Aide au pilotage d'activités d'ingénierie pour le développement distribué d'un système complexe

Iban Lizarralde

THÈSE

Préparée au

Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS

En vue de l'obtention du grade de

Docteur de l'Institut National des Sciences Appliquées de Toulouse

Spécialité : ***Systèmes Industriels***
par

Iban LIZARRALDE

Titulaire du Diplôme d'Études Approfondies en Systèmes Industriels

---

AIDE AU PILOTAGE D'ACTIVITES D'INGENIERIE POUR LE DEVELOPPEMENT DISTRIBUE D'UN SYSTEME COMPLEXE

---

soutenue le 09/07/2007 devant le jury:

Présidente	C. MERCE
Rapporteurs	A. BERNARD
	P. GIRARD
Examinateur	M. DUREIGNE
Directeurs de thèse	P. ESQUIROL
	A. RIVIERE

Remerciements

Les premières lignes des remerciements sont habituellement dédiées aux directeurs de thèse et aux rapporteurs.
Permettez-moi de transgresser cette règle et de remercier les gens qui ont rendu possible cette thèse par ordre chronologique.
En effet, ce travail est le fruit d’un long processus qui a démarré il y a environ 5 ans quand j’ai identifié certaines difficultés au sein du projet de développement de l’A380.
Avant le lancement de la thèse, j’ai eu l’occasion de discuter avec plusieurs responsables d’Airbus. Je tiens à remercier spécialement Jean-Claude Roussel, membre du EDD-Airbus et actuellement président de l’Association Française d'Ingénierie Système (AFIS), qui a porté au sein d’Airbus le projet de recherche SPEED dans le cadre duquel cette thèse a été réalisée.
Les discussions avec le LAAS-CNRS ont démarré grâce à Robert Valette, avec qui la chance a voulu que je partage le bureau quelques années plus tard. Cela a été un privilège pour moi de recevoir ses derniers conseils avant son départ en retraite.
Une autre personne qui part en retraite et qui a soutenu de cette thèse à ses débuts est Jacques Erschler, Directeur de la Recherche de l’INSA de Toulouse, qui m’a présenté les membres du groupe MOGISA parmi lesquels se trouvaient Pierre Lopez et Patrick Esquirol. Mil esker Jacky.
Merci Pierre pour ta présence dans le comité scientifique de ma thèse, même si nous n’avons pas gardé le rythme des réunions de la première année.
Et, bien évidemment, un grand merci à Patrick, grâce à qui j’ai eu l’occasion d’approfondir le sujet de ma thèse, même si souvent nos discussions n’étaient pas directement liées à la thèse. Je suis sur que nous retravaillerons ensemble sur un autre projet un jour.
Coté EADS, la première personne que je dois remercier est Yves Baudier. Il m’a accueilli au sein du Centre de Recherche et il a été une personne clé dans le montage de la thèse.
Et maintenant c’est ton tour Arno ! Je tiens à remercier Arnaud Rivière. Merci pour le temps consacré à ce projet, merci pour tes conseils, surtout au démarrage du projet et merci pour m’avoir aidé à gérer quotidiennement cette thèse, je pense qu’elle n’aurait pas été si bien organisée sans ton aide.
Dans le cadre du CCR, je voudrais également remercier Martine Callot.
Merci Maurice pour ton soutien et un grande merci à Agnès.
« Un fuerte saludo » pour les gens avec qui j’ai partagé le bureau, spécialement, Claude, Laurent et Pascal. Et un « gracias » généralisé pour toutes les membres du CCR, maintenant Innovation Works, surtout ceux et celles avec qui j’ai partagé un croissant le vendredi matin et ceux qui m’ont dépanné quand mon vélo avait un problème mécanique.
Un « thank you » pour les différents membres d’Airbus qui ont accepté de consacrer quelques minutes de leur temps précieux à mon projet de recherche.
Et un « mil esker » pour les membres de MOGISA avec qui j’ai eu l’occasion de découvrir des sujets passionnants de recherche. Je pense que cela va être difficile de pas continuer dans ce domaine.
Un merci aussi à Régine, Victor et Sernin, qui ont participé au projet SPEED.
Les dernières lignes sont habituellement réservées aux ami-e-s et à la famille. Pour ce point je ne vais pas transgresser la règle.
Un merci à tous et toutes, et sur tout à toi Sandra. Inutile de dire qu’il n’aurait pas du tout été possible de réaliser cette thèse si tu n’avais pas été là.
Eskerrik asko.

Contents

List of Tables

List of Figures

General Introduction

1- Context and Problem Statement

Shortening development lead times is now a constant objective for any industrial firm. The major benefit expected by a faster development of a new product is the competitive advantage due to the availability of the product before competitors. In aircraft industries the goal is not only to gain a competitive advantage but also to sustain profitability due to a fast return on investment. Concurrent Engineering based development arise in the 80’s in order reduce the development cycles. First adopted by the automotive industry, these methods have been widely deployed in major aircraft companies since the last 15 years. One of the key concurrent engineering concepts is the establishment of more efficient communication channels between skills and design teams.

Organisational structures have been progressively redefined to enhance exchanges. Consequently, autonomous design teams have been created integrating different skills into the same team, known as “plateaux”. From a process point of view, schedules are defined overlapping tasks and interactions between these tasks include non definitive data known as preliminary information. The organisation and processes are therefore becoming more complex and this is particularly true when investigating complex products development. Aircrafts can be considered as complex systems characterised by the high number of functions the final product has to perform, the high number of individual parts and equipments that have to be integrated onto the final product and the numerous interfaces between these components that support interactions between the teams. These aspects are further emphasized when considering extended and distributed organisation as observed in the organisation where this research has been carried out.

Scheduling design activities of multidisciplinary and distributed teams deals with resource allocation problems mainly based on human resources that need to be assigned to each team and different tasks. Moreover lead times need to be defined assigning time slots to each activity, taking into an efficient integration of the final product. Lastly, the synchronisation of the interdependencies between different design teams seems to be a key element for the scheduling of the design activities.

In a design process, the definition of lead times for the activities and the allocation of resources can not be supported by stable and complete information. Therefore, scheduling process need to deal with uncertainties that are inherent to the design process. These uncertainties can be managed from a proactive point of view or from a reactive point of view. While the former deals with methods to react before the schedule becomes inconsistent, the later investigates corrective methods in enabling re-scheduling after appearance of unforeseen events.

Uncertainty is also linked to the interdependencies. In concurrent engineering, data exchanged between design teams is usually base on preliminary information. The accuracy level of the data as well as the probability to be modified in further exchanges can be used to measure the uncertainty of the interdependency. These variables will evolve until an accurate data that will not be modified anymore. This evolution can be characterised by maturity level of the data which if founded on a human perception of the performance linked to each characteristic of a data.

2- Contributions

The present research project have focused on tasks scheduling and resources allocation process for engineering activities taking place at tactical level of the aircraft development program organisation. Therefore, the challenge is to support design team leaders managing design process uncertainties and to investigate the collaboration process between different design teams focusing on the interdependencies characterization and synchronisation.

One of the contributions is a decision support system tool based on constraints programming. In this specific project management model, activities have been characterised using an energetic based approach and different constraints have been modelled in order to validate the steering decisions through a rigorous tool. The constraints that have been modelled include the activity energy constraint, the cumulative resources constraint, the time windows constraint and two specific constraints linked to the task overlap problem and the interdependencies between design teams. These two constraints have been named as Energy Precedence constraint and the Contract Dependency constraint.

Steering decisions can be based on the modification of the activities assignment setting them forward or backward or the modification of the resources allocation plan. If no consistent solution is found, the managers of design teams can relax one or more constraints. Relaxation is basically renegotiating tight constraints with the actors involved in their management. Simulations of constraints relaxing effects are an efficient support for constraints renegotiation process. Each simulation is performed using constraints propagation algorithm and each simulation is considered as a scenario. These scenarii are built in order to deal with the uncertainties inherent to the design process.

These proposals have been used as a basis for the development of an application prototype. This prototype is a first step in order to describe the new project management practices to end-users, get their feedback and to validate contributions. Requirements captured during these phases will allow the development of a future operational application to be deployed among the different partners of complex system development project.

3- Reading guidelines

The report is divided into five chapters. First chapter introduces the context of the research project in order to highlight certain specificities of the aircraft product and aerospace business. This chapter includes the definition of the complexity of the product itself using some quantitative notions. It also describes the industrial approach which has led to a unique organisation that deals with the product design, manufacturing and integration.

Second chapter deals with a new aircraft design process. It describes the deliverables exchanged between different teams and that will evolve as the product progresses in its lifecycle. It also describes scheduling practices and the description of the different schedule approaches used by different actors of the design process. The aim of this chapter is to highlight the difficulties and the need that face these actors when scheduling design activities, allocating resources and managing data exchanges with other actors.

Third chapter introduces the approach that has been defined in order to deal with design tasks scheduling and design resources allocation process. It presents a state of the art on project management practices, and more specifically on planning and scheduling aspects and is focused on the collaboration process between design teams as well as on methods dealing with uncertainties of scheduling. Our aim is to define a rigorous framework enabling team managers to check consistency of its decisions with the project constraints through a Decision Support System (DSS). This DSS shall be supported by a Project Management model that allow the right exploitation of the constraints which include classical constraints such as deadlines and available resources but also emerging constraints due to the concurrent and distributed nature of product development.

Chapter 4 introduces our proposals based on the approach defined in the third chapter. It details the project management model based on the Constraint Satisfaction Problem. It includes the different constraints propagation devices and how these devices are used for validating simulations. It describes also the interdependency management process based on this model in order to support the collaboration process between design teams.

Chapter 5 describes the prototype that has been developed based on the project management model defined in Chapter 4. It includes the specification of the requirements defined along with actors involved in on-going aircraft development projects and the detailed description of each feature of the prototype. It also defines the experimentations and validations required to validate our contribution operationally.

Finally, we conclude with some open questions addressed by this project and detail the perspectives for a continuation of this initiative.

Chapter 1: The context: The development of a civil aircraft

1.1 Introduction

This research project is focused on the development project of a new civil aircraft for freight and passengers transport usage.

Our work investigates project management techniques related to the design and the development of these complex products. Before describing project management problems, we have to describe the context in which the work was carried out, in order to highlight certain specificities of the aircraft product and the industrial sector. Some of these main context-related issues concern the complexity of the product itself and are also related to the configuration of industrial organisations. It will also enable us to differentiate the generic aspects of our proposals from those that must remain specific to the aviation industry.

First, we will try to define the complexity of the product itself using some quantitative notions. Then we will describe the industrial approach which has led to a unique organisation that deals with the product design, manufacturing and integration. Finally, we will study some specific aspects of the organisation and architecture of the product system and the project related to its development.

1.2 Aircraft Classification

Aircraft is a generic term used to designed vehicles that are able to perform atmospheric flights. We propose to classify aircrafts by design, propulsion and usage.

A first classification based on design distinguishes lighter-than-air, aerostat, heavier-than-air aircraft and aerodyne. Lighter-than-air aircrafts include non-steering balloons and steering airships (sometimes called dirigible balloons), mainly known thanks to the success of the Zeppelin.

Concerning aircrafts heavier-than-air, there are two ways to produce lift: aerodynamic lift and engine lift. In the case of aerodynamic lift, the aircraft is kept in the air by wings or rotors while with engine lift, the aircraft defeats gravity by use of vertical thrust, like in the case of the rockets.

Another classification can be proposed according to propulsion means. At the beginning the different aircrafts built by human had no propulsion. Later on, appear vehicles with internal combustion engines. During cold war, both parties made tests with nuclear powered aircrafts, nevertheless, due to the problems associated with a crash landing of such an airplane, these programs were discarded. Nowadays turbine engines are common propulsion means to equip aircrafts for powerful and high-altitude uses. Moreover, and usually for research or recreation purposes, we can find human powered aircrafts as well as solar powered aircrafts (Figure 1).

Figure 1: Human powered aircrafts

Future aircraft developments include hydrogen based propulsion and synthetic kerosene issued from vegetables or coal.

Finally, a classification regarding the usage of the aircrafts can be realised. The main classification is between military purposes (such as combat, patrolling, search and rescue, reconnaissance, transport, and training) or for civil transportation. In the later category we include private usage as well as commercial usage. Some of the civil uses include fire fighting, medical transport, surveying, crop dusting, etc. But the most extensive use for aircrafts is for freight and passengers transport which include personal travel, business travel, or recreation.

In this document when we refer to the aircraft, we refer to a heavier-than-air vehicle, with aerodynamic lift, equipped with turbine engines type propulsion and mainly for freight and passenger transport usage. These characteristics correspond to the vehicles designed and manufactured by AIRBUS.

1.3 Complexity of the aircraft product

Complexity is certainly the first aspect one may think about facing the problem of designing and developing a new aircraft. Various research domains have investigated complexity [WEBER '05]. One can observe that complexity arises as soon as several items with different functions must be connected together and interact inside a system. Actually, the original Latin word “complexus” means “twisted together”.

From a product point of view, different levels can be defined to classify product complexity. It can be characterised by the large number of physical items to be integrated with multiple connections that might be difficult to control over time (structural complexity). This complexity is usually managed through modularisation and product structuring approaches.

From an activity point of view, process complexity deals with product development activities, taking into account items such as design procedures, skills organisation, work distribution, decision procedures, etc. Process complexity is generally induced by structural complexity. Indeed, products with high structural complexity need usually high number of activities linked to different functions and therefore with more probability of being distributed.

Thus, the development of a new civil aircraft can be considered complex both from a product and a process point of view.

1.3.1 Product complexity

A complex system can be defined as a network of interdependent elements, each one with its own functions, whose interaction determines the structure and performance of the final product [GINO '02]. However, this definition, which covers the notions of complexity and interaction, does not in itself enable us to understand some issues inherent to the project associated with complex products development. In order to integrate these two dimensions (product and project), Hobday, in [HOBDAY '98], lists the critical dimensions of the product that reveal its complexity. Based on Hobday's definition of these critical dimensions, we can propose a characterisation of the Aircraft product.

Table 1: Dimensions of Complexity

		CRITICITY				
		Very high	High	Medium	Low	Very low
DIMENSIONS	Unit cost/Project size	▨				
	Product size		▨			
	Technological innovation degree		▨			
	Software components within		▨			
	Components quantity	▨				
	Customisation degree		▨			
	Architectural complexity		▨			
	Available design alternatives quantity		▨			

One way to control this complexity is to specify a modular architecture. Ulrich, in [ULRICH '95], states that an architecture is modular when it *"includes a one-to-one mapping from functional elements in the function structure to the physical components of the product, and specifies de-coupled interfaces between components".* Opposed to the modular architecture, the integral architecture *"includes a complex (non one-to-one) mapping from functional elements to physical components and/or coupled interfaces between components".* This definition highlights the fundamental elements that contribute to the complexity of the product:

- The high number of functions the final product has to perform, through its physical components. The functions that the aircraft has to perform, as well as those that are inherent to the aircraft itself are incredibly varied. In order to ensure that all these functions are developed, aircraft manufacturers use the ATA [ATA '99] standard, which favours functional and physical breakdowns of the product. These breakdowns are also used for aircraft certification.
- The high number of components to be integrated. For example, an aircraft from the Airbus range contains four million parts (e.g. the A380). While the manufacturer's main concern is the development, manufacture or supply of these parts, it must also ensure that the physical components perform the functions of the final product. A distinction between physical and software elements is generally observed. The former may involve various specialities (mechanical, electromechanical, electricity, hydraulics, etc.), while the latter is composed only of computer code.
- The interfaces that govern the physical interactions between components. During the development of a complex product, management of these interfaces is very strategic, because the design and manufacture of dependent systems can be allocated to different entities within the organisation. Interfaces development is therefore at the centre of cooperation between the various actors in the company.

- The size and dimensions of the product. These factors lead to the production of a limited series of aircraft, due to the production costs implied by the size of the production infrastructure. Consequently, the risks associated with the decisions influencing the production means must be limited because these decisions are practically irreversible. Due to its size, aircraft development is seriously constrained by the development of manufacturing infrastructure and production schedules. The size of A380 for example, forces the parts to be brought to the assembly hall in Toulouse in France by sea and ground transportation, rather than by the A300-600ST Beluga aircraft used for other Airbus models.

1.3.2 Process Complexity

Aircraft lifecycles are rather unusual compared to other industrial products. The product's lifecycle is around thirty years. The manufacturer must therefore ensure, for a period of approximately thirty-five years that the aircraft can be operated correctly in service by airlines, and meet safety requirements set by regulations. Over the same period of time, the manufacturer must upgrade fleets continuously through the integration of engineering change request.

Maintenance and reliability requirements must therefore be integrated into the design cycle with a long-term vision. Decisions made during the design phase can affect downstream phases of the life cycle, which highlights the importance of integrating – into existing aircraft - any technical developments that occur with airlines, in order to build up models and knowledge that can be used during the design phase.

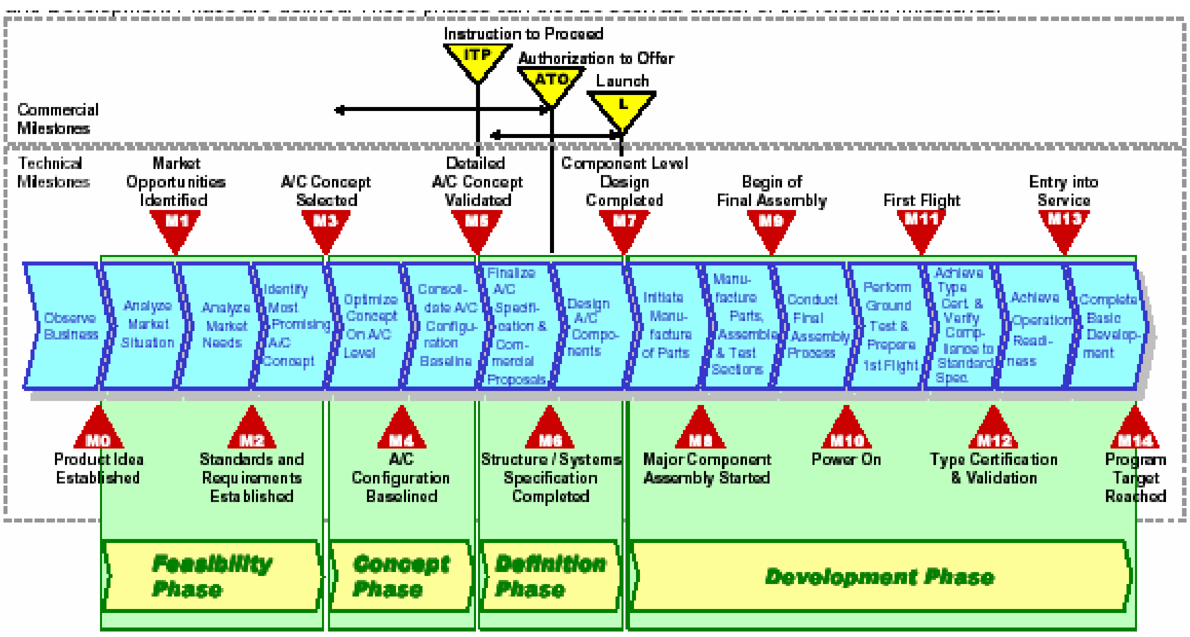

Instruction to Proceed
ITP
Authorization to Offer
ATO
Launch
L
Commercial Milestones
Technical Milestones
Market Opportunities Identified
M1
A/C Concept Selected
M3
Detailed A/C Concept Validated
M5
Component Level Design Completed
M7
Begin of Final Assembly
M9
First Flight
M11
Entry into Service
M13
Observe Business
Analyze Market Situation
Analyze Market Needs
Identify Most Promising A/C Concept
Optimize Concept On A/C Level
Consoli-date A/C Configu-ration Baseline
Finalize A/C Specifi-cation & Com-mercial Proposals
Design A/C Compo-nents
Initiate Manu-facture of Parts
Manu-facture Parts, Assemble & Test Sections
Conduct Final Assembly Process
Perform Ground Test & Prepare 1st Flight
Achieve Type Cert. & Verify Comp-liance to Standard Spec.
Achieve Operation Readi-ness
Complete Basic Develop-ment
M0
Product Idea Established
M2
Standards and Requirements Established
M4
A/C Configuration Baselined
M6
Structure / Systems Specification Completed
M8
Major Component Assembly Started
M10
Power On
M12
Type Certification & Validation
M14
Program Target Reached
Feasibility Phase
Concept Phase
Definition Phase
Development Phase

Figure 2: Airbus Aircraft Lifecycle

If we look at the development cycle of a new aircraft (Figure 2), we can see that the cycle may vary depending on whether the aircraft under development is an adaptation of an existing model or an entirely new reference. Depending on the case, the development cycle can last for 4-5 years (in the past, the product development cycle was around ten years and it expected to decrease significantly in the future). This exceptionally long development time is due to the complexity of the product and will involve a costly project management system requiring the allocation of a high number of resources. The risk associated to the high level of initial investments for this kind of project must therefore be controlled to maintain the company's competitive advantage

1.4 The industrial organisation and its environment

The objective of any civil aircraft manufacturer is to control the complexity of the product in order to tackle different market segments. As we will see in this chapter, this challenge has repercussions on the company's configuration and the distribution of development activities within the organisation.

1.4.1 Segmentation of the civil aviation market

The aviation market is generally broken down into four categories: aircraft of over 100 seats, corporate or regional aircraft of under 100 seats, the engine market and the market linked to the so-called aircraft MRO (Maintenance, Repair and Overhaul). In addition, we often refer to markets related to important aircraft subassemblies, such as avionics.

The 100 seats+ aircraft market is closely linked to the air traffic market for long journeys. This is the market that corresponds to the aircraft manufacturer on which our study is based (Airbus).

The market for aircraft of less than 100 seats corresponds to regional transport needs. Such aircraft offer limited capacity and range, since they are not designed to link up inter-continental hubs, but rather secondary airports, and are mainly aimed at business passengers.

The two main market leaders are not competing on this segment. Because the range of such aircraft is limited, and the airlines are generally smaller, the choice of engine is different from that of the 100+ seat segment. Engines are supplied to the manufacturer for a given aircraft. However, the engine is selected by the final customer, i.e. the airline. Up to now, engine sales were governed by two trends: airlines' desire to harmonise the engines selected within their fleet in order to minimise maintenance costs, and the choices proposed by the manufacturers. Recently, we have seen that airlines are prepared to diversify their choice of engines, whilst continuing to favour certain types of engine.. To sum up the situation rather loosely, we can say that the engine market is evolving in the opposite way to the MRO market, and that it is benefiting directly from the increase in sales of new aircraft.

The MRO market is dominant in this industry where product lifecycles are exceptionally long. The first Boeing 747, introduced in 1968, is still operational, for example. Thus, the main MRO customers are aircraft owners and operators. This market depends heavily on the number of aircraft in operation and the obsolescence of older aircraft, which are heavily constrained by safety regulations and prohibitive maintenance costs. For this reason, it is often more economical and less risky for an airline to invest in a new aircraft.

1.4.2 Market expectations and management of the offer

1.4.2.1 Modularity of ranges and products in the aviation sector

While modularity enables a significant reduction of product's complexity, its main purpose is to differentiate the types of aircraft offered, whilst maintaining an homogenous catalogue. By modularising its offer, the manufacturer can offer diverse products while taking advantage of major-scale production savings, and maintaining a consistent offer with respect to its customer airlines. Modularity can be observed at several levels and according to several points of view.

For passengers transport, airlines' demands are based on two major characteristics: the distance to be flown and the number of passengers. From the manufacturer's point of view, the objective is therefore to develop an offer of products that can cover all of these needs, whilst taking into consideration the existing offer of its competitors.

To manage a wide variety of needs while minimising production costs, the concept of product platform has appeared in the aircraft industry. A platform is called family and can be further customised through the options modularity.

While modularity enables a reduction of product complexity, its main purpose is to differentiate the types of aircraft offered, whilst maintaining a homogenous product portfolio. By modularising its offer, the manufacturer can offer diverse products while taking advantage of major-scale production savings, and maintaining a coherent offer with respect to its customer airlines.

