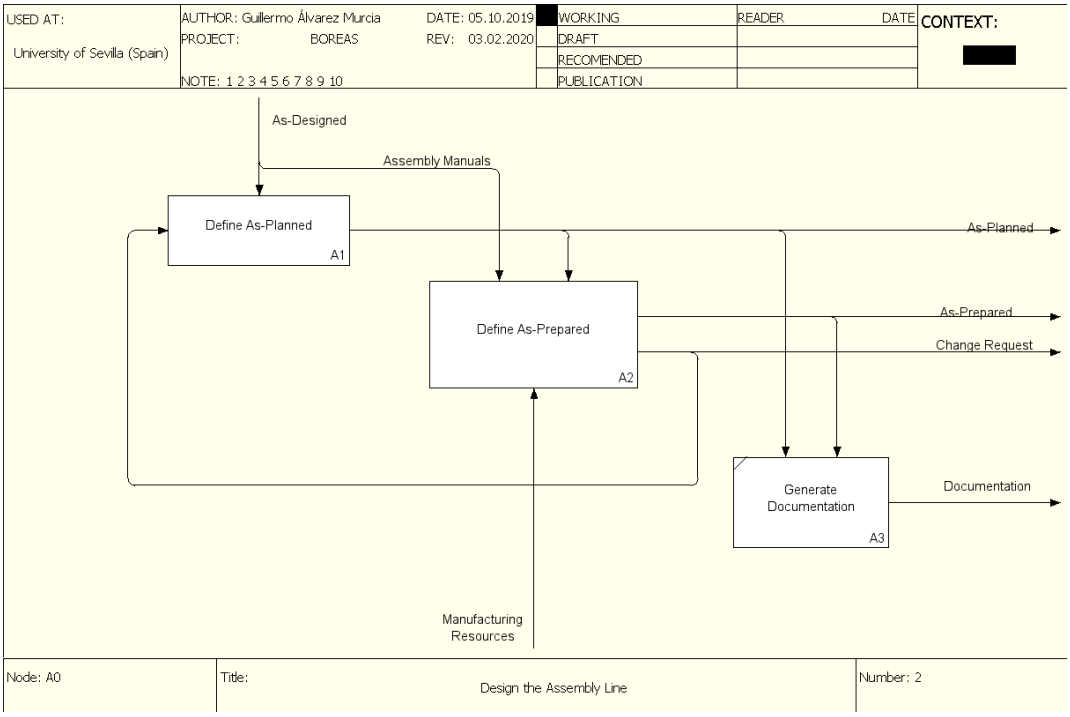


Figure 6. First level IDEF0 diagram, A0: 'Design the Assembly Line.'

In order to design an assembly line (see Figure 5 and 6), having the As-Design as a beginning point, and taking into account the Company know-how and customer specifications, as well as the Company resources (Manufacturing, CAx software and Quality management) As-Planned and As-Prepared need to be planified. Other parallel activities need to be done, such as generate respective documentation or send change requests in case any issues are detected.

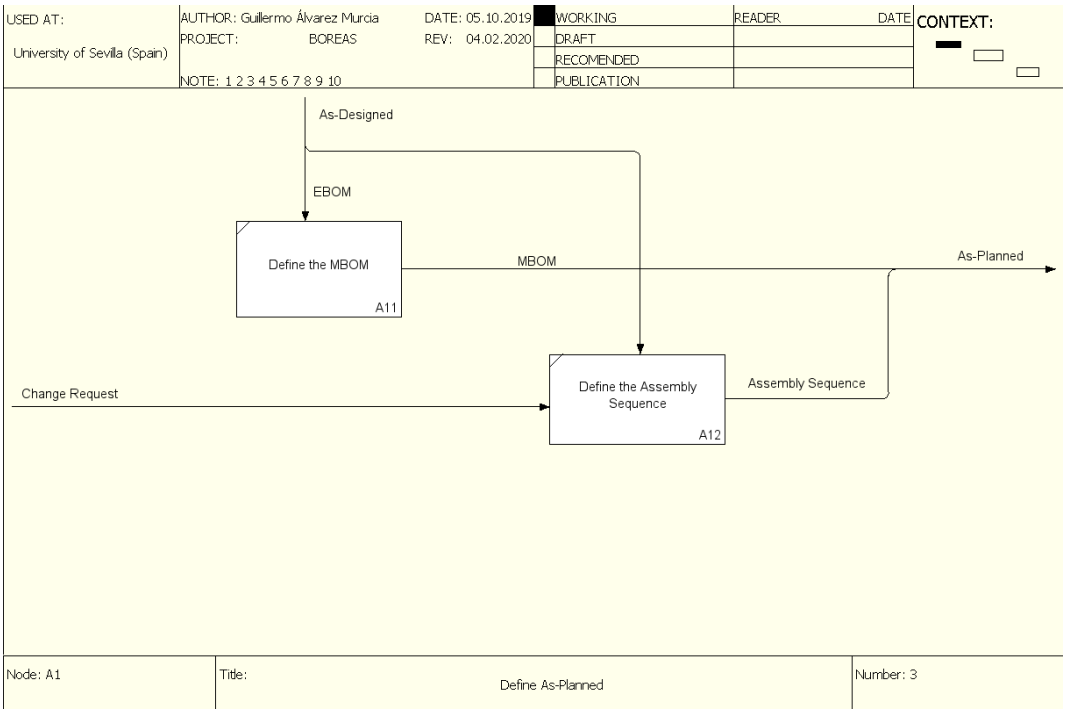


Figure 7. IDEF0 diagram A1: 'Define As-Planned.'

As-Planned is modelled with two main tasks, as can be seen in Figure 7: defining the manufacturing bill of material (MBOM) and the assembly sequence, conditioned by the change request proposal, if triggered.

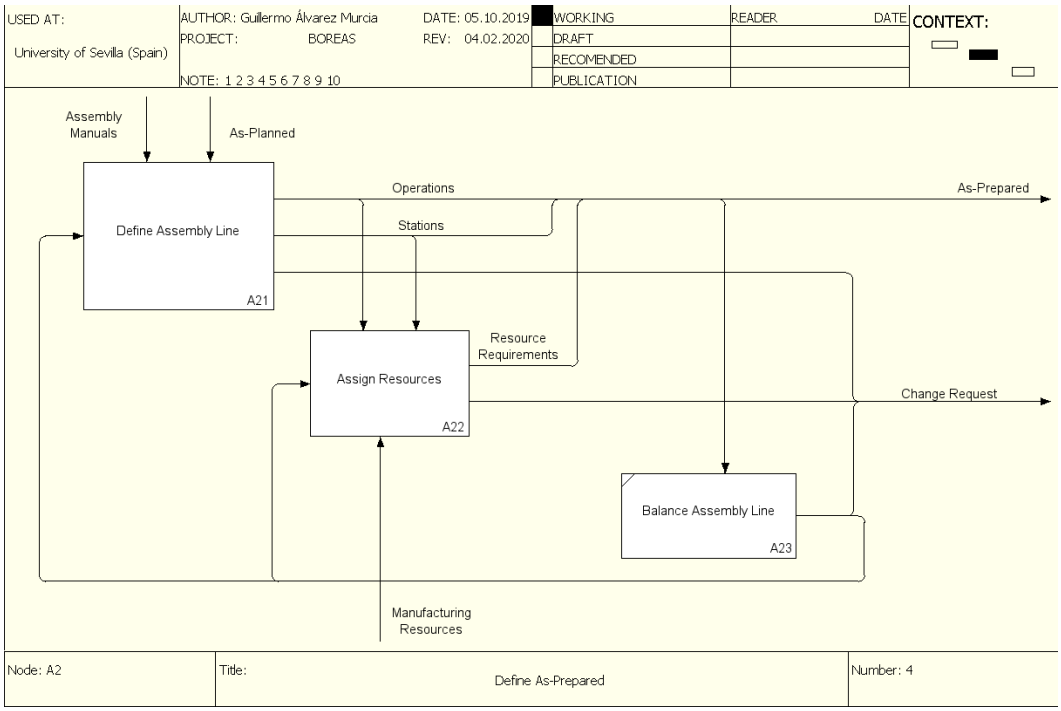


Figure 8. IDEF0 diagram A2: 'Define As-Prepared.'

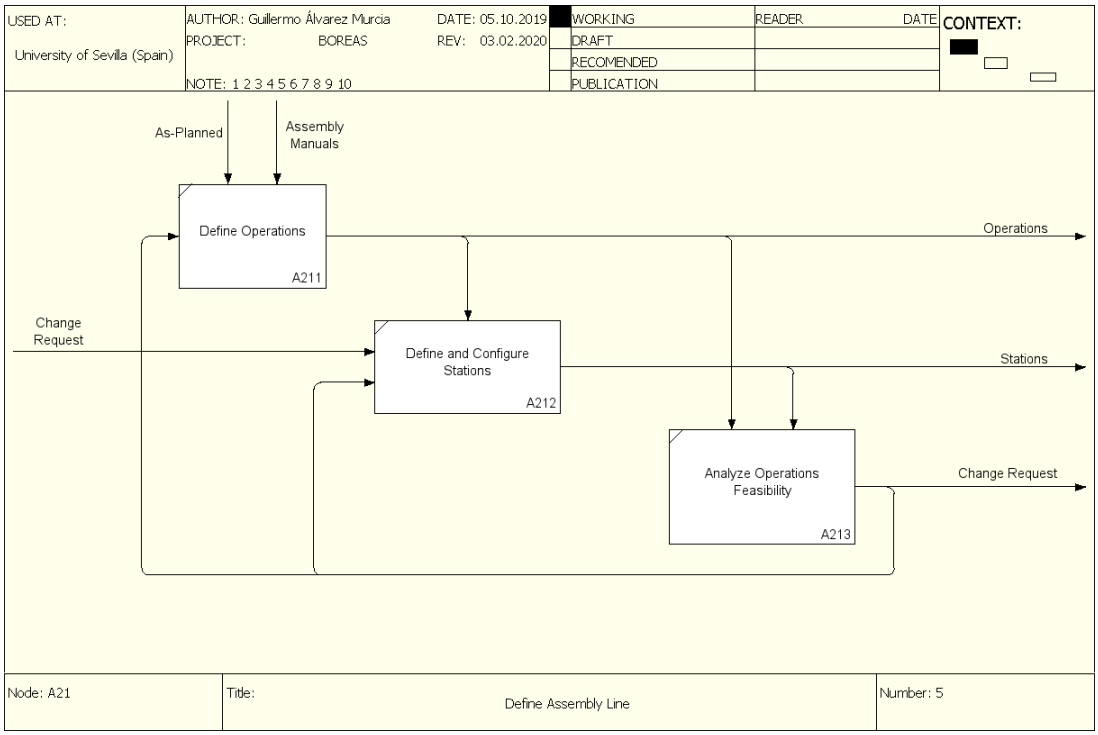


Figure 9. IDEF0 diagram A21: 'Define Assembly Line.'

As-Prepared uses As-Planned and the assembly manuals to define the assembly line, its resources and its balancing, as depicted in Figure 8. These also produces the change request proposal. The assembly line is built defining the different manufacturing operations (see Figure 9), that are then grouped in several stations, which need to be configured and sorted. Besides, feasibility need to be analyzed, both operation and station-level, being important in determining the change request conditions. Every of these is again conditioned by the possibility of a former change request trigger.

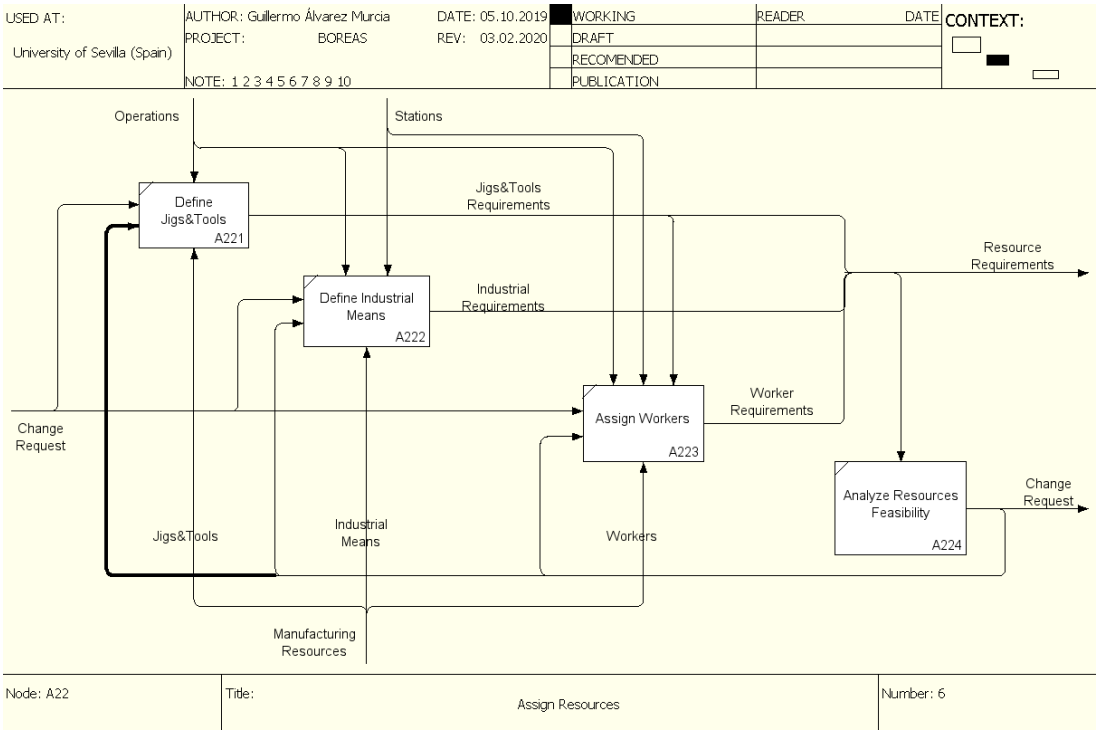


Figure 10. IDEF0 diagram A22: Assign resources

Resource assignment is made defining each of the different resources (see Figure 10), i. e., Jigs&Tools, industrial means and workers, as well as determining their requirements. Feasibility of these requirements is again needed, being a key factor when launching a change request.

As can be seen in Figures Figure 5-10, a detailed scope and boundaries are defined for the practical case presented. Transformation processes, resource and control definition, and feedback mechanisms are presented.

In order to better clarify the hierarchy between diagrams, Figure 11 presents an overview of all IDEF0 diagrams.

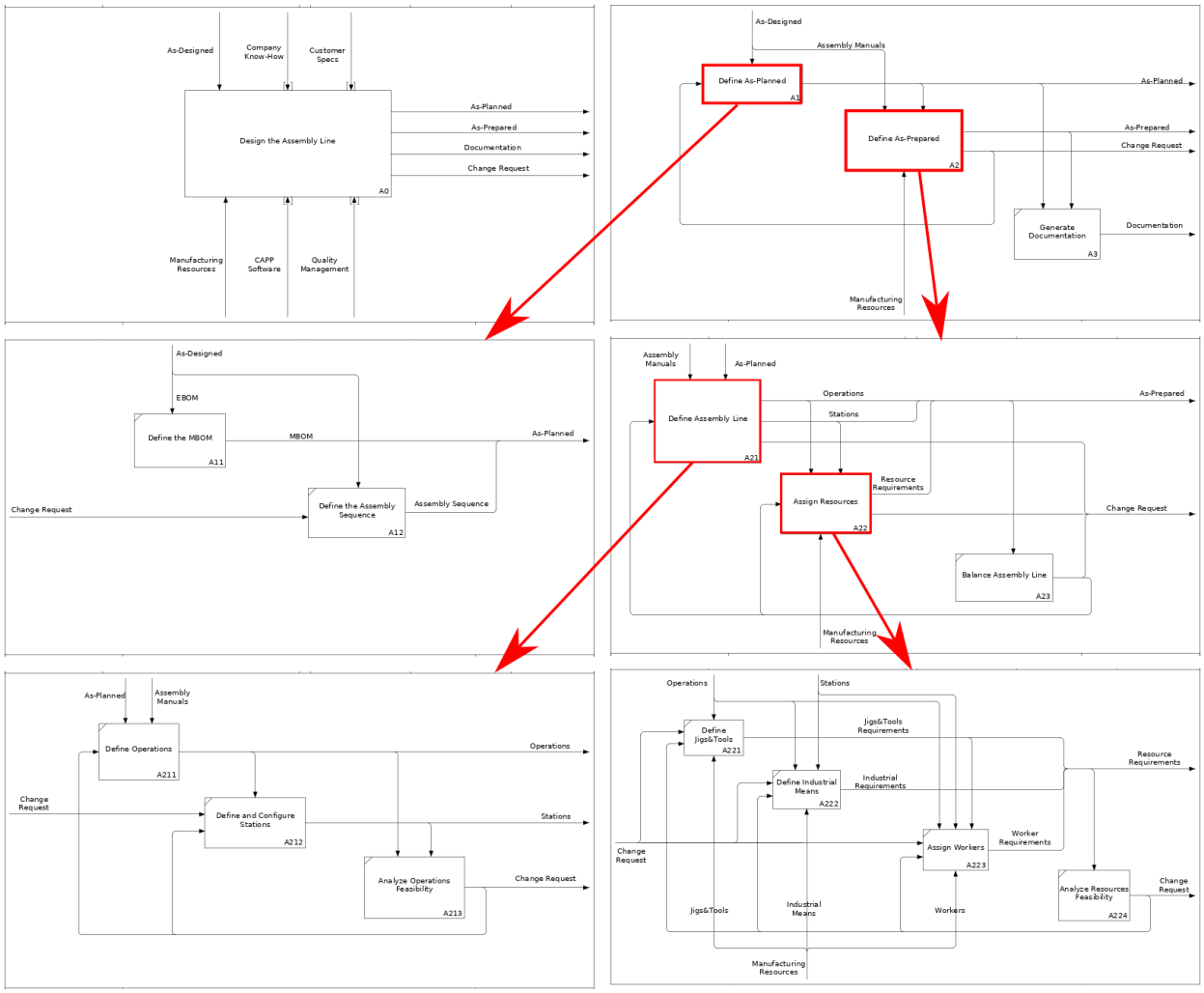


Figure 11. IDEF0 diagrams hierarchy overview

2.2 Data Model using Concept maps

Once the scope is set, the next step is building the Ontology core, that is, the Data model. All the Company knowledge is now embodied into the Ontology, via several general concepts and their relations between each other.

Because of the desired structure for the Data model, concept maps are chosen as the most suitable option. These diagrams have the same structure as the one pursued in Data model and are easily understandable by anyone no matter its relationship with the Company, nor its previous knowledge about ontologies.

2.2.1 Software for concept maps: CMap Tools and Graphviz DOT

As has been said before, two different approaches were made when building the Data model, due to major differences between the two software tools used. At first, CMap Tools was used. Its drag&drop behavior is intuitive and user friendly, and the results achieved are consistent and visually attractive when developing a simple model. The first disadvantage found is its lack of agility when working between iterations. This induced the use of DOT library in Notepad++. DOT is a text-based graphical tool, i.e., the graphs are automatically generated from a text code with the corresponding instructions. While results are still pretty good, its usage is much more advanced. Using DOT means losing some flexibility in the final diagram, in exchange of a much more powerful tool making changes rapidly between iterations.

This complementary character between both tools makes them ideal for its serial exploit. First, DOT is used to go through iterations until reaching the desired configuration. Then, so as to present the final result in a prettier way, CMap Tools is used.

DOT needs the user to be familiarized with its language. Once the specific commands are learned, its usage is as simple as typing the desired structure for the diagram onto a text file and then compiled and executed by DOT. Therefore, just rearranging code and executing it again lots of diagram variants can be generated immediately, being now remarkable the great value of this tool when iterating the model. One example of DOT scripting and its result can be seen in Figure 12 and Figure 13.

```
digraph datamodel {
    compound=true;
    colorscheme=pastel19;
    node [shape=record, style="filled, rounded", fillcolor=white, color=black]
    ranksep="0.5 equally";
    nodesep=0.5;
    concentrate=true;
/* DATA MODEL */


    /* As-Designed */
    subgraph clusterAsDesigned {
        label="As-Designed"; style=filled; color=1

        asde [label="{Wing Box \n (As-Designed)}"]
        ebom [label="(EBOM)"]
        p01 [label="{P01 (Part) | Spars \l}"]
        p02 [label="{P02 (Assembly Manual) | Joint Spars-Panels  \l}"]
        p03 [label="{P03 (Part) | Panels \l}"]
        p04 [label="{P04 (Assembly Manual) | Joint Panels-Ribs  \l}"]
        p05 [label="{P05 (Part) | Ribs \l}"]
    }
    asde -> {p01, p02, p03, p04, p05, ebom}
    ebom -> {p01, p03, p05} //[label="clasifies"]
//    p02 -> {p01, p03} [label=has]
//    p04 -> {p03, p05} [label=has]

    /* As-Planned */
    subgraph clusterAsPlanned {
        label="As-Planned"; style=filled; color=2

        aspl [label="{Wing Box \n (As-Planned)}"]
        mbom [label="(MBOM)"]
        sequ [label="(Mfg Sequence)"]

        subgraph clusterStep1 {
            label="Mfg Step 1"; style=filled; color=5

            s01 [label="{S01 (Mfg Step) | Assembly Spars-Panels \l}"]
            m01 [label="{P01 (Mfg Part) | Spars \l provided \l}"]
            m02 [label="{P02 (Mfg Manual) | Joint Spars-Panels  \l}"]
            m03 [label="{P03 (Mfg Part) | Panels \l manufactured \l}"]
        }

        subgraph clusterStep2 {
            label="Mfg Step 2"; style=filled; color=5
            s02 [label="{S02 (Mfg Step) | Assembly Ribs \l}"]
            m04 [label="{P04 (Mfg Manual) | Joint Panels-Ribs  \l}"]
            m05 [label="{P05 (Mfg Part) | Ribs \l acquired \l}"]
        }
```

Figure 12. DOT script defining the data model.

Figure 13. Concept map for the Data model, compiled from the DOT script.

As can be seen, there is some loss of flexibility, since arrows are automatically connected and cannot be rerouted, but results are quite good. As well as CMap, DOT has some customization features, such as independent background colors for subsections or box coloring, which will be shown in the model enrichment point.

The model presented is divided in four different sections, which mostly match the main activities previously shown in IDEF0 main diagrams. These are As-Designed, As-Planned and As-Prepared definition, and an extra module named as Feasibility and Balancing.

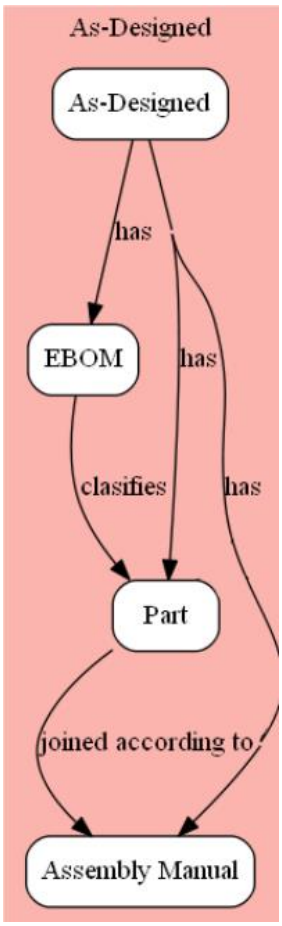


Figure 14. As-Designed definition of the Data model

As-Designed definition process is shown in Figure 14. It is an object made of the Engineering Bill of Material (EBOM), the Assembly Manuals needed to sort said EBOM, and the parts themselves constituting the final product. How these items are related is explained over the concept map itself.

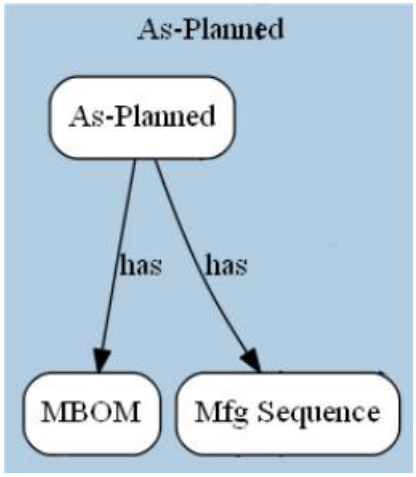


Figure 15. As-Planned definition of the Data model

As-Planned (Figure 15) is even simpler, at least by itself. It is just the sum of the Manufacturing Bill of Material (MBOM), obtained from the EBOM, and the Manufacturing Sequence used to carry out the Assembly.

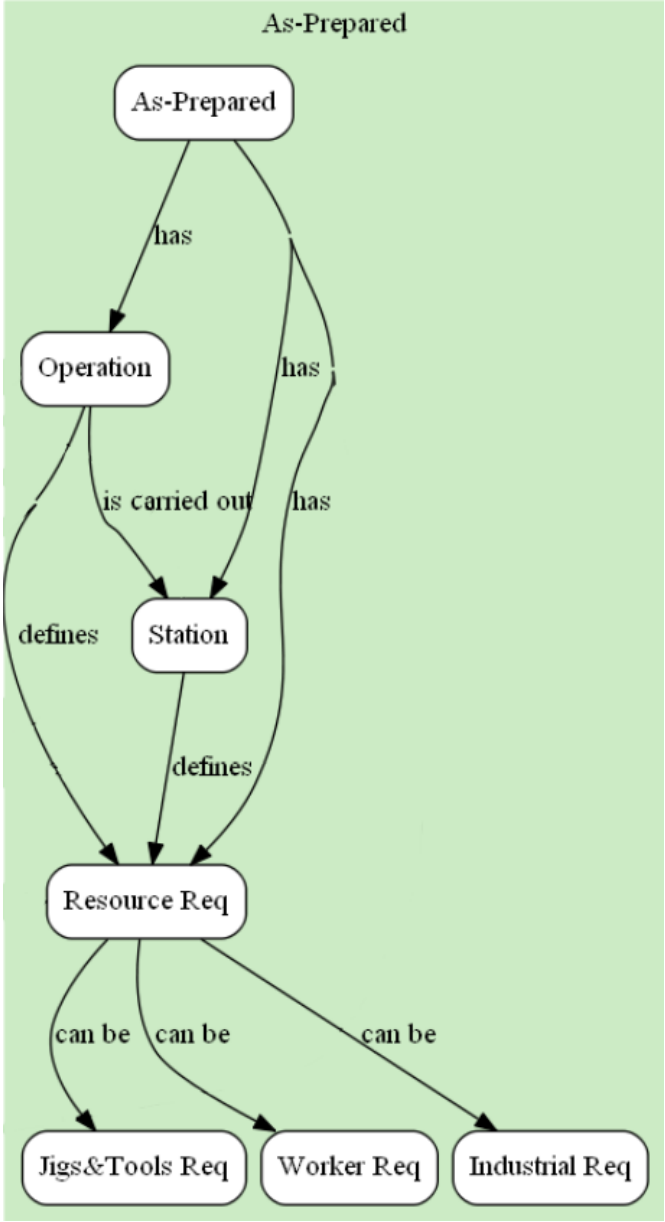


Figure 16. As-Prepared Definition of the Data model

As-Prepared (Figure 16) can be a bit more complex, due to the variety of objects involved. As-Prepared embodies all different operations made in order to fulfill the Assembly process. These operations are grouped in stations. The final sorted set of stations constitute the final assembly line. Both operations and stations are responsible for defining the resource requirements, that is, which specific investments are needed to be done so as to perform the different assembly steps. Resources have been grouped in three different categories: Jigs&Tools, Workers (manpower) and Industrial means.

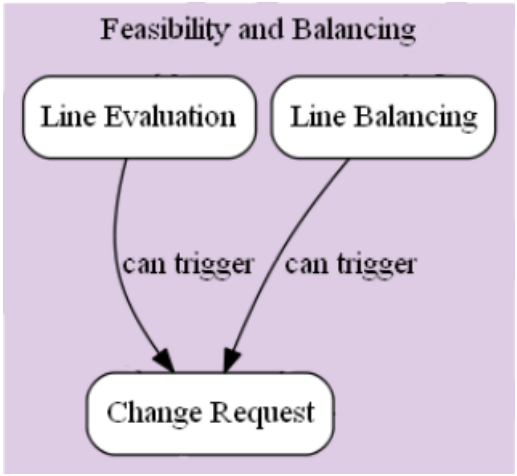


Figure 17. Feasibility and Balancing of the Data model

As have been said before, one extra module was needed to complete the whole concept map. This is the Feasibility and Balancing one (Figure 17). As can be seen, it is really simple, and is used only to ensure that change requests are triggered, and thus, iterations are made during the assembly line definition development. Change requests can be triggered due to issues regarding the line evaluation itself (non-feasible operations or sequences because of ergonomic reasons, for example) or due to balancing aspects (changing some operations or switching sequences may improve assembly time, for instance).

All these sections presented before are not self-isolated, but connected between each other, so that each different module feeds the other. The whole concept map has been presented in Figure 13.

CMap tools is, as said before and mainly, easy to use. When opening a new project, it is very fast to introduce the several concept boxes, and so is connecting them using arrows. Blanks are automatically placed and then easily filled. Once every single object is created, they can be rearranged simply by dragging and dropping them in the desired locations.

Some customization options are available, such as changing boxes color and outline, arrows type, text font and size or background color for diferenciating sections. An early diagram is shown in Figure 18.

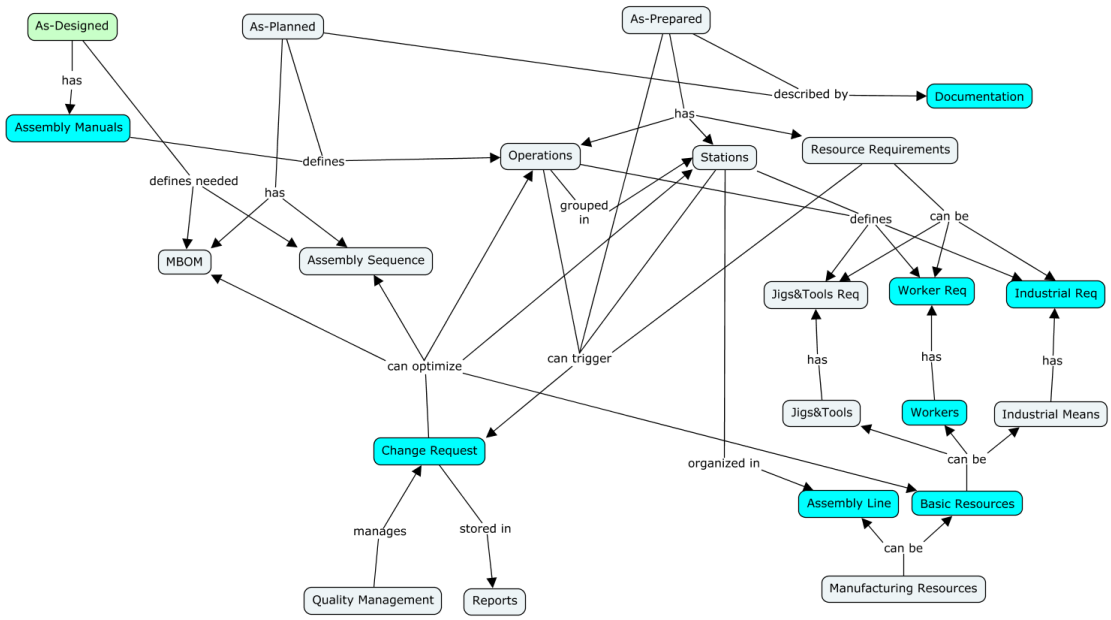


Figure 18. CMap Example

2.2.2 Data Model enrichment

The presented Data model is then completed and detailed, introducing new concepts and giving attributes to concepts. This allows a better and deeper definition of the model. A special codification is used to be able to differentiate between former and new elements: new boxes will be introduced in yellow. When using DOT, attributes will be implemented inside concept boxes, separated by a horizontal divider. Using Cmaps, attributes will be recognized because they are not enclosed in a box. An enriched model for clarification is shown in Figure 19. Said model is made using DOT and corresponds to the same practical case presented before, an aeronautical assembly line design.

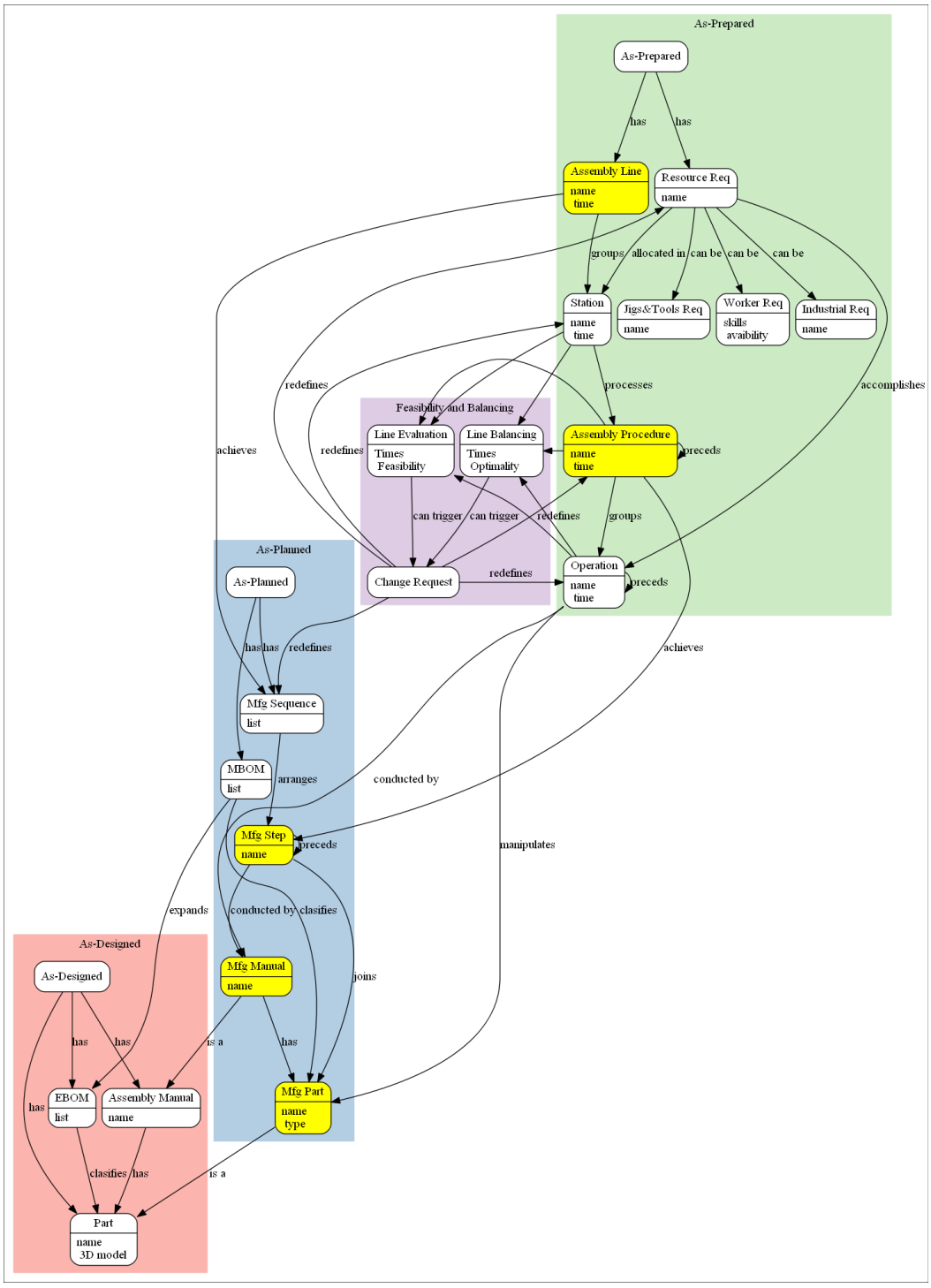


Figure 19. Enriched data model using DOT

Figure 19 shows the basic Data model structure for the said case, shared with the scope model already presented in 2.1. As-Designed, As-Planned and As-Prepared are located in different sections and related between each other, and also conditioned by the Feasibility and Balancing section, responsible for triggering a possible change request. Several “child” concepts hang from each of them. New concepts and attributes are introduced following the new codification. For instance, now the manufacturing sequence is described a little further, introducing the concept of manufacturing steps. In a similar way, the different stations are now made of assembly procedures, which properly arrange operations. Both new and previous concepts have some key properties, such as total time invested during an assembly procedure, or the manufacturing part type (bough - off the shelf- or made). It can be seen that the arrangement is not the best aesthetically speaking. This justifies the use of CMap for the final result.

2.2.3 Data Model instancing simulation

So as to exemplify how different layers inside the 3LM would work, an instancing simulation is presented.

Enriched data model shown in 2.2.2 would feed the Data layer via legacy applications or interfaces. Once this model is stored in the database, it can be automatically filled with concepts and attributes specific of a case of use. The filled diagram for a specific practical case is called an instance. Taking the wingbox assembly case as an example, the resulting instance that would be processed by the Data layer is shown in Figure 20

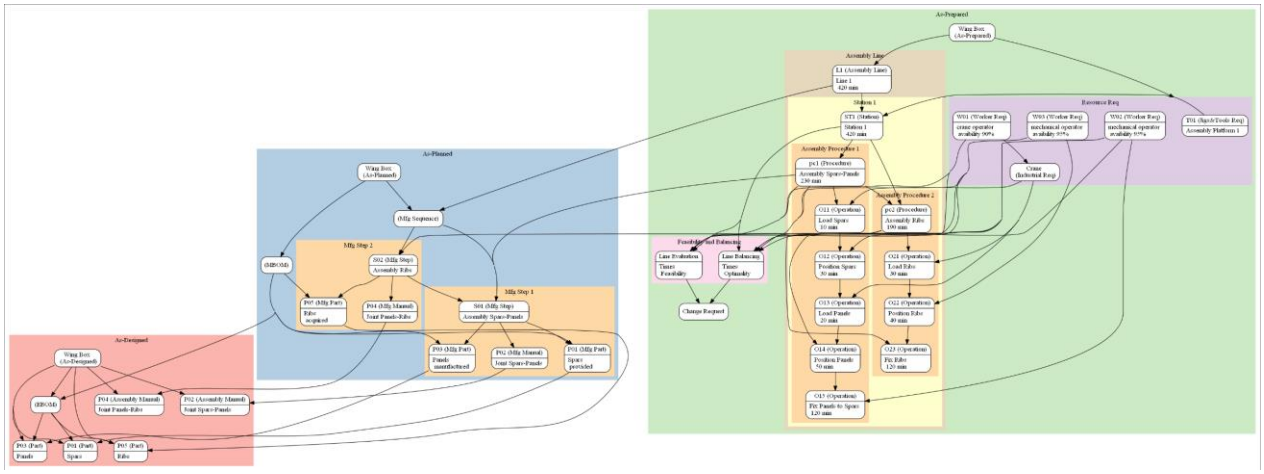


Figure 20. Example of Data model instance.

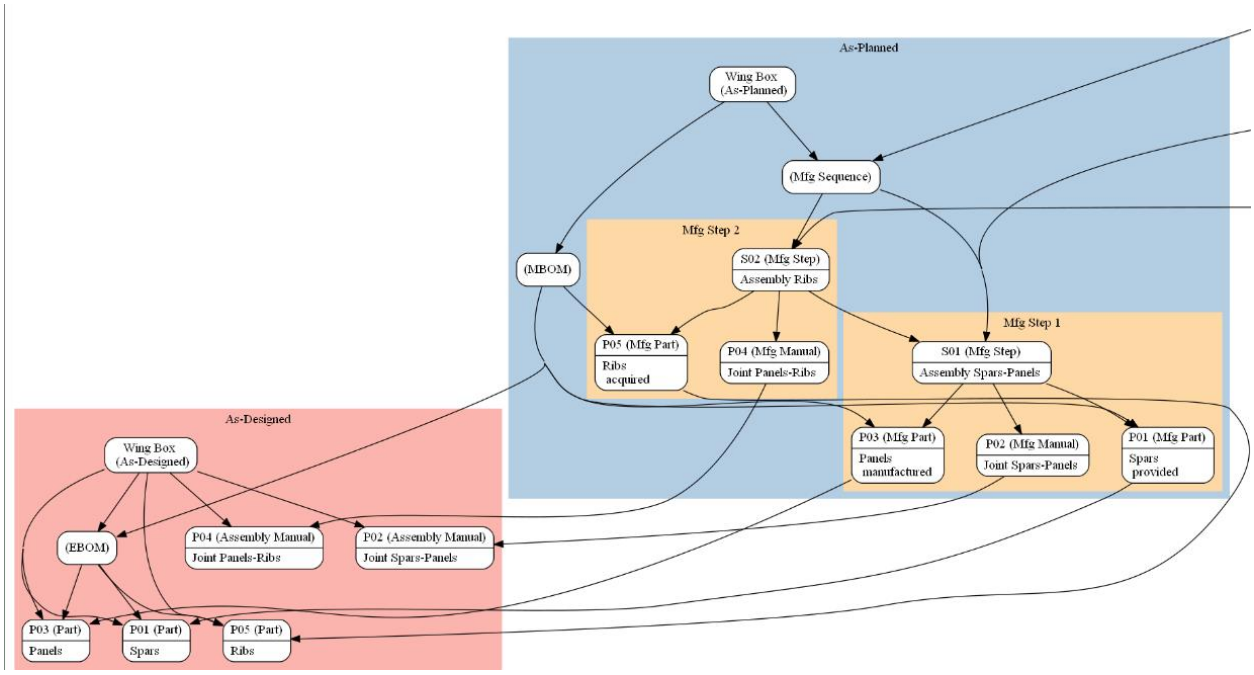


Figure 21. Example of Data model instance: detail of As-Designed and As-Planned.

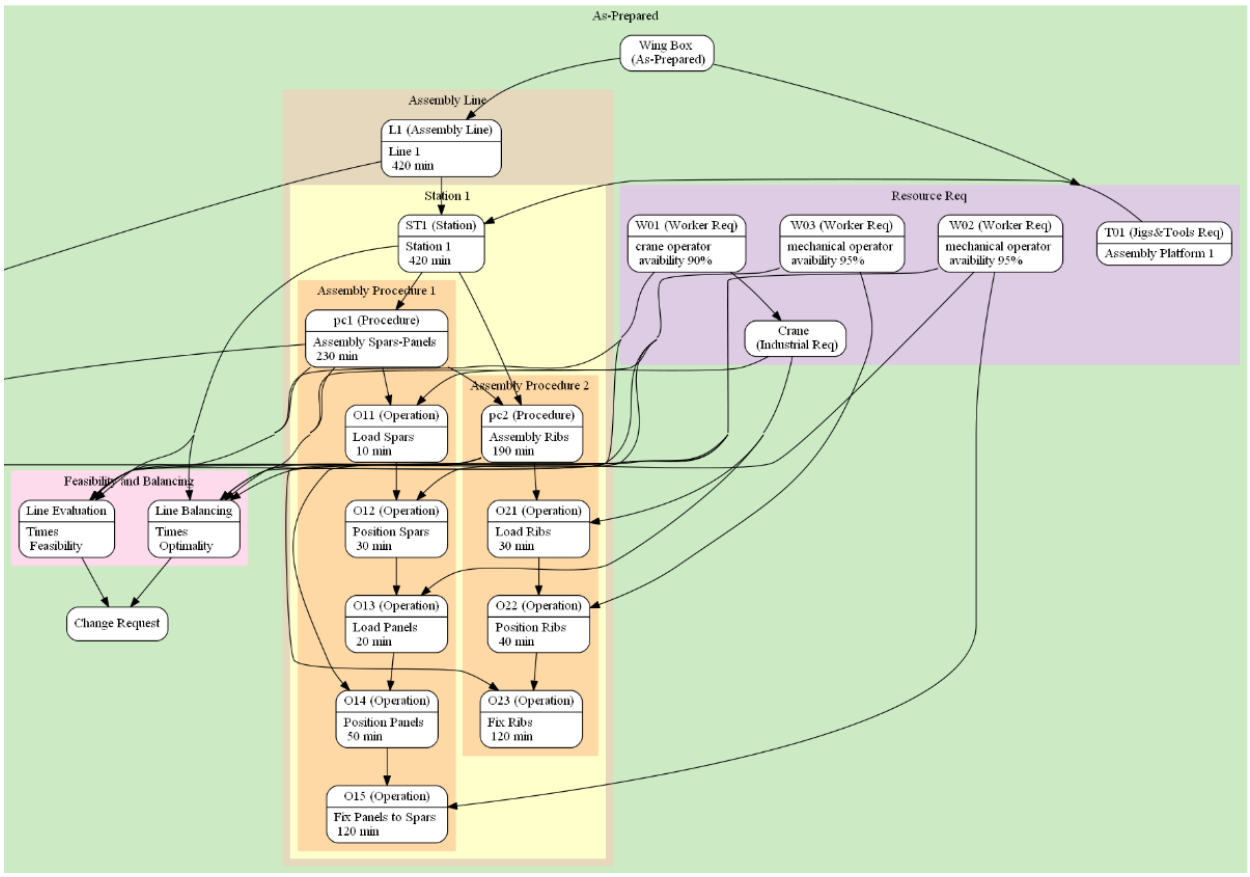


Figure 22. Example of Data model instance: detail of As-Prepared.

Figures Figure 20-Figure 22 show the full instanced model. The previous structure is now filled with specific data: each part and subasembblies are defined, as well as each manufacturing step, operation, station, procedure and resource requierement. Numerical values are given for the suitable attributes, such as the time needed to perform each operation. Other attribute fields are also filled. Feasibility and Balancing attributes have not been changed due to the impossibility of computing such analysis, but would return a Boolean value, and may or may not define and trigger a change request proposal. Theoretical concepts (the ones shown in Figure 19) are still present, marked between brackets.

For further clarification, some key concepts and their relationships are going to be described. Taking a look to the As Designed section, now it is the product itself (wing box) the concept in the box. Similarly, now parts are instanced, being noted as P01, P03 and P05. Each one is then defined via its name attribute (Panels, Spars and Ribs, respectively). P02 and P04 are used to instanciate the different subassemblies, that is, the joining of two parts.

Another good place to look is the assembly line section. Here, a single line made of a single station approach has been made. Inside this station, two different procedures are carried out, one after the other. The first one, denoted as pc1, is defined using its attributes, that is, its name, which clarifies what this procedure is about, and the total time needed in order to fulfil it, which is the sum of the times of every operation made inside the procedure. In this case, pc1 aims to join panels and spars of the wingbox, and is made of operations O11, O12, O13, O14, and O15. In the same way, these operations also are defined via their names and times, stored on their properties. Once pc1 is finished, pc2 can begin, joining ribs to the subassembly resulting of pc1. Its structure is very similar to the one explained for pc1 and thus is omitted. For simplicity, and just to exemplify and instancing process, non-productive times have not been considered, and numerical values have been estimated vaguely.

It is worth noting that this example was specifically chosen to be extremely simple and clear so that concepts could be easily understandable. On a real case the huge amount of parts and operations would make it difficult to apprehend everything at a glance.

2.3 Behavior Model using behavior diagrams

Once Data Model is developed, the next step is to create a structure that can explain how the different concepts are carried out, how every individual aspect of them behaves. It is then when the behavior model is needed.

An initial approach using different existing behavior diagrams was first considered, such as activity diagrams written using UML language. However, none of them seemed to completely fulfill the specific necessities and requirements that this work demanded. It was then decided that a new kind of diagram needed to be created from scratch. This diagram was defined using DOT, already explained, getting advantage of this language capabilities.

Different approaches and kinds of diagrams were discussed, each one with its own benefits and disadvantages. The main goal was to provide a simple yet complete enough tool so as to reliably reflect how concepts behave. How behavior diagrams are made is explained next, using Figure 23.

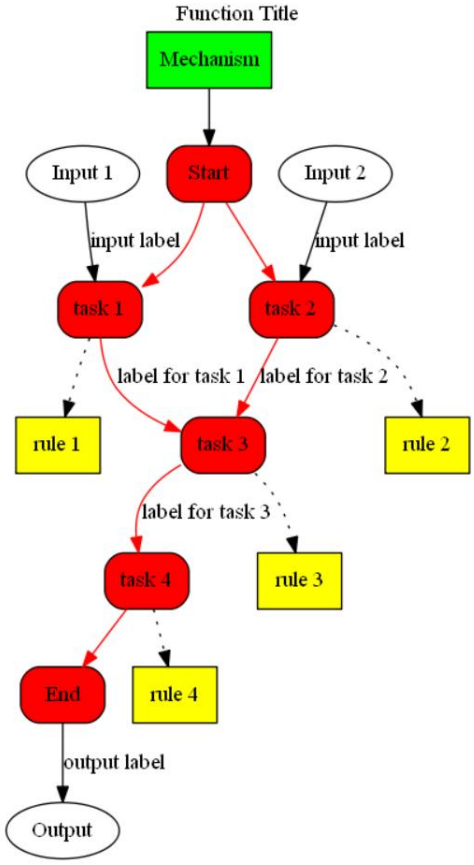


Figure 23. Behavior diagram basic example

Before even talking about each of the elements that are part of a behavior diagram, it is needed to explain the key elements used to build it. Behavior diagrams are made directly from the Scope Model, that is, the IDEF0 diagrams already shown in 2.1. Using the elementary functions (the ones that do not have 'children' activities associated), and their inputs, outputs and mechanisms, a new behavior diagram can be made. This can be seen in Figure 23, where the IDEF0 function is the title of the diagram, inputs and outputs are the beginning and end of the diagram and mechanisms enclosed in a green box and linked to tasks. When particularly speaking of mechanisms, it has been decided that when the mechanism affect every task, it will be linked to the Start box, in order to preserve some cleanness in the modelling.

Behavior diagrams are modelled like flowcharts often used when programming or building algorithms. A set of task needs to be done, from a start point to an end, and they can be performed either on a serial or parallel way. In this simple example, task 1 is carried out at the same time as task 2, once the two of them are completed, task 3 is made and then task 4 begins. The activity ends once task 4 does so. As well as flowcharts, tasks can be connected to previous ones and loop between them, so as to make an iterative process until a certain condition is met. If needed, labels can be added between tasks to clarify with aspects of each task are useful for the next one.

But apart from that, these diagrams also offer the information about how each task must be done. This is what has been called rules. It is therefore mandatory for every task to have a rule attached. Rules can be as simple as wanted but must provide clear directives about how to properly perform their respective tasks.

2.3.1 Software for behavior diagrams: Graphviz DOT

One of these diagrams needed to be done for each elementary function. Due to the fact that they are so similar between each other, only some of them are presented next. The other diagrams are collected in the Appendix.

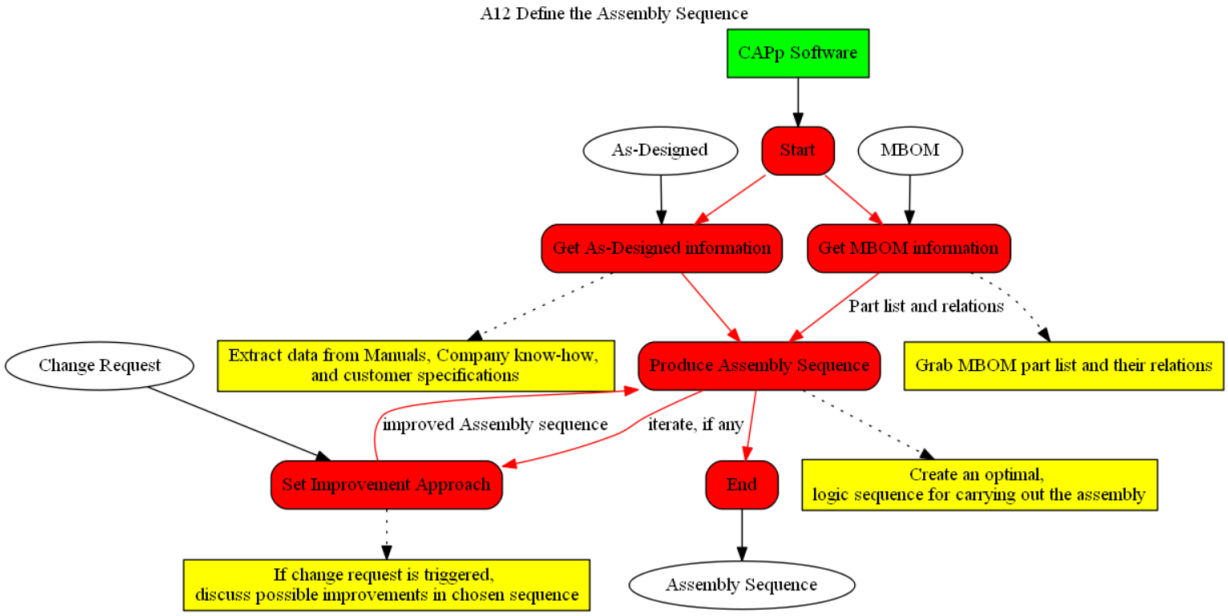


Figure 24. Behavior diagram for activity "A12 Define the Assembly Sequence."

Figure 24 shows the behavior diagram generated for activity A12, defining the assembly sequence. In order to create the assembly sequence, both information from both As-Designed and MBOM needed to be extracted. Then, after the definition itself, an iterative process was modelled as a closed loop between activities. These iterations will be triggered only if ordered by Change Request. This kind of diagrams generate some extra objects aiming set how inputs are turned into outputs, and how tasks are made.