Modularity can be observed according to two points of view. On the one hand, by defining product families within a range and on the other hand by defining standards and options. Product family type modularity involves defining the common characteristics that link the products in a manufacturer’s catalogue. It refers both to the characteristics of the “mission” that the aircraft has to fulfil and to the resulting configuration and technological choices (Figure 3).

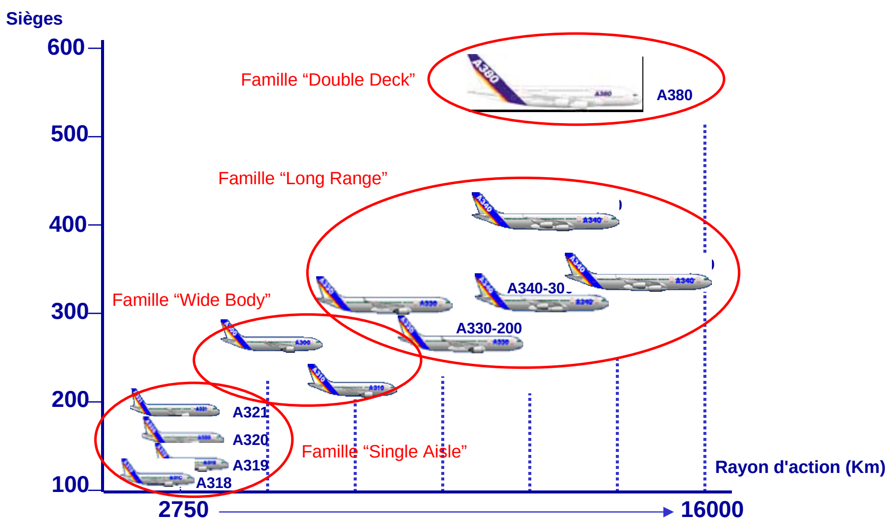


Figure 3: Airbus Family

For an airline, the definition of product families enables a reduction in the number of training hours for pilots and crew and an optimisation of aircraft maintenance and repair procedures [AIRBUS '03, KLANSNIC and DITTENBERGER '81].

For a manufacturer, extending a family with a new program might be considered as a re-engineering process since an existing platform design has to be adapted to a new set of requirements. Managing the consistencies between different sets of requirements with different rationales has to be taken into account during the estimation of a new program design lead-times. On the contrary, reuse of existing design may reduce these estimations

Baselines and options are two features of modularity which provide some flexibility in their offer with respect to the needs of airlines, particularly as regards the cabin configurations offered and their layout, and the choice of engine. Whilst trying to meet these diverse needs, the manufacturer must ensure that its proposals remain profitable. It therefore defines an aircraft according to a standard (or baseline), which is then enriched with options, and adds the engine selected from the list of choices [AIRBUS '01a].

Consequently, two aircrafts produced by the same manufacturer are rarely identical, except when an airline orders an entire fleet. And yet, the manufacturer has to manage the development planning of the different aircrafts taking into account their specificities and dealing with the modifications requested during the definition phase. Consequently, several schedules related to different aircrafts definition need to be managed in parallel. The duration of the activities linked to the definition of the standard parts will reduced at the same time as the firsts aircrafts are developed. Nevertheless, the duration of the activities linked to the definition of new options could be critical activities during the development of a new version of an aircraft whose standard parts has already been defined.

1.4.2.2 Competition criteria

The modularity of the offer is not the only competitive criteria between aircraft manufacturers. Other criteria have now become important when defining a commercial offer.

Some of these have led manufacturers to develop new products, of improved quality, more quickly and less expensively. This strategy is better known in the aviation industry as “Better, Faster, Cheaper” [MURMAN, et al. '00]. It has been adapted and used as a motto by a major European manufacturer, which chose to launch one of its new aircraft with the slogan: “Longer, Larger, Farther, Faster, Higher, Quieter, Smoother.” This slogan partly translates the demands of airlines, around which the competition between manufacturers is based (Figure 4).

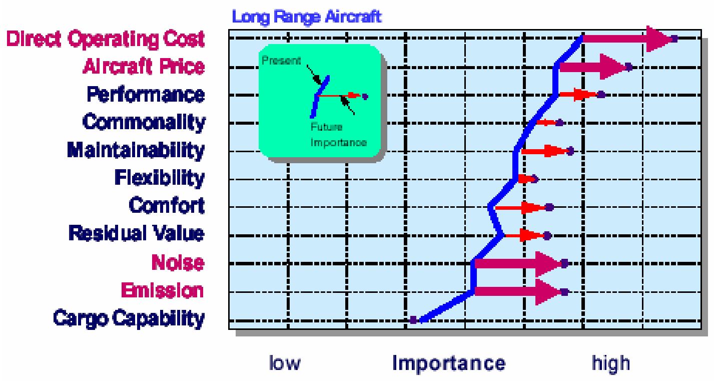


Figure 4: Competition Criteria

We present three criteria in order to identify the repercussion that these criteria has on new aircraft design activities.

First, manufacturer has to deal with the performance criteria which are mainly reflected by the reduction of weight and the drag. Performance criteria also include environmental constraints that include the reduction of emissions and noise. These constraints oblige the designers to work more closely to teams in charge of aerodynamics studies, new materials or engine manufacturer. As soon as a designer has the first sketch, aerodynamics studies will work on it in order to propose modification. These exchanges will follow during the definition phase, converging together into a efficient design. From a resource point of view, it must be stressed the fact that designer are usually allocated exclusively to the development of the new aircraft while the members of the aerodynamics studies teams and experts on new materials can work for different programs and research projects.

Secondly, manufacturer must deal with the cost of the product acquisition and use. For airlines, the cost of acquiring an aircraft or fleet can be prohibitive and considerably risky. This is justified if the airline goes on to make a profit on the routes on which the product is used. The profit margins will depend greatly on the conditions and running costs of the aircraft, the aim being to ensure that the aircraft is operational as often as possible, at the lowest possible cost, without compromising passengers’ safety. In order to remain competitive, a manufacturer must try to reduce the acquisition cost of its product and rationalise the design and production phases.

During the definition phase numerous trade off will deal between the cost for the airlines of the product acquisition and use, and the cost for the manufacturer of the product development. The allocation of extra resources during the definition phase can lead to an even better product from a running cost point of view (easier to realise maintenance activities, etc). Nevertheless, extra resources allocated to the same program can lead this program become less competitive and affect the development of further programs. Therefore, even if technically, the allocation of more designers seems justified, financial constraints can impose tight constraints on resources allocation..

1.4.3 The Extended Company

Sharing of design and manufacturing tasks according to a negotiated product breakdown – also known as “worksharing” – is one of the means at the manufacturers’ disposal to deal with the complexity of the product whilst respecting an industrial logic that guarantees the profitability of the organisation. This strategy enables us to designate an entity responsible for a part of the aircraft (a geographical zone of the aircraft and a series of functions, see Figure 5). Although this is often adopted for historical reasons linked to the construction of the industrial aeronautical organisations, it remains effective.

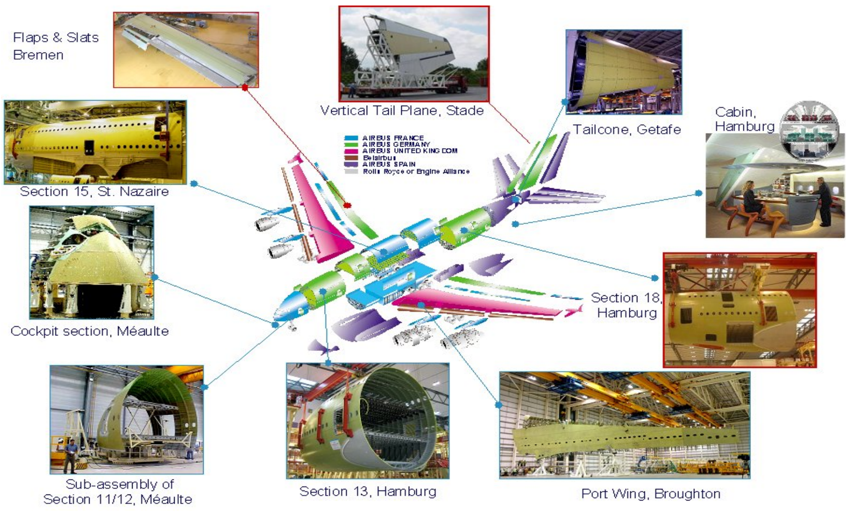


Figure 5: Worksharing of A380 aircraft

The major challenge of this strategy is to define accurately the product breakdown and identify clearly the industrial skills of each partner, in order to ensure a logical sharing of the tasks and the final integration of the product. Worksharing, coordination of activities and steering of the final integration are usually managed by the program management team. This team is also responsible for defining the management rules and the constraints linked to the interfaces between different subsystems.

Thus, development of a new aircraft is the result of numerous interactions between the program players, whose roles and activities can be diverse. This challenges the traditional model of the industrial organisation and introduces in its place the extended company model.

At present, the extended company does not only include the players involved in the organisation but also integrates external partners such as subcontractors. Since the boundaries of the organisation become difficult to determine, the extended company model considers all the players involved in the development project as a single organisation.

Although this phenomenon can be observed in most industrial organisations, the development of the extended company within Airbus is of a particular nature, as it is based on a series of organisational changes, which require further explanation.

1.4.3.1 Development of the extended company within Airbus

At the start of the 1970s, the first Airbus program – the A300/A310 – relied on the skills of major companies within the European aeronautical sector. Based on a modular breakdown philosophy, each partner was made responsible for the development and production of one or more of the aircraft subassemblies.

The product breakdown and the sharing of activities led to the development of specialised skills and knowledge for each partner. The specialisation of the partners (and that of their sites) occurred at two levels: a specialisation in terms of subassemblies, based on the technical homogeneity of the product, and a specialisation in terms of skills (machining, sheet metal work, composites, etc.). By recognising these specialities, the organisation avoided duplicating its industrial means, improved the utilisation rate of its manufacturing / assembly stations and developed specific skills, while minimising the number of trips between sites.

During the 1970s, in order to avoid any disparity in the allocation of resources, and above all for commercial reasons, the partners decided to centralise the commercial function, rather than allocate it to one partner in particular, which led to the creation of the AIRBUS GIE (Groupement d'Intérêt Economique in French, which is a category of Joint Venture)

Gradually, with each new program, the AIRBUS GIE became more autonomous and was able to resolve complex problems involving technical and commercial variables.

In addition, the GIE was able to reinforce the partners in their specialisations, by establishing itself as the program architect, capable of sustaining an overall approach to the production organisation. Such an approach also required perfect knowledge of the product at the interfaces between partners.

When the AIRBUS integrated company was created, this role of architect-integrator was reinforced, while the partners conserved the same responsibilities as previously. However, the interfaces between the organisations and with the program architects became less and less explicit, as each one became a unit within the organisation. This integration (which can also be considered as a step towards increased autonomy) was accompanied by an outsourcing drive, which saw an increase in the number of development and manufacturing tasks outsourced to subcontractors.

In June 2000 “Airbus Integrated Company” was created becoming a share-based company instead of the GIE. 80% of shares belonged to EADS (European Aeronautic, Defence and Space) and 20% to BAE Systems.

In September 2006 EADS bought the shares belonging to BAE Systems and nowadays holds the 100% of the parts of Airbus. Last news concerning the participation on Airbus includes the possibility of the participation of a Russian company.

1.4.3.2 Managing a subcontractor network

The performances of an aircraft manufacturer currently depend partly on the subcontractors’ network it relies on. For the development and production of the Airbus A380, for example, 70% of the activities have been carried out by the subcontractors’ network. From the contractor point of view, the players in this network can be classified into certain groups [ACHA, et al. '01, TALBOT '01]:

- *Firs level subcontractors*. These are designers and assemblers of complete systems, often known as multi-skill equipment manufacturers. They have resources and skills that may be quite far from those of the aircraft manufacturer but which are nonetheless essential for the product of an aircraft (landing gear, air conditioning systems, etc.). This position gives them a certain weight with regards to negotiation, since the work allocated to them by the aircraft manufacturer does not represent their only source of activity. They are in direct contact with the contractor over a fairly stable period of time, which allows for sufficient return on the investments made. The rules of “design and build” subcontracting are fully applicable in this case, apart from the fact that the equipment manufacturers often retain intellectual property rights for the products they design. Since the purpose of these rules is also to enable increased transfer of risks and workload, designers/assemblers of complete systems seek to apply the same rules to their own subcontractors, known as “second level” subcontractors.

- *Second level subcontractors.* These are suppliers of equipment for complete systems: they have recognised skills in the production domain but not as regards design capacity. In this respect, we may refer to “full production subcontracting”, given that research and development activities are not delegated. Eventually, these subcontractors will interact only with the first level subcontractors, but at present they still have some contact with the Principal – i.e. the aircraft manufacturer.
- Third level subcontractors. These are usually subcontractors with a limited capacity that work to order. It refers to traditional subcontracting that is linked to the industrial climate. They are usually in contact with the first and second level subcontractors and rarely with the aircraft manufacturer. However, the latter retains control over these subcontractors, as each company must receive an approval before it can enter into contact with the first and second level subcontractors.

According to this breakdown, the aircraft manufacturer, in its role as Principal, can be seen as the program architect, responsible for outsourcing certain elements of the program. It divides the product into units that can be developed or produced by different components within the organisation, and decides to outsource certain others to the subcontractor network. Its objective is therefore to ensure that the product breakdown is pertinent, and that the outsourced work is of the highest possible quality, so that the final product integration is a success.

1.4.3.3 The extended company and development of a new aircraft

If we focus on the development process for a new aircraft, the “extended company” configuration can present some problems. Coordinating the development activities associated with one or more assemblies of the aircraft or final product involves:

- Sharing the work accurately.
- Defining exchange procedures
- Implementing common development tools
- Synchronising activities to enable an on-time begin of the downstream activity
- Setting up a common information baseline to ensure that all the players can access correct, up-to-date information
- Managing the interfaces between systems whose development has been delegated to different partners or subcontractors
- Supervising development activities that concern structural activities and activities related to electrical and mechanical systems development.

Therefore, exchanges and interfaces between different actors of the extended company and their management taking into account time constraints, becomes a key issue for the development of a new aircraft.

1.5 Aircraft project structures

In the preceding chapter we have shown the reality of the extended enterprise encompassing a subcontractor network.

In the following we will focus on the Airbus Company, where this study has been realised. The scope of our study will not cover the different subcontractors.

As we will se in this chapter, in order to deal with the complexity of the product, different structures are defined related to the product, process and the organisation.

1.5.1 Organisation Breakdown Structures (OBS)

Inside Airbus, the organisation is characterised by a matrix organisation structure. On one hand activities can be hierarchically decomposed by into functions and knowledge domains; on the other hand a hierarchical decomposition describes the various programs and products.

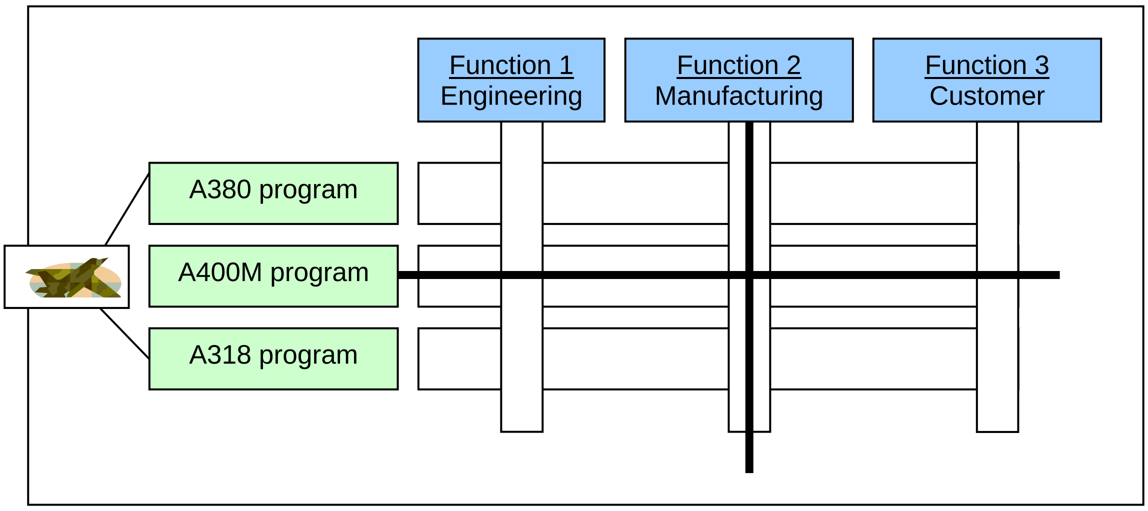


Figure 6: A matrix organisation structure.

Therefore, any actor working in an Airbus program will also belong to a functional hierarchy and a program hierarchy (Figure 6).

The balance between the two hierarchies evolves, enabling the company to be more efficient in new aircraft developments without loosing the knowledge of each function.

At the beginning of our project, Airbus modified this organisation in order to organise the enterprise in a more product oriented way, focusing the main subsystems of the aircraft and merging the different functions inside this centres.

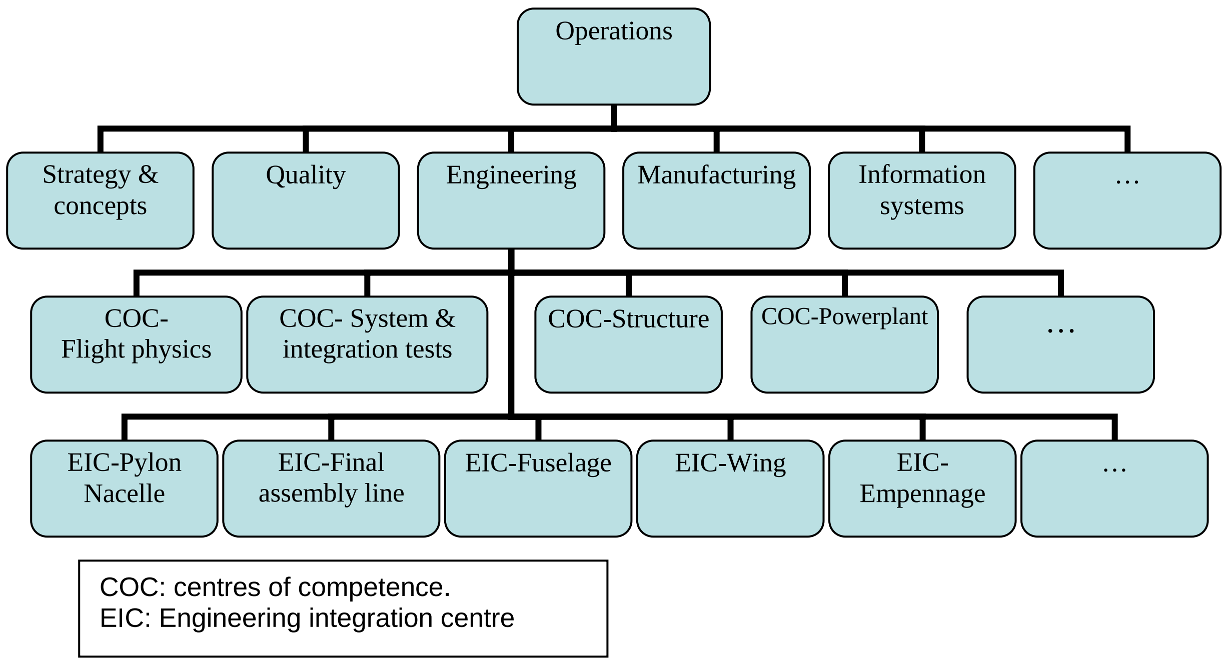


Figure 7: Previous Organisation Breakdown Structure

In Figure 7, we can see the former organisation where we can notice that Engineering and Manufacturing functions remain still separated.

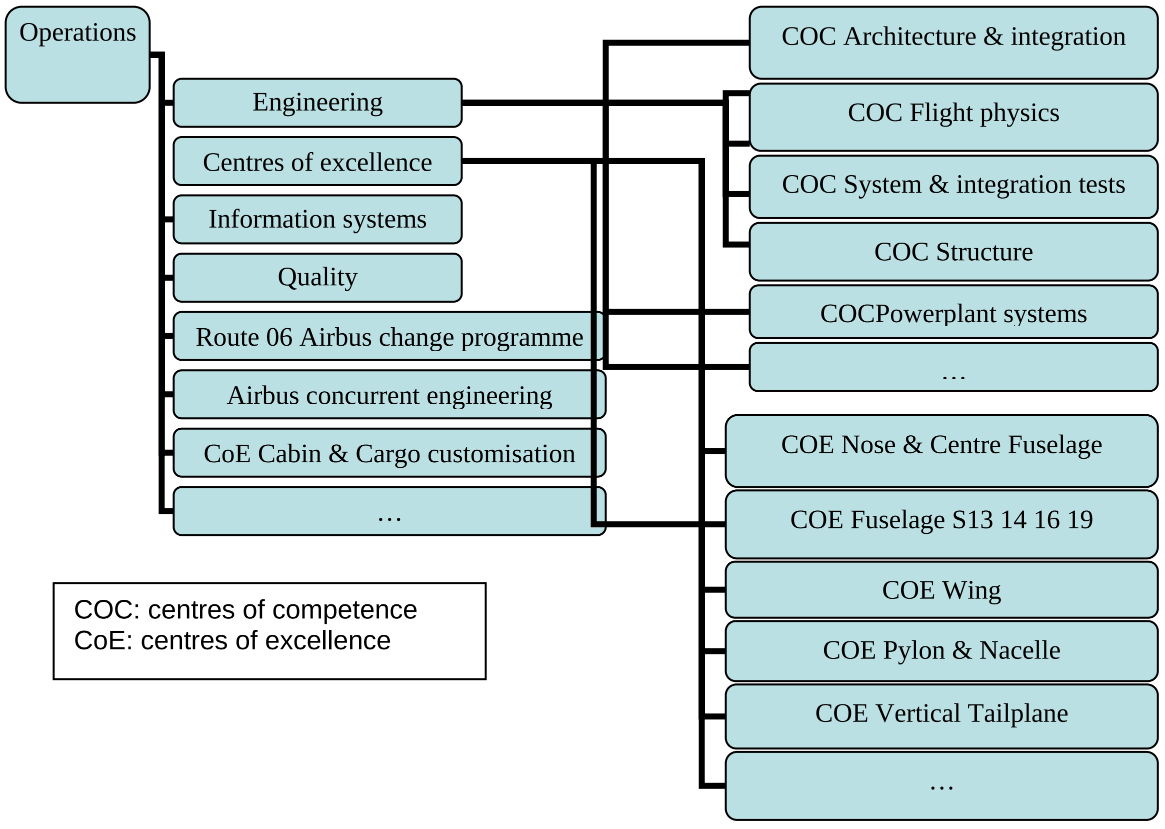

Operations
Engineering
Centres of excellence
Information systems
Quality
Route 06 Airbus change programme
Airbus concurrent engineering
CoE Cabin & Cargo customisation
…
COC Architecture & integration
COC Flight physics
COC System & integration tests
COC Structure
COCPowerplant systems
…
COE Nose & Centre Fuselage
COE Fuselage S13 14 16 19
COE Wing
COE Pylon & Nacelle
COE Vertical Tailplane
…
COC: centres of competence
CoE: centres of excellence

Figure 8: New Organisation Breakdown Structure

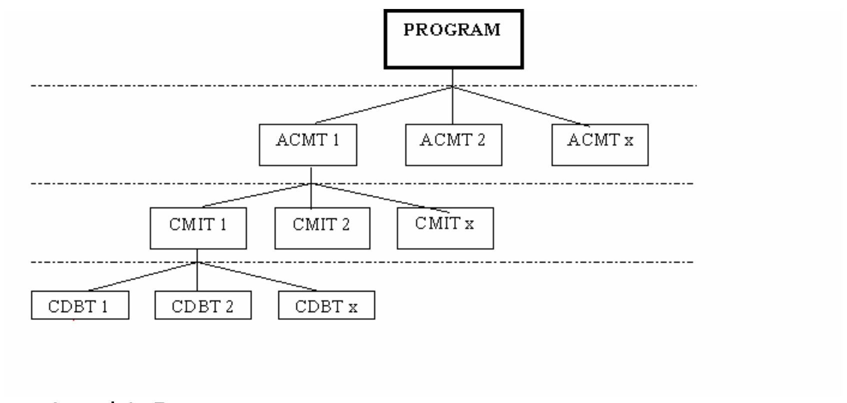
In Figure 8, we see the new organisation a new concept called Centres of Excellences (CoE) was introduced. The Centres of Competence (CoCs), set the Strategy, Standards, Policies, Methods, Tools, Skills in Design Work. Within the CoCs, the domains are the owners of all engineering disciplines and are responsible from the general functioning of individual processes. The Centres of Excellence (CoE) concept was proposed to model a transnational integrated organisational entity that interfaces Engineering and Manufacturing with Programs for all aircraft Programs in development or in serial phases.

The Centres of Excellence use the resources and capabilities of the Centres of Competence in conjunction with Programs. Each Centre of Excellence defines the need for and establishes the engineering grouping in order to:

- define the architecture of its section of the aircraft
- select the standards, tools, processes and methods
- integrate the Non-Specific Design Work (NSDW) and the Specific Design Work (SDW)[1]
- ensure the section design is harmonised
- ensure the design is manufacturable
- ensure the maintainability
- produce the manufacturing pack

This new organisation influenced the way schedules were built as we will see in this chapter.

On the other hand, if we focus on a program, we will identify different managerial levels as shown on Figure 9:



- Level 1: Program management.
- Level 2 : ACMT (Aircraft Component Management Teams)
- Level 3 : CMIT (Component Management and Integration Teams)
- Level 4 : CDBT (Component Design and Build Teams)

Figure 9: Program Organisation

[1] See 1.5.3

In this document we will propose a simplified view of this hierarchy. We keep only three levels: the program level, the managerial level and the design level. This last one corresponds to the CDBT level, which is the "working level" formed by design teams. In fact, CDBT include an engineering staff but also members of manufacturing and other functions. Therefore our "design team" definition seams is then an approximation of the real entity instances of this level. This approximation is acceptable since the scope of our project addresses only engineering practices. We will then use the term "design team" from now on.

In the case of the A380 program, the Organisation Breakdown Structure has been mapped to the Product Breakdown Structure, described in the next chapter.

1.5.2 Product Breakdown Structures (PBS)

The Product Breakdown Structure is a tree structure including the different subsystems that form the product. We can classify these subsystems into "End products " and "Enabling products" [GEIA '03].

While the former is the aircraft itself, the later deals with the systems associated with manufacturing/production, test, other stakeholders' deployment/installation, training, support, and disposal (including disposal) processes enabling product including enterprise capacities (facilities, equipment, tools, and staff) to accomplish these processes. Enabling product baseline documents include a wide range of documents that could include manufacturing plans, supportability planning, supply documentation, manuals, training plans, test planning, deployment planning, and others [DOD '01].

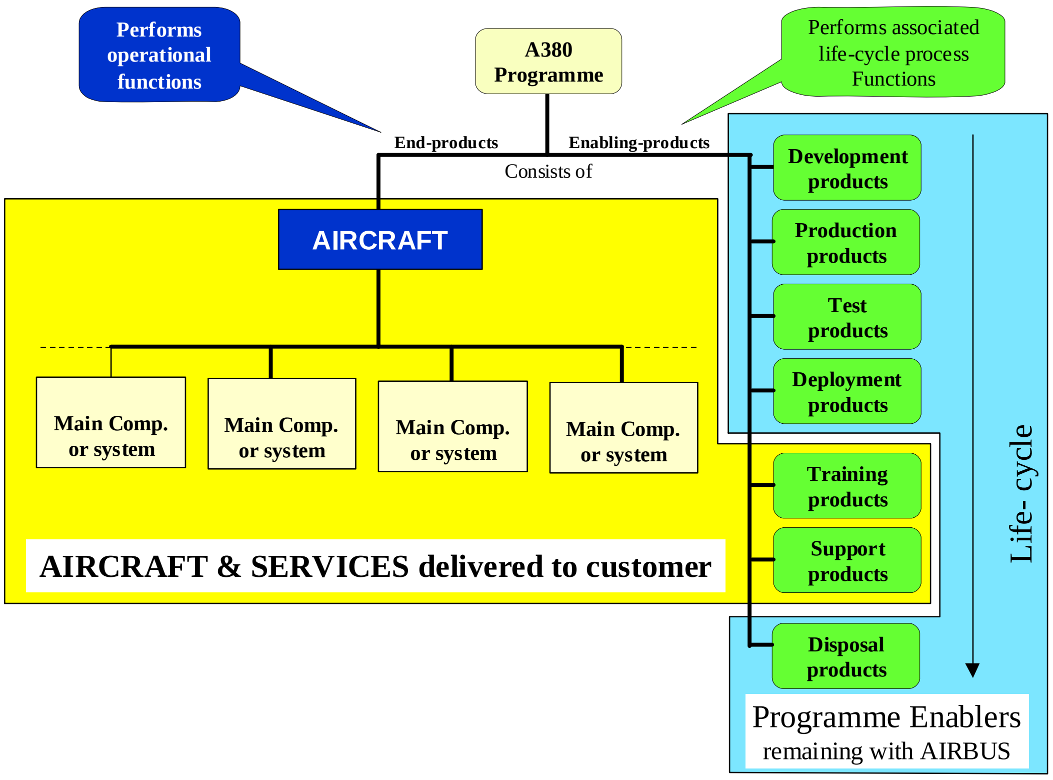


Figure 10: "End products" and "Enabling products" of an aircraft program

Figure 10 describes the complete PBS of an aircraft program detailing the end product part. As we have already seen, in last Airbus programs the organisation of the teams has been mapped onto the product breakdown structure. From a scheduling point of view, this change has introduced new type of schedules including different functions information and presenting more explicitly the simultaneous tasks performed by different functions responsible for the same subsystem.

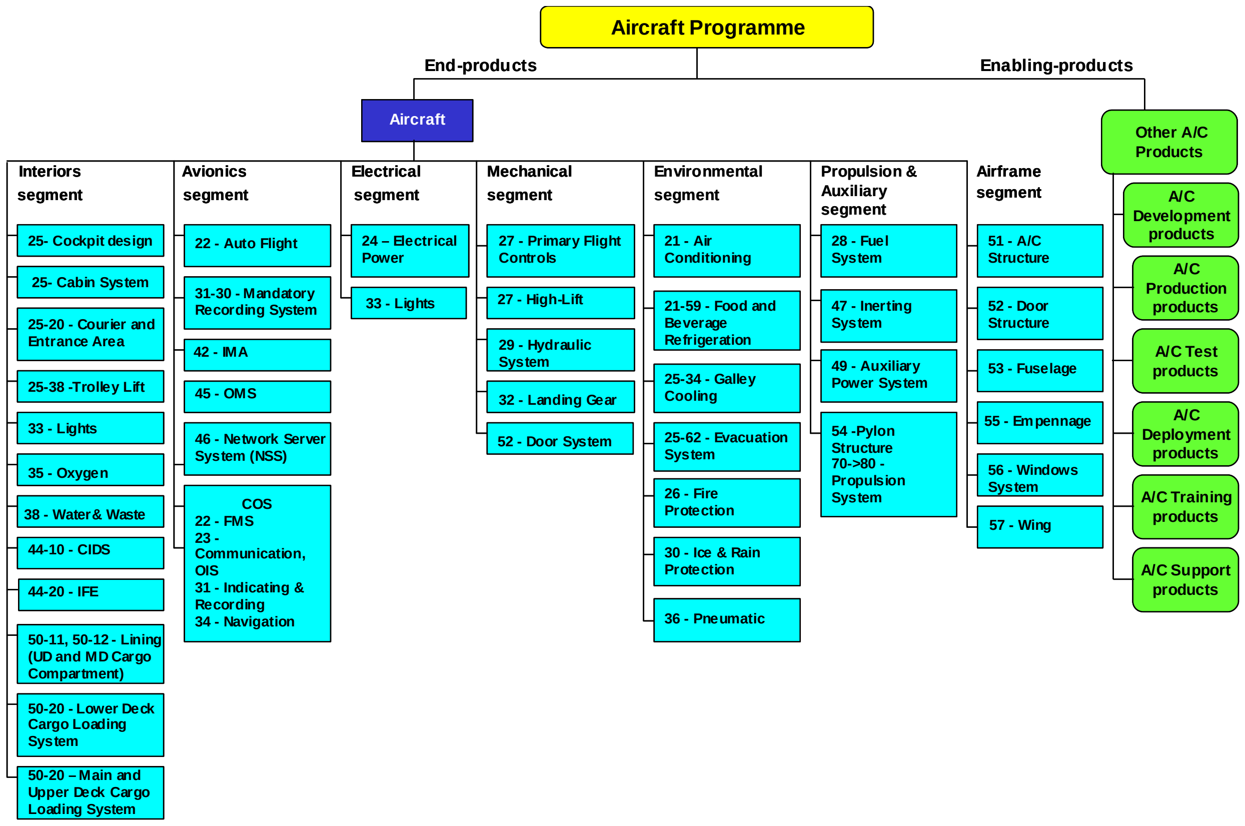


Figure 11: Product Breakdown Structure of an aircraft program

1.5.3 Work Breakdown Structure (WBS)

The Work Breakdown Structure is a hierarchical tree that includes the tasks to be done during the project. It is a fundamental representation in order to begin the schedules definition as well as to define the scope of each team. The goal of a Work Breakdown Structure is not to define all the actions that are needed to undertake during the project but to exhibit their outputs, in order to ensure the completion of the final product.

In the case of Airbus two types of documents characterize the WBS. On the one hand, the worksharing documents, which define the distribution of design activities between different teams. This include an accurate description of the subsystems and the scope of each team as well as the list of the outputs linked to the design supports, assigning each output to a team. On the other hand, tree structures will represent schematically the logic of the design and manufacturing activities. The leaf nodes of these structures are composed by Work Packages (WP), which represent a collection of work actions necessary to create a specific output.

In some cases each team defines its scope colouring the tasks of the tree structure that are responsible for (see Figure 12:).

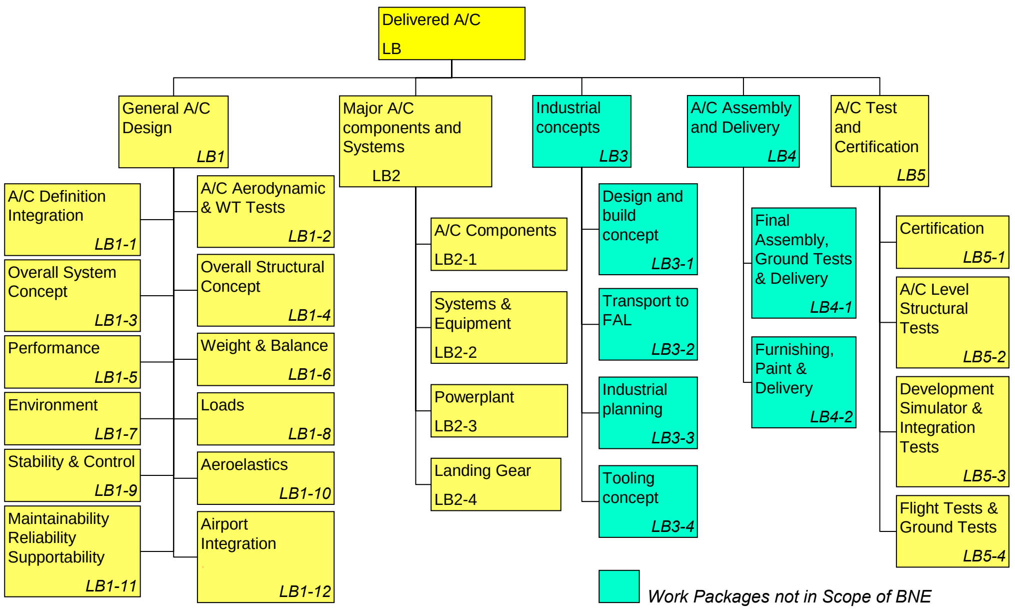


Figure 12: Work Breakdown Structure of an aircraft program

Design activities can be separated into two categories. On the one hand the Non-Specific Design Work (NSDW), which are activities linked to overall design of the system and which include the studies related to aerodynamics, materials, fight physics, etc. And on the other hand, the Specific Design Work (SDW), which are activities directly related to the definition of a subsystem or part of the product. WPs related to NSDW assemble the activities in the scope of the functions not directly related to the design process, while WPs related to SDW correspond partially to the subsystems that are defined in the PBS. Some of the WPs related to SDW include the integration of a transversal subsystem. This is the case of the subsystems that are included in each of the structural parts of the aircraft. Electric cables or pipes for different usage are good examples of this case. Considering a team who is in charge of the development of these pipes and another team who is in charge of the development of a structural part of the aircraft; the later will be responsible not only for the WPs related to the definition of the structure, but also for the WP related to the integration of the pipes in the structure. Consequently, two teams belonging to different branches of the OBS, will need to cooperate in order to integrate efficiently two subsystems of the final product.

1.6 Conclusion

Contextual analysis has highlighted the complexity of the product and the development project associated with it. In a market that is sensitive to variations in the economic environment, and subject to fierce competition, aircraft manufacturers have to exploit their experiences and all the skills at their disposal in order to introduce a new aircraft in a short time period whilst maintaining their profit margins.

While this challenge raises important technical problems, it also has an impact on the configuration and on the performance of organisations. The development and production processes for a new aircraft rely on a real network of partners and subcontractors, led by the manufacturer. An organisation based on the “integrated company” type has therefore been replaced by an “extended company” structure.

Taking into account the complexity of the product and this specific configuration of the organisation, project management system will need to deal with high number of resources, numerous detailed activities to be scheduled, coupled together by interdependencies, which causes numerous dynamic interactions between different teams. These aspects concern above all the design activities of the new aircraft development.

In the next chapter, we will go deeper into the characteristics of the design activities in order to understand the resource and the time constraints that influence the scheduling of these activities.

Chapter 2: Managing design activities of a new aircraft development

2.1 Introduction

As we saw in the previous chapter, development lead times are particularly long for an aircraft and it is crucial for the organisation to search for time savings at the various phases of this aircraft's lifecycle. The first objective of this chapter is to understand the design cycle for an aeronautical product which has been, for the past ten years, widely influenced by the concurrent engineering (CE) paradigm. CE aims to take benefit from opportunities of the parallel scheduling of activities implying distinct resources in order to shorten the overall duration of the design process. Analysing the design process will also highlight the issues linked to management of the product's informational heritage which consists in different documents exchanged between design teams. Before proposing a diagnosis, any improvement or any new tool, it has been necessary for us to observe the current scheduling practices focusing on the schedule management process and the description of the different schedule supports used by a variety of actors during design. We will stress the difficulties and the needs faced by these actors when scheduling design activities, allocating resources and managing data exchanges with other teams. The analysis of these difficulties will enable the definition of clear research problem statement.

2.2 CE: principles and application in the aviation industry

The AFNOR X50-127 standard specifies that "starting from the needs, the design process defines step by step the product that must meet the needs and expectations, through successive choices concerning increasingly detailed points" [AFNOR '02].

However, as Darses et al states in [DARSES F. '01], there is no predetermined path between the expression of needs and the specification of the solution, although solid methodologies and procedures can be useful, as can past experience.

Thus, the design process is not deterministic but aims rather to establish a framework in which designers' work can be managed, and their interactions controlled, whilst meeting the schedule constraints. Within this framework, the designers put into practice knowledge, procedures and methodologies.

While in the past sequential design processes enabled companies to respond to market needs, currently, models based on concurrent engineering concepts enable companies to meet the challenges of the civil aviation market.

In the late 1980s, the first major principles of concurrent engineering emerged in the automotive industry. This development was highlighted by comparative studies, in particular that of Clark et al [CLARK, et al. '87], concerning the development processes of new products produced by American, European and Japanese automobile manufacturers, in order to explain the success of the latter. Part of this work contributed to the success of Womack's study [WOMACK, et al. '91], "The Machine That Changed the World", which, for the first time, proposed an organisational and project management model that offered a real alternative to the traditional principles of sequential, “compartmentalised” engineering.

While this study brought these new trends regarding the organisation and management of development projects into the public eye; the US military industry had already revealed, in the early 1980s, certain concepts linked to information management for development projects. This initiative, known as CALS (Computer Aided Acquisition and Logistics which later became Continuous Acquisition and Logistic Support), presented by the Institute of Defense Analysis (IDA) to the Department of Defense (DoD), proposes a strategy concerning the management and exchange of electronic data (design, manufacturing and maintenance dossiers, etc.) between organisations involved in a military program. The objective of this strategy was to improve the management of information flows between players working on the same program and, consequently, indirectly promote the concurrent performance of activities. The innovation, proposed in the framework of the CALS initiative, is based on the implementation of integrated electronic systems for design, calculation and simulation, as well as data exchange and management systems enabling all players to access information related to the project. CALS therefore dealt with aspects linked to information technology, with the definition of exchange standards, before widening its scope to cover methodological and organisational aspects.

2.2.1 Origin of CE

Trends in product development processes were modified in the late 1980’s when the interrelated approaches like Concurrent Engineering [HAUG '93] and Integrated Product Development [ANDREASEN and HEIN '87] showed to be powerful approaches for maintaining competitiveness. Advantages of concurrency was mainly be illustrated by the significantly lead times reductions that some Japanese automobile companies could achieve [LARSSON '05, WOMACK, et al. '91].

In 1988, the IDA [WINNER, et al. '88] proposed the Concurrent Engineering approach as a methodological approach integrating the simultaneous development of products and processes (including manufacturing and logistics support). This approach takes into consideration, from the outset of the project, the product’s lifecycle, from design to use, including quality, costs, planning and user needs. This definition is strongly based on military scenarios. This is why the word “support” is replaced by “maintenance” in some studies.

In 1994, the AFNOR chose to favour the term “Integrated Engineering”, which it defines in [AFNOR '94] as “an approach that involves the simultaneous taking into account of different needs related to different phases of the product’s lifecycle.” This approach implies an integrated, simultaneous view of the products and associated processes,. It enables developers to take into account, from the start, the entire product lifecycle, from the initial expression of the need to the withdrawal from service.

The important element to retain from these two definitions is the integration, from the product design stage, of the constraints from downstream phases of the lifecycle. However, this element must not hidde all principles brought by concurrent engineering concept.

2.2.2 Principles of CE

The adoption of CE principles by a company has been driven by different factors:

The first factor is linked to the competition that exists between companies on some markets, and the resulting competitiveness factors. For instance in the aviation industry, whose main customers are airlines, these factors are the introduction of innovative elements in the design of new aircraft or in the aircraft in-service. In this case, it is the link between competitiveness and innovation that is highlighted, the pace at which new products are introduced onto the market to meet the expectations of airlines. A faster pace over a given period of time ensures that one manufacturer will gain an advantage over its competitors in certain market niches.

Given these issues, and with respect to the two dimensions discussed previously, it is possible to envisage the principles of concurrent engineering as affecting three major areas that are very closely correlated [PRASAD '95].

The first area concerns the organisation and management of the development processes, and the scheduling of the design tasks. In a traditional development process, the processes and tasks occur in a sequential way. The termination of one task therefore enables the start of the next activity. One of the essential concepts of concurrent engineering is the fact that tasks and processes can be performed *in parallel*. This approach means that information that is vital to the two tasks must be explicit and correctly managed. Thus, a downstream task can begin when the information produced by an upstream task is judged sufficiently mature, even before the task in question has not been completed [MARTIN '01]. Such a system involves a careful management of the overlap between tasks [TERWIESCH and LOCH '97]. Although there is a high degree of uncertainty with this type of management, the clear advantage of the approach is linked to the reduction in the number of modifications to be dealt with in the development process.

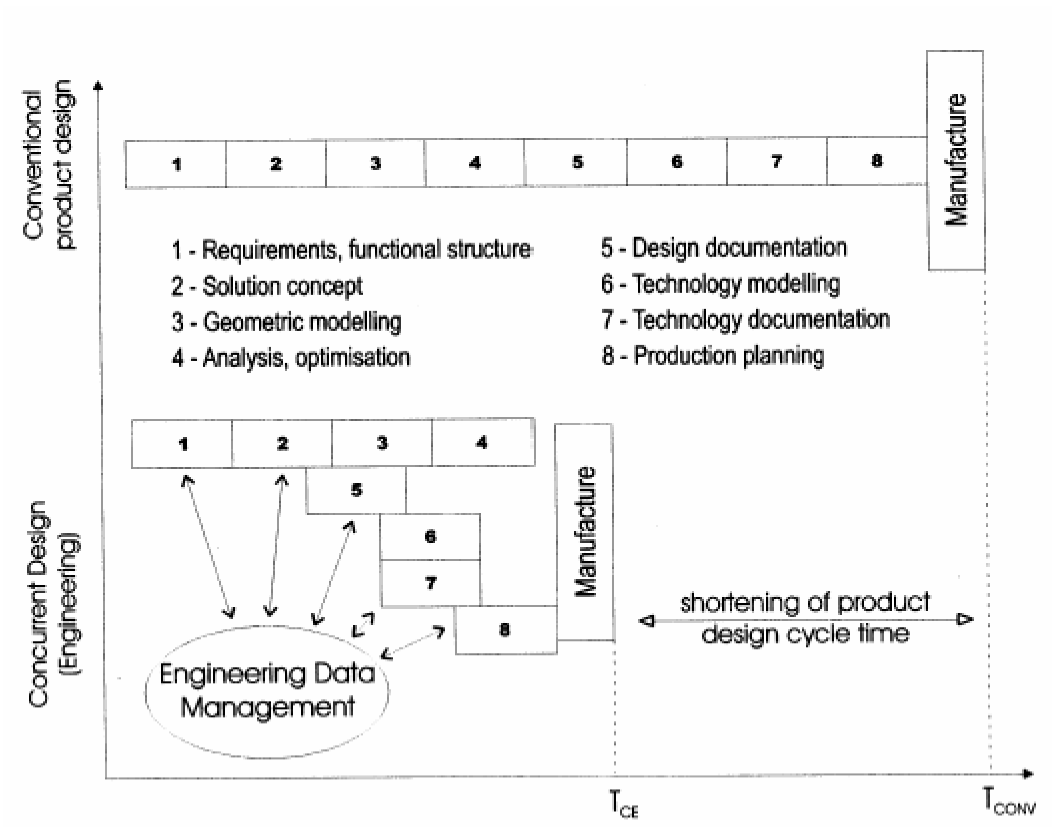

Conventional product design
1
2
3
4
5
6
7
8
Manufacture
1 - Requirements, functional structure
2 - Solution concept
3 - Geometric modelling
4 - Analysis, optimisation
5 - Design documentation
6 - Technology modelling
7 - Technology documentation
8 - Production planning
Concurrent Design (Engineering)
Engineering Data Management
shortening of product design cycle time
T_{CE}
T_{CONV}

Figure 13: Sequential Vs. Concurrent Engineering [CHLEBUS '98]

Management of the development processes based on the concurrent engineering model is therefore centred on task scheduling, coordination of players and information exchange. The last point concerns both the players involved in development and those involved in the other product lifecycle phases, since these constraints have to be taken into account from the start of product design.

The second area reconsiders organisational structures set up for development projects. Since effective coordination of processes and high quality informational exchanges are the proof of a successful concurrent engineering project, it is preferable to move away from the traditional vertical project structure and replace it with autonomous teams with cross-functional skills [SABBAGH '95]. To achieve these objectives, "project plateaux" have been set up, enabling all the players involved in the development of a new product to work together in the same place. In this context, the effectiveness of these players and the project performance can only be guaranteed if a sufficient decision-making power is left to project managers, and if efficient support processes for the plateau are available. This major principle of concurrent engineering supports cross functional teams to collaborate across traditional functional areas of expertise [WINNER, et al. '88]. A key objective is to *"improve communication between the many involved people including management, designers, product support, vendors and customers" [PRIEST and SANCHEZ '01]*.