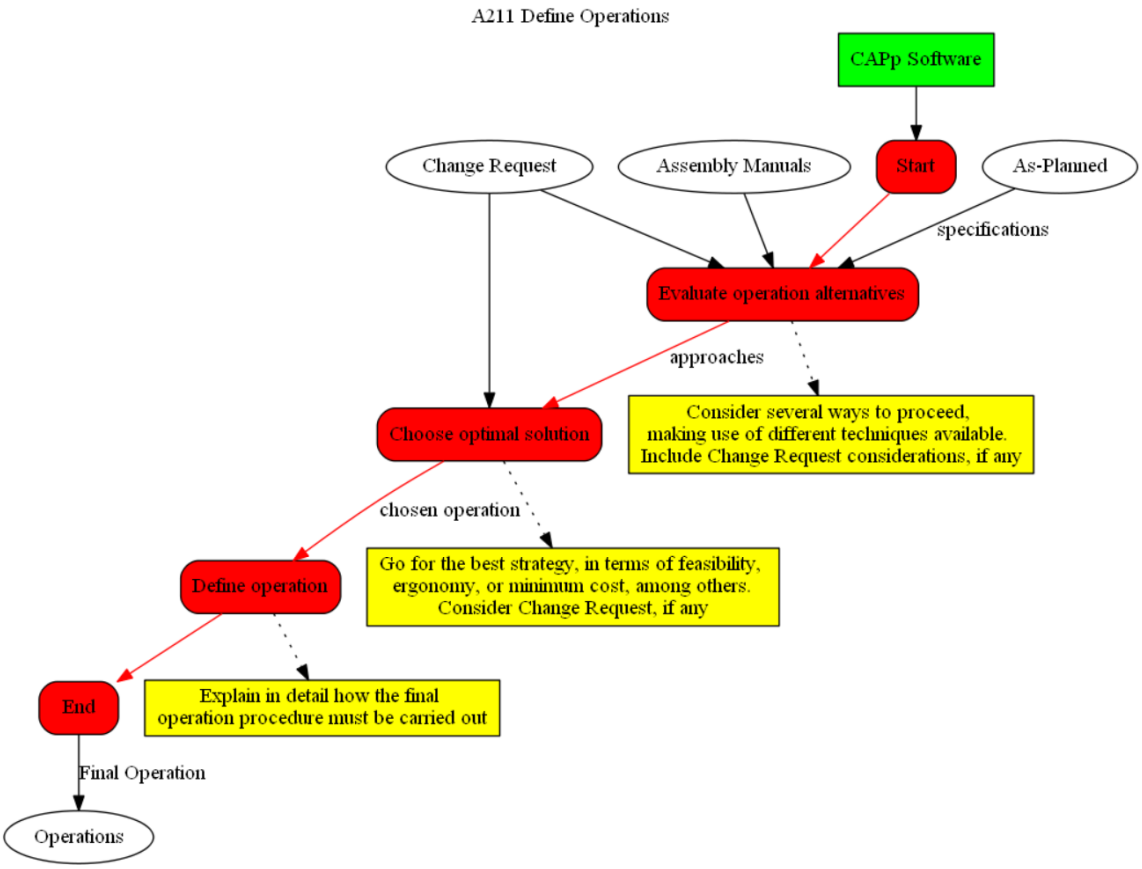


Figure 25. Behavior diagram for activity "A211 Define Operations."

Figure 25 shows a much more linear dynamic when following the different tasks. Using every input available, different situations are modelled and studied. Using analysis tools, it is needed to then select the best case possible, in terms of several different variables, and finally doing a better, specific definition of the chosen strategy. Again, some extra objects are needed in order to specify how to perform described tasks.

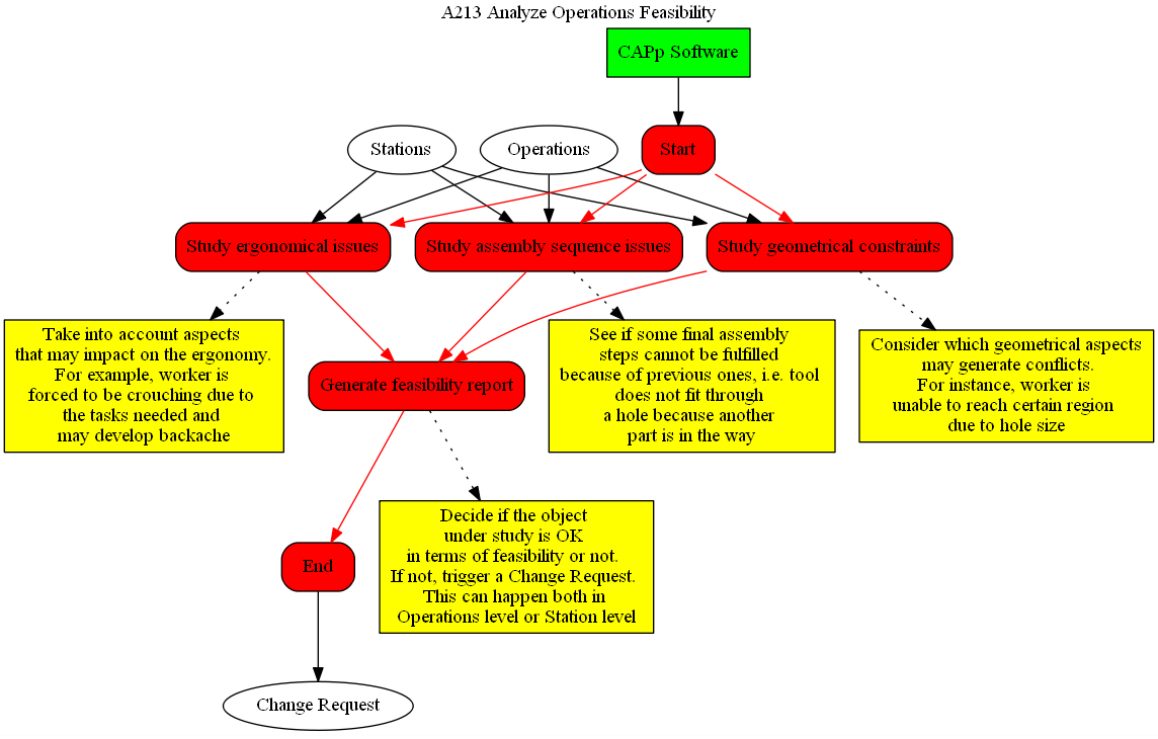


Figure 26. Behavior diagram for activity "A213 Analyze Operations Feasibility."

Figure 26 is an example of a parallel task diagram. When analyzing operations feasibility, several aspects need to be considered at the same time before arriving to any conclusion. Three different groups were considered in this case: ergonomics, issues associated to geometrical constraints, and bound to sequence steps.

As can be seen, there is a wide variety of diagrams depending on the activity chosen, as well as major differences between rules. Some are mere indications, while others (like the ones on Figure 26) include instanced examples for more clarification.

It must be remarked that some of the elements included in these diagrams are not presented in their respective IDEF0 representations (for instance, the CAPP software included in every behavior diagram). This is because once the different models are being developed, new considerations are being taken into account and a refining process begins. In order to build this Ontology properly, an iterative process should start now, iterating around Scope, Data and Behavior model, so that each model feeds the other back, until reaching the final solution. This iterative improvement strategy is out of the scope of this project and should be considered in future research.

2.4 Layers interaction

This section aims to explain what a theoretical fully functional 3LM model would do, and how the three different layers would interact between each other.

As has been explained during the last chapter, the core of the 3LM is the Ontology layer. It can be thought as a ‘cloud’ in which all the Company knowledge is stored.

But this collective knowledge needs to be embodied physically, becoming something tangible. That is what Data layer is meant for. It can be thought, if making an analogy, as if the Ontology is going from the abstract ‘cloud’ storage concept to a tangible ‘hard drive’ one, being now physically stored in databases. And as well as a real hard drive, all this stored information needs to be accessed. Data layer also develops tools so that the Service layer can communicate with the stored data, via interfaces or legacy applications.

Finally, Service layer holds software as authoring and simulation tools, visualizers, data analytics and dashboard and space design exploration tools. It uses all the Company knowledge collected in the Ontology layer and stored and served by the Data layer in order to create specific applications and use cases, previously defined as instances. Due to the nature and structure of this model, each layer is decoupled from the others, which in this case allows Service layer to be independent from them. This implies that the Company knowledge would be accesible no matter which software or program is desired to be used, being this one of the strongest advantages of the 3LM modelling philosophy.

Due to the huge complexity that developing a fully functional model involves, this work has always and only focused on developing more in depth the Ontology layer. The last functioning dynamic explained is merely academic and should be revised and expanded in future research.

3 USE CASE: AERONAUTICAL ASSEMBLY LINE DESIGN AND PLANIFICATION

You never change things by fighting the existing reality. To change something, build a new model that makes the existing model obsolete.

- Buckminster Fuller -

Assuming that the whole 3LM model is made, this chapter recreates what an instance made by the Service layer would look like. In future work, different tests will be required using a specific tool that can really demonstrate the viability of MfM in real use cases. Having this purpose on mind, it has been chosen a commercial program that has a close relationship with the aerospace industry and also because of its powerful different tools. Cited software is 3DExperience (from now on, 3DX), by Dassault Systèmes. The use case is already presented, an assembly line design and planification for an aeronautical component, specifically speaking, a wingbox assembly process. A more in-depth depiction of both 3DX and the use case selected will be presented during this chapter.

3.1 Introduction

As has been stated before, MfM methodology is currently being implemented and developed from an academical point of view. Several concepts have been tested and refined using Aras Innovator, an open-source PLM software.

ARAS, an American developer and publisher of product development software, are dedicated to providing a new approach to enterprise PLM that is as flexible and innovative as the products it helps to create. Aras Innovator is a PLM and PDM solution that will helps bringing innovative products to market quickly and support them throughout the product lifecycle.

Aras Innovator is based on a unique, service-oriented architecture that was built from the ground up to be flexible, scalable, and upgradable. It can be easily customized to align with every unique business need and can be seamlessly integrated with the software systems that the Company is already using.

It is because of these key characteristics, plus its open-source nature, that was first selected as the playground for carrying out the different tests needed in order to make progress on the MfM field and the 3LM methodology.

However, it was also required to prove that MfM models and 3LM methodology are feasible and profitable not only under the academic scope, but on a real use case level. In order to achieve that, a professional, reliable PLM software, widely used in the industry and close enough to the specific use case sector was needed. All these considerations, plus its innovative character, led to choose 3DX as the perfect candidate for such task.

It is worth noting that how the implementation of the methodology using 3DX would be made is yet far from

being sorted out, and the following model presented must be seen as an example used by future research to continue with testing and finally achieve some solid and automatic way of translating the Ontology owned by the Company into a fully functional model-based system.

3.2 The product: left wingbox of an aircraft

The use case in this project has not been chosen randomly, but because of its advantages when focusing just on modelling the manufacturing side of the whole work. Both components (left wingbox parts) and tooling were already CAD designed and facilitated, which translates in having already a complete As-Designed and avoiding to incur in extra time and effort.

These models have been given directly from aeronautical sources, which is a guarantee that every part of the subassembly, as well as tooling and jigs, are geometrically precise and correct. These files have been modified to delete parts that were not going to be used and enlighten file weights, which means less computing time when working with 3DX and cleaner trees and diagrams for better comprehension.

The product is shown in Figure 27, and is going to be disassembled so that the different key components can be shown and detailed.

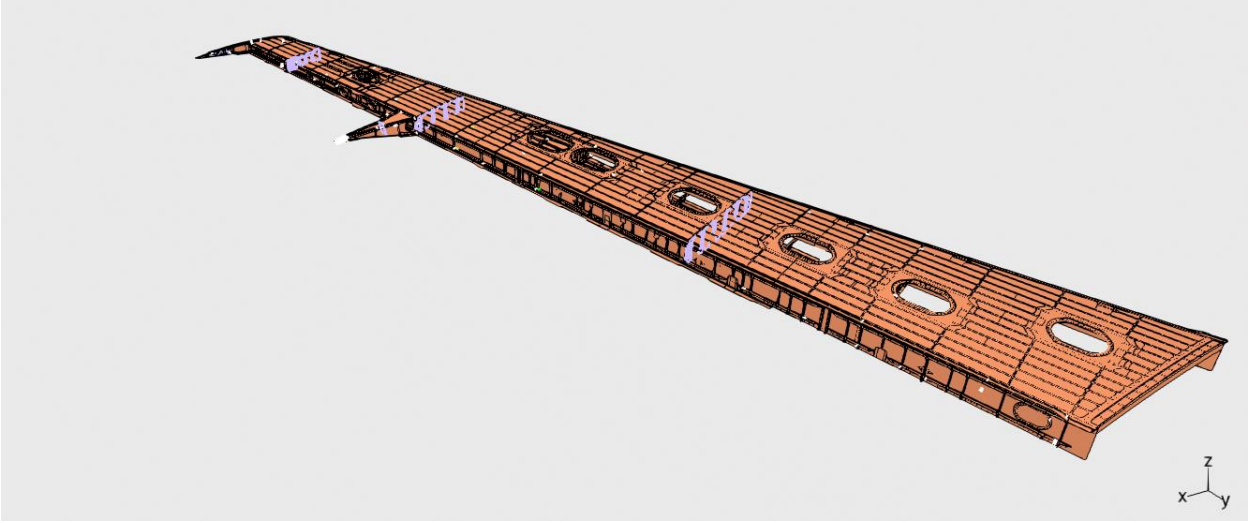


Figure 27. Left wingbox 3D view (courtesy by Nogales [16]).

The most user-friendly way of doing such dismantling is removing the very top section, known as the Top panel. This component is made of smaller panels, mechanically joined between each other, shown in Figure 28. These unions are just set, not needing any sort of riveting.

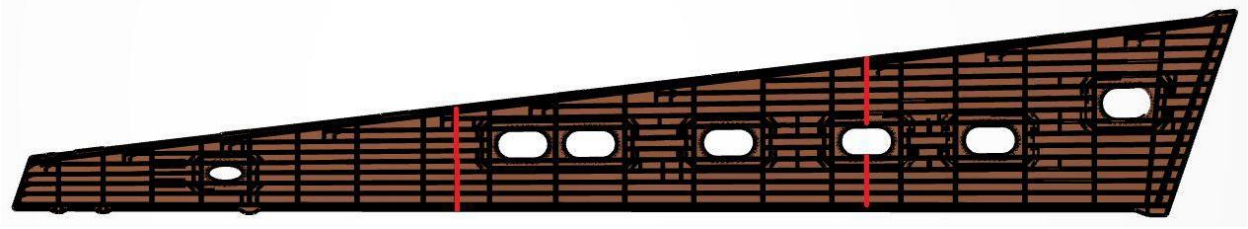


Figure 28.Top panel overview and its mechanical unions (red) (courtesy by Nogales [16])

As can be seen, this panel presents several manholes, for an easy access of workers during the main fuel tanks maintenance tasks, structural inspections, etc. So as to remark the space limitation and potential issues concerning workers' ergonomy, safety and health during its worktime, Figure 29 is presented. Most manholes have edge-protection mechanisms or are supplied via specific tooling.

Figure 29. Manhole access example (courtesy by Nogales [16]).

Similar to what have been already presented, the Bottom panel, opposed to the Top one, is also made of several smaller subcomponents. Unlike the Top panel, this one presents riveted unions between its parts, being also strengthened using gussets so as to get extra stiffness for the whole assembly. The Bottom panel, as well as its unions, is presented in Figure 30.

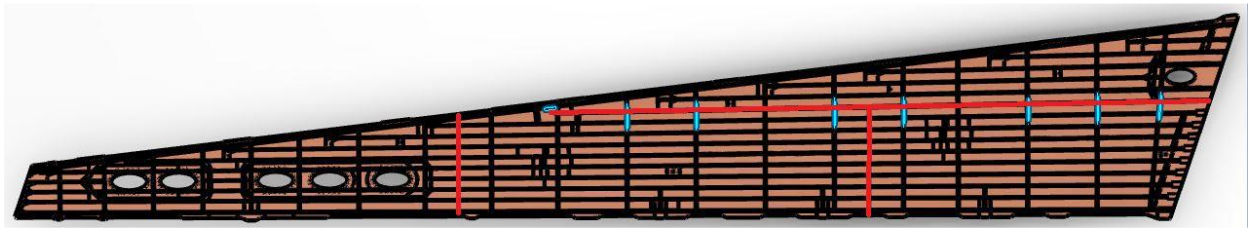


Figure 30. Bottom panel overview and its unions (red) (courtesy by Nogales [16])

In this case, there are only manholes placed on the outer half of the wing (see Figure 30).

Unlike cited parts, Spars are machined from a single preform with stiffners, edges, fillets, supports and accesses. They must take the airflow, facing the bigger loads. Taking into account their relative position in the assembly, they are named as Front Spar, placed on the leading edge, and Rear Spar, located before the high-lift devices. Both spars are shown in Figure 31 and Figure 32.

Figure 31. Front spar (courtesy by Nogales [16])

Figure 32. Rear Spar (courtesy by Nogales [16])

These four pieces constitue the so called wingbox, but it needs to be filled with other parts in order to be rigid and stable. These are the Rib, being this wing made of 6 sheet-metal ribs located throughout its span. It is worth noting that this does not mean a total of 12 ribs along the full aircraft span, because the main ribs, the ones joining wings to the aircraft fuselage, are not being taken into account.

Figure 33 shows that, as most common aircraft, ribs are enlightened and have some bulkheads used to allocate

systems interfaces. Supports and gussets are shown in blue. These allow a high-precision assembly between Front Spar and Ribs, not being present in the unions with the Rear Spar. Besides, the second rib also has a reinforcement in its union with the Bottom panel, presumably because of its closeness to the wingbox geometric centre.

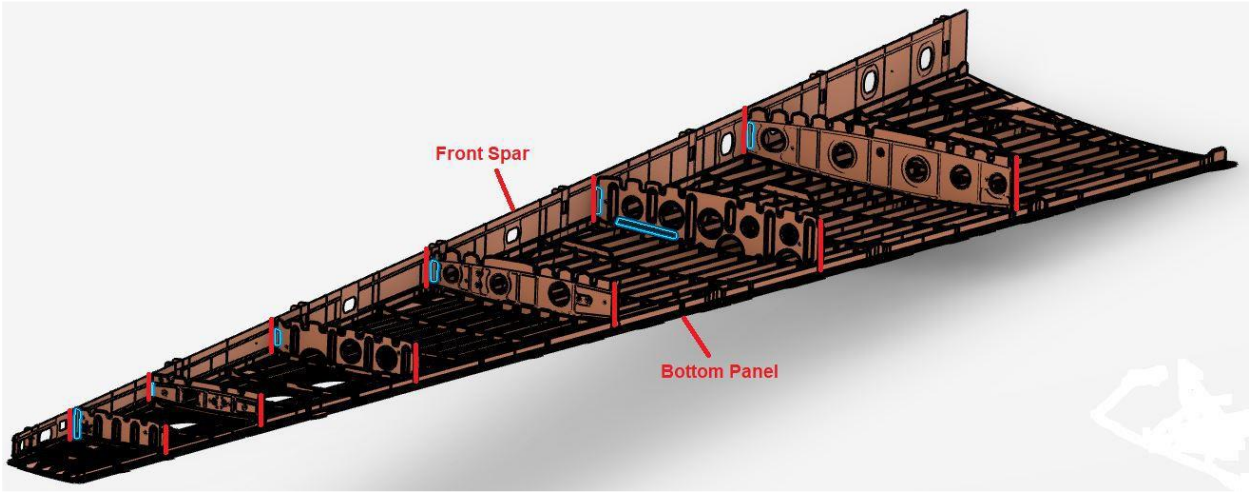


Figure 33. Box insides (courtesy by Nogales [16]).

This model also presents the wingtip rib, which is structurally similar to the previous ones, but is bigger in shape and exceeds the Rear Spar, so as to be used as a guide when positioning the spoiler.

The spoiler rail is included in the assembly, screwed to the wingtip rib (in grey on Figure 34). This must not be put in place until the whole box is assembled in order to avoid possible damage, and it is not needed until the spoiler assembly and regulation.

The wingtip rib has an external support used when mounting the wingtip fairing, is also enlightened and has a bulkhead, possibly so that is possible to do the wiring to the wingtip luminous signaling system. The wingtip rib is shown in Figure 34.

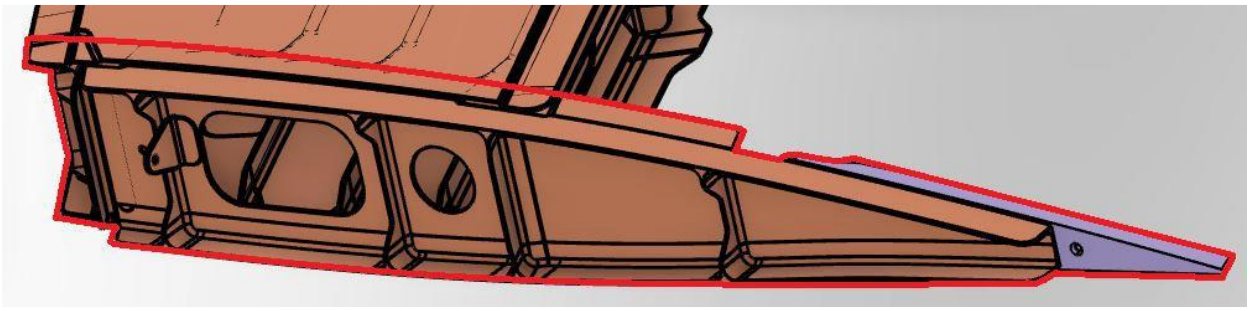


Figure 34. Wingtip Rib detail (courtesy by Nogales [16]).

In order to solve weight limitations and fuel tank accessibility issues, 13 trusses are added between the ribs (hidden in previous figures).

Each truss, made of triangular structures, sums an extra stiffness to the product, while simultaneously letting some flexibility to the structure due to the riveting between truss bars and panels. These bars, their number and lengths, depend on the location of each truss.

Figure 35 shows every truss in place on the Bottom panel. Only two of them (marked in red) were facilitated, being the remaining 11 modelled after the model acquisition in a previous work [16].

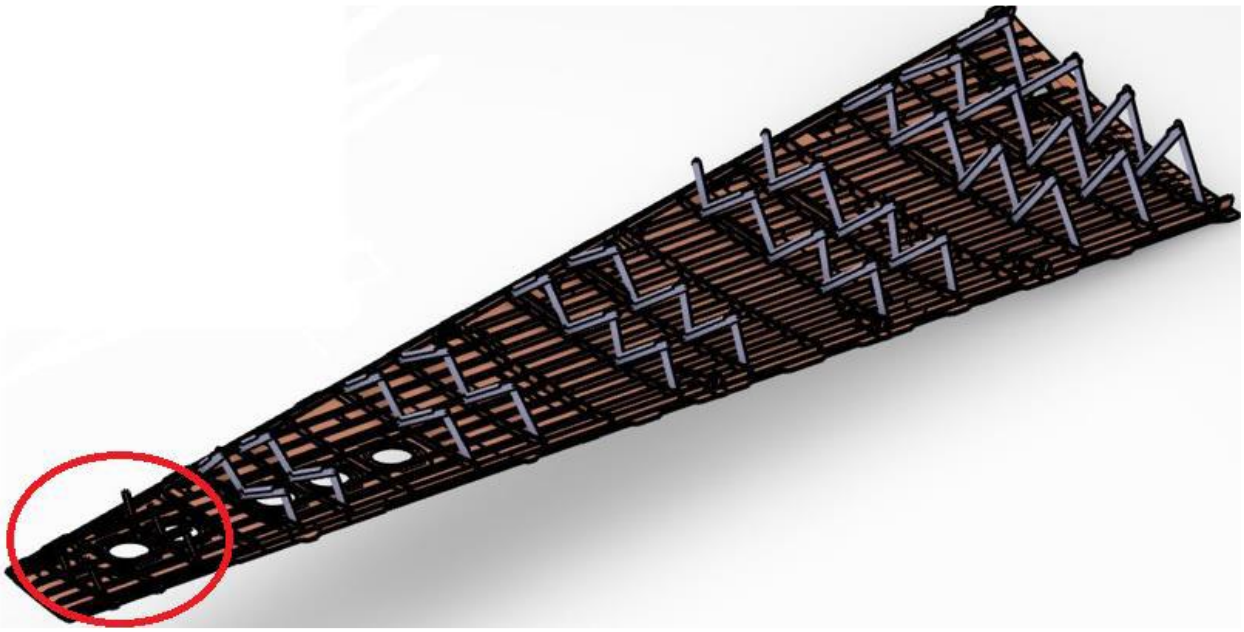


Figure 35. Trusses on the Bottom panel (courtesy by Nogales [16]).

All the previous parts constitute the assembly of the whole wingbox, save the area closest to the fuselage. Due to the assembly procedure, the 3D model also includes a structural mount for the flap movement.

The Flap Fairing is a pyramid-shaped support placed in the Rear Spar, being the flap resting on it during take off and landing maneuvers. The fairing is also used as a supporting and sheltering device for the electromechanic actuators, in charge of the movement of the high-lift device on its rail (in grey on Figure 36).

Unlike what happens with the spoiler rail, added on the final steps of the assembly procedure, the flap rail is assembled with the fairing. That is because the flap fairing is an independent sub assembly and it is also added at the end of the procedure. This Fairing can be seen in Figure 36.

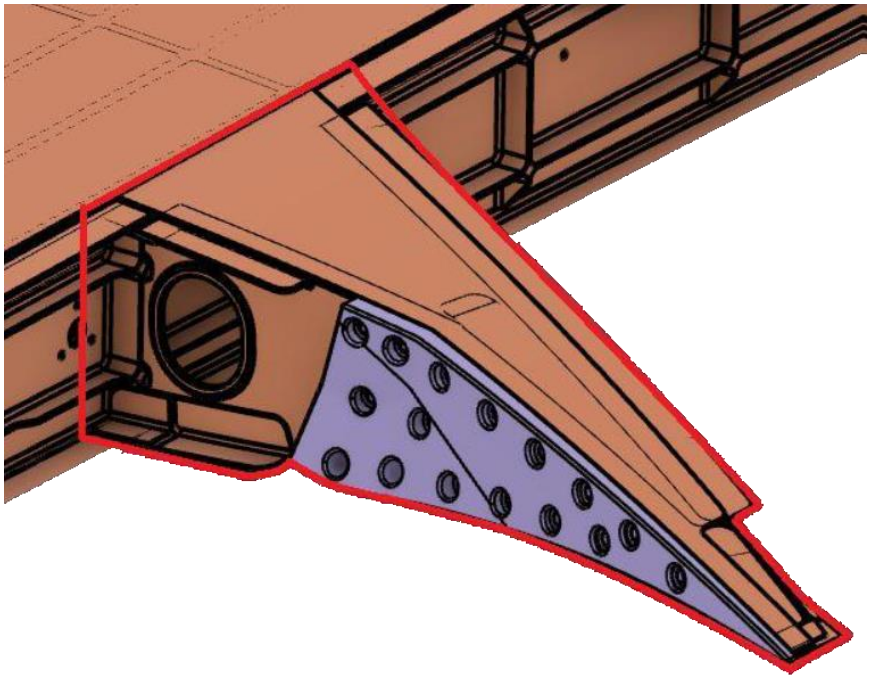


Figure 36. Flap fairing (courtesy by Nogales [16]).

3.3 The wingbox assembly process and resources

Once the final product has been shown, the different assembly strategies made, as well as the final approach selected, are presented. Similarly to what happened between the academic example used for making the As-Designed in Figure 20 and the final product previously presented, there is a big difference in complexity between the original assembly procedure used in the academic use in Figure 20 and the final configuration that is going to be developed on the next pages. This is due to the fact that, as has been said before, an aeronautical use case was needed in order to fully demonstrate the benefits and potential improvements that applying MfM implies.

Before reaching the final configuration, an initial hypothesis was made, and it has been wanted to have it collected here. This first approach was more typical of an aeronautical common use case, thus seemed reasonable at first. However, key issues appeared and forced to think of a different strategy when making the assembly procedure, that will be presented shortly.

At first, it was thought that the very first step consisted of joining both spars and ribs, which would make the 'skeleton' of the wingbox. Then, both panels would be riveted to the spars and ribs.

In order to exemplify this kind of assembly procedure, Figure 37 is added. As can be seen, both front and rear spars are in place and several ribs are fitted between them. Due to the lack of the panels in this stage of the assembly process, no trusses are mounted yet. After this skeletal structure is made, panels are drilled and riveted to it from the outer side, i.e., riveting direction is defined from the panel to the rib.

Figure 37. A400M HTP assembly (courtesy by Benasuly [17]).

Howevcr, it can be clearly seen in the 3D model that these kind of procedure does not match with the wingbox nor the industrial means used in the assembly.

A number of discrepancies found in both wingbox and tooling are labelled now:

- There is physical interference concerning the Rear Spar and the tooling used for positioning the ribs, which indicates that is unfeasible assembling both spars at the beginning of the process.
- The trusses' bars are not able to be riveted from the outside.
- Ribs are internally riveted to the panels' reinforcements and stifffners.

These issues led to change the original approach for a new one, that was then developed in much more detail, being made of four different stages, plus a 'zero' phase used as a pre-stage.

3.3.1 Stage 0: Components preparation

The first one, labelled as stage 0, is the preparation stage. During this one, both panels and spars are being prepared for their latter union.

Panels are assembled in their own tooling, their main parts, systems and support devices. These tooling stations (one for the left wing panels and one for the right ones) are symmetrical, being placed in front of each other and only separated by a central corridor. Each tooling station has a section for the bottom panel and another for the top one. A vacuum system is used to position every panel using the control jigs as guide. The tooling station has different tools supports, air supply, electric power and systems control panel. To carry out the riveting in the bottom panel, a semi-automatic drilling machine is installed. This configuration and specifications are clarified in Figure 38 and Figure 39.

Figure 38. Panels tooling station detail (courtesy by Benasuly [17]).

Figure 39. Panels tooling station (courtesy by Benasuly [17]).

Spars and their supports for flight and electric systems are assembled in their own tooling, being prepared for stage 1. Due to the fact that Rear spars are the component used as a reference for the whole assembly, this preparation stage may sometimes include deburring and final adjustments in said spar. Same as what has been said for the panels tooling, the spars one is symmetrical and is set in the same layout. Unlike panels, that are needed at the same time during the assembly procedure, spars are joined in different moments throughout the sequence, meaning it is possible to use this tooling station at half its capacity, both in terms of manpower and time consumption. A real spar tooling station is presented in Figure 40.

Figure 40. Spar tooling station (courtesy by Benasuly [17]).

3.3.2 Stage 1: Main assembly

Once the main components are prepared, the main phase is started. In this one, both panels, rear spar, ribs, wingtip and the flap rails are positioned, drilled and riveted. Hydraulic pipes for the flight control system are also added in this stage.

The tooling station used in this phase has a main structure (frame) used as a guide for positioning all the different ribs and the wingtip. This frame remains static, while two more mobile structures hold both panels via vacuum and put them in place, using some devices known as 'blades' for their positioning. This tooling can be seen in Figure 41.

Figure 41. Stage 1 tooling station main frame (upper left), mounted bottom panel (upper right) and blades detail (down) (courtesy by Benasuly [17]).

3.3.3 Stage 2: Spar riveting

This phase is remarkable because the box is now placed horizontally, opposed to the vertical positioning that was followed during stages 0 and 1. This way, both spars are freed, allowing the rear spar drilling, made using CNC technology, and the front spar riveting, already drilled during stage 1. The tooling station used during stage 2 is presented in Figure 42.

Figure 42. Stage 2 tooling station (courtesy by Benasuly [17]).

3.3.4 Stage 3: Ribs and panels riveting and closure

Then, the box closure is carried out, riveting the rear spar to panels, ribs and wingtip. Flight control rails are assembled, although only spoiler and flap ones were included in this model. Then, leading and trailing edge ribs are assembled, used as a structural support for the high lift devices, also not included in the model.

These operations are carried out in another tooling station, shown in Figure 43.

Figure 43. Stage 3 tooling station (courtesy by Benasuly [17]).

3.3.5 Stage 4: Aerodynamic surfaces assembly and final tests

Last but not least, leading edge and the rest of aerodynamic surfaces are added. Due to their criticity, several tests are needed to be performed before finishing the assembly. Among them, a 180-degree spin is made so as to check if there is any foreign object damage (FOD) existence, using the tooling machine shown in Figure 44.

Figure 44. FOD detection tooling machine (courtesy by Benasuly [17]).

3.4 MfM methodology using 3DExperience

Now that both the product and the assembly process have been explained, it is turn for the application of the MfM methodology to the use case.

As have been said in this chapter's introduction, a more professional software was needed to be used so as to gain credibility and prove the true potential and reliability of the MfM methodology. The software chosen was 3DX due to its close relationship with the industry, the broad experience in CAD/CAM/CAE software development by their creators, Dassault Systèmes, but also because of its huge variety of standalone tools collected on the same workspace. This versatility allows 3DX to be a great PLM software but also be able to be used as a CAD/CAM tool, and as a Process planning software.

3.4.1 3DExperience basics

It is believed to be useful to give a brief explanation about what exactly is 3DExperience, as well as its basic working philosophy, before getting into detail in this section.

The 3DExperience environment is described by its own creators as a 'Professional platform of experiences, which allows to create exceptional consumer experiences as part of its added value process […]. Based on 3D design, analysis, simulation and intelligence software within an interactive collaborative environment, it is available in both facilities and on a private or public cloud environment'. This definition suggests a new working philosophy, oriented towards a more collaborative workflow between different stakeholders, thanks to its unified interface and its cloud-based character. The 3DX platform is made of four different groups of applications, shown in Figure 45.

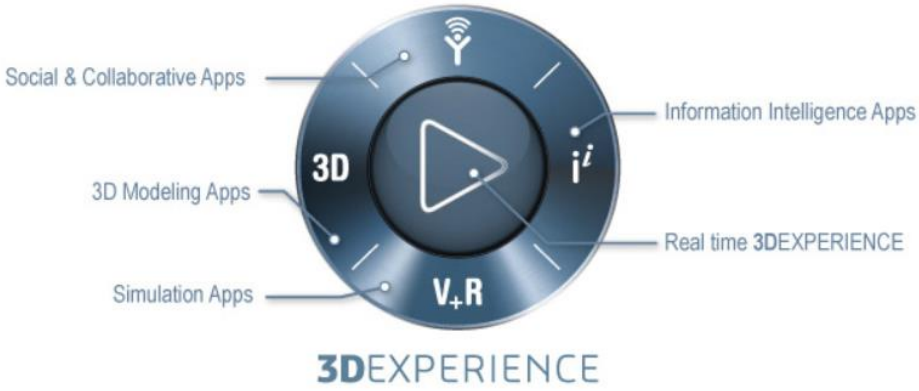


Figure 45. 3DX main groups

Each of these subgroups collects the former Dassault applications, that is, CATIA features are stored in what has been called 3D Modeling Apps, DELMIA and SIMULIA inside Simulation Apps, and ENOVIA characteristics into Social&Collaborative Apps. All these different functions have been complemented with others known as Intelligence Apps, as well as a live 3D environment play using Real Time 3DExperience function.

Due to the clear manufacturing-centered character of this project, only DELMIA apps will be used for developing the use case in the 3DX platform. These different apps, as well as their characteristics will be shown in 3.4.3.

More in-depth explanation of the 3DX dynamics, PPSR tree configuration and apps functions and characteristics can be seen in a previous work of the author [1].

3.4.2 3DExperience as PLM

In industry, product lifecycle management (PLM) is the process of managing the entire lifecycle of a product from inception, through engineering design and manufacture, to service and disposal of manufactured products. PLM integrates people, data, processes, and business systems and provides a product information backbone for companies and their extended enterprise. The vast number of applications included in the 3DX environment, inherited from specific functions present in CATIA, DELMIA, SIMULIA or ENOVIA, allows a complete following of the product throughout its complete lifecycle, from the concept phase to its massive production and later maintenance stages once it is in the market.

3DX apps have their own data structure, which is nowadays an obstacle for correctly applying MfM in this software. Future research needs to be done so as to check if it is possible to define the MfM data structure within the 3DX environment, or if it is necessary to develop some kind of interfaces that would be able to translate the MfM proposed data structure to the one used by 3DX. Given the huge potential and strength that 3DX has as a PLM software, in case these compatibility issues can be solved, it will be used to manage every object's lifecycle present in the methodology, including the model, the meta-model and every instance created using them. Nevertheless, 3DX's process planning side can always be used to implement models and instances, acting as the software in charge of the Service layer. This capacity of embodying the whole model inside a single platform will be proved in next sections. They will show the close relationships existing between objects of the models and 3DX data, interfaces, and commands.

3.4.3 3DExperience as process planning software

As said before, apart from a marvelous PLM software, 3DX also allows to make all the process planning via some of its numerous apps. This ultimately translates into the ability of carrying out a full implementation of Scope, Data and Behavior models into the 3DX framework. This section aims to show which elements if the specific 3DX modules (DELMIA, in this case) would be needed to do such implementation, as well as give a brief explanation about how such modules would be used. This will be made by directly relate objects and concepts from models, already presented in previous figures, with specific existing objects, items or properties inside 3DX applications.

3.4.3.1 Scope Model Relations

First, a brief comparison linking Scope Model to 3DX is made. As was presented in 2.1.2, Scope Model is defined using IDEF0 diagrams, which are made themselves of basic blocks with a main function, inputs, outputs, controllers and mechanisms (Figure 4). The first, obvious relation comes in terms of the mechanism's elements. All those functions using CAx software as a mechanism are directly related to 3DX, being the latter a full CAx tool (CAE, CAPP, CAD, CAM). Apart from that, it has been checked that every function defined in the IDEF0 diagrams of the model (Figure 5 to Figure 11) can be implemented in the several modules inside 3DX, and the software is able to manage inputs and outputs, as well as import the controllers. As an example, the As-Planned definition (MBOM and sequence) can be created and managed using the Manufacturing Item Definition app, As-Prepared (operations definition and assembly line configuration) and line balancing can be made with the Process Planning app, Resources Assignment and its optimization can be managed using Equipment Allocation or generating the documentation via Work Instructions app (although this last one would be complemented with a text processor -office- software).

3.4.3.2 Data Model Relations

Due to their similarity, next sections are going to detail the different relations existing only between certain functions (objects) of the models and their respective implementations in 3DX. Specifically, the As-Planned, As-Prepared and Resource Assigning are compared with the software apps. This will be made for both the Data and Behavior models.

At a Data Model level, the main trees of the app already conform a data structure similar to the one pursued by the MfM model. As was said, future research will say if this data structure would be adapted to perfectly match the MfM one. Nevertheless, 3DX data structure allows to embody all the features inside the MfM one, although organized differently. Thus, a little explanation of said trees is considered useful.

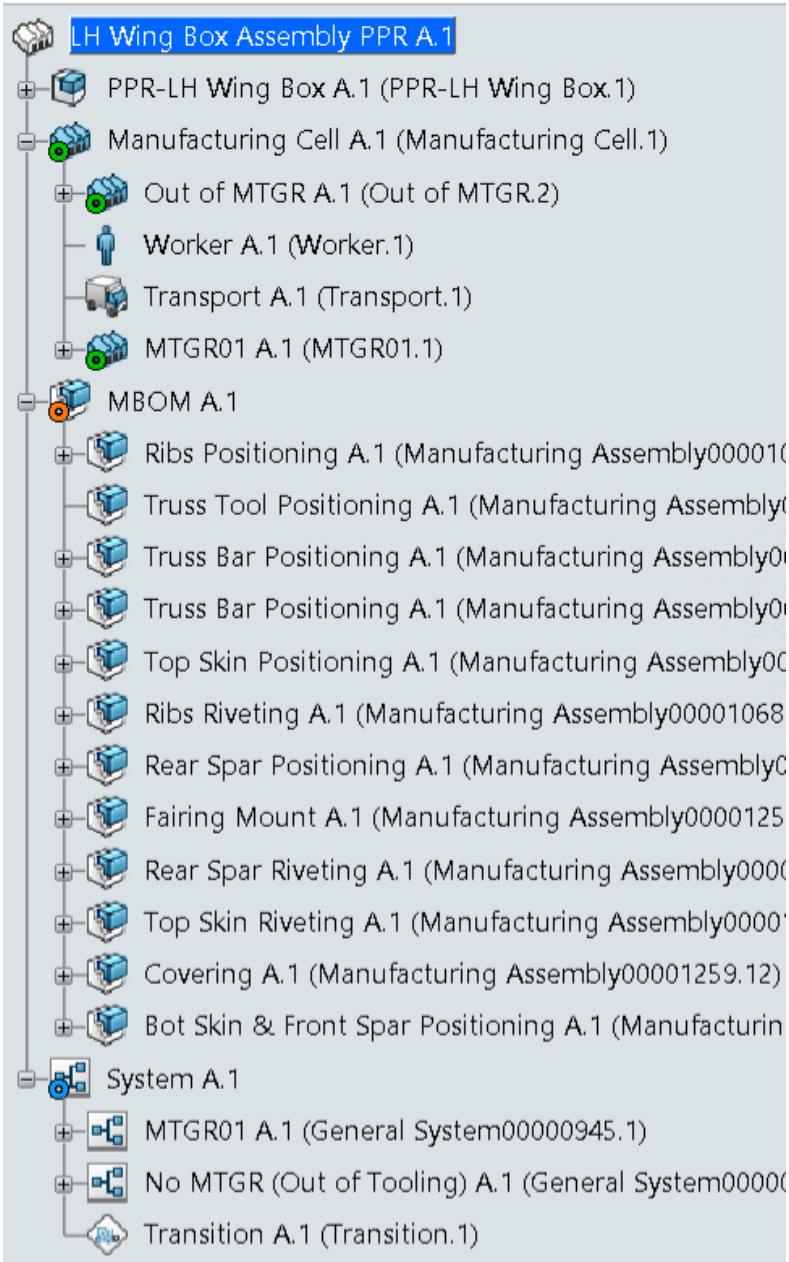


Figure 46. PPSR Tree.

Figure 46 shows a usual PPSR (Product, Process, System and Resource) tree. This tree is made of four different nodes, which give the tree its name. Each node collects its respective objects, whose type are also identified via icons. As an example, a truck icon is used to specify a transportation resource inside its node.

Apart from the tree hierarchy inside each node (inmediately comparable to the concept maps structure of objects and hierarchy), the links between nodes are also taken into account. For this purpose, scope links are created. These are represented with circles under the node icon. In Figure 46, a red circle marks the relation between Process and Product nodes, a blue one between System and Process ones, and a green one between System and Resource nodes. A better clarification of these links is made in Figure 47.

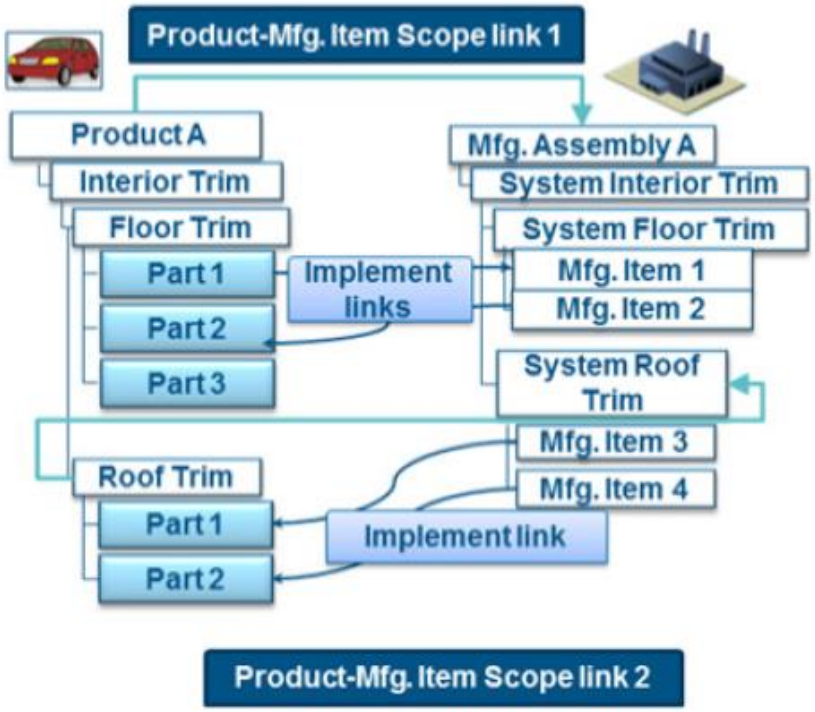


Figure 47. Scope link clarification.

As far as the As-Planned definition goes, Figure 48 (an As-Planned close-up from Figure 19) shows what is needed to create in order to completely define it, being made of the MBOM plus the Manufacturing sequence. The latter is also made of other objects, such as manufacturing steps, manuals and parts. Each of mentioned objects has its own equivalent one inside 3DX, all inside the Item Definition app.

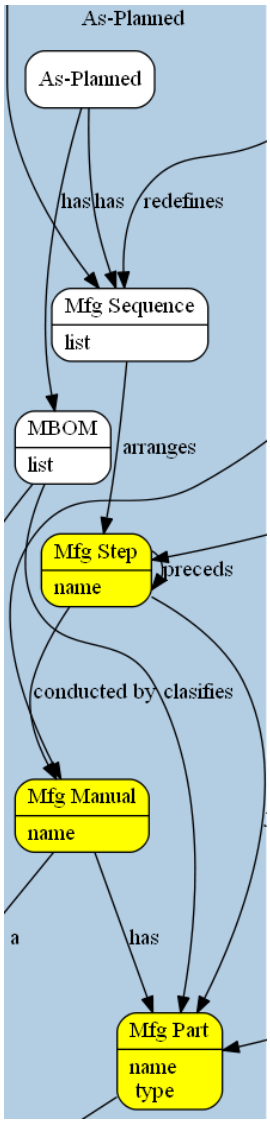


Figure 48. As-Planned close-up.

The necessity of creating a Manufacturing part from the As-Designed part or Manufacturing manuals from the Assmbly ones is justified with Figure 47. This relations are automatically covered once the scope link is created.

Speaking of the As-Planned definition itself, Figure 49 shows a full MBOM created using cited app. As can be seen, not only the part list has been created, but also the precedence relations between them, forming at the same time the assembly (manufacturing) sequence. Once the full sequence is created, its components are also defined, that is, every step, and every manufacturing part. Manuals can also be included in this app using the catalog function. Some detail can be also seen in Figure 50.

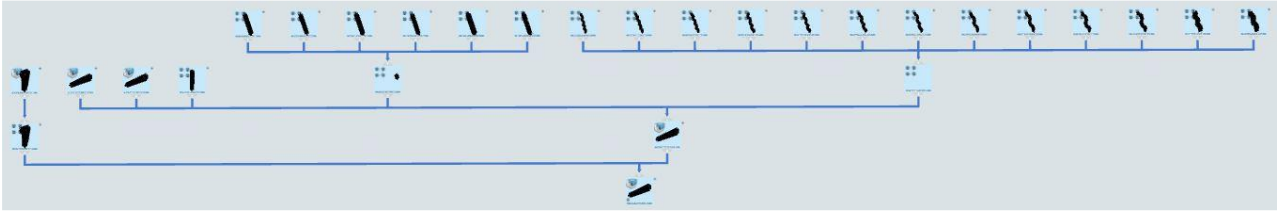


Figure 49. Full MBOM.

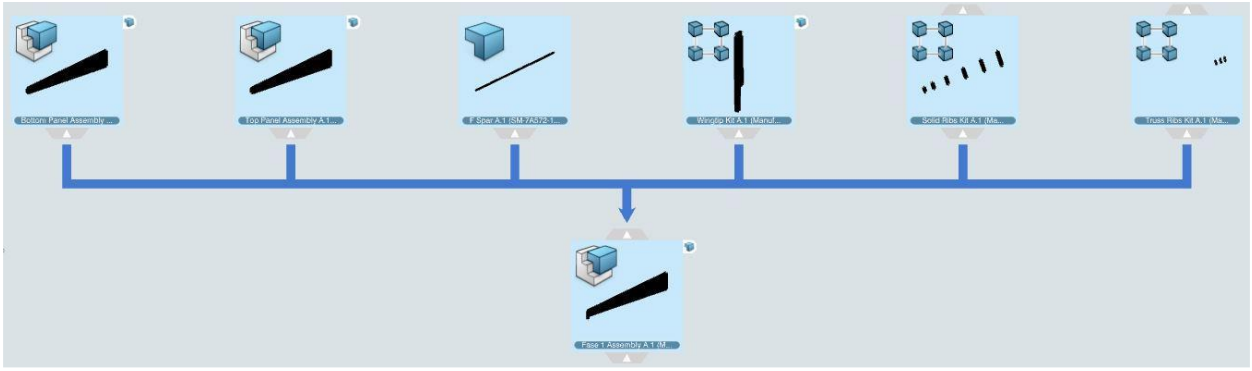


Figure 50. MBOM detail.

Figure 50 is used to explain this dual functioning of the app as both MBOM and sequence definition. Concerning MBOM, it can be seen that there are different icons on the top left of each of the tiles conforming the diagram. These represent the type of manufacturing part that each tile is (one of the object's attribute, see Figure 48). Both panels are product assemblies, rear spar is a provided part, and wingtip, ribs and trusses are manufacturing kits.

Thus, it can be concluded that 3DX can completely define, using just this specific module (as well as its own data structure) the As-Planned Data Model, both objects and their attributes.

Now moving on to the As-Prepared definition, Figure 51 (also a close-up of Figure 19) shows its key components, being the definition of the Assebly line, and the Resource Requirements determination. The first one can be created in 3DX using the Process Planning app, while the latter is defined via the Equipment Allocation app.

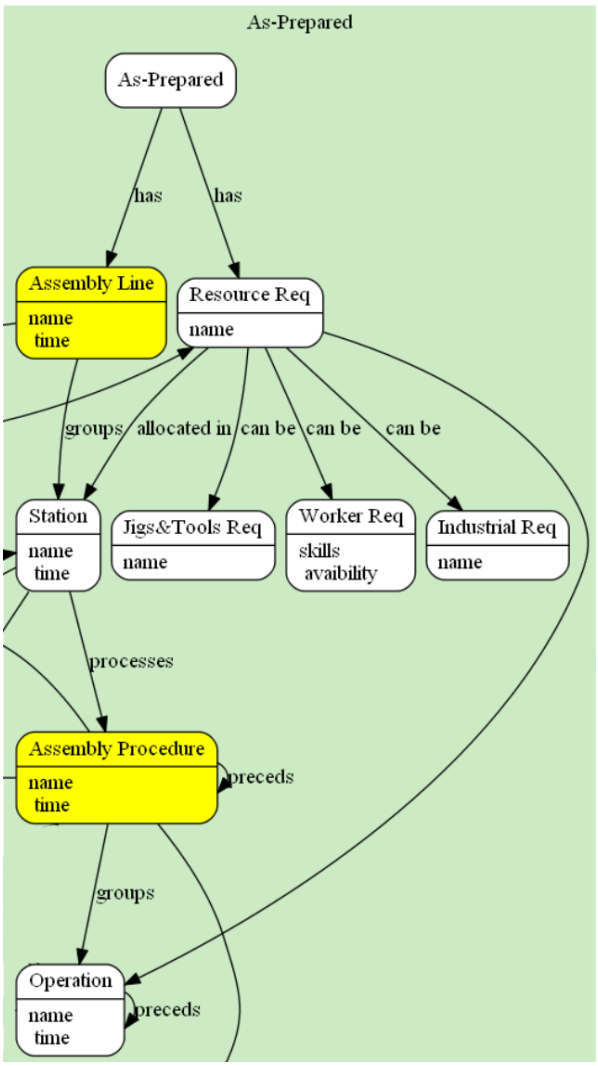


Figure 51. As-Prepared close-up.

First, a close look to the Process Planning app is going to be taken. The assembly line definition implies creating the different operations in the productive chain, grouping them into procedures inside stations, and organizing them to complete the full line. Each of these components need to be defined by specifying their names, types, and times.

The Process Planning app is presented in Figure 52. The application allows to create several lines, and insert stations inside them, filling the stations also with operations, and defining the precedence links between both operations and stations. However, not every group of operations (procedure) correspond to a station, nor every operation needs to be inside a station (Figure 52 shows that transportation operations can be considered on their own).

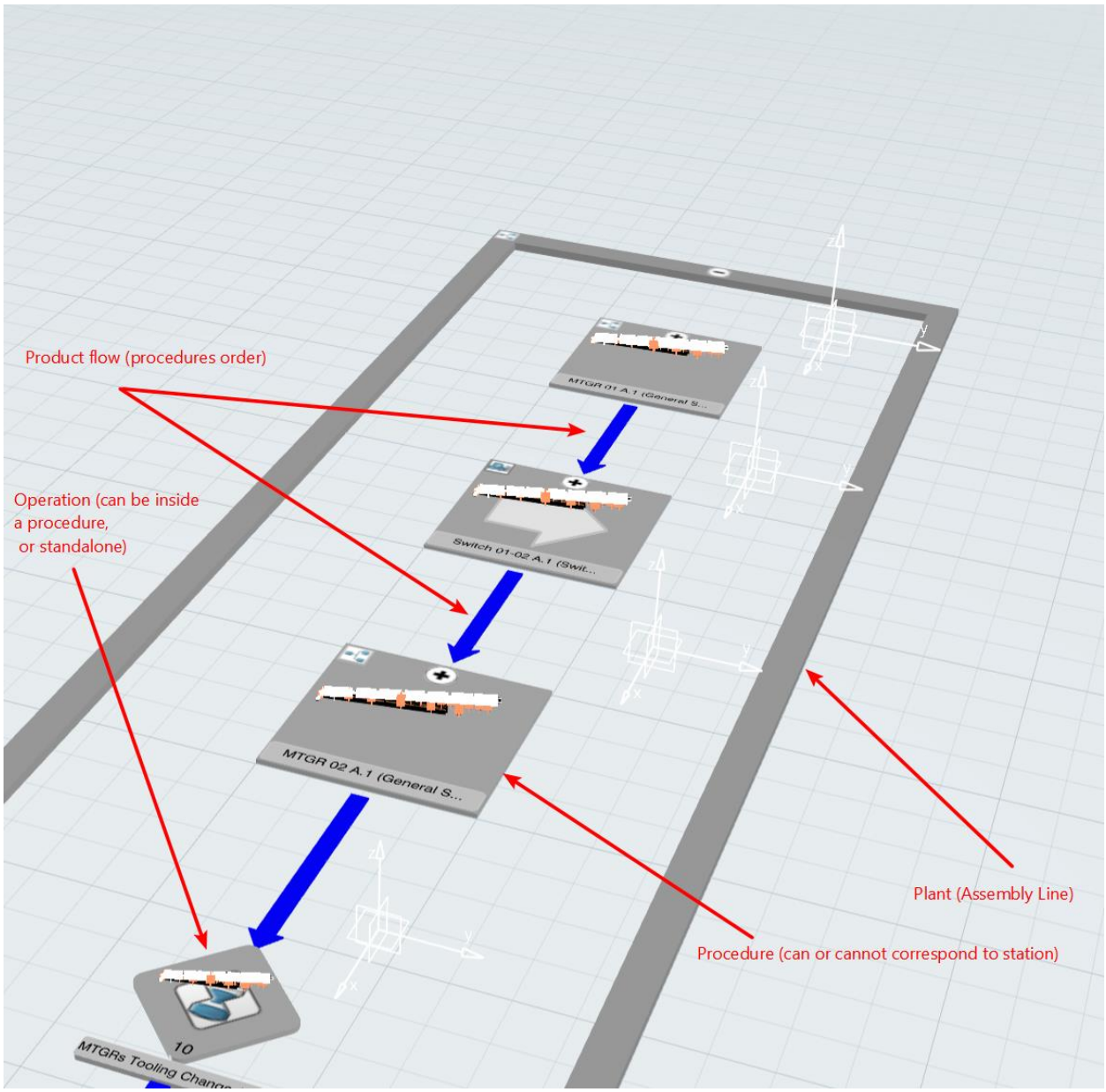


Figure 52. Process planning elements.