Finally, the third area deal with the definition of a single tool to ensure the consistency of the information handled during design, while centralising all the viewpoints of different skills involved in the process. This problem is now partially resolved thanks to the development of "product" information systems and the integration of TDM (Technical Data Management) tools which are nowadays deployed as PDM (Product Data Management) or PLM (Product Lifecycle Management) software.

As Françoise Darses highlights in [BOSSARD, et al. '97], "one of the advantages of concurrent engineering is that it offers a more valid organisational model with respect to the cognitive processes that underlie design activities," in particular because the proposed model takes into account – as early as possible – the constraints to be met by the product. From a strategic point of view, these cognitive arguments alone are not sufficient to adopt the principles of CE within an industry.

2.2.3 Application in the aviation industry: implications and results

If we refer to the experience of the European Aircraft manufacturer, it is possible to identify the advantages of such concepts in practice. In 1997, for the development of the 500/600 version of the A340, AIRBUS decided to move away from an activities-based organisation in favour of a more product-based organisation, through the launch of the ACE (Airbus Concurrent Engineering) project. Through the integration of CE concepts, the initial aim of this project was to reduce development times and costs, whilst ensuring an enhanced product reliability at entry into service. These concepts were introduced via the following principles [PAPAZOGLOU '01, SCHEIBLE '02]:

- setting up of a strong management structure led by the program.
- parallelising of sub-processes and standardisation of repetitive design tasks.
- setting up of integrated teams (design, manufacturing and support) with common responsibilities, located on the same plateau (up to 600 people, including all teams and disciplines). The integration of these teams led, in particular, to increased consistency between processes, while promoting the flexibility of such processes thanks to greater anticipation of decisions.
- integration of subcontractors into these teams with the definition of a breakdown of responsibilities.
- definition of procedures, methods and databases to be shared by all these teams.
- definition of a development schedule based around major milestones which, once achieved, give rise to a project review. The purpose of these reviews is not to establish the state-of-progress of the activities but rather the state-of-progress of the product design with respect to the reference configuration or baseline.

- definition of a single product information system that is shared by all the players involved in the project. The setting up of this product information system promoted exchange and communication between players, as well as the development of common PDM and CAD tools. Such an information system also authorised the reconciliation (homogenization) of the data produced, which in turn enabled development of the digital mock-up, of which the graphical component represents only the tip of the informational iceberg.

During development of the 500/600 version of the A340, the introduction of these concepts enabled a 25% reduction of the design time with respect to the A340 basic version, and a 15% reduction with respect to the A320. This reduction contributed partly to the 30% reduction in development costs with respect to the A340, which represents a saving of € 50 M. When the first aircraft went into service, its reliability rate was 99%. But perhaps one of the greatest successes of this project is the fact that it advanced the integration of the partners of this manufacturer which had been, until then, represented by several different entities.

For the development of the A380, the lessons learned from the A340 500/600 project enabled the deployment of more effective CAD and PDM tools that help to ensure the consistency of the design and help control the product complexity.

2.3 Development of a new aircraft

During development of a new aircraft, the players involved in the program intervene simultaneously (Figure 14) according to the objectives that have been allocated to their "swim lane". A "swim lane" is a group of activities dedicated to a specific phase of the product lifecycle or specialised according to a type of product sub-assembly. This breakdown helps us to identify skills and knowledge networks for each phase of the product lifecycle, and to identify the information handled by the players involved in the program.

These Activity Lines are as follows:

- "Overall Aircraft Design (OAD)":
- This includes the teams that propose concepts for the definition of new aircraft and analyse their repercussions on the mechanics of flight (aerodynamics, performance, loads, aero-elasticity, structural resistance, acoustics, etc.).
- "System":
- It covers all the activities related to the definition of electric, hydraulic and other devices not included in the structure.
- "Structure / System Installation":
- It consists in all activities related to the definition of the aircraft structure and fixed brackets for the installation of the systems within that structure.
- "Assembly":
- It groups all the activities involved in the final product integration and thus uses the results of the activities carried out by the industrial activity lines.
- "Industrial activity lines":
- These lines are dedicated to the manufacture and procurement of detail parts and tooling; they also relies on the intervention of teams such as painting and wiring.
- "Customer Support activities":

- They ensure that maintenance requirements are taken into account during the development phases; it covers then activities that produce support products (tooling, spares, documentation, etc.).

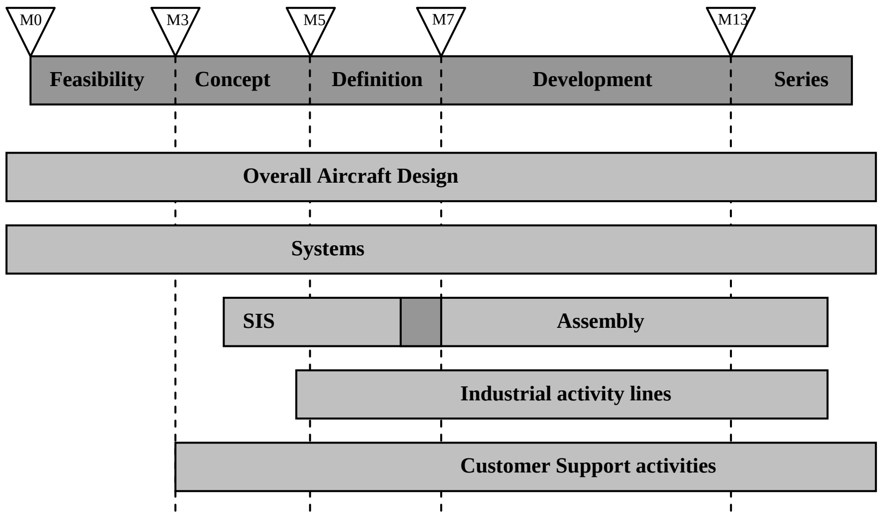


Figure 14: Skills in the lifecycle

Although these swim lanes can be dissociated, they carry out their work simultaneously. However, there may sometimes be more interactions between certain Lines.

For example, the “Structure / System Installation” (SIS) Line will only intervene once the preliminary concepts have been validated by the OAD Line.

Although some of these Lines intervene throughout the product lifecycle, this study will be restricted to the phases from design (Milestone M3) to entry into service of the first aircraft, i.e. the end of the development phase (Milestone M13).

2.3.1 The concept phase

The concept phase begins after closure of the feasibility phase and includes two important sub-phases, further detailed below):

2.3.1.1 Optimisation of concepts at aircraft level (Milestones M3-M4)

The objective of this phase is to optimise and fine-tune the concepts defined for the overall aircraft in order to establish an initial aircraft configuration during development.

After this initial configuration, it is possible to make contact with potential customer airlines and major suppliers. Thus, at the end of this phase, the following activities will have been carried out:

- study of the operational environment in which the future product will be used and gathering of opinions from potential customer airlines
- establishment of specifications at product level
- optimisation of the concepts linked to the major assemblies
- definition of the initial product reference configuration (Product Definition Level 0)
- first contact with potential suppliers

- technologies to be used are assessed as sufficiently mature and available.

2.3.1.2 Consolidation of the reference configuration by design (Milestones M4-M5)

The objective of this phase is to enrich the technical concepts related to the initial reference configuration, by associating them with information related to marketing, production, maintenance, support, certification, costs and recycling of the future product. It also aims to detail the specifications, define the risks associated with the project and determine work sharing between the different entities of the organisation, with the aim of drawing up a technical definition of the product. During this phase, the following activities will have been carried out:

- definition of work sharing
- definition of the project schedules
- approval of recurrent and non-recurrent costs
- second reference configuration made available (Product Definition Level 1)
- allocation of resources and budgets up to milestone M7
- approval of the manufacturing concepts
- definition of the operational infrastructure.

2.3.2 Definition phase

During this phase, the product progresses from the “detailed concept” stage to the stage of product fully defined by drawings and models. Usually, it includes two sub-phases:

2.3.2.1 Finalisation of the specifications and commercial proposals (Milestones M5-M6)

At this stage of the product lifecycle, the complete specifications of the future aircraft are available, the commercial proposals have been finalised and the first financial assessments of the major suppliers should be available. At the same time, the following activities are carried out:

- production of the detailed definition schedule
- drawing up of a performance guarantee with respect to the proposed concepts.

2.3.2.2 Definition of components (Milestones M6-M7)

Between these two milestones, the detailed definition of the physical product is produced, as well as its functional simulation, down to the basic components. Furthermore, during this phase, financing of the upstream phases (development and production – milestones M7 to M14) is carried out. Contracts are signed with the launch customers. As for the other phases, other activities are carried out during this period, such as:

- product definition made available (Product Definition level 2)
- decisions made regarding components (produced internally or outsourced)
- definition of financing and schedules for the development phase of the first aircraft.

2.3.3 Development phase

In the aviation industry, the development phase corresponds to the production, assembly and testing of the first aircraft. It enables the manufacturer to prepare for series production of the new aircraft. However, the actual development phase lasts from milestones M7 to M13. While it doesn’t contribute directly to the product definition, some of the players involved in this phase are those of the previous phase. Numerous iterations are therefore triggered between these two phases.

2.3.3.1 Start of manufacturing (Milestones M7-M8)

At this stage, the manufacturing standards are drawn up to enable production of the first aircraft (except for those components that require longer production times, and whose production therefore began earlier – centre section 15/21, for example). Consequently, the bills of material and the production processes are defined, as are the tooling and machining installations. This phase is considered to be complete when:

- the production-orientated product definition is made available (Product Definition Level 3),
- the production sites are preparing for the manufacturing launch,
- the aircraft certification program has been drawn up.

2.3.3.2 Manufacture of components, assembly and testing of fitted sections (Milestones M8-M9)

During this phase, the basic components are produced and the different aircraft sections are assembled by the company’s production sites. Functional tests are then carried out on each section before their delivery to the Final Assembly Line (FAL), according to the schedules defined. At the same time, the assembly sequences, capacities and operational rules are defined with the final product assembly in mind.

2.3.3.3 4.2.3.3 Final assembly (Milestones M9-M10)

When the assembly line and tools are available, assembly of the different sections can begin on the FAL (Final Assembly Line), where the engine is also integrated. Once this assembly is complete, the various aircraft sections are connected and tested to ensure that the systems linked to the aircraft flight controls perform correctly. The aircraft is then partially equipped for its test flight and the ground test procedure is available before the first aircraft is delivered to the test centre.

2.3.3.4 Ground tests and preparation for first flight (Milestones M10-M11)

At the end of this phase, the aircraft is ready for its first flight – with some limitations, to ensure that the operation is carried out in complete safety. To this end, all the functional and ground tests have been carried out, as have the safety tests. In addition, emergency systems and equipment are installed on the aircraft to ensure the safety of the flight crew in the event of failure of the aircraft’s vital systems during the tests.

2.3.3.5 Type certification and check of the conformity with respect to the standard specifications (Milestones M11-M12)

After a flight test campaign, a type certification request can be made to the Authorities. To obtain this certificate, the manufacturer must provide a complete set of information, containing: the production documents and inspection procedures, the technical, maintenance and repair manuals. At the end of this phase:

- the aircraft’s standard specifications are available
- the results of the flight tests are used to consolidate the aircraft’s basic configuration
- the aircraft’s maintainability is demonstrated
- the aircraft is certified by the Authorities.

The progress of the different phases involved in the development process of a new product is based partly on the development, negotiation and validation of the models that describe the artefact being designed. They are considered as informational supports, at the centre of cooperation between the players in the extended company. We describe the most important of them below.

2.4 Informational supports for the development of products using concurrent engineering

During the development of a new aircraft, numerous models are generated to describe the product configuration at a given moment in its lifecycle from a particular point of view (Figure 15). These models form part of the product's informational heritage that must be manage in configuration.

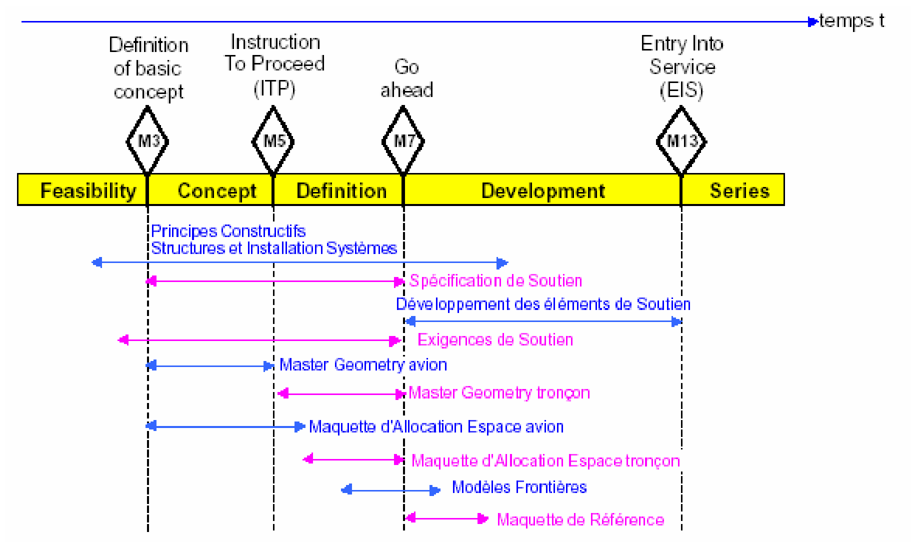


Figure 15: Aircraft Models

2.4.1 Formalisation of potential technical solutions (design principles)

There are often several solutions to a given problem. The design principles are used to propose several solutions for each problem and to enable the selection of the best solution from a technical and financial point of view. There are two types of design principles: generic design principles and specific design principles dedicated to a given program. Generic design principles can be divided into three categories:

- those related to the technologies used: materials, assembly techniques.
- those related to the product architecture: study of the certification regulations, position of the main sub-assemblies (LG, wings, engines, etc.)
- those related to the detailed design (geometry of the components, without specific dimensions)

The design principles are essentially design drawings for a zone, produced using CAD. Depending on the state-of-progress of the project and the elements available, the design principles can be 2D drawings (splicing design) or 3D models (routing design). Depending on the activity in question, these models can be used in different ways, as follows:

- the "Design / Engineering" activities will use them mainly for modelling of elements or assembly modes,
- the industrialisation activities will use them to analyse solutions, identify and determine the parts that have a long production cycle and anticipate procurement,
- the program will use them to validate the selected solutions.

2.4.2 Equipment specifications with space allocation (Structures and System Installations)

These specifications actually consist in two documents: the *system installation specification* (SIS) and the *installation dossier* (ID). The first document is a contract drawn up between the system designer responsible for a given equipment item and the equipment installers. It defines the initial volumes as well as the equipment's general and specific criteria. This document is drawn up by the system designer and is used as a basis for the equipment mock-up. It contains:

- the equipment space allocations and its environment (accessibility, maintainability, etc.)
- the overall dimensions of the equipment
- the frontier drawing with the dimensions
- the equipment configurations for flight testing.

Thus, we can distinguish four main actors that handle this type of model during equipment development:

2.4.2.1 The system designer

The system designer is responsible for the equipment specifications. The role of this player is to:

- determine the basic equipment design choices, which are used to decide the initial overall dimensions.
- coordinate the mock-up activities between the equipment mock-up designer, the system installers and the equipment suppliers,
- validate the equipment installation.

2.4.2.2 The equipment mock-up designer

The equipment mock-up designer shall:

- produce the 3D mock-up related to the equipment installation based on the installation specification
- study and propose installation solutions
- resolve, with other partner working in the same zone, any installation problems (interference, conservation, etc.)

2.4.2.3 The system installer

The system installer produces a mock-up of the hydraulic pipes, the electrical harnesses, the mechanical controls, etc. that are linked to the equipment).

2.4.2.4 The structure installer

The structure installer produces a mock-up of the structure in the zone where the equipment is installed.

2.4.3 Taking into consideration support requirements during aircraft development

The idea of this approach is to express, fully and efficiently, during the early design phases, the constraints and requirements of the customer airlines as regards the support of their future aircraft. It guarantees to the airlines that their product will be delivered with maximum availability, economical to use and maintain, whilst complying with all the safety requirements. These requirements and constraints are formalised in the form of specifications that can be used by any people involved in engineering.

2.4.4 Definition of the product shape and baseline (Master Geometry mock-up)

The Master Geometry mock-up is developed to have an overall vision of the aircraft's theoretical references. It is the official reference for external profiles and geometrical references. Consequently, it combines all the external forms of the aircraft, the main geometrical references (references of frames, stringers, ribs and spars, major interface points, etc.) and all the associated items. These reference elements are produced in wire-frame or surface mode. Since it defines the aircraft references, this mock-up can be used by a large number of actors in the development process, including those responsible for aerodynamics, structure designers, toolmakers, etc.

2.4.5 Space allocation for definition of components (Space Allocation Mock-up)

The space allocation mock-up is created in order to have an overall vision of the aircraft represented by simplified volumes. It therefore represents a complete theoretical aircraft with its structure and systems, according to a homogeneous representation (simplified solid) based on the solutions proposed by the design principles.

The space allocation mock-ups can be used by several activities:

- "design / structure" activities
- "design / systems Installation"
- activities linked to support engineering
- activities linked to industrialisation

2.4.5.1 The "Design / Structure" activities

"Design / Structure" activities can use the space allocation mock-ups in order to:

- Represent, in a simplified volume, the essential parts of the structure (main envelope)
- Define the bases of the interfaces.

2.4.5.2 The "Design / Systems Installation" activities

"Design / Systems Installation" activities can use the space allocation mock-ups in order to:

- make a preliminary ergonomics study: ergonomics of the cockpit, access to equipment, etc.
- pre-install the overall system volumes, integrating the circuit segregation rules and their different physical properties
- check any interfaces between the systems
- pre-install, in a simplified way, the main routings (electrical, hydraulic, pneumatic)

2.4.5.3 The activities linked to support engineering

These engineering activities can be used in order to:

- Validate the logistics support objectives
- Simulate the maintenance tasks (removals, general maintenance).

2.4.5.4 The activities linked to industrialisation

The activities linked to industrialisation can be used in order to simulate the possible section breakdowns and plan the final assembly tools.

The space allocation mock-up is usually validated during meetings known as "design reviews".

2.4.6 Management of interfaces (frontier models)

A frontier model defines the responsibilities at the frontier between two sections of the aircraft that are placed under the responsibility of two different units within the same organisation. These models can be considered as contracts between two units of the organisation. They are therefore used as references for the validation of definitions, as support for the industrialisation definition and as references for the analysis of assembly problems. To ensure they can be used effectively during development, the frontier models must contain certain information, such as:

- the final functional requirements of the product
- the detailed design at the frontier (detailed design principles)
- sharing of responsibilities
- physical baselines (accessible for measurement after assembly)
- the functional dimensions of the various batches that constitute the aircraft
- sharing of tolerances
- provisions for drilling, assembly, etc.
- definition of the space reserved for installation and removal of spare parts (key dimensions)
- adjustable parts and the values of their gaps if applicable

2.4.7 Representation of the aircraft, defined through the geometric reference mock-up and the definition dossier (DD)

The product's progress in its definition phase leads us to replace the initial models represented by the space allocation mock-up by definitive part models. This advanced representation of the product is called the *geometric reference mock-up*. It provides a geometrical representation of:

- the detail parts (parts models, used for production of parts drawings)
- the assemblies (with the tree enabling representation of assembly drawings)

Once validated, the geometric reference mock-up constitutes the geometrical baseline of the definition dossier (DD). A DD is produced for every aircraft by the Design Office, prior to its manufacture and assembly. To be complete, the dossier must contain three drawing sets and all the modifications that have been made to the aircraft in question. These drawing sets are:

- *The mechanical drawing set* provides the mechanical definition of all the detail parts and the assemblies, with their accompanying parts lists. Also distinguishes between standard and non-standard parts and provides indications concerning the installation of the different items.
- *The electrical drawing set* provides the definition of all the systems, installations and electrical circuits. Also contains the definition of all the harnesses, cables, relays and

terminal blocks. The drawings it contains are therefore principle and wiring diagrams accompanied by the list of equipment used.

- *the equipment references* (purchased, outsourced, etc.) that are to be installed on the product.

The development of the product via the creation and successive collaborative fine-tuning of the models involves defining and implementing a strategy for managing the product's informational heritage.

2.5 The scheduling of design activities

Development of the aircraft's various subsystems can be placed under the responsibility of several entities within the extended company. However, the aircraft manufacturer must ensure that these developments are coherent, bearing in mind the final product integration.

In order to schedule the definition of the different design information support developed in the framework of the development of each subsystem, a specific scheduling management process is deployed based on different types of schedules.

By setting up such a process and the resulting mechanisms, the manufacturer should be able to:

- supervise the progress of the design process for a complex product.
- guarantee managers access to coherent, up-to-date information concerning internal and external subsystems progress.
- check that, during the design phase, the product development is consistent with customers' expectations (internal and external customers).
- enrich and control the product's informational heritage over time.

2.5.1 Different levels of schedules

2.5.1.1 The master schedule

The master schedule at project level shows the majors phases, activities and events of the project, compliant with the project delivery target. These phases are cascaded down to the relevant lower levels to define the lower level master schedule. Each entity of organisation has its own master schedule compliant with the upper level master schedule.

The aim of the master schedule is to identify and communicate throughout the project team the overall project time objectives. The master schedule is limited to:

- major phases
- major Activities
- main products
- main interdependencies (outputs & inputs)

The master schedule does not include information on achievement or on progress. The master schedule size is typically one page of A4 paper.

2.5.1.2 The steering schedule

The steering schedule consolidates the detailed activity and establishes the interdependencies between project organisation entities. It provides the baseline for the monitoring & and the control of the progress. The target frame of steering schedule is given by the Master schedule.

The steering schedule shall contain:

- the milestones of master schedule without changes.
- the interdependencies milestones.
- output deliverables for other teams.
- the input deliverables requested from other project teams.
- the internal milestones as halfway milestones with appropriate maturity, useful to ensure preventive mode regarding outputs and product issues.

Interdependencies (outputs, inputs), products and internal milestones shall be clearly identified in the steering schedule.

The steering schedule shall cover:

- all phases
- all activities
- all interdependency milestones (outputs and inputs)
- all internal milestones
- all products if relevant detailed schedule doesn't exist or the main products if relevant detailed schedule exists.

The steering schedule includes all the necessary information to compare the current status and the reference. It is built to allow progress measurement of the activities performed and it is regularly updated.

The resources allocation is sometimes defined in the steering schedules. Nevertheless, defining design activities that include the human resources is not a generalised practice. Most of the teams prefer to treat the resource allocation problem after the definition of the schedule. This point will be discussed later since we propose to improve the quality of schedule by taking the resource constraints as soon as possible.

2.5.1.3 The detailed schedules

Detailed schedules are specific schedules which could be used at each level of organisation to perform the day by day management for their own activities. Usually detailed schedules are mainly used at the lowest level of the organisation.

Detailed schedules are used to track specific work products, for example the design principle, project drawings, the worksheet realisation in line with the schedule requirements as defined in the relevant Master schedule. Detailed schedules include all details necessary to compare the current progress status and the reference. It is then built to allow the progress measurement of the work performed and it is regular updated. A Regular report of detailed schedule enables to update steering schedules.

Detailed schedule shall cover:

- one or some phases,
- one or some activities,
- the entire product linked to specific activities or phases and all relevant internal milestones.

2.5.2 The scheduling management process

Two major phases define the scheduling process. On the one hand the definition of the schedule that will be used to control the project and on the other hand the control or steering phase where real progress will be compared to the schedule defined in the first phase. Moreover, the first phase can be separated into two sub-phases, a first step that deals with the creation of the first schedules mainly dealing with internal constraints and a second sub-phase that consolidates the schedule and establishes a schedule consistent with internal and external constraints. It is validated by external actors. Therefore, we have separated the scheduling management process into three steps: Schedule Built-up, Schedule Consolidation and Schedule steering.