The operation creation process, their definition, time assignment and sorting is shown in Figure 53. This specific model is quite linear, due to the linear character that most aeronautical processes have, but parallel dynamics can also be modelled. Figure 52 also shows that the Process Planning app allows to create different types of operation (general operations, loading, unloading drilling and riveting ones shown), assign them an estimated or calculated time (number under each of them) and sort them creating a product flow. Once every operation is created, the grouping task of them can be made easily, thus creating the procedures, stations and ultimately line presented in Figure 52. Their estimated times are computed just by the sum of each operation time.

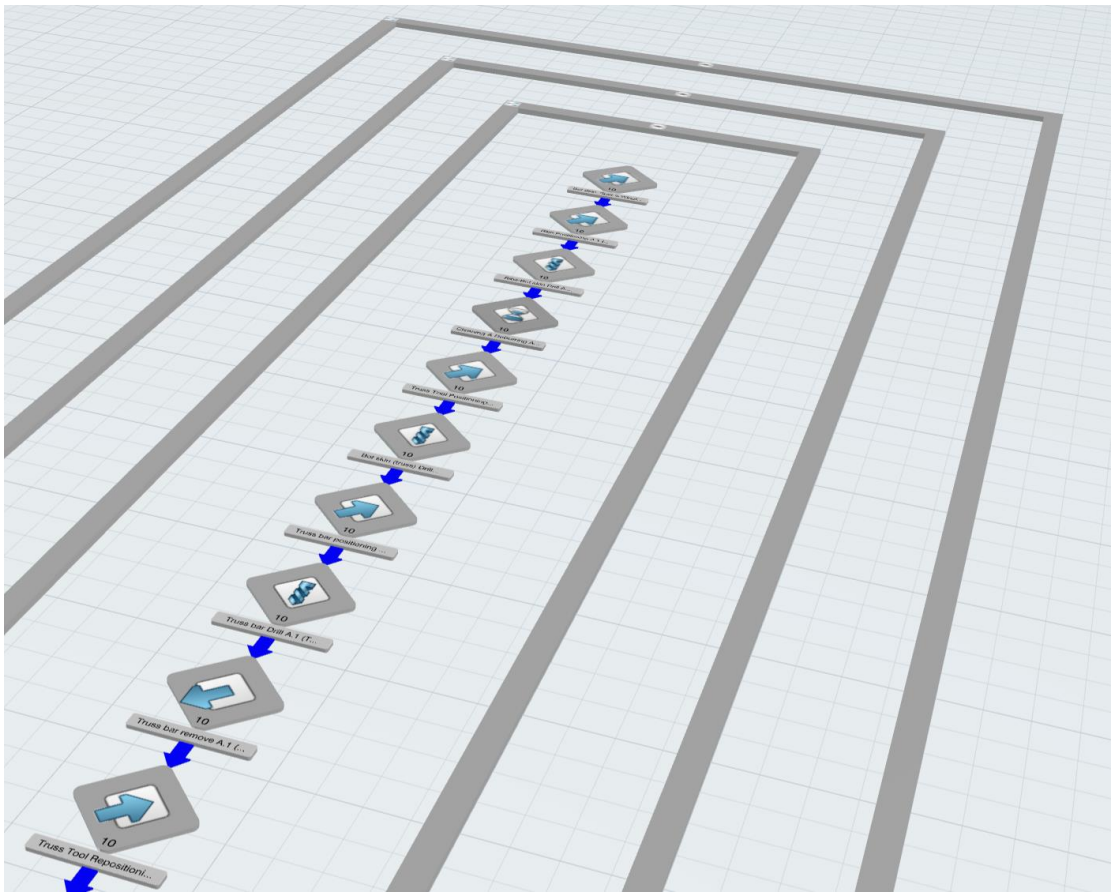


Figure 53. Operation creation detail.

Now moving on to the resource definition, the Equipment Allocation app is used. Figure 54 shows this app, in which have been represented one of the tooling stations described in Figure 41. The app tree shows more different types of resources, that although were not physically modeled, allow to demonstrate the ability of the softwate of creating different kind of resources, as well as their properties (attributes). Every resource type considered in Figure 51 are created in the app: Jigs&Tools (tooling station MTGR shown), Industrial Means (manufacturing cells) and Workers (both worker and transport created). These resources can be created for each station, and analysis tools (such as workshift creation, workload balancing, and others) can also be applied, allowing to define the Feasibility and Balancing objects in Figure 17. An example of this is presented in Figure 55.

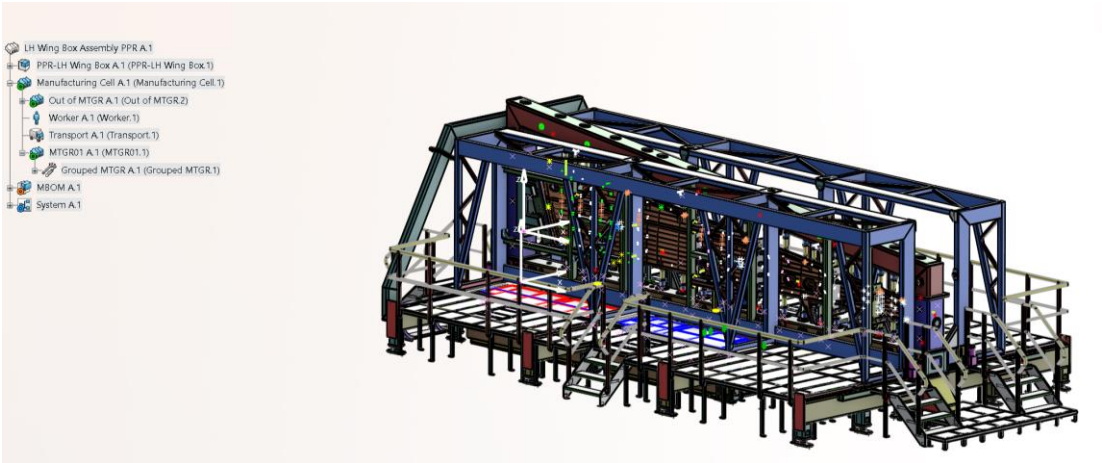


Figure 54. Equipment Allocation tooling overview.

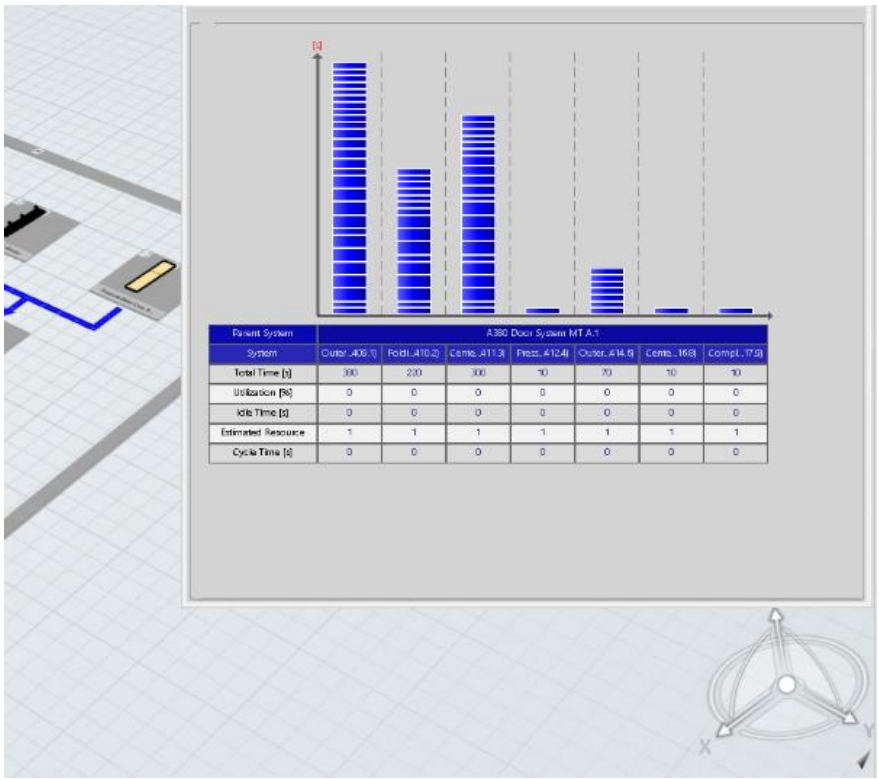


Figure 55. Workload balancing tool.

So, it can be concluded that using both Process Planning and Equipment Allocation apps, the As-Prepared modelling can be fully implemented, as well as the balancing and feasibility analysis objects. Although no more model objects are going to be put here so as to avoid the overextension, it has been checked that the rest of the objects, their attributes and links between them can also be implemented with 3DX apps, and thus, the whole Data Model can be stored using 3DX as the software in charge of the Service Layer.

3.4.3.3 Behavior Model relations

This section aims to follow the same model-3DX objects comparison made in the previous one, this time focusing on the Behavior Model. The equivalent behavior diagrams for the As-Planned, As-Prepared (both assembly line and resource assignment) and balancing and feasibility analysis are considered.

Behavior models are centered about the dynamics of the process, the 'how' things are made. Thus, it seems logic to compare behavior diagrams with the different commands and tools inside the 3DX apps, for they are used to carry out the different actions. The consecution of these actions allows to transform the model's inputs into its outputs.

It is worth noting that due to one of the key requirements proposed for the 3LM models, that is, the ability to be independent of the software used as a Service layer, they are quite vague. This ultimately means that one single activity put in the diagram could sometimes translate into several activites/commands inside 3DX, or the need to accomplish previous activities in 3DX before beginning the 3LM model creation.

In the same way as when talking about the Data Model, the first comparison made is focused on the As-Planned. The behavior diagrams concerning this are activities A11 and A12. Both are presented next (although activity A12 was already shown in Figure 24).

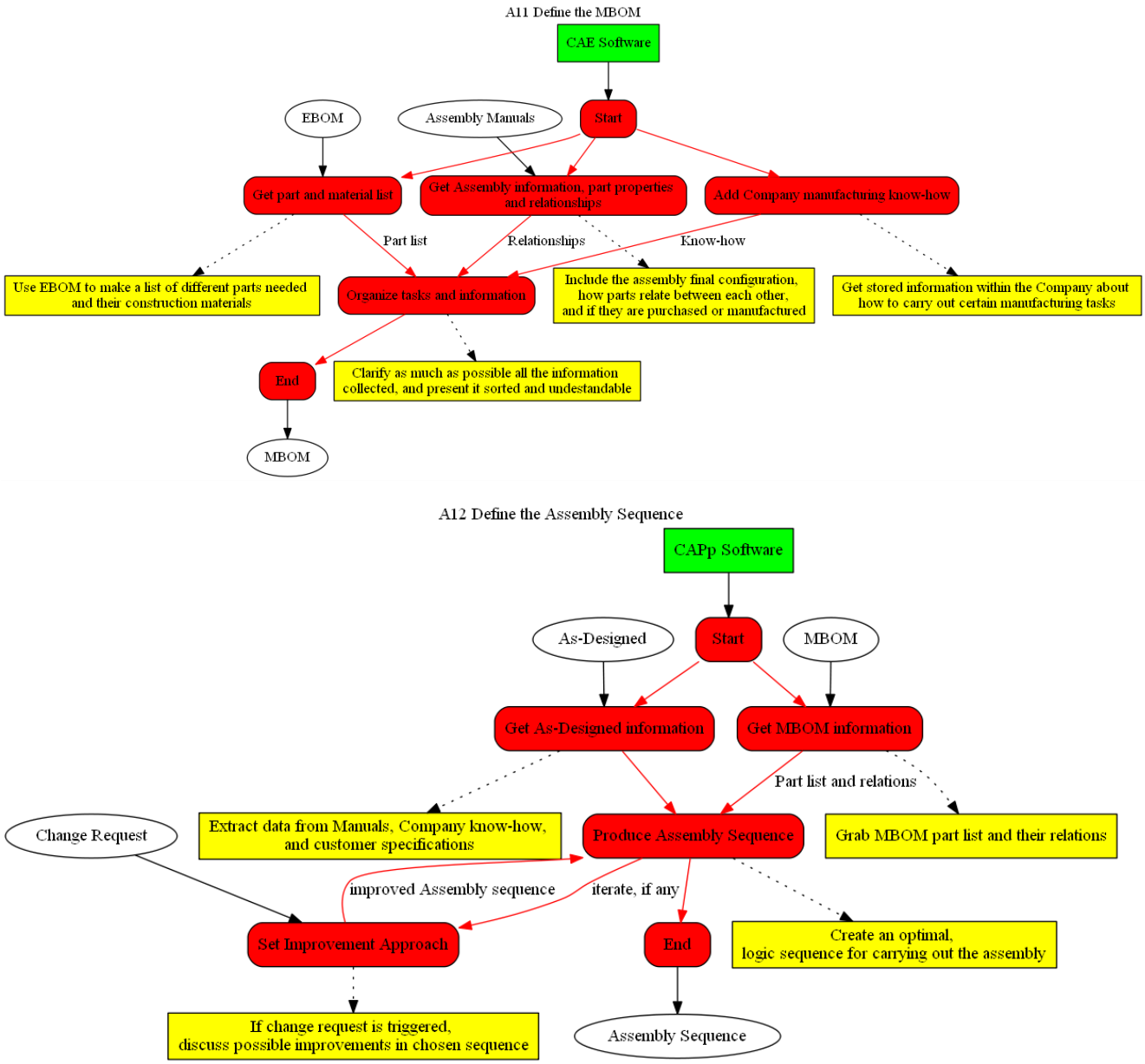


Figure 56.As-Planned behavior diagrams: MBOM (A11, up) and Assembly Sequence definition (A12, down)

The most immediate comparison, seeing Figure 56, is the mechanism used in each diagram, marked in green. Both diagrams note that their whole behavior needs to be done using a CAE/CAPP software, respectively. It has already been stated that 3DX can be used as both, and much more. So, both mechanism labels could be fulfilled with 3DX.

Due to the dual character of the Manufacturing Item Definition app, which defines MBOM and Sequence at the same time, a joint comparison of some of the key activities, noted in red, is going to be made.

First, empty manufacturing tiles need to be created, each of them corresponding to a manufacturing part and step. This is an example of what was commented before, an intermediate step when creating the model that is not collected in the activity diagram, because of the general character of said diagrams. The creation of these tiles also defines their type, being them provided parts, assemblies, manufacturing kits, among others, detailed when explaining Figure 50. Tiles creation is simple, just using the corresponding command.

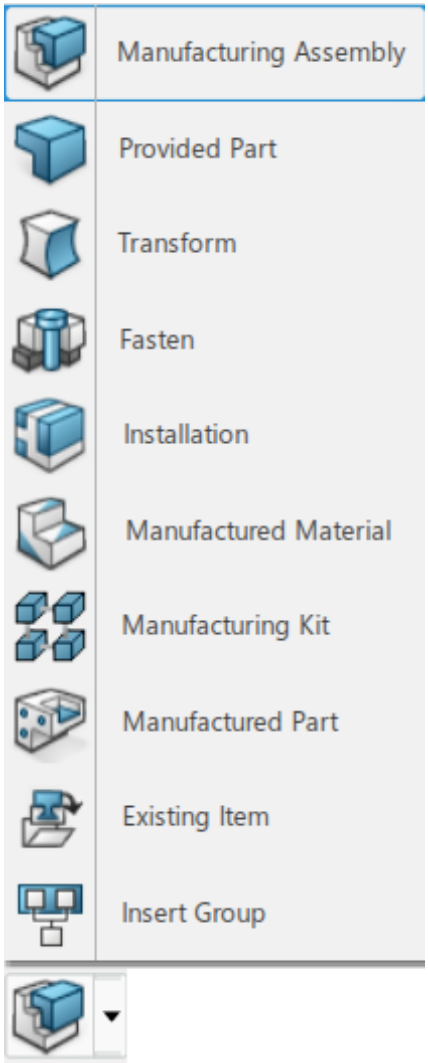


Figure 57. Manufacturing tile (and types) command

Once tiles are made, on one hand, parts and material list (A11) are gotten via the direct link existing between process and product, which has previously defined as a scope link. This also allows to get the information of the As-Designed (A12). The scope link creation process is simple, being made just by clicking its command, then selecting both process and product nodes in the PPSR tree (Figure 58).

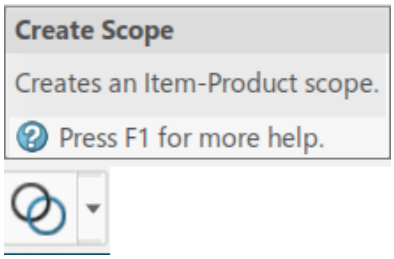


Figure 58. Scope link definition command.

Once the scope is created, product information can be exported to the MBOM/sequence definition just by assigning one or more different parts to each tile created. This can be made just by drag&drop or using the assignment manager command (Figure 59).

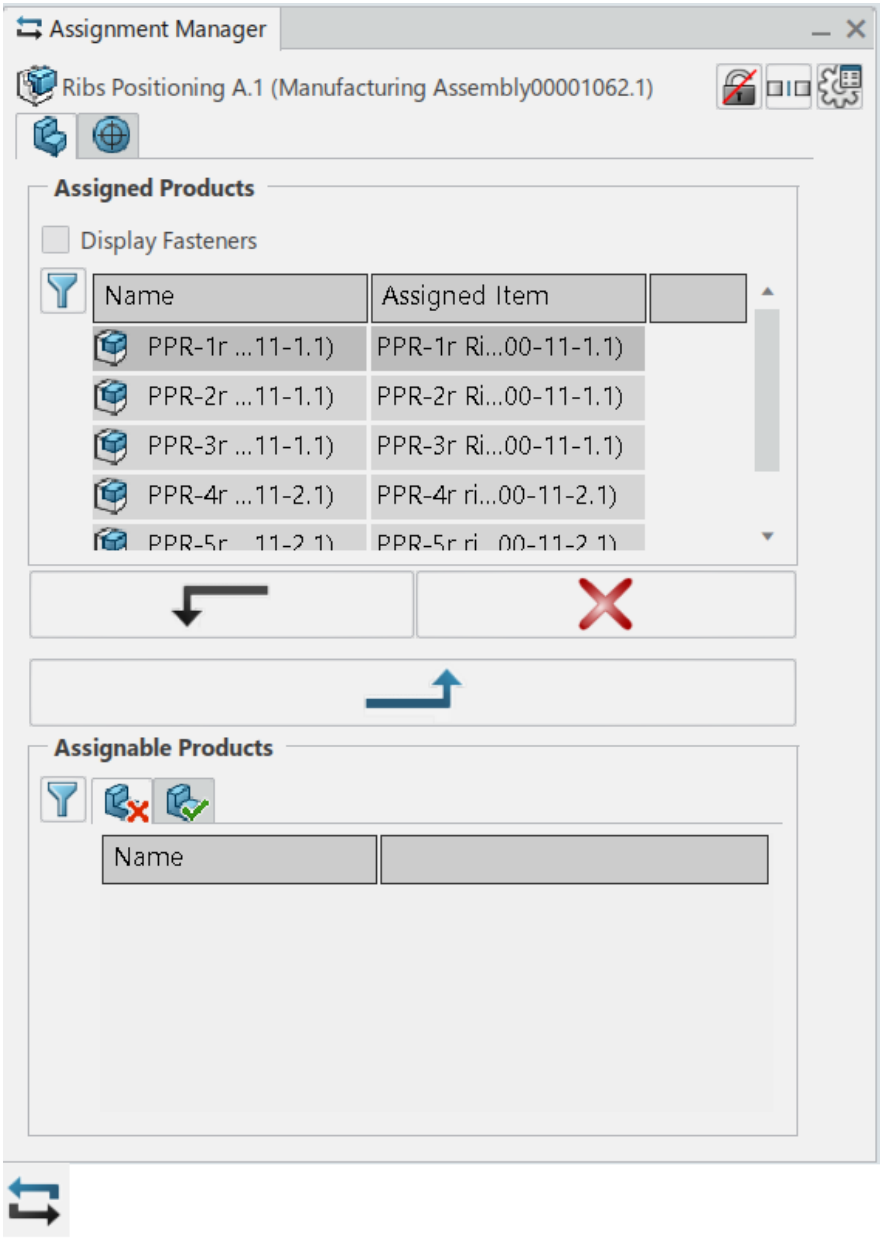


Figure 59. Assignment Manager command and interface.

Manuals information can be exported using catalogs and independent 3DX modules, but have not been fully explored, and so they will be excluded for now. Tasks organization is made easily, by drag&drop mechanics, which allow an immediate rearrangement of the different tiles. Once rearranged, precedence links, and thus assembly sequence itself definition, are created using the precedence constraint command, which introduces the blue arrows seen in Figure 49 and Figure 50.

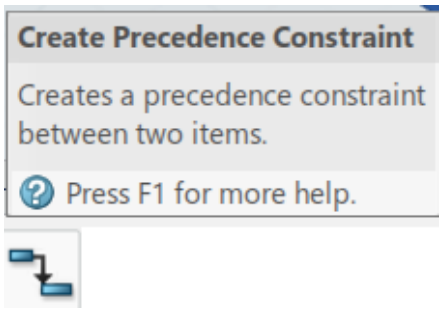


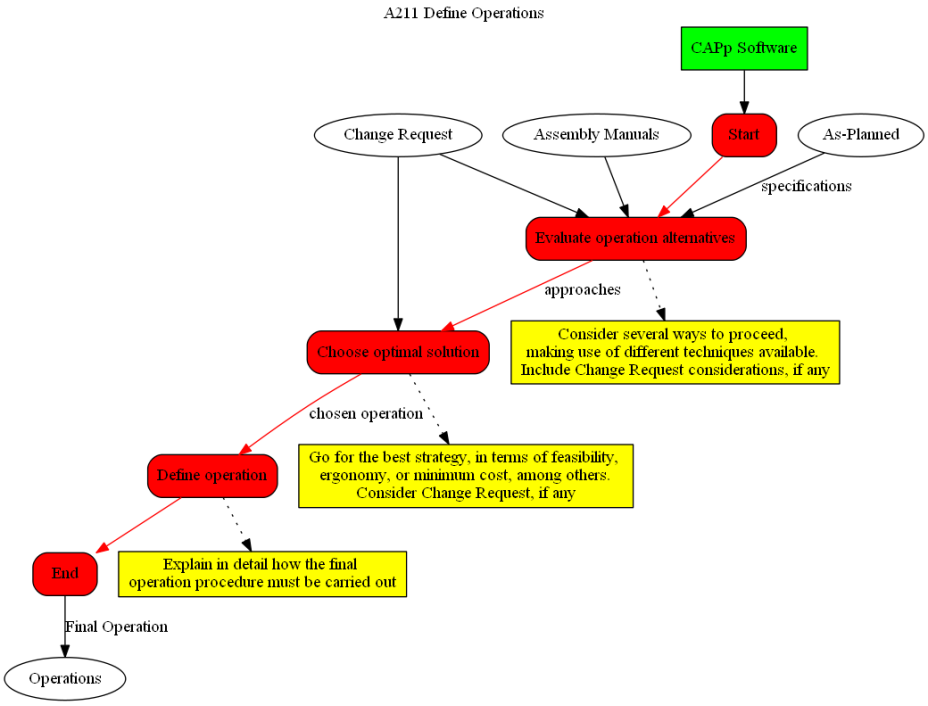
Figure 60.Precendence constraint command.

Following this procedure and using the commands shown, a full MBOM/Sequence (i.e, the As-Planned as it has been conceived) is generated completely. It can thus be concluded that 3DX allows the execution of the As-Planned Behavior Model using just the Manufacturing Item Definition app.

Moving on to the As-Prepared definition, in this case much more diagrams are needed to its full determination. Due to their similarity and intending not to overextend the comparisons, only a few of them are going to be studied. As have been said before, all different diagrams are collected in the Annex, in case they are required.

Chosen diagrams for this comparison are also the main ones: Define Operations (A211), Define and Configure Stations (A212), and Define Jigs&Tools (A221). The former two are part of the Assembly line definition and are going to be implemented using the Process Planning app, while the latter is inside the Resource Assignment, and will be implemented with the Equipment Allocation Module.

Operations and Stations definition, while initially conceived as separate activities, are carried out simultaneously in the Process Planning app in 3DX, in the same way as MBOM and Sequence are created in Manufacturing Item Definition in the As-Planned. Again, behavior diagrams were designed to be very general so as to not be tied by the use of a single software and need to be slightly modified or adapted when using a specific program when on the Service layer. The procedure followed for this comparison will be identical as the one adopted for the As-Planned. Both behavior diagrams are presented in Figure 61.



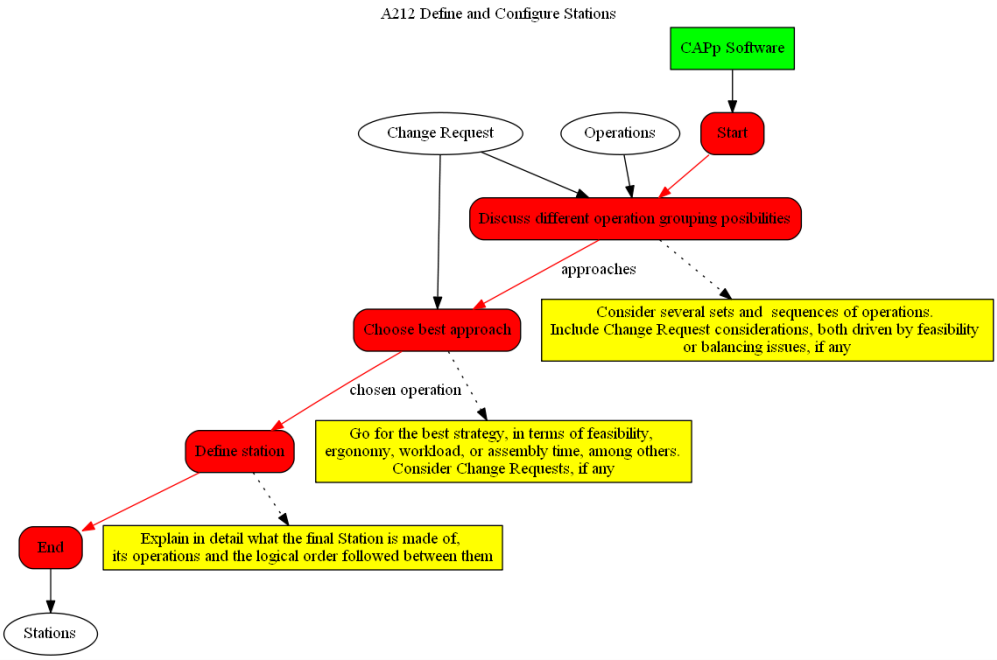


Figure 61. A211 and A212 behavior diagrams

As have been said when comparing diagrams in the As-Planned section, the first immediate relation comes when looking at the mechanism labels (in green). Again, 3DX is used as the CAPP software in which everything else holds up.

Apart from that, it can be seen that the main structure is preserved in both diagrams. Several strategies need to be considered, then studied so as to find the best, and once this decision is made, store the optimal solution in the model.

As happened with the Manufacturing Item Definition app, some previous actions need to be done in 3DX before accomplishing the diagram itself, being the first of them the creation of the workplan and importing the As-Planned information. The former is made by its correspondent command (Figure 62), while the latter is achieved creating another scope link, this time between the Process and System nodes in the PPSR tree.

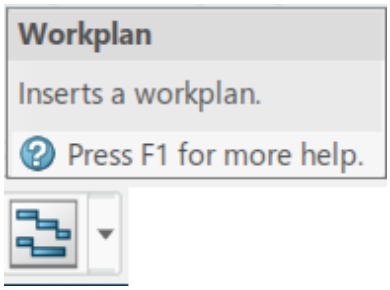


Figure 62. Workplan creation command.

The workplan serves as a frame in which the different operations, system tiles (representing the assembly procedures/stations) and lines are created. Using different workplans, several approaches could be made and analysed, in order to decide which of them is optimal, and then implement it as part of the As-Prepared model.

Operations are created using the command in Figure 63. A lot of different types of operation are available to model, noting the loading, unloading, fastening (riveting) and transfer ones.

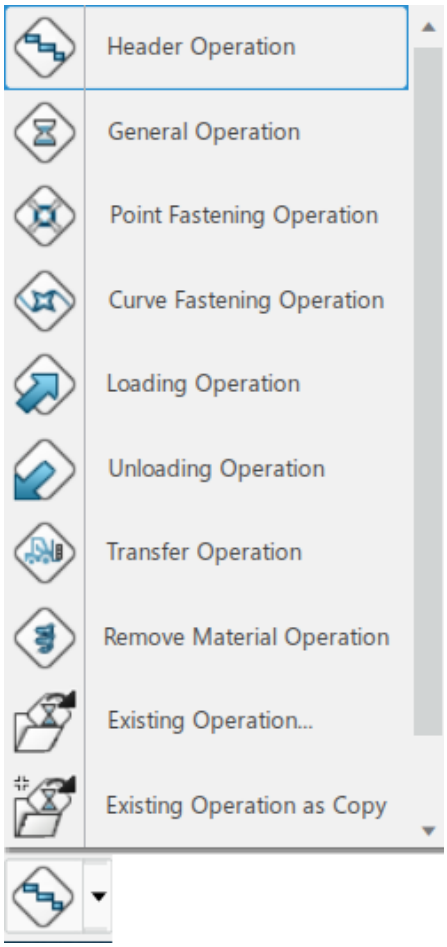


Figure 63. Operation creation command.

These operations can be placed inside the workplan, and rearranged using drag&drop, or put them inside system tiles, which are created using the command for this purpose (the same used to create the initial workplan shown in Figure 62). Different kinds of system tiles are also considered, being used for these comparisons just the general system ones.

Once all the operations and tiles are created and arranged as desired, relations between operations and tiles need to be made so as to create a productive chain. To achieve this, precedence link creation command (identical to the one present in Manufacturing Item Definition app) and product flow creation (Figure 64) are used. Precedence links are exclusive of operations, while product flow relations can be made between systems and operations alike. Tree reordering command is also included, for it is useful when rearranging information inside the model.

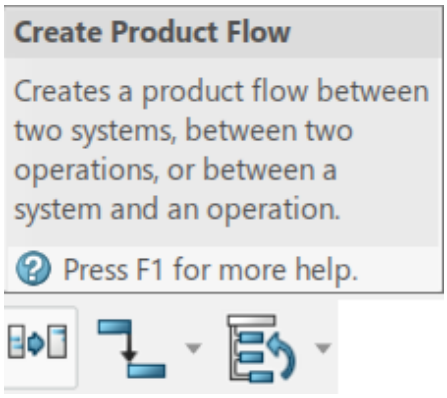


Figure 64. From left to right, Product Flow, Precedence Link, and Tree Reordering commands.

After this arrangement and linking process, a full assembly line scenario is created. Several scenarios need to be modeled, and then some analysis tools can be used to decide the best approach. These tools are the time analysis, which allows to assign the duration of each operation, workload balancing, computing the relative workload of each system tile or group of tiles (already seen in Figure 55), or Premises Usage, which allows to set a production demand and shift models to compute a weighted time of operations.

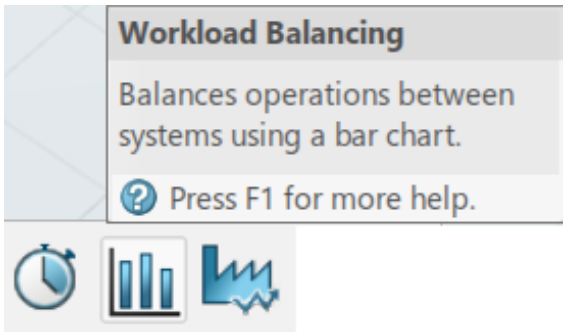


Figure 65. Analysis tools. From left to right, Time Analysis, Workload Balancing, and Premises Usage.

The conjoint use of these tools gives metrics to use when deciding which scenario is going to be finally implemented. The final result will be a configuration similar to what have been seen in Figure 52 and Figure 53, being said configuration the best amongst the different initial solutions considered.

Focusing now on the Resource Assignment activities, due to their great similarity, only the Jigs&Tools definition is going to be studied (A221), whose behavior diagram is shown in Figure 66.

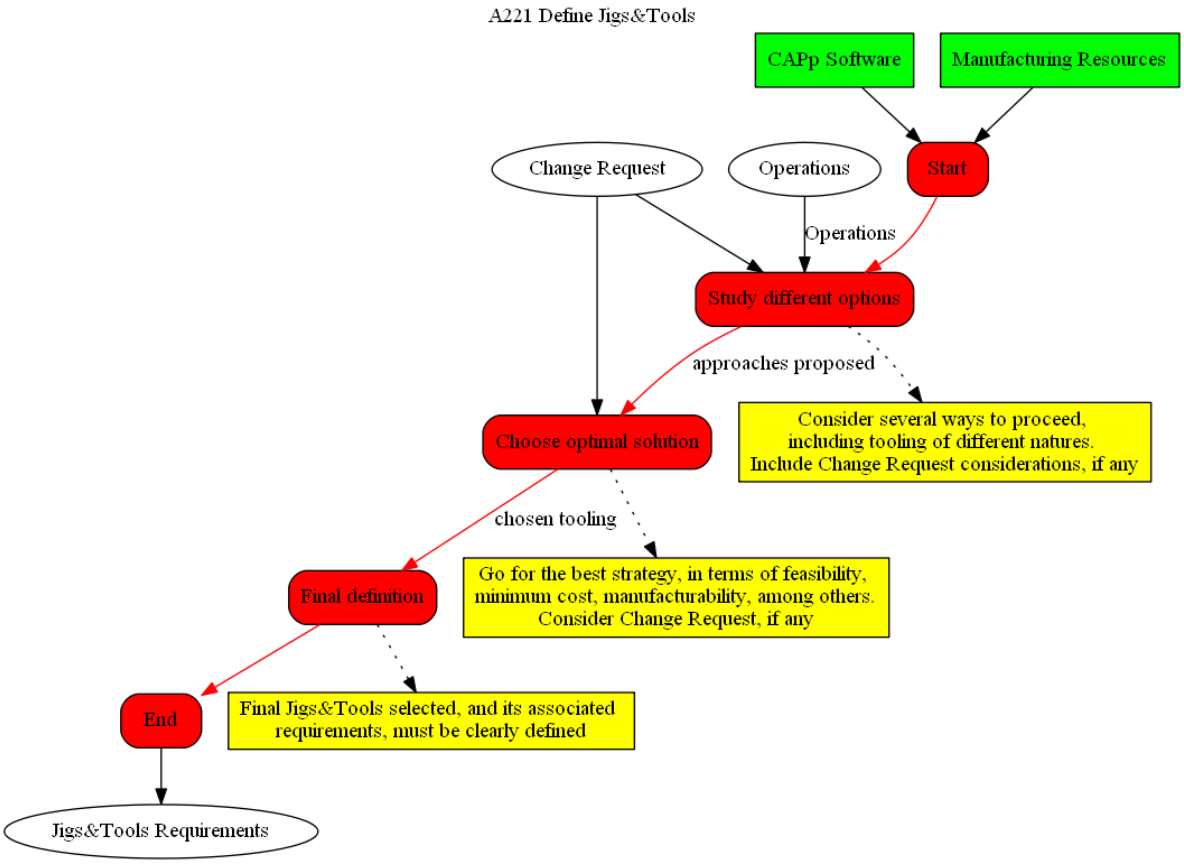


Figure 66. A221 behavior diagram.

As can be seen, its main structure is identical to the one presented in the diagrams in Figure 61. Again, the diagrams' structure has been maintained general so as not to compromise their effectiveness when migrating software. However, due to its belonging to the resource node, another 3DX app is used and thus the commands needed for its creation are slightly different.

Once again, mechanism labels are immediately related. 3DX is used as the CAPP Software to model the diagram, and Manufacturing resources are the actual resources existing prior to the modelling itself. Comparing the actual resources with the ones estimated necessary once the final model is created allows to specify the real resource requirements. These previous existing resources are imported using the 3D models of the resources via scope links and Assignment Manager commands, already shown.

As happened with the rest of diagrams, prior to implement the model, an initial tile creation is required, and information from the previous node is needed. In this case, no such tile exists, but a whole containing resource which embodies the rest of resources, usually one defined as 'Area' or 'Manufacturing Cell'. Lots of other resource types can be defined then inside, presented in Figure 67 (note especially workers, robots, transport, machines, and tooling -hidden in the figure-.). Other familiar commands, such as the scope link creation, used in this case as a relation between System and Reource nodes in the tree, Assignment Manager, to export the system information onto the Resource model, or precedence links creation, whose purpose is identical to the one in other modules, are also shown. Their functioning has already been explained and it is not going to be repeated.

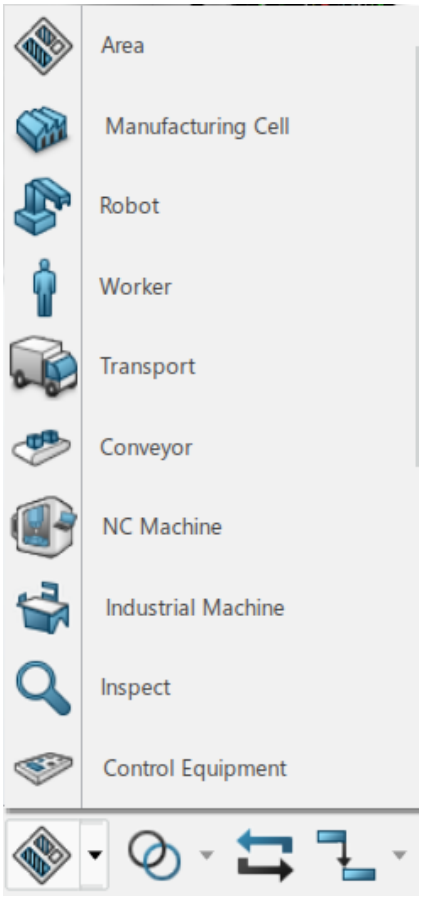


Figure 67. Resource tyopes insertion commands and other useful ones.

The use of said commands, in a similar way as when creating the process planning structure, ends up with a complete resource assignment scenario. Once several scenarios are created, analysis tools come into play so as to decide the best approach, as happened when choosing the best assembly line.

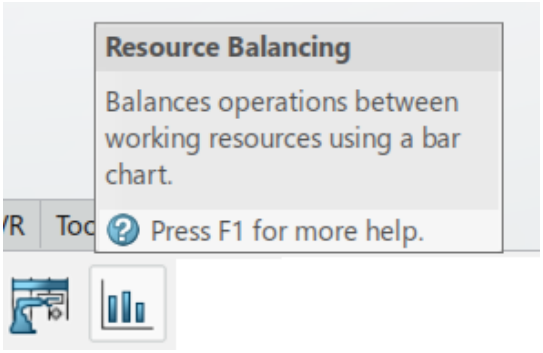


Figure 68. Resource Analysis tools.

Said tools are presented in Figure 68. These are the resource balancing command, which gives as a result a bar chart similar to the one seen in Figure 55, but focused on resources load, capability and availability; and the Resource utilization Gantt, which displays a tree and flow view of resources and operations, providing information about operations assigned to resources, operation times, resource utilization and their associated parts. Using both tools it is possible to determine which scenario is most suitable for the tasks pursued, and fully implement it.

After going through all these commands and functions, it can be concluded that 3DX, using a couple of its apps, is perfectly able to execute the As-Prepared Behavior Model in its totality.

From all seen in the previous sections, the last conclusion is that 3DX can implement the whole Ontology Model (Scope, Data and Behavior) using just some of its apps. The use of more advanced apps and their co-working could define the different Models with more precision, detail and complexity, but due to the academic character of this project, this task is left for future research.

4 SUMMARY, CONCLUSIONS AND FUTURE RESEARCH

Science can amuse and fascinate us all, but it is engineering that changes the world.

- Isaac Asimov -

This chapter first presents a summary explaining the phases carried out during this project, the problems faced and the improvements proposed for the development of the MfM methodology. Then the main conclusions are presented along with a proposal for future lines of research related to this project.

4.1 Summary

MfM has been recently proposed as a way of creating industrial models from manufacturing point of view. This project aimed to collaborate in the development of the MfM methodology and to develop an use case for the design of aeronautical products assembly lines.

The MfM methodology is based on a 3-layer framework and simple and easy-to-use supporting software tools. The layers are known as Data layer, made of databases in charge of physically storing the models; the Ontology layer, core of the methodology, models all the Company know how and holds it so as to prevent its accidental modification or wear over time; and Service layer, which holds the software and tools exploited by the user to access this knowledge. The main advantage of this kind of modelling is the ability to store all the knowledge on an independent construct which does not rely on any specific software to be accessed, modified or exploited, thus making immediate any migration, if necessary. All three layers are connected and fed between each other. This project was only focused on developing the Ontology layer.

3LM Ontology meta-model is based on four different models, connected between each other: Scope, Data, Behavior and Semantic model. This work has studied the three former ones.

Scope Model defines the model boundaries, which is needed so as to be sure about the degree of detail given to the model and is the founding for the rest of the models. It was modelled making use of the IDEF0 diagrams, which have been made using RAMUS software. These diagrams are simple yet powerful and allow a clean and understandable model from the very beginning.

Data Model stores the different concepts or objects extracted from the collective knowledge, as well as their relationships. It is the very core of the Ontology, for it embodies all the know-how owned by the Company. In order to build the Data Model, concept maps were chosen, due to their simplicity, versatility and ease of use and understanding. Two different tools were used when creating the Model, CMap Tools and Graphviz DOT. Each of them has its advantages and issues, so their conjoint use was necessary so as to exploit their benefits. Several Data Models were made using DOT because of its quickness, and CMap Tools was used for

aesthetical reasons.

Behavior Model tries to save all the activities needed in order to perform certain tasks, the way every concept inside Data Model behaves, the 'how' when things are made. Different approaches when building Behavior Models were considered, finally designing a new type of diagram specifically for the MfM methodology. Different color-coded maps were used and named as behavior diagrams, trying to collect the different behaviors of the objects in the models. Behavior diagrams were developed using DOT.

Some obstacles were faced during the Ontology building process. Scope Model was developed without major difficulties, but Data Model proved to be really time consuming due to the way concept maps creation process was. This was what motivated the use of a quicker, more dynamic software which allowed to make minor changes with ease, before adopting a final configuration. DOT came to solve this problem, and concept maps definition speed was greatly improved, allowing a larger number of iterations before arriving to the final map desired.

Behavior Model was also an issue. A lot of considerations were taken into account when trying to determine how a model which stores object's behavior should look like. Some much more complex approaches were discarded because they were really difficult to understand if not used to seeing them, and even when creating these models, their degree of abstraction made them useless once some certain amount of time had passed without being in close touch with them. This led to develop an easier approach, trying to keep things simple without impacting in the model accuracy. Behavior diagrams were conceived, and although they have quite improvement margin, they have proved to be good enough for this preliminary study.

All these models were created for a specific subject, that is, designing an assembly line. Once models were developed, it was necessary for them to be applied and check their effectiveness. This was done instancing the models into a use case, the assembly of a wingbox. This example was considered due to its aeronautical character, plus its balance between simplicity and understandability. A small number of components was chosen so that the diagrams were not so complex, and estimated values were assigned to the different objects' attributes. This exercise allowed to have a preliminar, academic instance to demonstrate the model viability.

Once this preliminar instance was made, a real instance was developed. This tries to be the theoretical result of what a fully implemented MfM methodology would return. Ontology knowledge, physically stored in the Data layer, would be accessed by any software inside the Service layer, and any desired instance could be made using said software. For this example, due to its closeness to the aeronautic sector, as well as its powerful and versatile tools, 3DExperience was used as the program inside the Service layer.

3DExperience is a very powerful software that can be used both as a PLM program and a process planning one. Due to its particular data structure, it is not clear yet if its structure could be adapted to match MfM one and thus could be used as PLM software. Only its process planning side was explored in this project.

The use case selected was now much more complex, aiming to demonstrate the model robustness when working with 'real' cases. A much larger number of components, steps and procedures was considered, both in the product and the processes needed for its assembly. All the different tools inside 3DExperience proved to be able to embody all the models, as well as any instance created from them. This demonstration was made by comparison between each object inside the models and functions or characteristics inside the program.

4.2 Conclusions

As far as it has been seen, the Ontology building meta-model proposed is solid, versatile, and easy to carry out and understand. It is based on four different models, which is a middle point between complexity and accuracy when modelling. These models are also easy to understand, modify and exploit, without losing their inherent value.

The meta-model can be adapted to any degree of detail desired, implemented no matter which software is used, and capable of producing instances with several degrees of complexity. This was proven delivering two different instances, varying their complexity in terms of number of components of the assembly, its steps and resource management. Each instance was modelled using different software, simple tools for the preliminar instance, and a powerful suite for the more complex one. Both instances resulted to be satisfactory in terms of these objectives.

Models were made using open-source, clear, simple, and user-friendly programs, and possible integration with powerful commercial software has also proven to be possible. All software shown in this project meet these requirements, and last chapter demonstrates that a powerful set of commercial tools, such as 3DX, is able to completely implement the whole meta-model and the instances generated.

4.3 Future research

After the developing of this work, some lines of future research have been opened. It is expected that this project can be a starting point for future projects and more in-depth studies about some of the concepts mentioned in this document.

First, Semantic Model was left out of scope in this work, but it is believed that it should not be underestimated when building the Ontology. Having a common glossary so that the different stakeholders can understand each other, avoid concept errors and misunderstandings, and keep pace with the rest of the team when developing new technologies are key for the sake of the Company. A future research should study how this Semantic Model is going to be made, and thus complete the Ontology building meta-model.

As have been said before, 3DExperience was chosen, among other reasons, because of its ability to be used as a PLM as well as a process planning software. Although the latter was proved to work when embodying and instancing the models, the former depends on the capability of adapting its own data structure to the MfM one, or the creation of specific interfaces that could translate one into another. A thorough investigation needs to be done so as to check if this is possible and achievable. As a side note, previous CATIA versions have this function, allowing advanced users to create their own applications and adapt them to their very own necessitites. Hence, it is reasonable to think that 3DExperience should have a similar feature.

On the other hand, a really complex and accurate model and instance was made when evaluating 3DExperience usage as Service layer software. In order to avoid this great effort to be wasted, more future tests should be made using this instance, achieving new milestones in this MfM development.

Finally, during the development of this work it has been noticed that there is an even greater iterative character in building ontologies and its meta-model. Big iterative processes were carried out when refining Data and Behavior Models, but it was also realized that some models changed the perception of how previous models should have been made. This feedback between models suggests a bigger loop of iteration, which was not made in this project. A more in-depth revision of all the models as a whole unit, as well as its refining via iterations, should be made so as to achieve a solid, coherent and accurate Ontology building meta-model.

REFERENCES

[1] G. Álvarez, Planificación con 3DExperience del proceso de ensamblaje de las compuertas del tren de aterrizaje del Airbus A380, Sevilla: Degree Final Project (in Spanish), University of Seville, 2018.

[2] A. Ramos, J. Ferreira and J. Barceló, "Model-Based Systems Engineering: An Emerging Approach for Modern Systems," *IEEE Transactions on Systems, Man, and Cybernetics, vol. 42, no. 1, 2012.,* vol. 42, no. 1, 2012.

[3] S. Mellor, A. Clark and T. Futagami, "Modern driven development," *IEEE Softw.,* vol. 20, no. 5, pp. 14-18, Sep./Oct. 2003.

[4] M. Uschold and M. Gruninger, "Ontologies: principles, methods and applications," *Knowledge Engineering Review,* vol. 11, no. 2, 1996.

[5] J. Bergenthal, «Final Report Model Based Engineering (MBE) Subcommittee. NDIA Systems Engineering Division, M&S Committee,» 2011. [Online]. Available: https://www.ndia.org/-/media/sites/ndia/meetings-and-events/3187-sullivan/divisions/systems-engineering/modeling-andsimulation/reports/model-based-engineering.ashx. [Last access: November 2020].

[6] S. Friedenthal, R. Griego and M. Sampson, "INCOSE model based systems engineering (MBSE) initiative," *INCOSE International Symposium,* 2007.

[7] N. Ivezic, B. Kulvatuyou, D. Brandl, H. Cho and L. Yan, "Drilling down on Smart Manufacturing - enabling composable apps," *Manufacturing Letters,* vol. 10, pp. 14-17, Oct. 2016.

[8] N. Ivezic, B. Kulvatunyou, Y. Lu, Y. Lee and J. Lee, *OAGi/NIST Workshop on Open Cloud Architecture for Smart Manufacturing. Report Number: NISTIR 8124.*

[9] F. Mas, J. Ríos, L. Menéndez and A. Gómez, "A process-oriented approach to modeling the conceptual," *The International Journal of Advanced Manufacturing Technology,* vol. 67, pp. 1-4, Jul. 2013.

[10] F. Mas, J. Racero, M. Oliva and D. Morales-Palma, "A Preliminary Methodological Approach to Models for Manufacturing (MfM)," in *Product Lifecycle Management to Support Industry 4.0*, Turin, Jul. 2018, pp. 273-283.

[11] R. Rizzi Starr and J. Parente de Oliveira, "Concept maps as the first step in an ontology construction method," *Information Systems,* vol. 38, no. 5, pp. 771-783, May. 2012.

[12] M. Austin, P. Delgoshaei and A. Nguyen, "Distributed System Behavior Modeling with Ontologies, Rules, and Message Passing Mechanisms," *Procedia Computer Science,* vol. 44, pp. 373-382, 2015.

[13] A. Szejka, O. C. Jr. y F. Mas, «A Preliminary Method to Support the Semantic Interoperability in Models of Manufacturing (MfM) Based on an Ontological Approach,» de *Product Lifecycle Management in the Digital Twin Era*, Moscow, Springer, 2020, pp. 116-125.

[14] D. Morales-Palma, F. Mas, J. Racero and C. Vallellano, "A Preliminary Study of Models for Manufacturing (MfM) Applied to Incremental Sheet Forming," in *Product Lifecycle Management to*

Support Industry 4.0, Turin, Jul. 2018, pp. 284-293.

[15] F. Mas, J. Racero, M. Oliva and D. Morales-Palma, "Preliminary ontology definition for aerospace assembly lines in Airbus using Models for Manufacturing methodology," *Procedia Manufacturing,* vol. 28, pp. 207-213, 2019.

[16] R. Nogales del Valle, Definición con 3DExperience de la secuencia de montaje del cajón del ala de una aeronave, Sevilla: Degree Final Project (in Spanish), University of Seville, 2020.

[17] S. Benasuly Labuz, *Utillaje de Montaje,* ETSI, Seville, Mar. 2017.

[18] J. Ríos, F. Mas and J. Menéndez, "A review of the A400M Final Assembly Line Balancing Methodology," in *AIP Conference Proceedings - 4th Manufacturing Eng. Society Intl. Conf.*, Cádiz, Sep. 2011.

[19] R. Pinquié, L. Rivest, F. Segonds and P. Véron, "An illustrated glossary of ambiguous PLM terms used in discrete manufacturing," *International Journal of Product Lifecycle Management,* vol. 8, no. 2, pp. 142-171, Jul. 2015.

APPENDIX A: BEHAVIOR DIAGRAMS

All behavior diagrams not shown in this work are collected here, for any reader to consult.

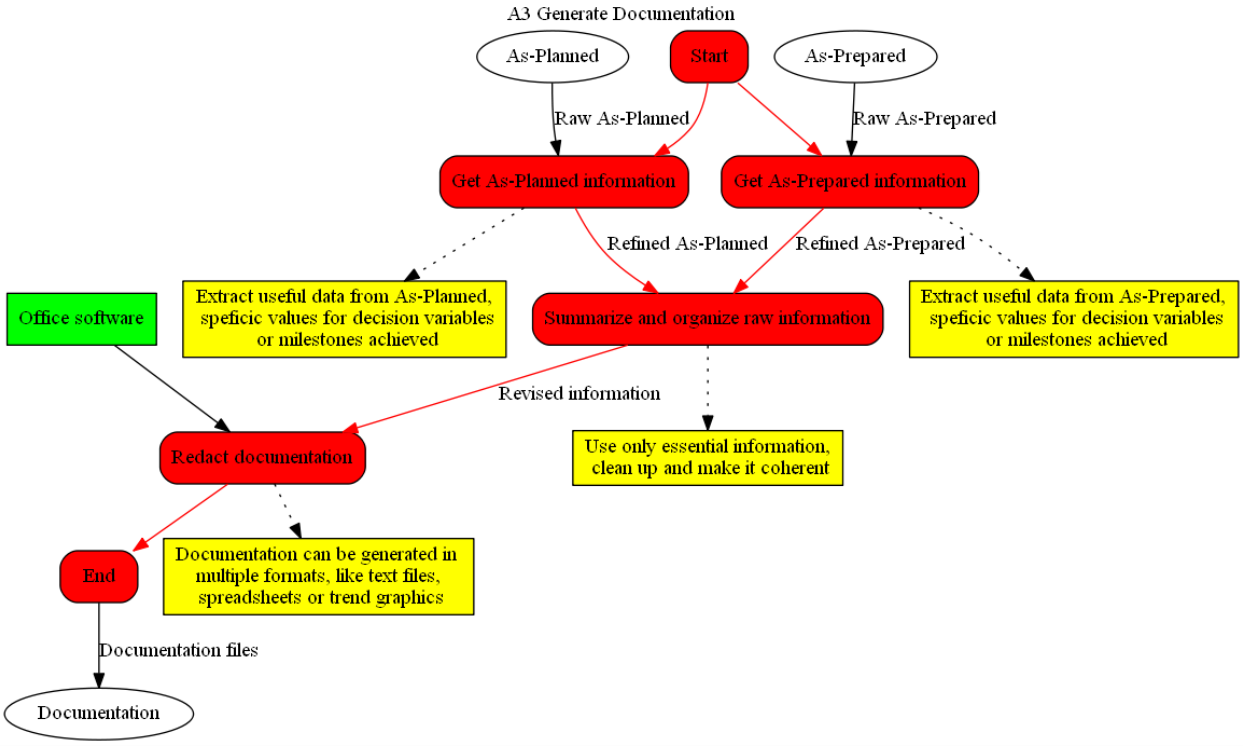


Figure 69. Behavior diagram for activity “A3 Generate Documentation.”