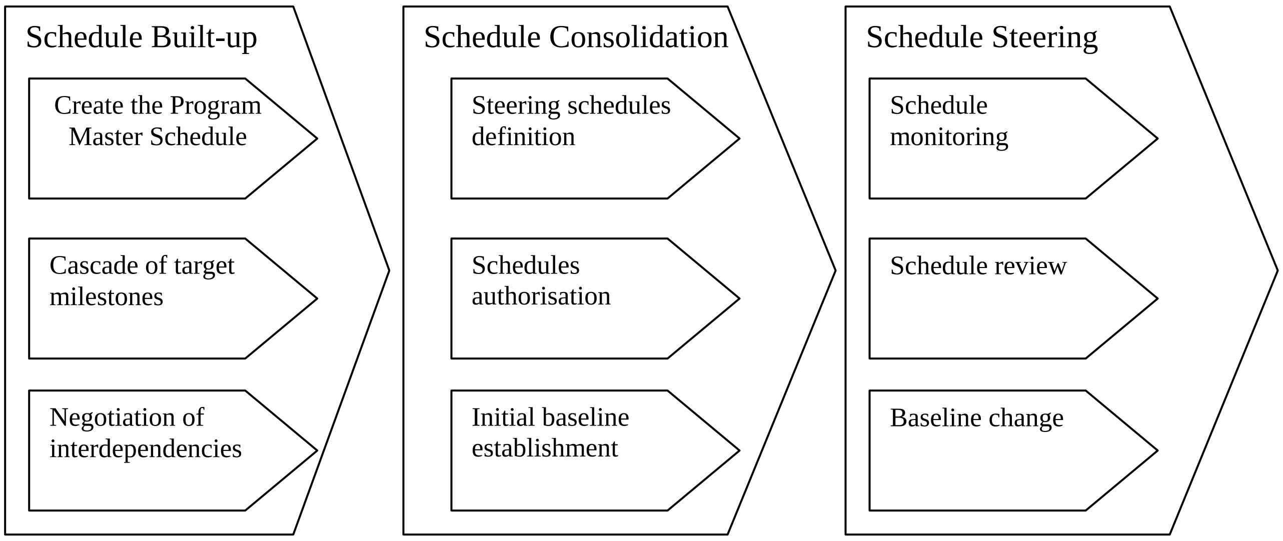

Schedule Built-up
Create the Program Master Schedule
Cascade of target milestones
Negotiation of interdependencies
Schedule Consolidation
Steering schedules definition
Schedules authorisation
Initial baseline establishment
Schedule Steering
Schedule monitoring
Schedule review
Baseline change

Figure 16: The Schedule Management Process

2.5.2.1 The schedule built-up

The built-up step begins by creating master schedules at different managerial levels. The major inputs to build the project master schedule are:

- the project delivery date,
- the scope of work,
- the major assumptions.
-

The process to build the project master schedule is the following:

- identification of the critical path.
- estimation of the duration of critical path activities (based on previous project, entities expertise, ratio.)
- challenging critical path activities' duration and balancing between others project requirements (Costs, Quality & Performance). It includes iterative negotiation activities between functions /entities and project stakeholders to define the right duration.
- definition of targets for the project team.

To build the master schedule of a project team at lower level, the process is similar but the first input is the master schedule of the upper level unit.

Figure 17 illustrates the target cascade process where major milestones are detailed by target milestones for lower level project teams.

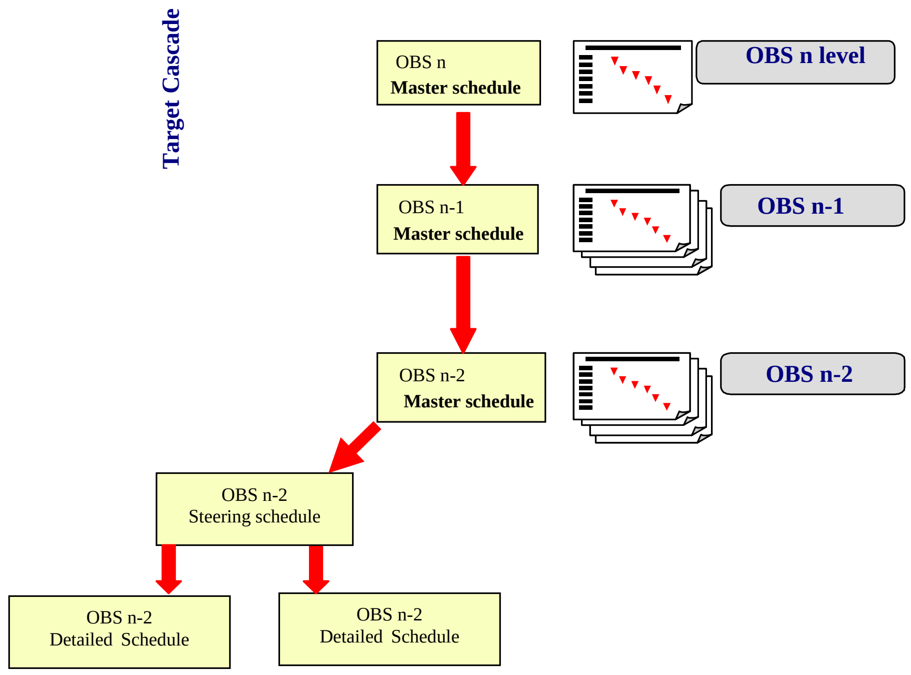


Figure 17: The target cascade process

On the one hand, the major inputs to build the steering schedule are the master schedule and the scope of work of the relevant organisation entity. The steering schedule consolidates the detailed activity and establishes the interdependencies between project organisation entities. It provides the baseline for the monitoring and the control of project status.

The target frame of steering schedule is given by the master schedule.

The duration up to deliverables completion depends on the allocated resources. It has to be challenged and negotiated with relevant entities, mainly with upper level managers in the project organisation. The major input to build detailed schedule is the steering schedule.

When an activity induces an important number of internal and product deliverables (for instance, the definition of some activity induces an important number of drawings), it has to be scheduled at the detailed schedule level. But relevant steering schedule shall contain milestones to measure the definition progress (25 % drawings done, 50% drawings done…etc).

Once the master, the steering and the detailed schedules have been defined for the lowest managerial level, interdependencies at this level need to be negotiated. Indeed, the fact of cascading independently the targets might cause discrepancies in the interdependencies between two entities belonging to different upper level managerial entity. This exercise allows the readjustment of lower level schedules and the possibility to begin the consolidation process.

2.5.2.2 The schedule consolidation

Once the lower level schedules have been readjusted and all the actors agree concerning their interdependencies, the scheduling built-up process can continue up-stream. This step is known as consolidation and includes three points. The definition of the steering schedules above the lowest level, the authorisation of the different schedules and the initial baseline establishment.

Based on the lowest level steering schedules, upper levels merge the most important tasks and milestones in order to define a new steering schedules.

The steering schedule includes all the necessary information to compare the current status and the reference. It is built to allow progress measurements on the activities performed and is therefore regularly updated.

Figure 18 shows the example of this step in the case of the A380 program.

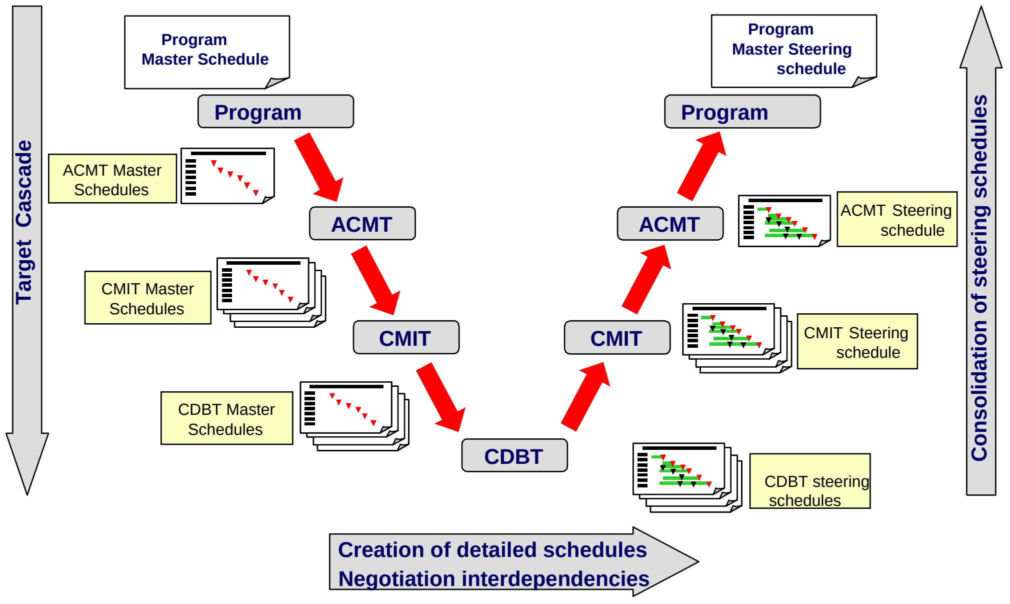


Figure 18: The schedule consolidation

Targets are cascaded down to the "working" level i.e. program to ACMT/OAD, ACMT to CMITs (OAD to domains), CMIT to CDBTs.

Once the working level (normally CDBTs) develop detailed plans and negotiate interdependencies, all involved plans are discussed with the level above (normally CMITs). Key milestones will be selected and consolidated into a CMIT Steering Plan together with the CMITs management milestones. This Steering Plan therefore becomes the common reference between the two levels and will be communicated to the ACMT management and others affected by the plan.

Once the steering schedules are created in each level, project teams among project organisation have to ensure that their steering schedule:

- is validated by the relevant other project team entities
- is taken into account & validated by the relevant functions /entities
- is authorised by the upper level of project organisation

Finally, a schedule authorisation phase enables the baseline to be established. This action consists in:

- recording all the dates included in master schedule as target date.
- recording all the dates included in steering schedule as target date.
- recording all the dates included detailed schedule as target date.

2.5.2.3 The schedule steering

Once a consolidation step is over, we consider that the project is in a running phase where basically the real progress can be compared to the baseline. This step is known as the schedule steering; the main points are the monitoring aspects and the baseline modification. These actions are realised during reviews dedicated to scheduling issues.

The schedule monitoring is based on:

- a project progress assessment
- a periodic comparison between achievements & the baseline
- an identification of the relevant corrective actions to build a recovery plan if any deviation or potential drift is identified

In normal conditions, the project leader organises regular schedules reviews with the project team task owners. These reviews are organised with a predefined frequency. Schedule review supports the project progress assessment and the recovery plan building.

The general process in the project progress assessment aims to check the completion of each milestone of the previous period. For a milestone, it is usually a binary progress definition: 100% achievement is obtained when the milestone is done and 0% in any other case. For a task progress, a percentage progress is associated for the ongoing task.

When a delay is detected by comparing to the baseline, the project leader can decide to organise a special meeting to monitor the progress more accurately. These reviews can be considered as “crisis” reviews if the project leader considers that the delay can affect the global progress of the aircraft development. These reviews are not realised in predefined time periods and they usually answers to project leaders daily decisions. During these extraordinary reviews, “recovery plans” can be defined. A recovery plan shall include new recovery dates with associated description of means and processes. Basically, if a milestone is delayed then a recovery date called a “forecast date” shall be proposed. Forecasted dates are considered as a strong commitment and shall be realistic and reliable.

In order to avoid numerous successive recovery plans, it is highly recommended to assess the progress of the next period (assessment of future progress) to implement the schedule management preventive mode. It enables:

- to define the potential drifts,
- to implement early recovery plans.

Finally, major change events with impact on the main project target dates, involves a change of the schedule baseline. A New baseline shall be established, defining new program targets and rebuilding the overall master schedule, steering schedules and relevant detailed schedules if needed.

Sometimes, the baseline might be modified. It is the case when the project has a long duration, i.e a new aircraft development. In that case, it is difficult to develop the steering schedule with the same granularity for all the project phases and especially for the last ones. Therefore a steering schedule could be completed as necessary respecting the following rules:

- no change on the master schedule,
- no change on baseline dates for the milestones included in the previous issue.

2.5.3 *Quality Gates (QG)*

2.5.3.1 Definition

Quality gates aim to avoid a “black tunnel” development and subdivide complex development projects into phases with a valuable significance.

The nature of a quality gate is quite different to that of a milestone. A milestone is essentially an absolute time reference, a quality gate always refers to the content-related maturity of a project, respectively of a deliverable.

From a scheduling point of view, quality gates subdivide the global development time horizon and create independent and consecutive time windows. Consequently, the scope of a scheduling problem is reduced considerably since the problem is divided into smaller ones. These phases are created following the criteria associated to the product development process steps. For instance if the whole system is estimated to 30-40 months, for a standard design team responsible of a subsystem, phases between the quality gates will be typically 20-30 weeks long.

Three fundamental characteristics define a quality gate. Firstly, a quality gate is a “rendez-vous” that synchronizes the processes of different subsystems. It obeys to a systemic vision of the project and it helps to highlight the subsystems that need an extra effort in order to respect the “drumbeat” of the global development. This issue is especially important in highly integrated complex processes, such as for the development of a civil aircraft.

Secondly, a quality gate is an essential issue for the customer/suppliers relationship. At a quality gate, performances agreed both by customers and suppliers at the beginning of a phase are assessed with regard to their compliance (quality and completeness). For the customer, a quality gate offers the possibility of protecting his own activity against malfunctions in the supplier’s activities. On the other hand, for the supplier, the quality gate offers the opportunity of safeguarding his own activities to match exactly the requirements of the customer’s activities.

Lastly, quality gates are closely linked to important decisions that may affect the definition of the project These decisions concerns either the project itself (in the case of a go/no go decision), or the product, for example when it states definitively the system architecture facing several alternatives. In most cases, it is then necessary to verify if tasks and requirements for the next phase are correctly defined taking into account the decisions validated at the quality gate. We detail now the general QG process.

2.5.3.2 The general QG Process

Quality gates are generically defined for all programs through specific documents. This generic definition needs to be instantiated to program specific processes; customer-supplier relationships and agreement criteria (see Figure 19).

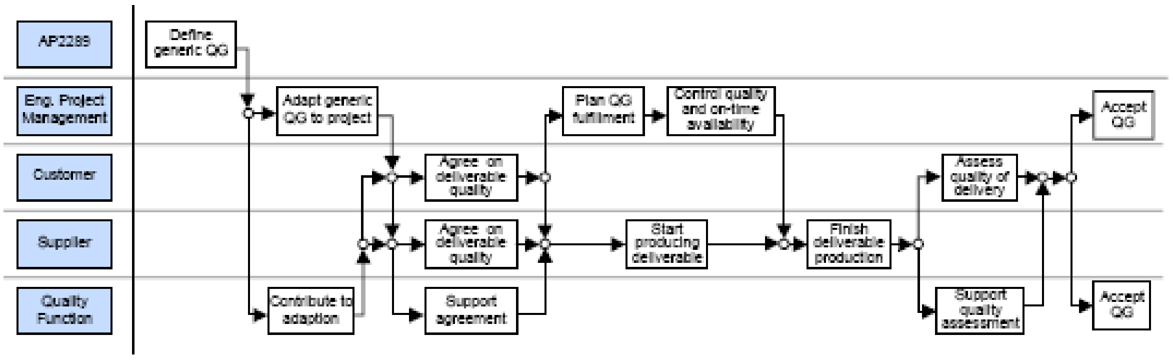
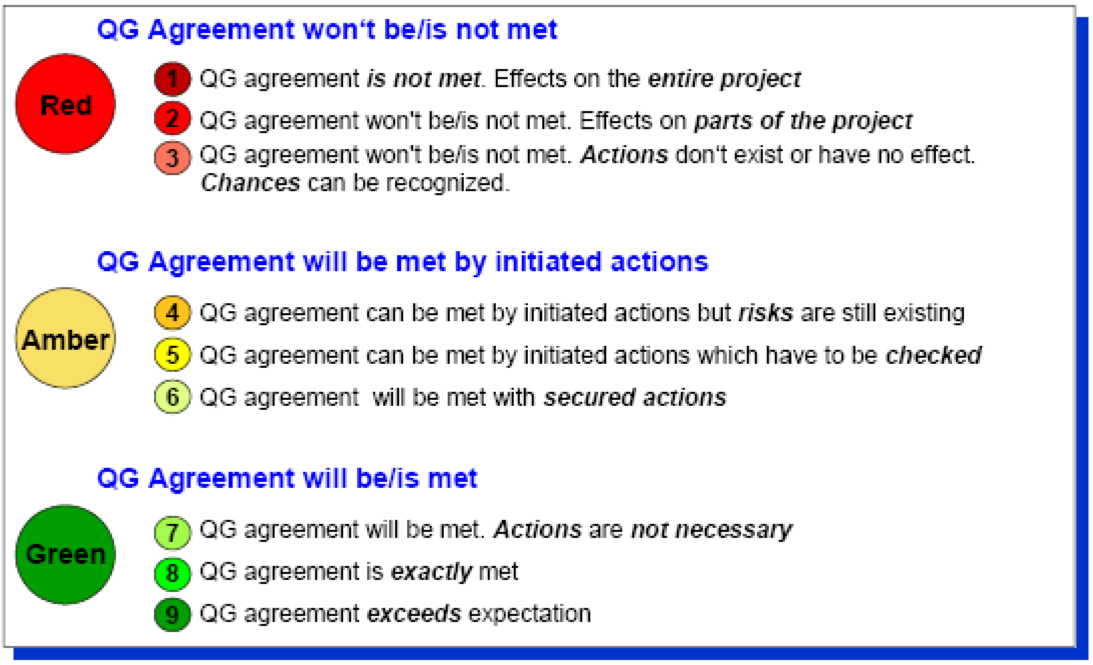

AP2289
Eng. Project Management
Customer
Supplier
Quality Function
Define generic QG
Adapt generic QG to project
Contribute to adaption
Agree on deliverable quality
Agree on deliverable quality
Support agreement
Plan QG fulfilment
Control quality and on-time availability
Start producing deliverable
Finish deliverable production
Assess quality of delivery
Support quality assessment
Accept QG
Accept QG

Figure 19: The Quality Gate process

2.5.3.3 General quality assessment

As shown with Figure 20: Assessment of the deliverable quality, the basic assessment of the deliverable quality is indicated by a colour (green, amber, red). This colour scale is subdivided in numbers (1…9).



QG Agreement won't be/is not met
Red
1 QG agreement *is not met*. Effects on the *entire project*
2 QG agreement won't be/is not met. Effects on *parts of the project*
3 QG agreement won't be/is not met. *Actions* don't exist or have no effect. *Chances* can be recognized.
QG Agreement will be met by initiated actions
Amber
4 QG agreement can be met by initiated actions but *risks* are still existing
5 QG agreement can be met by initiated actions which have to be *checked*
6 QG agreement will be met with *secured actions*
QG Agreement will be/is met
Green
7 QG agreement will be met. *Actions* are *not necessary*
8 QG agreement is *exactly* met
9 QG agreement *exceeds* expectation

Figure 20: Assessment of the deliverable quality

Generally, green status represents a complete fulfilment of the agreement; amber indicates that additional actions have to be initiated in order to still reach the quality target. And red colour stands for a complete non-fulfilment with corrective actions initiated.

These statements are based on facts captured from the field and discussed in the following section.

2.6 Selected approach to collect facts from the field

This paper is based on real industrial case studies provided by AIRBUS The capture was articulated around different kind of actions:

2.6.1 Airbus "Lessons-Learnt" (LL) activities

We have been involved in the company's internal LL process that aims to capitalise on design practices during the development of Airbus programs. These LL activities have mainly been related to the A380 program. We had access to information extracted from questionnaires addressed to operational units' managers as well as the results extracted from data bases LL team.

The process for LL activities is divided into four main phases: decision, organisation, collection/validation, and reuse.

In the first step, the decision factors where defined before launching LL actions. These factors can be related to:

- the situation description (foreseen problems, key actors availability, etc).
- budget.
- resources needed.
- time schedule link to the level of analysis required.
- risk assessment for specific topic.

Before starting with the collection of the data, LL actions where organised. This step includes:

- LL needs Identification.
- customers identification.
- experience providers identification.
- LL team definition
- definitions of actions to capture experience as interviews, workshop, data investigation, etc

Once these points defined, the collection of data began, having in mind the planning decided. In other words, actions to capture experience as interviews, workshop, data investigation where launched.

The validation of data collected does not need to wait for the end of the data collection; to be more efficient in time, the validation was organised in parallel of collection actions.

Nevertheless, a validation step is necessary before defining specific recommendations. Moreover, to ensure that solutions are not context specific, it was necessary to compare solutions with related experience.

Finally the last step deals with the reuse of the data collected during the exercise. First, the results where presented to the identified customer. Secondly, in order to share results as widely as possible, additional actions where launched to improve the exchange and to make the information accessible to customers that have not been identified or potential customers from future projects.

2.6.2 Transfer operations

We were involved in transfer operations activities (at both managerial and operational levels) that aim to share planning knowledge and experience between current and future projects. These transfer operations were not organised in the framework of “Lessons-Learnt” activities. The goal was not to collect data in order to share widely the results, but to organise specific meetings between two teams of different programs (where one of the programs is more advanced from a development point of view) and to exchange about the best practices of the more advanced program. Usually these meetings were held during the launching period of a new program. The actors of the new program present the solutions, methods and the tools that are considering for application and listens to the suggestions of the actors who have lived that experience recently.

We participated to meetings where project management methods and tools where debated. These meetings were an excellent way to identify add-hoc tools developed to specific needs for the project management.

2.6.3 Procedural documentation

We analysed the aircraft manufacturer internal procedures related to project management activities as well as planning elaboration guidelines.

The merging process of such different companies like Daimler-Chrysler Aerospace (Dasa), Aerospatiale Matra SA and CASA, has left different definition of procedures related to project management. These procedures include different approaches for planning design activities as well as different approaches of the methods that support the planning procedures.

Since the formation of the European Aeronautic, Defence and Space company (EADS) and later on the Airbus company, a great effort has been realised in order to harmonise the guidelines and procedures.

The first result of this harmonisation process linked to project management practices was the Airbus directive “AP1002, Aircraft Project Management” [AIRBUS '01b], which aimed “to provide the basic rules for Aircraft Project Management within the Airbus organisation”.

This directive gave the general rules to be applied in the Airbus Organisation in order to set up and conduct Project Management on any Aircraft development (new project or derivative).

The document accepted the numerous processes covered by Project Management and decided to focus on the following ones:

- project establishment
- project organisation
- project planning
- risk management
- resource management
- project monitoring & control
- information management
- configuration management
- management of supportability and support products/services
- project closure

For each topic, the directive described the objectives and the guidelines to be applied. Moreover it defined the milestone outputs from each aspect of project management activity as shown on Figure 21.