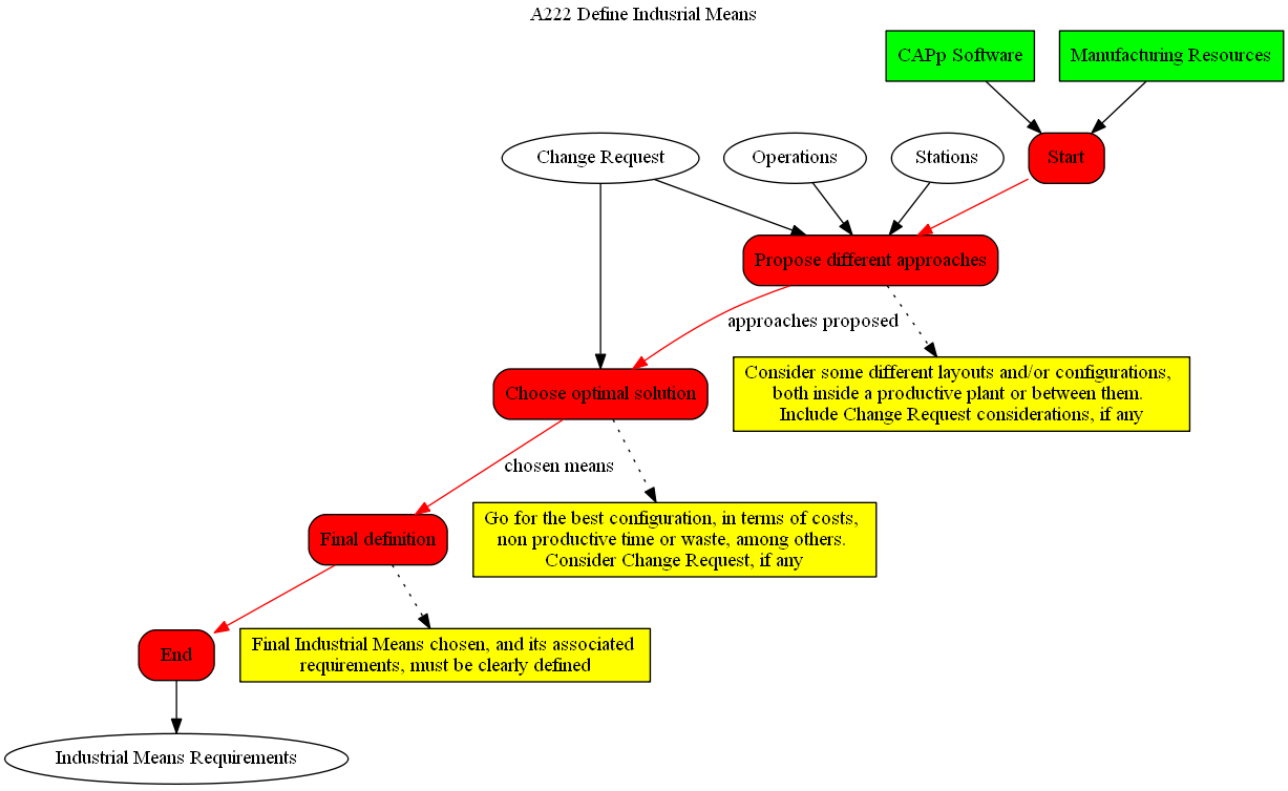


Figure 70. Behavior diagram for activity “A222 Define Industrial Means.”

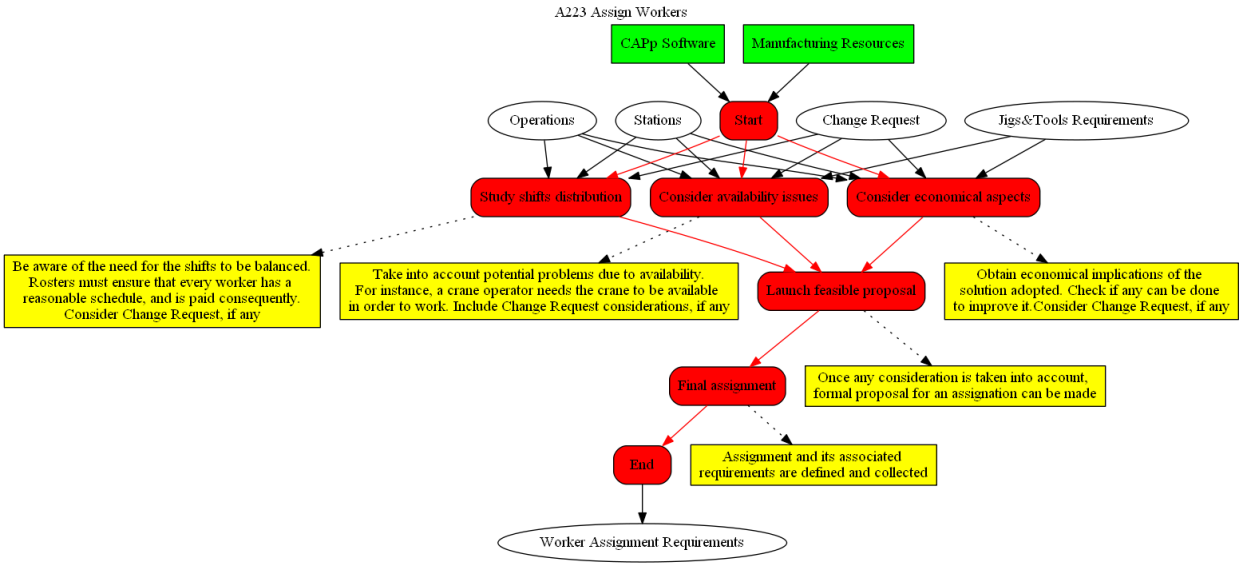


Figure 71. Behavior diagram for activity “A223 Assign Workers.”

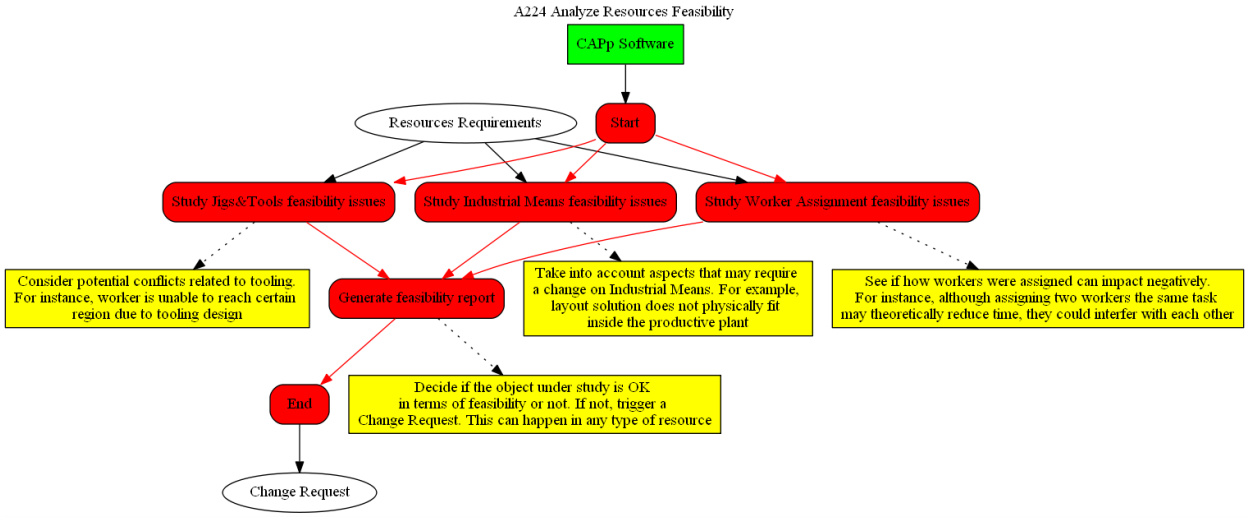


Figure 72. Behavior diagram for activity “A224 Analyze Resources Feasibility.”

APPENDIX B: RESUMEN AMPLIADO EN ESPAÑOL

1. Introducción

Models for Manufacturing (MfM) es una aproximación preliminar hacia una metodología que pretende suministrar una serie de procesos, métodos y herramientas asociadas para ayudar a los ingenieros a cimentar la disciplina de la fabricación en un contexto basado en modelos. Es una propuesta de un equipo multidisciplinar de la Universidad de Sevila, en colaboración con profesionales del sector aeronáutico. El MfM está actualmente en fases tempranas de desarrollo.

La metodología MfM recae en el desarrollo de un marco de referencia, el 3LM (3-Layers Model: capas Data, Ontology y Service), basado en la definición de una ontología de fabricación y habilitando las capacidades de simulación, comportamiento y análisis, priorizando el conocimiento de la empresa. La capa de Ontología es el núcleo del modelo. Contiene todos los procesos de la compañía y su alcance, los modelos de datos y semántica, y las simulaciones asociadas y requisitos de comportamiento. Antes de la realización de este trabajo existían propuestas para los modelos Scope y Data, pero no para los Behavior y Semantic.

Este proyecto pretende colaborar en el desarrollo de la metodología MfM. Las principales contribuciones en este trabajo son: (1) se propone un esquema de representación de conocimiento para modelar el comportamiento del sistema bajo estudio; (2) se analiza la aplicación de la metodología al proceso de diseño de una línea de ensamblaje aeronáutico, y (3) se construye un modelo del proceso de ensamblaje de un cajón de ala de una aeronave en 3DExperience, para que pueda ser usado en futuras pruebas de rendimiento de la metodología MfM.

Como se ha comentado, la metodología MfM surge como una modificación a los enfoques clásicos de la ingeniería de sistemas basada en modelos (MBSE), siempre centrada en los procesos de diseño, orientándose en este caso hacia la fabricación. Esto permite tener una metodología de aplicación durante todas las fases del ciclo de vida del producto, en contraposición al MBSE, solo usable durante el diseño funcional: Diseño Industrial, Producción/Fabricación en serie, y Soporte en servicio. En productos como una aeronave típica, estas fases pueden suponer más del 75% de su ciclo de vida total.

Durante el mencionado ciclo de vida, en el mundo aeronáutico se emplean cuatro tipos de software diferentes para generar, modificar, gestionar o emplear los datos: Computer Aided applications (CAx), Product Lifecycle Management (PLM), Enterprise Resource Planning (ERP) y Manufacturing Execution System (MES). Toda esta información se encuentra dispersa en bases de datos propias de cada programa. Aunque cada uno de ellos garantiza la consistencia de sus datos, no puede decirse lo mismo de los datos empleados entre distintos softwares. Hoy en día, la mayoría de los programas empleados necesitan de interfaces intermedias para su interoperación, a menudo de terceros, o mediante hojas de cálculo de Excel, lo que compromete en gran medida la robustez de los datos empleados y las simulaciones llevadas a cabo.

La solución que se propone pasa por emplear el marco 3LM dentro de la metodología MfM. Se pretende crear una Ontología común para definir, gestionar y mantener el conocimiento de la empresa, así como el metamodelo que permite generar la metodología para la construcción de dicha ontología. Dicha línea está en fases tempranas de desarrollo, y ha sido aplicada con cierto éxito en algunos escenarios industriales, obteniéndose resultados prometedores.

2. La metodología MfM

La metodología MfM propuesta se basa en un modelo de tres capas, el 3-Layer Model. Este modelo garantiza la independencia entre capas, consiguiendo que las capas de Datos y Ontología puedan quedar aisladas, lo que implica la capacidad de definir la ontología sin ninguna interacción con las capas de Datos o Servicios. Esto se traduce finalmente en la obtención de una ontología que puede ser operada, modificada y preservada con independencia del software que se quiera emplear para su consulta, lo que dota al modelo de una enorme flexibilidad e interoperabilidad. En resumen:

- La capa de Datos recoge todas las bases de datos e interfaces, nubes, y varios otros. Dentro de ella están las bases de datos que almacenan la información instanciada usando la capa de Ontología.
- La capa central es la de Ontología, y es el núcleo del modelo. Recoge todo el conocimiento de la compañía, sus procesos, alcance, modelos de datos y semánticos, y las simulaciones asociadas y requisitos de comportamiento. Debido a su importancia, el proyecto se ha centrado en desarrollar esta capa, profundizando en la definición de cada uno de sus componentes, y dejando las capas de Datos y Servicios para trabajos futuros. Dicho proceso se realiza haciendo uso de las aplicaciones definidas anteriormente.
- La capa de Servicios comprende los programas, herramientas, softwares, simuladores, analizadores de datos, o dashboards empleados. Estos servicios se usan gracias a la información almacenada en la capa de Datos, instanciada según la capa de Ontología.

Modelar el know how completo de una empresa no es una tarea fácil, ya que es de gran complejidad y grado de abstracción. Para poder almacenar este conocimiento, se ha dividido el modelo de la ontología en cuatro componentes principales: Scope model, que pretende definir el marco de referencia de la ontología, así como sus reglas básicas; Data model, que recoge los diferentes conceptos conocidos por la empresa, y las relaciones que existen entre ellos; Behavior model, que da a estos conceptos y relaciones un carácter dinámico, como se comportan, y su evolución en el tiempo; y Semantic model, que busca crear un glosario de términos técnicos común que evite los malentendidos causados por las diferentes interpretaciones del lenguaje y las distintas acepciones de algunos conceptos clave en los campos de interés.

- **Scope Model**

  Para poder crear una ontología acerca de cualquier campo, lo primero es definir su *scope*, esto es, su alcance, cuales van a ser sus limites y el grado de detalle que se va a dar a los contenidos de la ontología. Esto asegura que cada una de las partes interesadas trabaja bajo un mismo marco de referencia cuando se construye la ontología, y evita la persecución de cada vez más detalle en los modelos, concepto conocido en desarrollo del producto como *feature creep* y que en este caso podría llamarse *detail creep*.

  La definición del Scope model se ha hecho con diagramas IDEF0, porque permite crear diagramas simples, limpios, claros y fáciles de entender a simple vista.

- **Data Model**

  Después de crear el Scope Model, se crea el llamado modelo de datos o Data Model. En este modelo es en el que se almacena como tal la información que posee la compañía. La técnica más común para esto es usando representaciones gráficas, es decir, haciendo uso de la teoría de grafos, concretamente usando mapas conceptuales. De este modo, los diferentes conceptos se guardan dentro de formas, y las relaciones entre ellos se señalan visualmente incluyendo flechas que los conectan. Normalmente a dichas flechas se añade un conector (habitualmente un verbo), que dotan de mayor información a la conexión. Para que estas representaciones sean efectivas, es necesario crear primero una codificación de colores y formas dentro del mapa conceptual.

  Inicialmente, el software empleado fue CMAP tools. No obstante, debido al alto carácter iterativo que presenta la creación del modelo de datos, con el fin de agilizar dichas iteraciones, se optó por emplear de forma complementaria DOT. Mientras que el primero es un programa basado en una interfaz (GUI) y mecánicas drag&drop, que lo hacen fácil de usar y altamente personalizable, el segundo es un compilador de gráficos basado en texto. Si bien su personalización de los mapas es algo más limitada, permite generar mapas de calidad suficiente sin más que escribir un pequeño *script*, y lo que es más importante, realizar modificaciones sobre ellos con muy poco esfuerzo, lo que ha permitido que las

iteraciones entre versiones del modelo de datos se lleven a cabo con gran rapidez, algo impensable empleando CMAP Tools.

Tras la creación del modelo de datos, este puede ser desarrollado en más detalle. Se dice entonces que el modelo se ha enriquecido con nueva información. Puede añadirse cuanta se desee, siempre teniendo en cuenta no sobrepasar lo establecido anteriormente en el Scope Model. Este enriquecimiento pasa por añadir nuevos conceptos que, aunque inicialmente no se creen necesarios para la definición del modelo, pueden aportar información extra a la hora de dotar de detalles. Además, dichos conceptos pueden poseer atributos, propiedades usadas para definir mejor cada concepto.

- **Behavior model**

  El Data Model agrupa muchos conceptos y sus relaciones, conformando el conocimiento de la compañía. Sin embargo, estos conceptos y sus relaciones causales deben ordenarse de acuerdo a algún criterio. Por ejemplo, pensando en un proceso de ensamblaje, aparte de conocer qué partes han de ensamblarse y sus posiciones relativas, también ha de saberse la secuencia de ensamblaje, esto es, el orden temporal en el que las piezas se unen entre sí, y también cómo se llevan a cabo exactamente las operaciones de ensamblaje. Estos "cuándos" y "cómos", entre otras cosas, se recogen en el modelo de comportamiento o Behavior Model. En esencia, almacena cómo se comporta cada concepto y relación dentro del Data Model, con respecto a sí mismo y a otros. Por tanto, el modelo de comportamiento debe ser una evolución del modelo de datos, y utilizar a este último como punto de partida.

  Se han seguido diferentes estrategias para obtener un modelo de comportamiento que, si bien fuese simple y fácil de entender, permitiese también cumplir los objetivos marcados para el mismo. Finalmente se opta por lo que se ha denominado diagrama de comportamiento. Debido a sus similitudes, se ha empleado DOT como herramienta para modelar estos diagramas, al igual que con los mapas conceptuales.

- **Semantic model**

Es necesario crear un modelo semántico al construir una ontología. Como se ha comentado, el principal objetivo de crear una ontología es preservar el conocimiento de la compañía de forma que pueda ser empleado por cualquiera, desde cualquier lugar, haciendo uso de cualquier software. Para garantizar eso, es imprescindible desarrollar un lenguaje común de forma que miembro del equipo que vaya a usar la ontología entienda del mismo modo (tenga la misma acepción) de cada concepto y definición cuando se hable de cualquier tema. El modelo semántico pretende conseguir un total entendimiento entre las partes interesadas, evitando que los malentendidos se produzcan y proporcionando un marco semántico común.

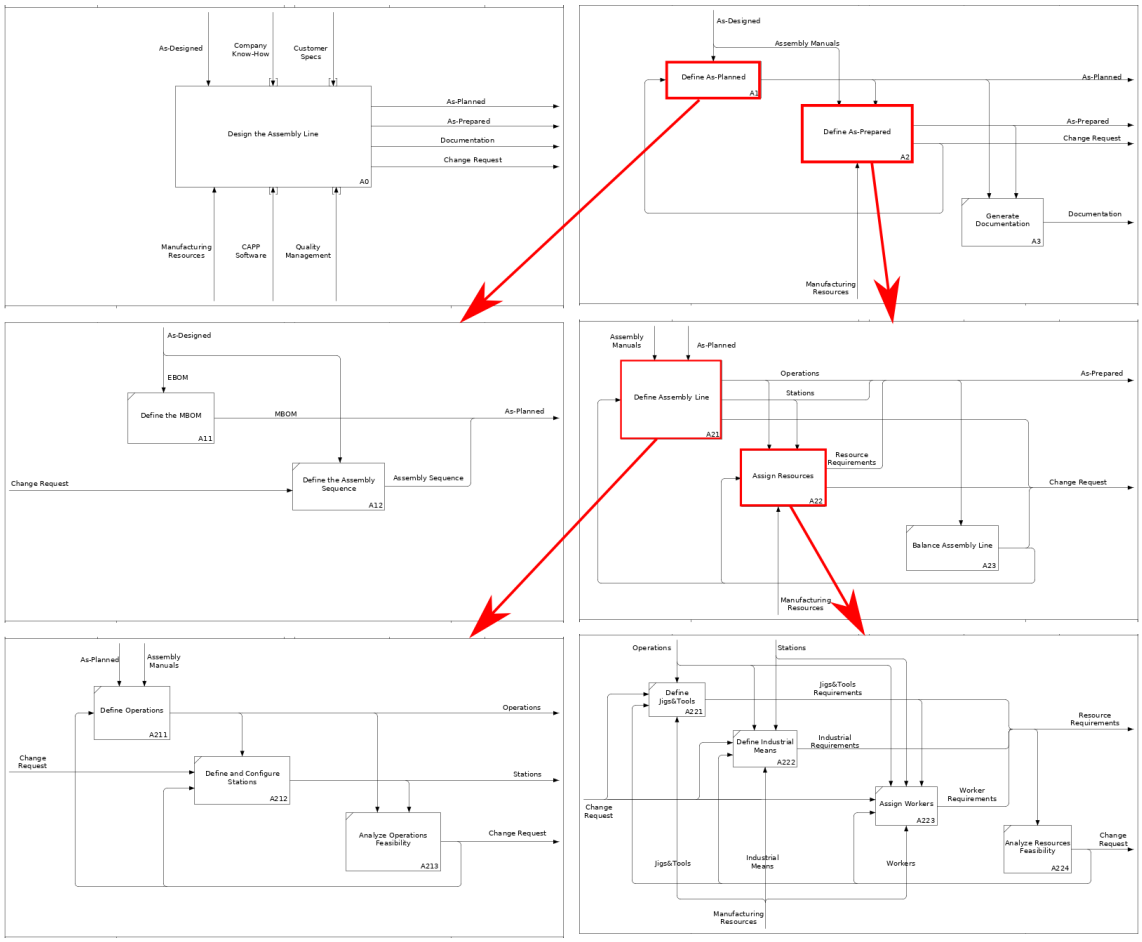
Este modelo se queda fuera del alcance del proyecto, y debe ser desarrollado en trabajos futuros para completar el modelo de la ontología.

3. Construcción de la capa de Ontología

El resultado de aplicar IDEF0 a un sistema es un modelo que consiste en una serie jerarquizada de diagramas interrelacionados. Los diagramas están compuestos por funciones (cajas) y los datos y objetos que las conectan (flechas).

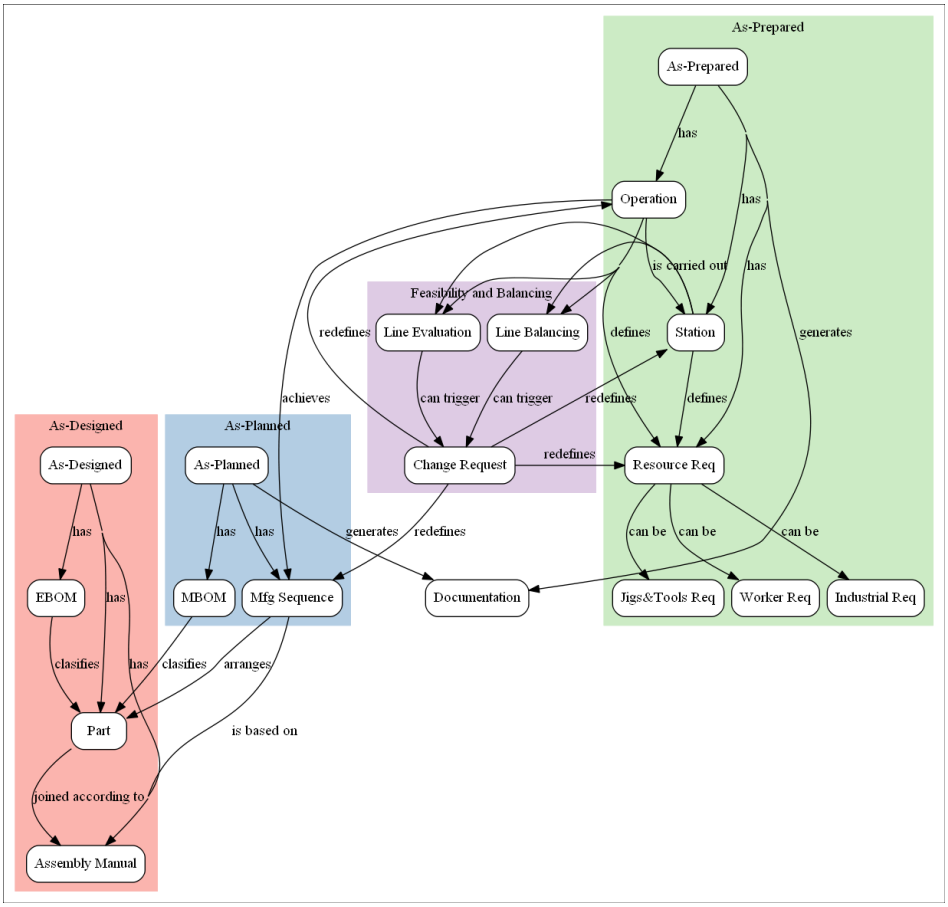
Cada función dentro del diagrama puede tener varias conexiones. Desde la izquierda, las entradas son empleadas para producir las salidas, que aparecen a la derecha. Desde arriba, los controladores ayudan a especificar como las entradas se convierten en salidas, pero permanecen inalteradas durante el proceso. Desde abajo, los mecanismos son los recursos empleados para producir las salidas a partir de las entradas. Estos recursos pueden ser industriales, humanos, software, etc.

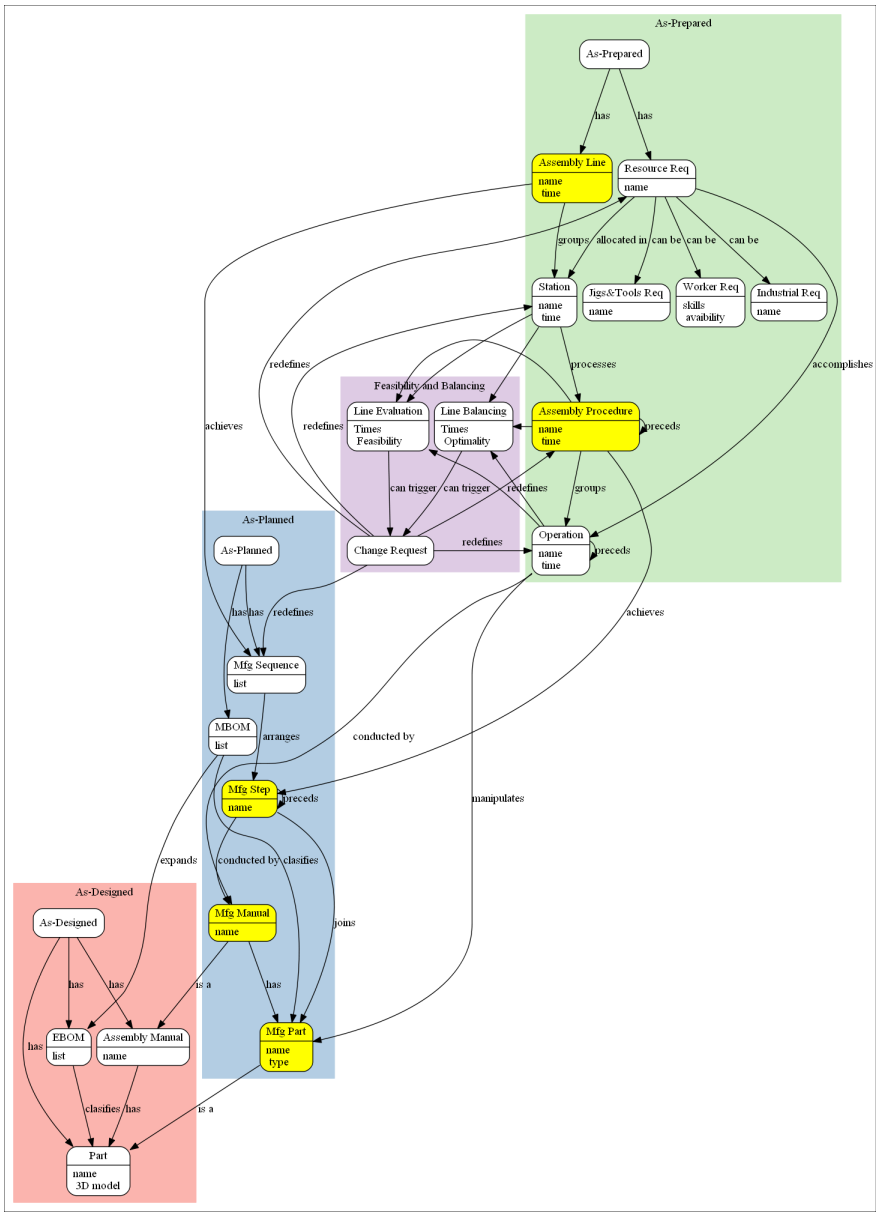
Los diagramas IDEF0 se encuentran anidados, de forma que un diagrama padre puede albergar varios hijos, y así sucesivamente, en varios niveles de complejidad. El conjunto completo de los diagramas desarrollados en el Scope model se presenta seguidamente.