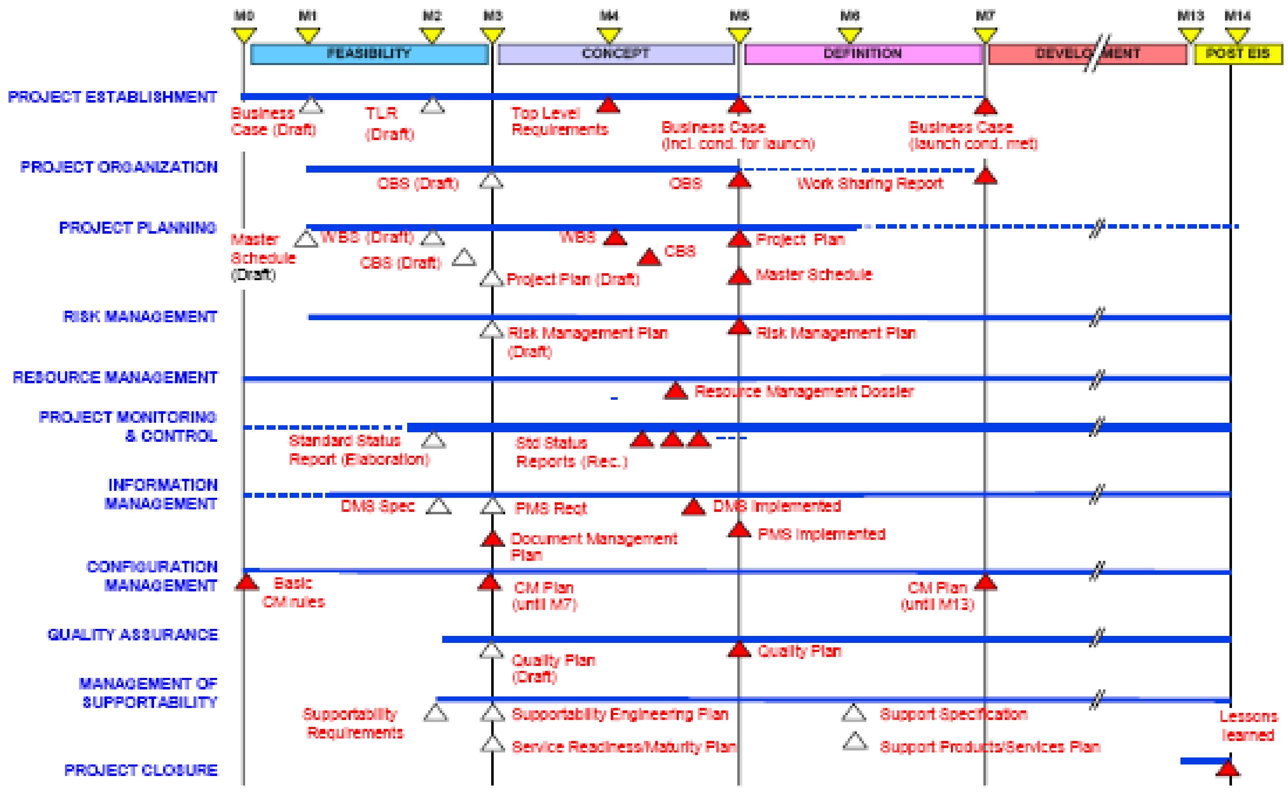


Figure 21: Project management activity's outputs

This directive was succeeded by other documents that defined more accurately the common project management practices to be applied in Airbus [AIRBUS '05a, AIRBUS '05b, AIRBUS '05c].

The documents analysis has been necessary in order to understand some of the practices identified in the field. Moreover, discussions with their authors were very fruitful. On one hand they had the difficulty to make trade offs concerning different practices derived from former managerial structures. On the other hand, they are in a position to collect reactions and difficulties to apply these directives and guidelines on each organisational structure.

2.6.4 Semi-structured interviews

Semi-structured Interviews with team managers and project managers have also been carried out. The specific issues where related to the different questions studied during this research project. Figure 22, shows the structure of the different actors involved in the A380 program teams.

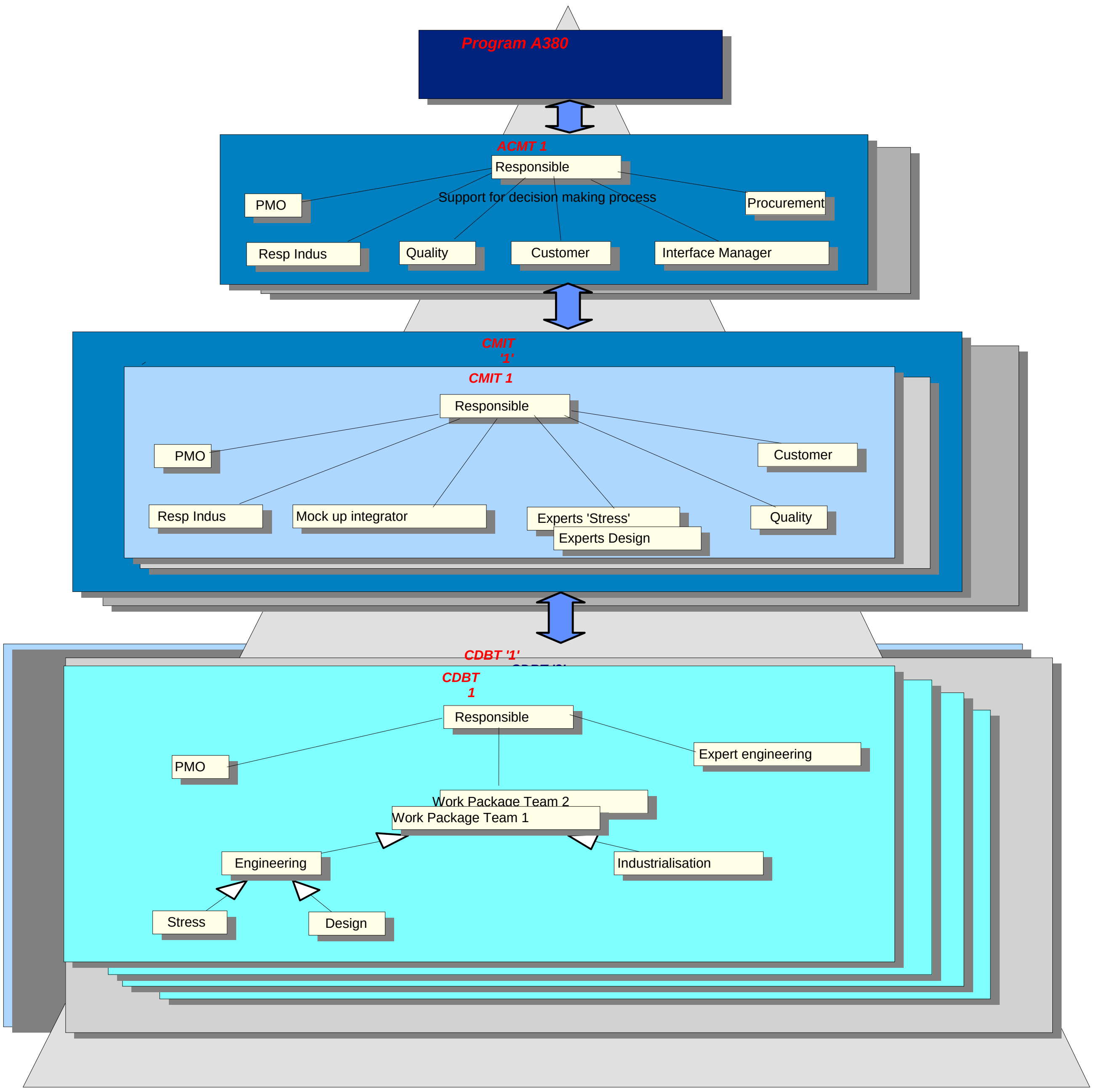


Figure 22: Structure of the actors involved in the A380 program teams

Table 2: Function description of the actors involved in the A380 program teams

Actors	Function
Responsible	The main role of the responsible of a managerial team is to take decisions taking into account the inputs coming from different support functions. Usually these decisions are trade offs concerning opposed views of two functions.

PMO (Project Management Officer)		He or she supports the responsible of the teams on aspect linked to cost and time issues. He or she is usually the schedule builder and he or she measures the team progress.
Industrial responsible		He or she supports the designer in order to take into account manufacturing and assembly aspects on the design of the product. Moreover he or she participates actively on the managerial level scheduling issues.
Quality responsible		He or she deals with the definition of internal processes as well as the risk management issues and requirements management.
Customer Focus		He or she supports the designer in order to take into account customers needs on the design of the product.
Interface Manager		He or she defines the responsibilities of the work to be done. It organises bilateral meetings between different teams and manages the interdependencies between teams.
Procurement		He or she supports the responsible of the team concerning the contract with subcontractors as well as concerning the materials buying aspects.
Mock up integrator		He or she is responsible of the virtual assembly of parts as well as the configuration management aspects.
Expert Engineering	Expert Design	He or she supports the design process with its experience.
	Expert Stress	He or she supports the design of the parts from a materials strength point of view. It is also responsible for the certification.

We had at least one interview with each of the actors at the beginning of the research project in order to identify scheduling practices and needs for improvement. Most of them were members of the Centre and Nose Fuselage ACMT of the A380. During the research project more interviews where held mainly with managers of CDBTs and PMOs at the three managerial levels. For specific issues related to interdependencies and risk management practices, Interface manager as well as Quality responsible where asked to explain their practices in one phase of the project. These interviews allowed defining accurately the interdependencies and risk management processes as well as the relationships that these processes have with scheduling practices.

Finally, after analysis of collected information, the first observations on current project management (PM) issues have been presented to program management and PM functions for validation. Some of them are presented in the section below with a specific focus on scheduling activities.

2.7 Difficulties and needs of a new aircraft development project scheduling

Based on information captured during interviews and LL, we propose to characterize some major issues related to design scheduling practices which can be considered as rooms for improvement for the development a new aircraft.

2.7.1 Uncertainty of information needed for scheduling

The majority of project scheduling methods assume that information to build schedules is available, stable and complete at the beginning of the project definition. However, facts show that design process is exposed to a significant level of uncertainty, particularly with design and development activities.

This uncertainty lies partly in the necessity to choose among several alternatives (imprecision) but also in the partial controllability of the events associated to the start and the end of the chosen activities. An imprecise variable in preliminary design is a variable which may potentially assume any value within a possible range because the designer does not know, a priori, the final value that will emerge from the design process. Stochastic uncertainty arises from a lack of exact knowledge of a variable due to some process the designer has no direct control or choice over [HERROELEN and LEUS '04].

Activities duration is a well-known case of the uncertainties that are linked to the design process. Uncertainties related to the duration can be both imprecision type uncertainty and stochastic uncertainty. Indeed, uncertainty depends on the innovation level of the activity we are scheduling.

For the activities with high degree of innovation and that are carried for the first time, the definition of the duration is a difficult task at the beginning of the project. Indeed, the imprecision among the different durations that can be chosen force the project leader to define its schedule with a high degree of imprecision.

On the opposite, an activity the project leader is familiar with can be defined in a more deterministic way. If the execution of this activity is realised in a conventional way, the duration of the activity that has been defined at the beginning will probably be respected. Nevertheless, events that have never appeared in former project can always arise, modifying the duration of the activity.

As the design process evolves, the imprecision of each design variable is reduced. Nevertheless, the uncertainty related to unforeseen events remains.

Other variables disturbed by uncertainty include resources definition. The amount of resources that need to be assigned to an activity can be uncertain. Moreover and when human resources are concerned, these uncertainties can be expanded to other variable related to human resources: repartition of skills, experience level, combinations between different resources, events that avoid the resource to be present etc.

Uncertainty is also related to the data that is needed by the actors in order to complete the activity. The date this data should be available as well as the level of maturity is often an uncertain decision.

2.7.2 Heterogeneity of the scheduling tools and supports

Planning tools are heterogeneous and schedules are designed with different methods. As a consequence, exchanges between teams but also with program management are affected. It points out the fact that planning and schedules are used for different purposes and based on different types of information. While some are used for communication inside a design team, others can be used as a tool to steer the design process.

In the lowest level of the organisational structure we observed that some schedules with a very short time window, are mainly utilised for communication purposes. These schedules are sometimes built with a drawing software that allows the representation of a solution but which does not provide any scheduling functionality.

In management teams between the lowest level of the organisational structure and the program level, schedules are efficient means to steer the progress of the subsystem development. Comparing baselines with real progress is an exercise frequently performed by the project manager. In order to efficiently measure the project progress, the schedules types and the method used to define these schedules should not permit any misinterpretation or any ambiguity. But in addition to the steering exercise, project managers usually need to manage the inputs and outputs exchanged with other teams. He will often check if the necessary inputs will be delivered at the agreed milestone. On the contrary he might not pay too much attention to the outputs release. This issue depends on the objectives that have been assigned to the project leader. Unfortunately, these objectives usually deals with the cost, quality and time related objectives linked to the subsystem, and the importance of the outputs for other teams is minimised. Therefore, when tradeoffs need to be done, the project leader will advantage the objectives linked to its subsystem rather than the outputs defined through the process of its internal activities. Depending on their relative importance schedules will be built differently. In some cases input and outputs will not be part of the schedule of internal activities at all, while in other cases these input and outputs will be defined as milestones (usually with predefined forms) between the tasks and milestones related to internal activities. While some tools allow the accurate definition of the milestones, others do not, and therefore, information about these inputs and outputs need to be managed externally.

Lastly, management teams between the lowest level of the organisational structure and the program level need to deal with the resources allocation problem. Even if most of the tools offer the possibility to manage tasks charged with resources, the reality is that the management between tasks definition and resources allocation is done with two different tools, which generates both an extra work (replication of data) and the risk of a loss of consistency due errors during data replication (particularly if done manually).

2.7.3 Processes synchronisation

The synchronisation shall be realised in two levels. Firstly, as explained in chapter 1.5.3, it is necessary that development of major subsystems follow a similar pace. This point is usually managed taking as reference the first assembly of the subsystems. Similarly, the synchronisation needs to be realised at higher levels. This is the case between aircraft and engine, which are developed by different companies but which processes are highly dependent.

Finally, the synchronisation of the process can also be seen from a multi-levels point of view. Indeed, even if the main subsystems' processes, do not respect exactly the developing phases defined at aircraft level, they need to respect some milestones constraints defined in order to guarantee the correct integration of each subsystem onto the aircraft. This issue is especially important in highly integrated complex processes, as for the development of a civil aircraft.

Figure 23 shows the main phases and milestones of an aircraft development, the engine development and a major subsystem development, in this case the landing gear.

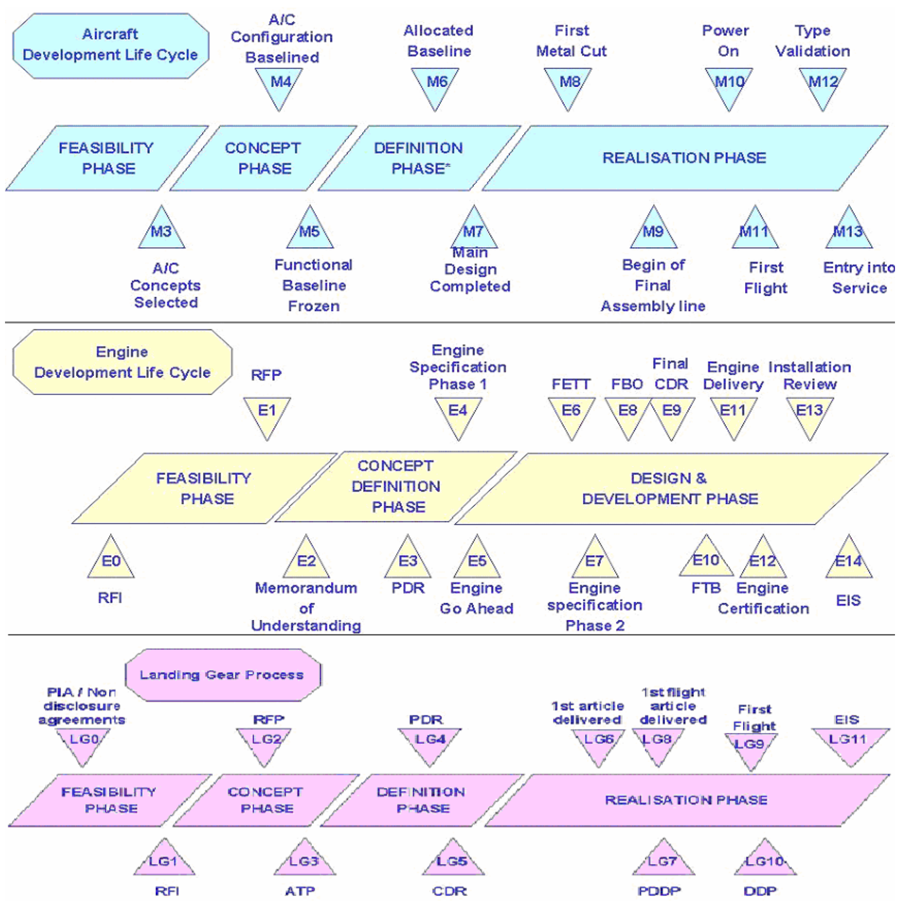


Figure 23: Development lifecycle of aircraft, engine and landing gear (from [ENHANCE '02])

2.7.4 Multilevel scheduling process

Schedules used in the design process need to be designed for and managed at different levels of the organisation. Links between different levels should be used to cascade project milestones to lower levels but also to identify the team's constraints and communicate them to upper levels. Both cascading and escalation processes can be time consuming and sources of errors, particularly if scheduling tools are different and not interoperable.

At project launch, the cascading process enables defining accurate schedules at lower managerial levels being sure that they respect the main program milestones. Depending on the type of schedules used at each level, the cascading process may vary. It is necessary also to pay attention when, due to the need of increasing details, a target milestone is converted into a task. In detailed schedules it is possible to keep main program milestones and to create links between them and accurate milestones in order to stress how internal activities might affect the global progress of the project.

Once the project is launched, the real progress of each lowest level team needs to escalate its own constraints and achievement in order to check constantly if its outputs are in line with project objectives. The escalation process usually aims to be an assistance for decision making processes at different management levels. In order to support this process, management levels have usually their own schedules built with their own targets but also with most significant milestones coming up from lower levels. Another procedure that allows building schedules in these levels is the aggregation procedure. Aggregation consists for example on making the sum of the durations of successive different tasks coming form lower levels to compute the duration of a composite task. In order to stress the most important tasks participating in the aggregation procedure, it is possible to realise weighted sums.

Lastly, key performance indicators (KPI) can be built from schedules in order to support the decision making process of the upper managerial level. These indicators allow the manager to focus on the teams being in late at a glance.

2.7.5 Schedule robustness

Static schedules are accurate and useful, but very sensitive to unforeseen events. Indeed, a design team might have to re-evaluate a new context and update the schedule each time an unforeseen event is detected. The update frequency can be high considering the dynamic characteristic of the design process. The baseline schedule might be modified frequently, loosing a reference framework and discrediting the goals fixed for the team members. This point is related to the uncertainty of the information we find in a schedule. If the uncertainty information is taken into account when defining a schedule, we make the schedule more robust. Robust schedules are also known as proactive schedules and can be defined as:

- likely to remain valid under a wide variety of disturbances [LEON, et al. '94].
- the violation of the assumptions upon which it is built are of no or little consequence.
- the ability to satisfy performance requirements predictably in an uncertain environment.

The utility of these approaches depends, to some extent, on whether the uncertainty in the environment can be qualified in some way. If so, this information can be used by proactive scheduling techniques.

2.7.6 Alternatives management

At the design stage, different alternatives have to be managed simultaneously through different planning scenarii. Schedules should support the evaluation of the different scenarii and be used within the decision-making process. Schedules are currently considered as objects to be updated as a consequence of decisions rather than objects enabling the preparation and consolidation of decisions. These scenarii are issued due to two factors. On one hand, from a design point of view, there might be different options for the subsystems definition. Until a decision is taken concerning the design solution, the design team might choose to study both solutions and from a scheduling point of view, this means that usually the manager builds two different schedules. The different design options could be related to the possibility of using different materials (i.e. Carbon fibre or aluminium parts, this is an example that usually appears in last aircrafts development), different architecture or linked to the manufacturing technology.

When a trade off is made between different design options, different criteria are used for the decision making process. Schedules defined for each scenario, support this process, making available the information needed for time criteria definition. In most of the cases, there are some common milestones defined in both schedules. These milestones allow comparing not only the completion date for each design option but also the key outputs that could influence the definition of other subsystems. On the other hand, the scenario could be issued form the same design option. In this case this is not a technological aspect that influence the development process but a managerial decision that might modify the order in which the tasks are performed or how the resources are allocated.

Some design teams concentrate the design efforts on the final phase of the development process. Even if they respect the final delivery of their work, their rarely respect the delivery of preliminary information to other design teams. Managing the resources allocation and the task performance order is a matter of respecting internal time and costs commitments, but also the contracts that link them with other design teams. In the next chapter, we will discuss more accurately this type of needs.

2.7.7 Dependencies management between design teams

Dependencies between design teams can be observed through deliverables exchanges, which are often subject to negotiation. Consequently, dependencies management often refers to interfaces management, deliverables management, contracts management or interdependencies management. The dependencies that are formalised through deliverables, build a network whose nodes are teams and edges are information flows. Such a network enables to understand the constraints/contracts established by a design team on another. Unfortunately this network is rarely made explicit.

Taking into account the dynamic nature of the design process, the issues of constraints and planning changes propagation becomes crucial. Each time a design team modifies its own schedule, especially after the occurrence of an unforeseen event, the information should be transferred to other teams it has some dependencies with.

In order to anticipate changes in dependencies, a close link between dependencies management methods and schedules should be established. Finally, one of the key observations considering the management of the multiple dependencies between design teams is that managers are usually accountable for internal commitments such as the delivery date of the final design or part, or the internal budget. Nevertheless they are less accountable for the contracts agreed with other design teams in the framework of the dependencies. The consequence of this issue is that, managers usually build the schedules stressing the date of the final delivery, or the internal budget, rather that the respect of the deliveries related to the contract with other design teams.

2.8 Definition of the research problem

In this chapter, we have highlighted some of the problems which be found by a project manager during the development of a new civil aircraft. To reduce the scope, we have selected the problems dealing with engineering activities performed before the official project launch (preparation phase) and in the running phase once the project is launched.

We strived to understand the limitations of current methods for schedules creation before the project launch. During this phase, resources need to be allocated to each team taking into a count the work to be done. Therefore, the first question that the team manager has to deal with is: “**Are the allocated resources sufficient for the completion of all the assigned activities?**”.

Moreover, at this stage uncertainties related to the activities definition but also to concurrent design alternatives maturation needs to be taken into account. In this work, we will try to answer to the following question: “**How can these alternatives and uncertainties be managed from a scheduling point of view?**”.

We also focused on project running phase following the project launch. We have investigated the dynamic aspects of a new aircraft development and how the schedules are impacted. Due to these dynamic aspects, team managers constantly face baselines modifications and they should be able to determine continuously whether the allocated resources and defined time slots for each activity are consistent according project constraints. “**Is it possible to define a rigorous framework enabling team managers to check consistency of its decisions with the project constraints?”.** These constraints encompass classical constraints such as deadlines and available resources but also emerging constraints due to the concurrent and distributed nature of product development. Therefore, the following research questions are stressed: “**How can interdependencies between different teams be managed efficiently?**” and **“how can team manager guarantee that deliverables will be released with the requested maturity?”**

But efficient project steering does not only deal with horizontal exchanges between teams being part of the same managerial level. In complex product development projects, vertical exchanges between different managerial levels shall be considered when dealing with resources allocation and activities synchronisation actions. Therefore, the project management framework we are looking for shall also answer to the following question: **“How schedules related information can be managed upstream and downstream in the project organisation structure, in order to offer the necessary information related to time criteria and resources allocation for each managerial level?”**

As we can see, is not only a fact of defining some guidelines for individual project managers involved in the project, it is also a fact of supporting interactions between these actors when the project does not progress as expected. Therefore, we will conclude with the following research question: **”Can collaboration between design teams and between different managerial levels be enhanced by offering an accurate reference for renegotiation of constraints?”.**

2.9 Conclusion

The introduction of concurrent engineering practices enables aircraft manufacturers to take advantage of collaboration between the different entities of the organisation. In practice, collaboration between these actors can be observed during successive exchanges of information supports (also known as formal or informal deliverables). The maturity of these supports evolves and the same support can be exchanged several times with different maturity levels. Furthermore, the activities that are defined in order to develop these supports are scheduled following a process that includes different types of schedules. Each schedule type answers different and specific needs.

New practices include quality gates management which subdivides complex development projects into phases focusing not on the activities control but on the fulfilment of the requirements associated to quality gates. Therefore, the project manager in charge of the fulfilment of these requirements has no restrictions to manage the activities between two quality gates leading to an autonomy that should be exploited in a consistent and rigorous way.