As-Designed
Company Know-How
Customer Specs
Design the Assembly Line
A0
As-Planned
As-Prepared
Documentation
Change Request
Manufacturing Resources
CAPP Software
Quality Management
As-Designed
Assembly Manuals
Define As-Planned
A1
Define As-Prepared
A2
Generate Documentation
A3
As-Planned
As-Prepared
Change Request
Documentation
Manufacturing Resources
As-Designed
EBOM
Define the MBOM
A11
MBOM
As-Planned
Change Request
Define the Assembly Sequence
A12
Assembly Sequence
Assembly Manuals
As-Planned
Define Assembly Line
A21
Operations
Stations
As-Prepared
Assign Resources
A22
Resource Requirements
Change Request
Balance Assembly Line
A23
Manufacturing Resources
As-Planned
Assembly Manuals
Define Operations
A211
Operations
Change Request
Define and Configure Stations
A212
Stations
Analyze Operations Feasibility
A213
Change Request
Operations
Stations
Define Jigs&Tools
A221
Jigs&Tools Requirements
Define Industrial Means
A222
Industrial Requirements
Resource Requirements
Change Request
Assign Workers
A223
Worker Requirements
Analyze Resources Feasibility
A224
Change Request
Jigs&Tools
Industrial Means
Workers
Manufacturing Resources

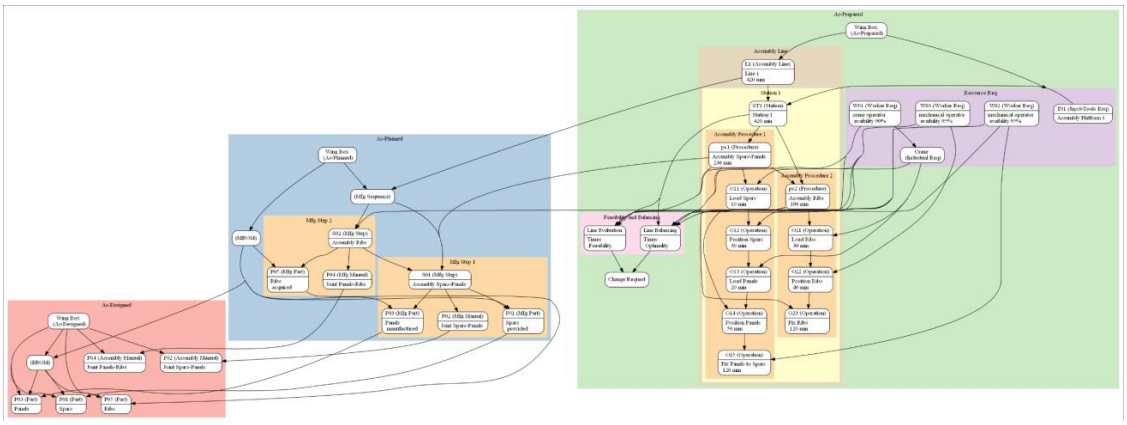
Se han empleado dos programas a la hora de desarrollar los mapas conceptuales que dan forma al modelo de datos. Inicialmente, se empleó CMAP Tools debido a su carácter intuitivo y sencillo, y a sus resultados atractivos y consistentes cuando se genera un modelo simple. El principal inconveniente al usar este software es que no permite agilizar las pequeñas modificaciones realizadas sobre el modelo, haciendo cada iteración más lenta de lo deseable en un proceso tan largo como el seguido. Es por esto que se quiso contemplar una alternativa diferente y se optó por compilar, mediante la librería DOT en Notepad++, los mapas conceptuales. Aunque no permite tanta flexibilidad en el resultado final y visualmente puede dar resultados extraños en algunas ocasiones, su facilidad a la hora de iterar ha resultado ser clave para desarrollar el modelo de datos. Si bien requiere de una previa familiarización con el lenguaje, una vez los comandos se conocen, su uso es inmediato sin más que redactar el script y compilarlo.

En cuanto a los mapas conceptuales en sí, se componen de diferentes conceptos, situados dentro de unas cajas, y relacionados entre sí por flechas, que poseen un conector extra para dar más información de la conexión entre conceptos. Una vez completado, el proceso de enriquecimiento, añadiendo nuevos conceptos y atributos, se realiza con un código de colores. Los nuevos conceptos se introducen como cajas amarillas, y los atributos aplicables a cada concepto se separan del concepto en sí mediante una línea divisoria. Ejemplos del modelo de datos y de su resultante enriquecida, compilados a partir de DOT, se presentan a continuación.





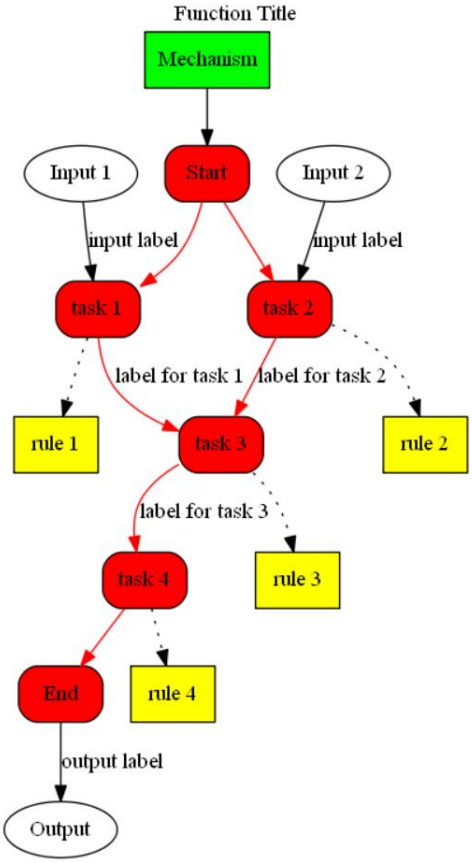
Llegados a este punto, se cree conveniente ejemplificar como funcionan las diferentes capas del modelo 3LM, usando para ello una instancia del modelo. El modelo enriquecido mostrado se pasaría a la capa de datos mediante aplicaciones de terceros o interfaces. Una vez almacenado en la base de datos, puede rellenar automáticamente los conceptos y atributos del modelo con valores específicos del caso práctico. Este diagrama relleno con valores concretos para un caso de uso se llama instancia. Tomando como caso práctico el ensamblaje de un cajón de ala, se obtiene lo mostrado seguidamente.



Tras crear el modelo de datos, hace falta una estructura que explique como se comporta cada uno de los conceptos, cómo se llevan a cabo las diferentes actividades. Para ello se crea el modelo de comportamiento.

Inicialmente se consideraron estrategias para generar diagramas de actividades, empleando el lenguaje UML. Sin embargo, ninguno parecía satisfacer por completo las necesidades y requisitos que se pedían, lo que llevó a decidir crear un nuevo tipo de diagrama desde cero, llamado diagrama de comportamiento. Este se ha creado usando DOT, aprovechando las ventajas ya comentadas.

Se consideraron diferentes tipos de diagramas, con sus respectivas ventajas e inconvenientes, con el fin de hacer diagramas que fuesen simples y al mismo tiempo permitieran reflejar con cierta precisión los comportamientos de los conceptos. Finalmente se opta por un diagrama del tipo que aparece a continuación.

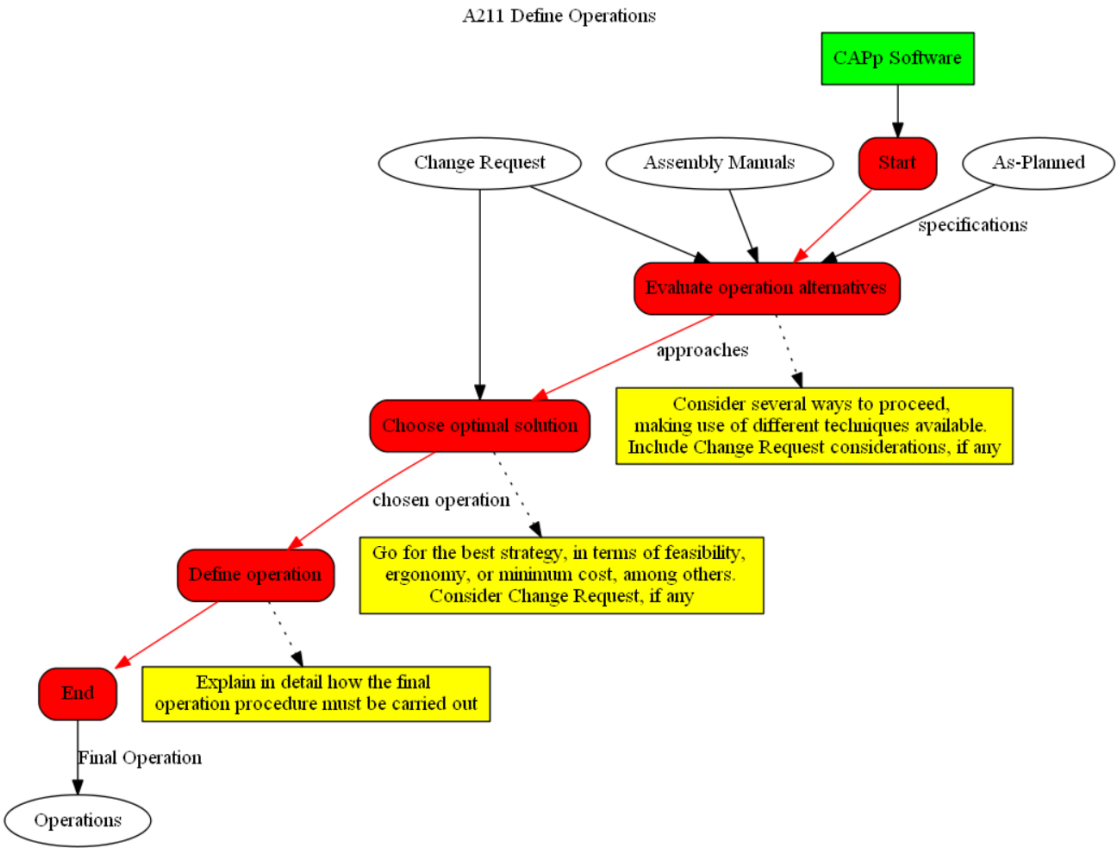


El modelo de comportamiento se crea directamente a partir del Scope Model, usando las funciones elementales (aquellas sin funciones hijas) y sus entradas, salidas y mecanismos. Como puede verse en la figura, cada función de los diagramas IDEF0 es el título del diagrama de comportamiento, las entradas y salidas tienen su homólogo en este diagrama y se encuentran al inicio y al final de este, y los mecanismos se representan mediante cajas verdes. Se ha decidido que cuando el mecanismo afecte a todo el diagrama, con el fin de no afectar a la claridad del mismo, se referirá el mecanismo como aplicable al nodo Start.

Los diagramas de comportamiento se modelan como los diagramas de flujo comunes en la informática. Una serie de tareas deben hacerse para llegar de un inicio a un fin, y pueden darse de forma secuencial, o en paralelo. Al igual que en los diagramas de flujo, pueden darse bucles que creen una serie repetitiva de tareas hasta que cierta condición se cumpla.

Aparte de eso, este diagrama también ofrece información acerca de cómo cada una de esas tareas debe hacerse. Para ello se introducen lo que se ha llamado reglas. Por tanto, es obligatorio que cada tarea lleve asociada una regla. Estas reglas pueden ser tan simples como se desee, pero deben proporcionar directivas claras sobre el procedimiento para llevar a cabo la tarea asociada.

Debido a su gran similitud en cuanto a representación gráfica con los mapas conceptuales, se ha empleado también DOT para su creación. Se ha generado un diagrama por cada una de las funciones elementales, pero debido a su similitud, solo una se refleja a continuación.



Cabe destacar que no todos los elementos de estos diagramas se encuentran en sus correspondientes IDEF0. Esto se debe a que durante la realización de este proyecto se constató la gran iteración que existe también en la creación del modelo de ontología completo, como conjunto. Esto implica que consideraciones tenidas en cuenta a la hora de crear el modelo de comportamiento afectan directamente a la definición de los modelos Scope y Data, que ya se encontraban cerrados. Esta iteración global se ha quedado fuera del alcance del trabajo y debe llevarse a cabo en futuros trabajos.

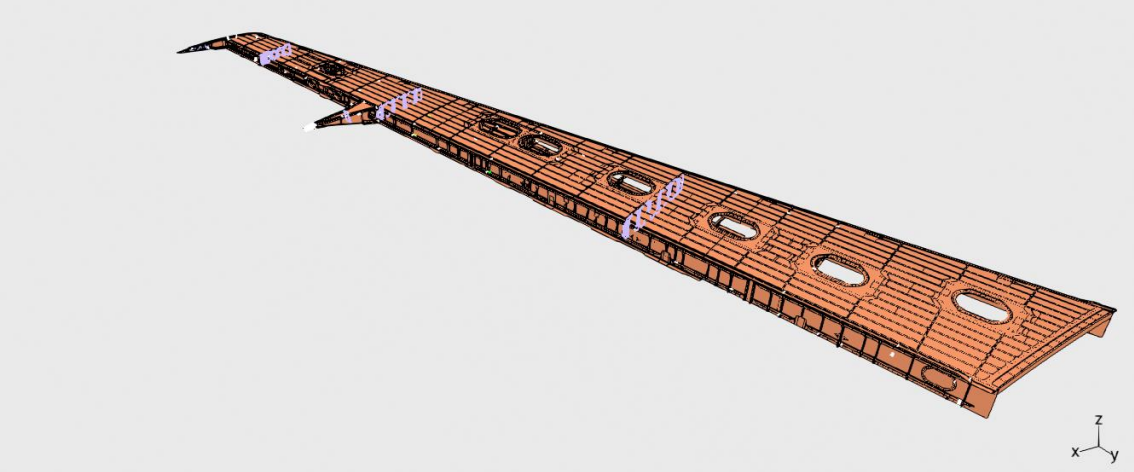
4. Caso práctico: Diseño y planificación de una línea de ensamblaje aeronáutico

Esta sección recoge el aspecto que presentaría una instancia concreta creada una vez el modelo 3LM se encontrase completamente implementado. En futuros trabajos, deberían realizarse varias pruebas usando herramientas especificas que demuestren la viabilidad del modelo para casos reales. Con este fin, se ha elegido un programa comercial con una estrecha relación con la industria aeronáutica, así como una gran variedad de herramientas muy potentes. Dicho software es 3DExperience, de Dassault Systèmes. El caso práctico seleccionado para esta instancia es el diseño y planificación del ensamblaje de un cajón central de ala.

Como se ha comentado, la metodología MfM está actualmente implementándose y desarrollándose desde un punto de vista académico. Varios de estos conceptos se han ido probando y refinando haciendo uso de Aras Innovator, un software PLM de carácter open source, que presenta varias ventajas en cuanto a su flexibilidad, facilidad de actualización y personalización, sin renunciar a funciones PLM básicas. Estas características fueron las que hicieron que se eligiera como primer terreno de pruebas para avanzar en el campo del MfM. Sin embargo, también quiere probarse que la metodología es factible y ventajosa en casos de uso reales. Para ello se elige el software ya comentado, 3DExperience. Si bien la implementación de la metodología usando este programa está aún lejos de completarse, el modelo que se presenta se ha empleado como ejemplo para que futuras líneas de trabajo realicen más pruebas y consigan uan forma sólida y automatizada de traducir la ontología de las empresas en un sistema totalmente funcional.

El modelo es del cajón de un ala de avión, y se presenta en la figura adjunta.

El cajón central se encuentra compuesto de dos largueros o spars, anterior y posterior, y recubierto por los revestimientos o paneles (panels), superior e inferior. En el interior de este cajón, para dotar al conjunto de rigidez, se encuentran seis costillas (ribs), que están intercaladas con varias estructuras de tipo celosía (truss), aportando flexibilidad a la estructura sin renunciar a la estabilidad del cajón. El modelo también incluye algunos componentes como la costilla de punta de ala, que cuenta con herrajes para la fijación e instalación de las correderas de algunos mandos de vuelo, así como el carenado de las correderas de flaps, con sus correspondientes herrajes.



El proceso de ensamblaje del cajón se realiza en las siguientes fases:

- **Fase 0: Preparación de los componentes**

  En esta fase, tanto los paneles como los largueros son preparados para la posterior unión entre ellos.

  Los paneles se montan en su propia grada, tanto sus componentes como sistemas asociados. Estas gradas (una para los paneles de ala izquierdos y otra para los derechos) son simétricas, se encuentran una frente a otra y están separadas por un pasillo. Cada grada tiene una sección para el panel inferior y otra para el superior. Cuentan con sistemas neumáticos, potencia eléctrica, sistemas de vacío para posicionar, y una remachadora semiautomática en el caso de los paneles inferiores.

  El caso de los largueros es totalmente análogo, contando con una grada propia para su preparación, que en ocasiones implica un repasado y ajustes finales.

- **Fase 1: Ensamblaje principal**

  En esta fase, los dos paneles, el larguero posterior, las costillas, la punta de ala y las correderas del flap son posicionadas, taladradas y remachadas. En esta fase también se incluyen los sistemas hidráulicos y de mandos de vuelo.

  La grada empleada en esta fase consta de un marco principal usado como guía para posicionar todas las costillas y la de punta de ala. El marco permanece inmóvil mientras dos estructuras móviles sujetan los paneles mediante vacío y los colocan en su posición, usando las llamadas “cuchillas” para este posicionamiento.

- **Fase 2: Remachado de largueros**

  En esta fase es de notar que el cajón se coloca horizontalmente, frente a la orientación vertical de las etapas previas. De esta forma, los largueros están libres, lo que permite el taladrado del larguero posterior, usando CNC, y el remachado del anterior, ya taladrado en la fase 1. La grada empleada para ello se muestra seguidamente.

- **Fase 3: Remachado de costillas y paneles y cierre**

  Después, se lleva a cabo el cierre del cajón, remachando el larguero posterior a los paneles, costillas y punta de ala, y se montan las correderas de los mandos de vuelo. Se ensamblan las costillas de borde de ataque y salida, usadas como apoyo estructural para los dispositivos hipersustentadores.

- **Fase 4: Superficies aerodinámicas y pruebas finales**

  Por último, se añaden los bordes de ataque y salida, así como el resto de superficies aerodinámicas. Debido a su criticidad, varias pruebas deben llevarse a cabo antes de dar por finalizado el ensamblaje. Entre ellas, se realiza un giro de 180 grados para comprobar si existe algún daño por objeto extraño (FOD).

En cuanto al uso de la metodología MfM mediante la plataforma 3DExperience, ésta apuesta por una nueva filosofía de trabajo orientada hacia una mayor colaboración entre las partes interesadas, resultando en un flujo de trabajo concurrente, gracias a una interfaz unificada y a su carácter basado en la nube. El entorno consta de cuatro grupos de aplicaciones principales, donde cada subgrupo contiene aplicaciones equivalentes a las funciones presentes en CATIA, DELMIA, SIMULIA y ENOVIA, siendo complementadas con otras aplicaciones destinadas a la gestión de la documentación en este nuevo entorno, así como la posibilidad de visualizar el proyecto en 3D en tiempo real. Debido al carácter claramente centrado en la fabricación de este proyecto, solo se han empleado aplicaciones DELMIA.

Las aplicaciones de 3Dexperience tienen su propia estructura de datos, lo que a día de hoy es un obstáculo para aplicar correctamente esta plataforma como software de PLM para gesitonar objetos se´gun la metodología MfM. Trabajos futuros deberían comprobar si es posible definir la estructura de datos del MfM en el entorno del programa, o si debiera desarrollarse algún tipo de interfaz que tradujese la estructura propuesta por MfM a una entendible por 3DExperience. Debido al enorme potencial que este software tiene como PLM, en caso de poder resolverse estos problemas de compatibilidad, podría ser empleado para gestionar cada objeto presente en la metodología, incluidos el modelo, el metamodelo y cada instancia creada a partir de ellos. Con independencia de esto, la parte de 3DExperience enfocada al process planning puede ser empleada para implementar modelos e instancias, actuando en este caso como el software elegido para la capa de Servicios. Esta capacidad para albergar el modelo completo en una sola plataforma se demuestra a continuación, evidenciando la estrecha relación existente entre los objetos propios de los submodelos de la ontología y los datos, interfaces y comandos de 3DExperience, mediante comparación directa.

- **Relaciones en el Scope Model**

  Para comenzar, se hace una breve comparación entre el Scope model y 3Dexperience. Como ya ha sido presentado, el Scope model se define usando diagramas IDEF0, que están a su vez hechos de bloques básicos con una función principal, entradas, salidas, mecanismos y controladores. La primera relación evidente viene en términos de los mecanismos. Todas esas funciones cuyo mecanismo aparece marcado como CAx software está directamente relacionado con 3DExperience, ya que es un software CAD/CAM/CAE/CAPP. Aparte de eso, se ha comprobado que cada función definida en los diagramas del modelo puede ser implementada en los diversos módulos dentro de 3DExperience, y el software es capaz de gestionar entradas y salidas, así como importar controladores. Como ejemplo, la definición del As-Planned se puede crear y gestionar usando Manufacturing Item Definition, el equilibrado de la línea mediante Process Planning, la asignación de recursos y su optimización mediante Equipment Allocationm y la documentación puede generarse usando Work Instructions (complementado con algún software de ofimática).

- **Relaciones en el Data Model**

A nivel Data model, los propios árboles de objetos PPR (Product, Process, Resources) de la aplicación ya conforman una estructura de datos similar a la que persigue el MfM. Como se ha dicho, investigaciones futuras dirán si esta estructura de datos puede ser adaptada para ajustarse a la del MfM o si es necesario generar una intefaz que las conecte. En cualquier caso, la estructura de datos del programa permite abarcar todas las características que requiere la estructura del MfM, aunque organizada de forma diferente. Por tanto, se cree conveniente explicar brevemente dichos árboles.

- **Relaciones en el Behavior Model**

Los modelos de comportamiento se centran en la dinámica del proceso, el "como" se hacen las cosas. Por tanto, parece lógico compararlos con los diferentes comandos y herramientas dentro de 3DExperience, ya que son los que se usan para llevar a cabo las diferentes acciones. La concatenación de estas acciones permite transformar las entradas del modelo en sus respectivas salidas.

Cabe destacar que debido a uno de los requisitos fundamentales propuestos para los modelos 3LM, esto es, la capacidad de ser independientes del software que después sea empleado en la capa de Servicios, estos modelos son muy vagos. Esto significa en última instancia que una sola actividad del diagrama puede traducirse en ocasiones en varias actividades o comandos dentro del programa, o necesitan de tareas previas antes de ser acometidas.

Puede decirse que la plataforma es capaz de ejecutar a la perfección el modelo de comportamiento del As-Prepared en su totalidad, siendo la conclusión final que 3DExperience puede implementar el modelo completo de Ontología (Scope, Data y Behavior) solamente usando algunas de sus aplicaciones. El uso de herramientas más avanzadas y su funcionamiento conjunto podría definir los distintos modelos con más precisión, detalle y complejidad, pero es algo que se deja para trabajos futuros.

5. Conclusiones

El metamodelo de construcción de ontologías es sólido, versátil, y fácil de llevar a cabo y entender. Está basado en cuatro modelos diferentes, lo que está a medio camino entre la complejidad y la precisión cuando se modela. Estos submodelos también son fáciles de entender, modificar y emplear, sin perder por ello su valor.

El metamodelo puede ser adaptado a cualquier grado de detalle que se desee, implementado sin importar el software empleado, y capaz de producir instancias de distintos grados de complejidad. Esto se ha demostrado generando dos instancias distintas, variando su complejidad en términos de su número de partes, pasos, y gestión de recursos. Cada instancia se ha modelado usando software distinto, herramientas simples para la instancia preliminar, y una plataforma potente para el más complejo. Ambas instancias han resultado satisfactorias en términos de los objetivos citados.

Los modelos se han hecho usando programas open source, claros, sencillos y amigables, y la posible integración con software comercial más potente también ha quedado constatada. Todo el software mostrado en este trabajo cumple estos requisitos, y el trabajo demuestra que una serie de herramientas potentes, como las que posee 3DExperience, es capaz de implementar el metamodelo completo y las instancias generadas a partir de él.

6. Trabajos futuros

Tras el desarrollo de este trabajo, se abren nuevas líneas de investigación, que se espera empleen este documento como punto de partida.

En primer lugar, el modelo semántico ha quedado fuera del alcance de este proyecto, pero se cree que no debería ser subestimado al crear una ontología. Tener un glosario común que permita evitar malentendidos entre las partes al desarrollar cualquier tecnología se antoja clave para cualquier empresa. Futuros trabajos deberían estudiar cómo realizar este modelo, y completar así el metamodelo de la construcción de ontologías.

Como ya se ha dicho, se eligió 3DExperience, entre otras razones, debido a su capacidad de ser empleado tanto como PLM como CAx. Aunque ha podido comprobarse la correcta integración de esta última vertiente a la hora de contener e instanciar los modelos, la primera depende de la capacidad de adaptar la estructura de datos de la plataforma a la especifica que necesita MfM, o bien la creación de una interfaz que actúe como puente entre ambas. Debe llevarse a cabo una investigación exhaustiva para ver si esto es posible o no. Como comentario aparte, se sabe que versiones anteriores de CATIA sí daban la posibilidad de hacer esto, permitiendo a usuarios más avanzados crear sus propias aplicaciones y adaptarlas a sus propias necesidades. Por ello, parece razonable pensar que 3DExperience debe contar con una función similar.

Por otro lado, se ha desarrollado un modelo e instancia muy complejos y precisos, a la hora de evaluar el uso de 3DExperience como software para la capa de Servicios. Se espera que este esfuerzo sea aprovechado y que puedan llevarse a cabo futuras pruebas usando esta instancia, con el fin de alcanzar nuevos hitos en el

desarrollo del MfM.

Finalmente, durante el desarrollo de este trabajo, se comprobó que existe un carácter iterativo aún mayor del inicialmente supuesto a la hora de construir ontologías y su metamodelo. Se llevaron a cabo grandes procesos iterativos cuando se refinaron los modelos de datos y comportamiento, pero también se pudo constatar que algunos modelos cambiaban la percepción de como modelos anteriores deberían haberse hecho. Esta retroalimentación entre modelos sugiere la existencia de un bucle de iteración mayor, que comprende al resto, y que no ha sido incluido en este proyecto. Debería llevarse a cabo una revisión en profundidad de todos los modelos como un todo, así como refinarlo mediante iteraciones. De esta formas, se conseguiría un metamodelo mas sólido, coherente y preciso a la hora de crear ontologías.