Nevertheless, these scheduling practices do not answer all needs that arise in order to steer design activities. Some of these needs, like uncertainties of design activities are inherent to every new product development. Others like multi-levels schedules management needs or difficulties to manage different design alternatives are due to the specific nature of the organisation we are looking at.

Lastly, one of the issues that is considered as a key element for a efficient product development and that has been identified as source of conflicts and scheduling disarrangement is the dependency management process between design teams.

Dealing with these difficulties and focusing mainly on engineering activities, we defined several research questions we will investigate in the next sections.

Chapter 3: Building an approach to solve the aircraft design scheduling problem

3.1 Introduction

In the previous chapter, we have analysed current project management practices and highlighted issues related to design activities scheduling for new aircraft development. It has also presented some lesson learnt from a major European aircraft manufacturer.

This chapter is focused on three aspects that will help us to tackle the research problem we have selected for investigations, and for each of them, we provide a state-of-the-art.

First, an overview of the different project scheduling management models used in large project developments is provided. Then, we discuss some approaches enabling the management of uncertainties at the design stage. Afterwards, we focus on tasks scheduling and resources allocation processes taking place at the tactical level in the aircraft development program organisation. Finally, we analyse the problem of managing dependencies between different design teams working in concurrent engineering, and the necessity to foster collaboration among team managers.

Our aim is to cover these three aspects of the design activities management problem through a decision support system (DSS). This DSS shall be supported by a project scheduling management model that allows efficient integration of the three aspects in the same framework.

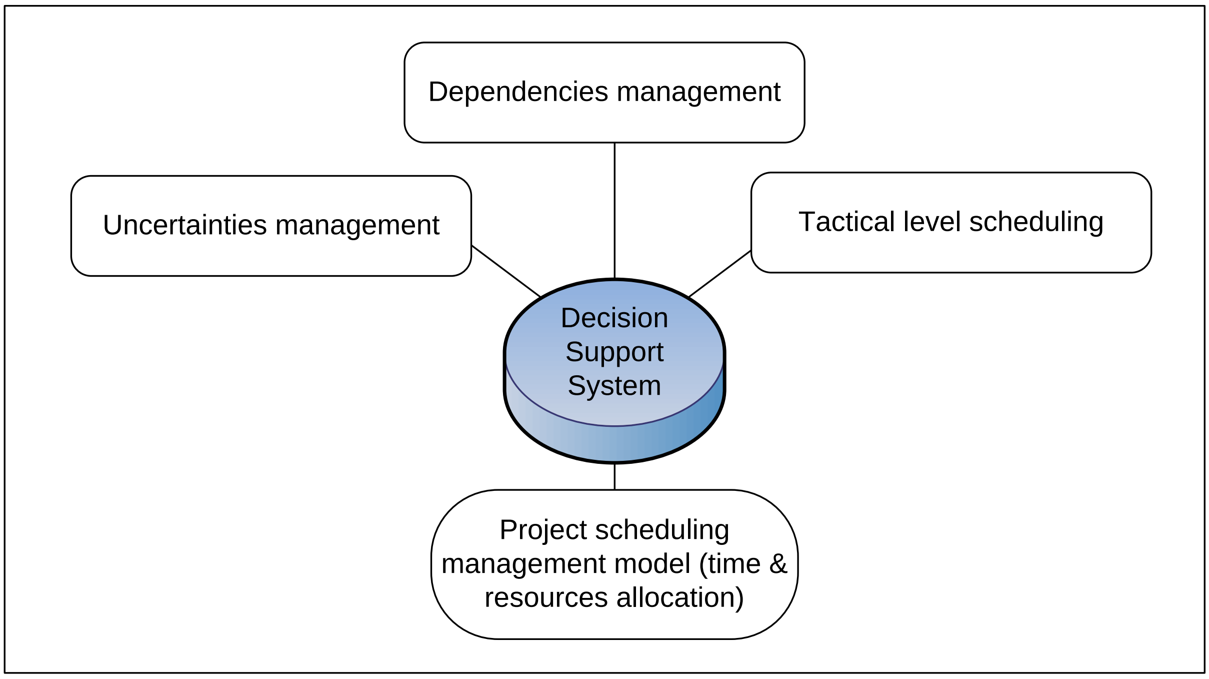

Dependencies management
Uncertainties management
Tactical level scheduling
Decision Support System
Project scheduling management model (time & resources allocation)

Figure 24 : Preliminary view of the decision Support System

3.2 Analytical models for project scheduling management

3.2.1 Classical project network models

Project network models, also known as graph based project models, represent the project as a directed graph that links activities and time constraints. More precisely, there exist two graph models that support the representation of a project: Activity on Arrow (AoA) graphs and the Activity on Node (AoN) graphs.

In AoA graphs arcs edges represent activity durations and nodes figure time events involved in the expression of a time constraint between two activities. Time events are start events, end events, or any intermediate event measurable during the realisation of an activity. In Activity on Node (AoN) graphs, nodes represent activities and arcs edges figure minimal duration time-constraints (most of time precedence constraints) that link two activities. Generalised precedence relations graphs are extensions of AoN graphs in which arcs edges represent any maximal or minimal duration constraint between two events (start or end).

Originally, both AoA and AoN models can express a partial ordering of activities and help to visualize the possible parallelism of some activities. Let us notice that dummy activities are often required to express rigorously any set of time constraints between activities in AoA graphs.

These models are well known since the 1950’s mainly because of their capacity to support the calculation of the minimal duration of a project and the set of the so-called critical tasks, through the determination of the critical paths on the graph [GIARD '97].

Two well known project scheduling methods are the Critical Path Method (CPM) and the Program Evaluation and Review Technique (PERT). The former was developed in the 1950s by the Dupont Company for managing plant maintenance projects. It enables to compute the earliest starting time and the latest starting time for each activity under the hypothesis that the project is completed at the earliest time. Critical activities are such that their earliest starting time and their latest starting time are identical). For others tasks, time margins are also computed.

The PERT model also enables to model some uncertainties on tasks durations in order to determine the most probable realisation of the project, whilst the CPM model can associate cost functions to activities in order to minimize the project global cost. In France, the use of project management models were initiated by the Metra Company through its method called MPM (Metra Potentials-based Method) in which linear “potentials” inequalities represent the time constraints that link variables associated to activities start and/or finish events [ROY '62].

Later more elaborated models were developed aiming to incorporate new capabilities to these models taking account of complementary characteristics related to the projects (GAN, GERT, Q-GERT, VERT, RAILH …) [GRUNDER '98].

Nevertheless, these models are more suitable for a time analysis of a stable project and are not supporting a decision help when deployed in a dynamic product development; indeed, project network models take the assumption that all the tasks shall be realised and only time related constraints are modelled. Concurrent engineering based product developments need to deal with loops related to rework and the beginning of a task is not merely linked to the completion to the precedent or precedents tasks.

3.2.2 Design structure matrix (DSM)

The DSM is an analytical method introduced by Steward [STEWARD '81] [EPPINGER, et al. '94]; it is also referred to as Dependency Structure Method, Problem Solving Matrix (PSM), incidence matrix, N-square matrix or Design Precedence Matrix. It is a matrix representation of a system which can be a product, a project or an organisation. In the first case, it is usually used by system engineers to model binary relationships (communications, physical dependencies) between subsystems of a complex product. In the case of the project modelling it has been used for studying tasks dependencies or precedence for decision points during a new product development. Lastly, from an organisation point of view, DSM is used for defining new organisational structures based on clusters that minimise interactions between teams [EPPINGER '01].

Table 3: Different types of DSM from (www.dsmweb.org)

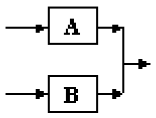
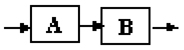
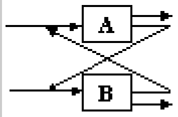
DSM Data Types	Representation	Application
Component-based	Multi-component relationships	System architecting, engineering and design
Team-based	Multi-team interface characteristics	Organizational design, interface management, team integration
Activity-based	Activity input/output relationships	Project scheduling, activity sequencing, cycle time reduction
Parameter-based	parameter decision points and necessary precedents	Low level activity sequencing and process construction

These four different types of data can be represented in a DSM and offer the manager a large variety of possibilities to deal with the complexity of a project. In our case, we have studied the activity-based DSM applications and how its representation can support our research questions. The main advantage of activity-based DSM is to provide a more compact visualization of the structure of an activity network, since graph models are become difficult to read when applied to complex projects with a high number of activities

In an activity-based DSM, the same list of tasks that compose a project is assigned to rows and columns. A mark located at the intersection of a row *i* and a column *j* of the DSM *i* states that task *i* requires information from task *j*. It is then easy to form for any task the set of tasks that require [deliver] information from [to] it. Marks below the diagonal represent forward information transfers to later tasks, while marks above the diagonal represent feedback information transferred upstream.

Relationship between system can be represented by the DSM in three ways: parallel (or concurrent), sequential (or dependent) and coupled (or interdependent).

Table 4: Activity-Based DSM

Three Configurations that Characterize a System			
Relationship	Parallel	Sequential	Coupled
Graph Representation			
DSM Representation	$\begin{array}{c\|c\|c} & A & B \\ \hline A & & \\ \hline B & & \end{array}$	$\begin{array}{c\|c\|c} & A & B \\ \hline A & & \\ \hline B & X & \end{array}$	$\begin{array}{c\|c\|c} & A & B \\ \hline A & & X \\ \hline B & X & \end{array}$

DSM is useful at a tactical level, at the beginning of the design process. It is a static model that helps design managers to highlight and understand interfaces between activities in order to structure and group them into clusters. Nevertheless, an activity-based DSM in which 80% of the cells are marked denotes a highly interdependent network of tasks within project. In this case, DSM would not be very useful for representing key interdependencies and activity clusters as they are masked by the high number of information represented.

Even if DSM can point out iterations, it is not a tool for a dynamic steering of the activities and does not support the resources allocation problem.

3.2.3 System dynamics (SD)

Interactions between design teams can also be modelled using System Dynamics (SD) models. System Dynamics becomes interesting when numerous feedbacks characterise a project.

The System Dynamics (SD) is a modelling method introduced by Jay Forrester of the Sloan School of Management at Massachusetts Institute of Technology [FORRESTER '61]. This modelling method has been used in different situations. Concerning project management applications, the first complex project where SD was utilised was *Ingalls* shipbuilding in the late 1970s [COOPER '80]. Since then, SD has been used in many complex projects in order to understand the reasons of schedules and cost overruns and also as a policy making tool [LYNEIS, et al. '01].

System Dynamics offers the possibility to model non-linear relationships. In project management applications, this can help to model relationships where cause and effects are not simple. For example, increasing the workweek for designers in 10% can have a positive consequence of the augmentation of drawings production by 10%. Nevertheless, if the workweek is still increased, the drawings production will not follow in the same rate.

Concerning task completion models, complex models are usually based on few main variables showed in figure X:

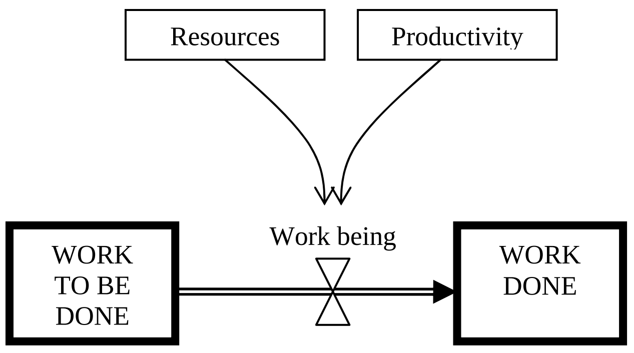


Figure 25: Basic variables on a SD model for activities management

System Dynamics will uses characteristics and availability of the resources in order to measure the completion rate of the work. Therefore, the manager will not only calculate the completion date of the task but also how the progress was realised. Complex System Dynamics models include several loops linking resources to quantitative aspects but also to qualitative managerial aspects.

While System Dynamics can be interesting to model any constant process with numerous similar tasks, it is not a suitable tool for representing a project with many tasks of different nature. Moreover, SD does not allow the explicit representation of tasks or their interdependencies. Because of these characteristics, SD is considered as a complementary project management tool by most of project management experts [PMI '00].

Some authors have proposed to merge SD with other methods in order to counterbalance its drawbacks [bulbul]. Nevertheless, no formal hybrid or combination technique has emerged. The main idea behind these proposals is to define a repository for managing a project where SD models will use inputs from other methods taking, for instance, project progress into account.

3.3 Uncertainties management in design activities

3.3.1 Different types of uncertainties and some approach to control them

Uncertainty is a term that can be related to the predictions of the future, the measures we are making in reality or simply to the unknown past and future events. Therefore the utilisation of the uncertainty varies in different fields as engineering, economics, philosophy, quantum mechanics, etc.

Meyer [MEYER, et al. '02] has studied different kinds of schedules and management styles according to innovation level of the development project. Taking into account different types of industries, he made a classification of four types of uncertainty: variation, foreseen uncertainty, unforeseen uncertainty and chaos. For each of them he defined the most suitable tools and methods.

Unforeseen uncertainties are related to risks and opportunities that can not be identified during the project planning. It can be issued from the non planed interactions of subsystems or events. Even if most of the cases these types of uncertainties can be catastrophic for the projects, there are examples of fruitful projects which were based on opportunities not identified during the project planning.

Moreover, projects with chaos type uncertainties are described in [MEYER, et al. '02] as having the same results as unforeseen uncertainties but, these types of projects were not launched with reasonably stable assumptions and goals. Our research work seems more related to the first two types of uncertainties: Variation and foreseen uncertainty.

Variation uncertainties are related to imprecision in selecting a value. The uncertainty of a variable is stated by giving a range of values which are likely to include the true value or the final value. Stochastic processes are also related to this type of uncertainty. Indeed, these imprecise values are often treated as random variable and may be grouped into two categories according to the method used to estimate their numerical values. On the one hand the statistical methods and on the other hand the variables evaluated by assigning a probability distribution or other means. Related to scheduling processes, these theories have been applied mostly to deal with unknown durations of activities. In next chapter we will introduce some of the method used to build schedules with uncertainty in the duration of the activities.

Then, the foreseen uncertainties are linked to identifiable and understood events that the team cannot be sure will occur. The calculation or estimation of the probability that the event will occur is linked to the stochastic term, from the Greek “stochos”, meaning “guess”. Therefore a stochastic process is opposed to a deterministic process, meaning that defining a state of the process does not determine the next state.

The probability theory in mathematics or different methods in artificial intelligence (neural networks, etc) have dealt with stochastic processes.

The calculation or estimation of the probability of the risk, as well as the impact perception of the risk is an essential factor for decisions making, and product development process is not an exception.

3.3.2 State of the Art on uncertainties management approaches

In order to deal with uncertainties inherent to a development project, different methods have been developed. In this chapter, we classify these methods in to three categories: proactive scheduling, reactive scheduling and risks management.

3.3.2.1 Proactive schedules

Proactive schedules are based on statistical knowledge to deal with uncertainties and they are built to be valid even if a disturbance occurs. These type of schedules are also named robust schedules and they often look for a balance between schedule stability and makespan optimisation [VAN DE VONDER, et al. '05, VAN DE VONDER, et al. '06].

Many techniques for proactive scheduling are based on extra time allocation to the activity duration in order to counterweight possible unforeseen lateness events that will make the schedule unusable; this is the case for redundancy-based techniques [HERROELEN and LEUS '04].

In production management temporal protection techniques are used to take into account the possibility of a machine breakdown. A temporally protected activity is composed of two types of intervals for the same activity. Both have the same end time and are differentiated by a slack between the start times [CHIANG and FOX '90, GAO '95].

Extra time is also one of the key points of the Critical Chain Project Management proposed by Eliyahu M. Goldratt in 1997 in his book “Critical Chain” [GOLDRATT '97]. The Critical Chain Project Management is based on the Theory of constraints (TOC) which is a management technique focusing the constraints that prevents the organisational system from achieving a higher performance relative to its goal. It deals with internal constraints mainly linked to resources and to external constraints linked to market.

On the contrary of the Critical Path methods, Critical Chain Project Management focus on the insertion of extra times called buffers and it does not look for the minimisation of the makespan but rather for a solution that respect the defined constraints.

Examples of project development time reduction in companies that have applied Critical Chain Project Management method can be found in [LEACH '99] and [CASEY '05]. The later made an accurate study of how Critical Chain Project Management could be implemented in a new aircraft development.

The use of project buffers as well as resources buffers aims to protect the key milestones but it also provides an interesting technique to distinguish progress critical activities or project constraint from progress on non critical ones. This separation enables managers to have a more realistic progress status of the project.

Critical Chain Project Management emphasises on-time project goals accomplishment while traditional project management methods focus on on-time accomplishment of individual tasks within the project [ROGERS '03]. Nevertheless, the main pitfall of the Critical Chain Project Management method is the difficulties to mathematically define the buffers in order to pertinently model the uncertainties of the schedule [HERROELEN and LEUS '01, HERROELEN, et al. '02].

Other techniques include probabilistic or stochastic approaches which imore more a diagnostic tool rather that a solution oriented schedule. Indeed, it does not produce robust schedules, but it enables realising scheduling simulations. If the user is able to measure the probabilities, it may build deterministic schedules. Moreover, probabilistic techniques can be used in parallel to other techniques. They can allow, for example, to stress the activities for whom the definition of an extra time is suitable.

Some authors have developed probabilistic approaches based on a DSM model. These models incorporate stochastic elements focusing on uncertainties in tasks duration and probability of design process change [CARRASCOSA, et al. '98].

Other authors have emphasized not only time aspects but also the resources allocation problems under uncertainty [NOZIK, et al. '01, TURNQUIST and NOZIK '03].

Most of these techniques are issued from the classical PERT model. Indeed, since the PERT technique was first used for Polaris project in 1958, probabilistic methods have evolved to become more and more complex and focusing on different scheduling aspects.

The PERT is able to incorporate uncertainty by making it possible to schedule a project not knowing precisely the details and durations of all the activities [MODER, et al. '83].

Different authors have evolve the basic PERT to include other functions, like [MALCOLM, et al. '59] and Dimitri Golenko-Ginzburg [GOLENKO-GINZBURG '88] who proposes some modifications to the PERT model under assumptions aiming a more realistic approach for project scheduling.

Later, this function evolved and became a continuous probability function defined as the Beta distribution. The Beta distribution can be used to model events which are constrained to take place within an interval defined by a minimum and maximum value. The continuous Beta distribution has been used for stochastic PERT applications [BACELLI '93, RAMAT '97].

Another evolution of the basic PERT model is the PERT Problem with Alternatives (PPA). In this model two types of nodes are included in order to deal activities that can be realised in parallel or a choice to be realised between two or more activities. Even if this method allows the manager define different scenarios, the number of scenarios that can be built with these models are limited because the user has to model each modification as a new path of the PERT. PERT Problem with Alternatives models can therefore be considered as a contingent scheduling method [CHAUVET, et al. '98]. Contingent schedules also named multiple schedules techniques. Contingency is related to the fact that some variables (i.e. duration) can not be decided since they are provided by the external world [VIDAL and FARGIER '97].

Contingent scheduling techniques do not aim creating only one schedule that shall be robust but they deal with the uncertainty creating multiple schedules or parts of them. When unforeseen events happen, the user changes the schedule and chooses the one that better fits on the real progress of the project. This set of schedules can be generated using different type of models. The Bayesian Network is one of them. These types of networks are probabilistic graphs where each node corresponding to one variable, and each variable corresponding to the individual rule by which a schedule will be constructed step by step [LI and AICKELIN '03].

In the case of the GLORIA method [NAIM, et al. '04], developed in the frame of a R&D project of EDF, the kernel model is created using Bayesian Networks, which enables the decision makers to measure the “domino” effect that an event could have on the project. In this project, the Bayesian Networks do not give any accurate information concerning time and cost aspects and is mainly used to create different scenarios and to measure how each scenario respects the project’s objectives. Therefore, we can considerate, that the GLORIA method is more linked to Risk analysis methods.

These scenarios are created using the Bayesian Network model and following a Risk Analysis method. Moreover, the Bayesian Network does not only define the likelihood of the risk defined for this project but it also supports the definition of the impact value measuring the outcomes of the network.

Another model for contingent scheduling is the Markov Decision Process, which is a discrete time stochastic control process characterized by a set of states. These states build a mathematical framework for modelling decision-making as in each state there are several actions from which the decision maker must choose. Markov Decision Process was developed in the 1950’s and is widely used nowadays in robotics or manufacturing scheduling problems.

3.3.2.2 Reactive scheduling

On the contrary, reactive scheduling implies the redefinition of the schedule if an unforeseen event occurs. In most of the cases, even if a proactive schedule has been chosen to steer the project, some events will force a revision of the baseline schedule. In this case, there is a need to modify the former schedule and add the changes introduced by the unforeseen event. Some authors have used the frequency of rescheduling as an indicator for measuring the performance of the scheduling process [CHURCH and UZSOY '92, VIEIRA and HERRMANN '03].

Smith proposes a reactive scheduling system named OPIS designed to incrementally revise schedules in response to changes to the defined constraints [SMITH '95].

Just-in –case scheduling can also be considered as reactive scheduling. It is a technique to generate schedules in a field with uncertain duration activities.

Drummond et al. have used this technique to the telescope observation scheduling, where there is only one resource and the observation activity has a time window determined by the possible observations periods [DRUMMOND, et al. '94].

Finally, different authors have worked on algorithms aiming at effective rescheduling. These practices are part of the reactive scheduling and the algorithms need to find a good balance between the quality of the new proposed schedule and the rapidity of the calculation.

The algorithms for the insertion of new activities in schedules that have already been built are part of the algorithms for reactive scheduling [ARTIGUES and ROUBELLAT '00, ARTIGUES and ROUBELLAT '02, DRUMMOND, et al. '94].

3.3.2.3 Risk management

Risk management tools are closely associated to classical project management tools. Risk management is the process of measuring risk and building plans to manage it. These plans or approaches include trying to avoid the risks or reducing the negative effect of a risk.

The notion of risk includes the source of the potential harm and the consequences of that event. Most of the definitions stress the negative aspect of the risk, linking it to dangerous chances or chance to loose. Nevertheless, risk notion can be understood as a hazard that can offer positive opportunities to the project or the company.

Some methods try to generalise risk management practices at different managerial levels, see [BENABEN, et al. '04], considering risks management practices as a core practices in the day to day project steering.

Risk as calculated from its risks of appearance and the gravity or impact of its consequences [AVEN '03, WINKLER '96]. Therefore, risk can be defined as:

Risk= Probability x Impact

The risk analysis process includes the following phases: Risk identification, risk assessment, risk management, risk control and risk lessons learnt.

Risk identification involves identifying sources of potential problems. The earlier these sources are identified lower are the probabilities of modification occurring in the project progress and the better will be prepared the actions to handle that risk. In order to identify an important number of risks, different actors of the product development process should be involved in the identification phase.

Risk assessment includes evaluating the likelihood of occurrence of potential problems and their consequences if they occur. The evaluation is then usually performed following the formula described previously. The goal is to define a risk value in order to compare and assign treatment priorities on identified risks. This classification can for example stress the most important risks in a first level and the less important in a second level. A third level is usually defined for the risks that will no be treated in this process.

Another practice is to classify all the risks following the same criteria and to focus only in the 20% of the most important risks. Following the Pareto rule, it can be considered that managing these 20% of risks means to manage 80% of the risk inherent to the project.

Risk management phase selects and implements the plans or actions that are required to ensure that those risks are controlled. This includes the actions to decrease the impact of the risks but also to reduce the likelihood of occurrence of problems.

Risk control deals with the steering process of the actions defined in the former phase. Actions can be classified between those which have been performed and those which have not still been performed. After implementation of a corrective action, the original risk needs to be re-evaluated in order to define the success of the action implementation.

Risk lessons learnt phase includes actually the different phases if the risks have been managed using a predefined framework and if they are ready to be saved for future use. Information related to risk identification can be a key element to further identification phases. Moreover, the efficiency of the actions that have been performed will help next project define the most efficient actions for each type of risk.

Several industrial companies have developed their own risk management methods taking into account their specificities [BEDILLION and ORR '99, ELKINGTON and SMALLMAN '02]. In the framework of Airbus new product development, the document AM-2457 [AIRBUS '05d] aims to be a guide for managing risks within projects. Nevertheless, it is a generic document that introduce the risk analysis basic concepts but it does not detail the relationships between design activities scheduling and risks management.

3.4 Tactical level scheduling

The second type of functions we are focusing in this project are the Tactical level scheduling and resources allocation.

First, we briefly position the tactical level in a new aircraft developing project. Then, we describe the constraints that we need to respect at this level and finally we provide a state of the art on the different methods dealing with scheduling problems taking into account these constraints.

3.4.1 Tactical level Positioning

In complex product development project three levels of management activities can be defined: strategic level, tactical level and operational level.

Strategic level mainly deals with long term decisions related to the enterprise. In the Airbus case, managerial decisions dealing with the project launch or the type of project that should be developed are decided at strategic level. It also includes long term investments and subcontracting policy. Strategic level problems can include facility location and workforce planning, which are usually solved using Linear Programming techniques [EPPEN, et al. '89, ESCUDERO, et al. '93].

Opposed to the strategic level, we find the operational level which is related to the activities that have direct added value on product. Day to day management as well as conflicts management are activities performed at this level. From a scheduling point of view, detailed schedules used at this level aim to manage accurate progress of each design principle, drawing or part. Managers usually coordinate their groups using action based schedules which are redefined on a weekly basis.

Tactical level management activities include the different managerial levels related to an aircraft program. On the top level the program managerial level will have the responsibility to develop and integrate aircraft major systems and sections under time, cost and quality constraints.

Global product architecture as well as resources allocation problems will be major tactical level manager decision making variables. Therefore, each level shall respect key milestones and the budget defined at higher managerial levels, making sure that quality and security aspects of the subsystems they are responsibly for will not be affected. In order to support the decision making process, the manager needs accurate information coming form the levels below. This includes activity progress, resources utilisation, identified risks, mayor technical issues affecting the on going process, dependencies management with other teams, etc. Therefore, vertical information flows becomes a key issue for correct steering of the subsystem development.

Some =models have defined a decision-making system that deal with the three levels. This is the case of the GRAI-R&D model [GIRARD and DOUMEINGTS '04], which is an extension of the GRAI model developed originally for manufacturing systems. This model does not only deal with three levels but it supports decision making process into two fields: the object field and the action field. While the former allows the transformation of product requirements into the product definition, the later considers the availability of project information and the necessary resources to perform the activities defined in the project plan.

Process and methods to transform product requirements into the product definition has been investigated by the Systems Engineering community [EISNER '02, FAULCONBRIDGE and RYAN '03, MEINADIER '02]. In our case, we focus mainly on the action level, while taking into account the relationships between object and action field [LIZARRALDE, et al. '06b, LIZARRALDE, et al. '07a]. Moreover, we will position our work at the tactical level which mainly deals with the definition of synchronisation milestones and the resources allocation problem. For these reasons, our approach can fit easily in a framework like the GRAI-R&D model.

3.4.2 Constraints definition for tactical level scheduling

Lets place in managerial level called “X managerial level” of the OBS (Organisation Breakdown Structure) of a new aircraft development project. We have identified five type of constraints related to this managerial level as described by Figure 26:

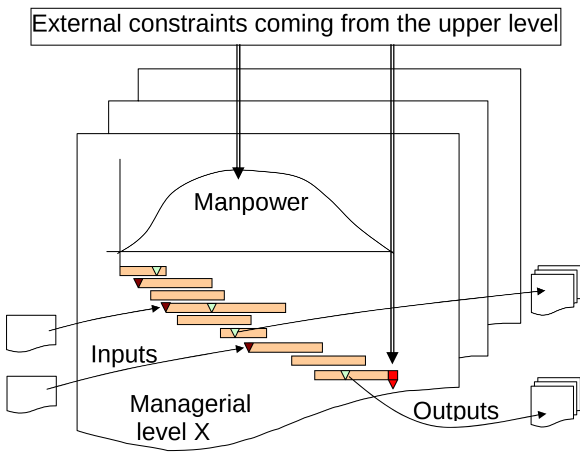


Figure 26: Constraints considered at tactical level

On the one hand, we have defined two internal constraints linked to the activity definition and to the precedence relation between activities. On the other hand, two types of external constraints are taken into account. First, we focus on decisions flows, which are related to constraints coming from the upper level of the organisation. Secondly, we consider information flows at the same level of the organisation (i.e. dependencies between teams at the same organisational level). Technical data exchanged between design teams is the main component of this information flow. To define schedules, we take into account time constraints related to the delivery of these data.

The upper level of the OBS defines major project milestones for project completion at a target date and allocates manpower until this target date, based on experience gained on former projects. The available resources will be the fifth constraint. Major project milestones are then cascaded to lower OBS levels defining target milestones for each team. Global manpower is also cascaded to lower OBS level defining the part of available manpower reserved for each team.

These two kinds of constraints (milestones and allocated manpower) will be the external constraints related to the interface with the upper level and we make the assumption that the manpower is dedicated to one specific project and can not be assigned to multiple projects.

In the literature constraints are classified into temporal constraints and resources constraints. We will add to this classification the constraints linked to the activities which can be considered as part of the definition on the scheduling problem or a constraint that eventually could be relaxed.

3.4.2.1 Constraints linked to the activity

Activities are the basic entities in project scheduling problem. They can be defined using durations or energies amount or work quantities.

In most of scheduling cases, an activity is linked to a resource and the duration is fixed. If the activity is defined by an energy amount then the duration is not fixed and it will depend on the number of resources allocated to the activity. Nevertheless, usually a limit is defined to avoid non realistic definition of activities in this case. Therefore a minimum duration or a maximum number of allowable resources can also be defined for an activity. In the case of activities defined by an amount of energy, we can distinguish the activities that do not allow a resources number modification once the activity is launched and the activities that allow to use different resources quantities during the activity progress. The former can be defined as rectangular activities, vertical distance being the resource quantity and the horizontal distance of the rectangle being the duration. The latter are defined as elastic or fully elastic activities. Other types of activities include the partially elastic activities which accept the allocation of different amount of resources during activity performance under selected condition [BAPTISTE, et al. '99].

Schedules can also be classified taking into account if activities are preemtive or not. In non-preemptive scheduling, activities cannot be interrupted. On the contrary, in preemptive scheduling, activities can be interrupted at any time.

3.4.2.2 Temporal constraints

Temporal constraints include precedence between internal activities, constraints related to the interdependencies with other teams and milestones fixed by the upper managerial level.

Precedence between internal activities arises when technically two activities can not be performed completely in parallel. Furthermore, there can be a precedence relation between activities from different teams. This constraint is known as organisational constraints and beyond the fact of a technical limit, this constraint can be due to geographical separation of subsystems. In our case we will link these constraints to the interdependencies between teams.

These two types of constraints will not be treated in this chapter but in the next one dedicated to the dependencies.

The third temporal constraints are milestones fixed by the upper managerial level. This milestones are due dates. In production scheduling due dates are more frequent than in project scheduling. Indeed, from a scheduling creation point of view due dates are very strong constraints since activities related to this constraints can be placed in the schedule and have already an upper bond. Due date constraints can also be defined as economical constraint. Indeed, usually penalties are defined if these dates are not respected and as consequence some scheduling objectives try to minimise the number of non respect due dates or look for a minimisation of the average violation of the due dates.

3.4.2.3 Resources constraints

A first classification between resources types can be done between renewable and non-renewable resources [WEGLARZ '81].

When resources used by an activity are renewable, they will be released and therefore they will be again available for the next activity as soon as the former is finished. Considering a renewable resource capacity of nine units, if four units are used from t_0 to t_1 time period for an activity A, during this period only five units will be available for the other activities. Nevertheless at t_1+1 period, the full capacity (nine units) will be again available for the different resources performing at this period. In production scheduling machines are usually considered as renewable resources. In project scheduling problems manpower is an example of renewable resource.

On the contrary of the renewable resources, non-renewable resources can not be reused and therefore are consumed by processing an activity. A classical example of a non-renewable resource is money.

Another classification regarding resources is a widely used classification that distinguishes schedules into two different problems: On the one hand, disjunctive scheduling problems, where a resource cannot be used for more than one task during a given period. On the other hand, we have cumulative scheduling problems, where a resource can be used for more than one task during a period if the maximum available resource quantity is respected. For cumulative scheduling problems with rectangular tasks, the “tallest” rectangle is therefore defined based on the available resources.

3.4.3 State of the Art for tactical level scheduling and resources allocation

Taking into account the different constraints described in the former chapter, the manager can perform two essential functions of the project management process: Scheduling activities and resources allocation. The goal is to realise all the defined activities, respecting the constraints or most of them. In order to do that, it will be necessary to define a schedule where activities will be placed, with a start date and end date, and where the resources utilisation is defined. Scheduling techniques vary depending on the objectives and the list of constraints to take into account.

Concerning the objectives, the most used objective is the minimisation of the makespan or to minimise the end date of the last activity. Other objectives can be related to cost aspects, minimising the resource utilisation or the overall project costs. If due dates are defined during the project, an objective can be defined in order to minimise the number of delayed due dates.

When the goal is the minimisation of the makespan, Activity On Node project networks can be very useful and visible. At the beginning the resources constraints where not taken into account for the scheduling process. The network techniques that we have already explain where widely used for illustrating the activities realisation process. Using techniques like the Critical Path Method, managers where able to calculate the starting and ending times for each activity, determine which activities where critical to the completion of a project (called the critical path), and reveal those activities with "float time" (less critical). In order to do this calculation, it was only necessary the list of the activities with their duration and the dependencies constraints between them.

Nevertheless, the fact of not taking into account the resource constraints limit these techniques to some specific phases and there are not complete technique to deal with scheduling and resources allocation problems of a new aircraft development project tactical level. If we take into account the resource constraints, we deal with problems known as Resource Constrained Project Scheduling Problems (RCPSP) that will be described in the next chapter.

3.4.3.1 Resource Constrained Project Scheduling Problems (RCPSP)

RCPSP are NP-Hard problems, this means that there exist no algorithm enabling to optimally solve the problem in an amount of time bounded by a polynomial function of the size of the data.

3.4.3.1.1 Resource Constrained Project Scheduling Problems classification

Resource Constrained Project Scheduling Problems can be classified following different criteria. Usually we consider a classical RCPSP and additional assumptions are added if the classical RCPSP is not sufficient to model the real problem.

Classical RCPSP includes an acyclic Activity On Node project network with non-preemptive activities and scarce renewable resources. These data is considered to be integer non-negative values. A solution is given by a schedule which assigns start times to all activities respecting the different constraints. The objective is to minimise the makespan.

A variant to this problem is the multimode RCPSP. Opposed to the classical RCPSP, which is a single-mode problem, in multimode RCPSP, non-renewable resources are considered. On the contrary of the renewable resources, non-renewable resources are limited for the entire project.

Without considering non-renewable resources, there is also a variant to classical RCPSP which deals with the execution modes only considering renewable resources. Different modes are defined considering the duration and resources allocation. Linking to the rectangular task definition given in chapter X, we can summarise that the multiple execution modes RCPSP is a problem where each activity can be defined following different rectangles. Preemption is not allowed in this variant and if the activity starts in one mode it can not change the mode during the execution.

If we consider the capacity of the resources, we can classify different problems considering constant available resources or available resources that vary during the project. We can also make a classification considering single project or multiple simultaneous projects.

Lastly, a classification can be made between highly disjunctive and highly cumulative problems. A scheduling problem is highly disjunctive when many pairs of activities can not be performed in parallel using the same resource. On the contrary, highly cumulative problem accept the fact of executing many activities in parallel using the same resource [BAPTISTE and LE PAPE '00].

3.4.3.1.2 Solving methodologies

Numerous methodologies have been developed in order to solve Resource Constrained Project Scheduling Problems [OZDAMAR and ULUSOY '95] [HERROELEN, et al. '98, KOLISCH and PADMAN '01, TAVARES '02]. These methodologies can be classified into exact approaches and heuristic and metaheuristic approaches. Exact methodologies give optimal solutions to the problem [KOTSIOPOULOS and CASSAIGNE '02], while heuristics and metaheuristic approaches are based on the concept of a guided algorithm whose purpose is to solve complex problems where the exact algorithms are not sufficient.

Exact algorithms include dynamic programming, zero-one programming and implicit enumeration with Branch and Bound [BRUCKER, et al. '94 , HARTMANN and KOLISCH '00, MINGOZZI, et al. '98, STINSON, et al. '78, WU, et al. '99].

Heuristics are approaches for directing one’s attention on problem solving. It is originally derived from the Greek “heurisko” which means “I find”.

Metaheuristics are heuristics to solve some computational problems like RCPSP using black-box procedures [DEPUY and WHITEHOUSE '01, KOLISCH and HARTMANN '99].

Heuristic and metaheuristic algorithms have been successful because of the fact that they give rapid answers with relatively good quality for large problem. The main pitfall of these algorithms is that they are usually developed for a specific problem. For the same reason it is difficult to realise a classification of this algorithms.

We will include in this category the local search metaheuristic which can be used for problems that can be formulated as finding a solution maximizing or minimising a criterion among a number of candidate solutions. Local search algorithms move from solution to solution in the search space until a solution supposed to be optimal is found or a time bound is elapsed. Based on local search method, another widely used metaheuristic is the tabu search. Tabu search uses a local or neighbourhood search procedure to iteratively move from a solution x to a solution x' in the neighbourhood of x, until some stopping criterion has been satisfied [PINSON, et al. '94, THOMAS and SALH '98]. Other local search algorithms include simulated annealing [KIRKPATRICK, et al. '83].

Other heuristic and metaheuristic algorithms are inspired on biological processes, like the ant colony optimization algorithm or genetic algorithms. Ant colony optimization uses many artificial ants (or agents) that incrementally build solutions. Artificial ants deposit artificial pheromones (to this ant-inspired behavior is due their name) that are used by later ants to guide their search [LUO, et al. '03 , MERKLE, et al. '02]. Genetic algorithms maintain a pool of solutions rather than just one. The process of finding superior solutions mimics that of evolution, with solutions being combined or mutated to alter the pool of solutions, with solutions of inferior quality being discarded [WANG, et al. '05].

3.5 Dependencies management

3.5.1 *Defining dependencies*

Dependencies appear when reliance or dependence exists between two or more objects.

Dependencies between actors or teams are usually considered as interactions. In general, in order to cover the dependencies between different kinds of objects, we will use the term relation or relationship. In order to characterize dependencies, one should start defining the type of dependencies and between what kind of object act the dependency.

Marle [MARLE '02] defines seven types of objects participating in a project and seven types of dependencies interacting with the objects. The objects defined in his work are:

- Actors
- Activities
- Goals
- deliverables
- Project
- External decisions
- Process and organisations external to the project
-

The first five objects are defined in the framework of an accurate project ones the object “project” has been defined. Nevertheless, the project is not an isolated object and there are external objects that should be taken into account. In the work of Marle this external environment is taken into account by the last two objects. Considering these objects, Marle defines seven possible dependencies between these objects:

- Hierarchical relation
- Resource utilisation relation
- Sequential relation
- Contribution relation
- Influence relation
- Similarity resemblance relation
- Exchange relation

The first three relations are the most common relations that we can find in a project. Furthermore, contribution relation and influence relation are more related to the goal of the project, how this goal is cascaded downstream, and the decisions that are taken by different actors. Similarity relations are linked to best practices actions and have a temporal aspect, as the relation is established between objects being part of different periods. Finally, the exchanges relations are those relations that are not necessarily hierarchical or that have not lead to any influence between the objects. This is the case of relations that we can find in some information systems. Nevertheless it will not be a relationship that we will study in our work. In our case we will focus on hierarchical relations, resource utilisation relation and sequential relation.

Hierarchical relations and resources utilisation have already been treated in the chapter related to the constraints. Indeed, hierarchical relations included due dates constraints defined by the upper managerial level, as well as the resource capacity given by this level. Moreover, during the project running, hierarchical relations are usually based on an information flow which is often the basis for the decision making process. Key Process Indicators (KPI) are very useful tools for “at a glance” representation of the project progress and is part of the day to day hierarchical relations.

Resources utilisation can be seen as relations as it has been done on Marle’s work or as constraints as we have introduced for our work. Indeed, two team that are constrained to use the same resource will have a relationship that will force them find a good balance between the usage as the common resource on each team. Nevertheless, if the viewpoint of this problem is situated inside one of the teams, the common resource usage is seen as a constraint as we have defined in the 3.4.2.3 chapter.

Sequential relations have been widely studied in scheduling literature. It has usually been named as temporal constraints. In our work, we will consider two types of sequential relations. On the one hand we will deal with precedence between internal activities and on the other hand with constraints related to the interdependencies with other teams.

3.5.1.1 Precedence between internal activities

Due to technological requirements, some activities can not be performed independently from each other. Precedence between activities can be found in different scheduling problems and the RCPSP is not an exception. In literature, these kinds of constraints can be defined as temporal constraints or generalised precedence constraints. Figure 27 shows thirteen possible temporal relations between two activities [ALLEN '83].

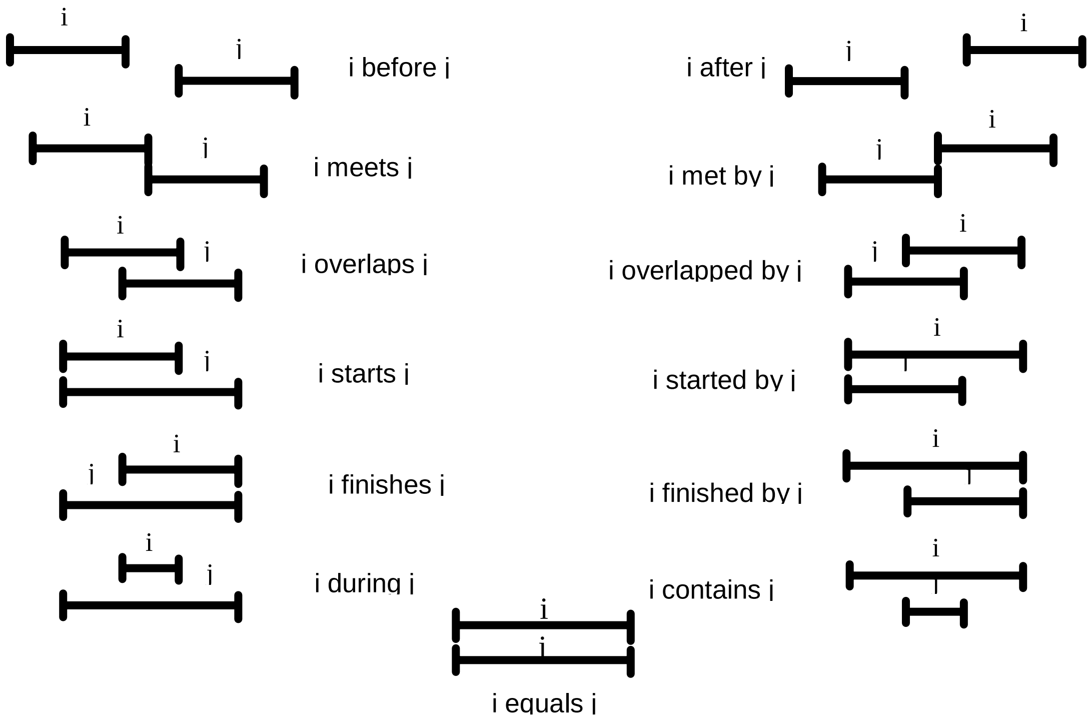


Figure 27: possible temporal relations between two activities

In reality a pair of activities will be performed in one of these modes. Nevertheless; the most used precedence constraints take into account only the simple finish-start precedence type. Therefore if activity *i* precedes activity *j*, this constraint is usually modelled as follows:

$S_i + d_i \leq S_i$

Being *S* the start time of the activity and *d* the duration of the activity, usually defined as a non-negative integer. Nevertheless this type of sequential activities model does not support parallel activities which are very common, as we have already seen, in Concurrent Engineering.

3.5.1.2 Interdependencies with other teams

Interdependencies between other teams are sequential relations, but on the contrary of the precedence between internal activities, interdependencies with other teams are not due to technological limits but rather to the fact of a geographic distribution.

During the design process, interactions between design teams appear while exchanging data (Figure 28).

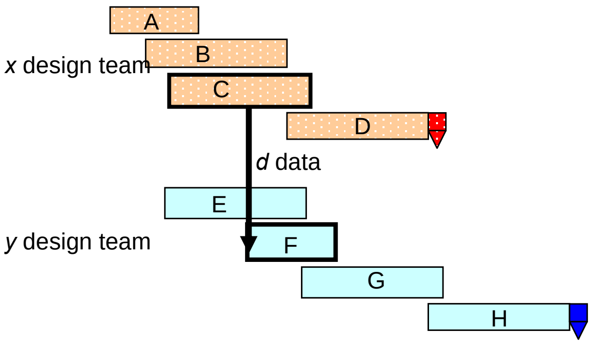


Figure 28: Data exchange between two activities.

Data is the generic term used to describe deliverables that are exchanged between design teams. These deliverables can be models, drawings, mock-ups, requirements specification document, calculation results, sketches, test results, etc. They are produced in order to answer to a specific requirement (or set of requirements) from a stakeholder involved the development process. Data is provided by the supplier as an output of a design activity and will be used by the customer as an input for its own design activity.

From a supplier point of view, characteristics of data evolve and get closer to final data. From a customer point of view, reliability of provided data will increase at the same time as supplier's design activity progress. Generally, the likelihood of modifying a data decreases and its maturity increases until the completion of the design activity.

If we refer to the time-location matrix [RODDEN and BLAIR '91] distinguishing between synchronous and asynchronous work on the one hand and co-located and distributed settings on the other hand; we will link interdependencies management to distributed settings.

Data exchanges are nowadays largely supported by product oriented information systems that support distributed work. Some tools support also engineering process models [LASMIS, et al. '03]. Managing workflow of the engineering process together with information concerning the product itself can lead to efficient management of interdependencies.

Most popular groupware technologies for data exchanges are based on asynchronous communications tools [GUTHRIE '04, LABORIE '06, LARSSON '05, MACGREGOR, et al. '01].

Work carried out at the same time need to focus on data exchanges practices in real-time; this is the goal of the shared workspace technology that is supported by some Computer

Supported Cooperative Work (CSCW) tools [THOMSON, et al. '00a, THOMSON, et al. '00b]; it is also the goal of the concept of collaborative spaces [GIRARD, et al. '03].

3.5.2 State of the Art for dependencies management

3.5.2.1 Dependencies management and concurrent engineering

Terwiesch and al. [TERWIESCH, et al. '02] focused their work on the content of the exchanged between design teams and mainly on the preliminary information exchanges. They categorise information based on two variables: information precision and stability. The former is linked to the accuracy of the data exchanged while the latter defines the probability of given information of being modified later on.

Based on these variables, different strategies of data exchanges can be defined, these strategies are included in a large variant of possibilities between two boundary strategies defined as iterative and set based strategies. On the one hand the iterative strategies focus on accurate data delivery since the beginning. This strategy involves a high risk of modifying the information on the next delivery. In others words it focus high stability rather than high precision. On the other hand, set based strategy deals with a range of values related to a value. This value range will converge until an accurate value is reached. This means that the data customer will not have accurate information in preliminary deliveries. Nevertheless, stability of the information will be high.

Strategy chosen in order to exchange information will depend on the technology of the subsystem to be developed but also on data exchanges management practices. Moreover, actors involved in these exchanges can define different collaboration modes depending on the rework risk that they are organized to consent.

3.5.2.2 Design Structure Matrix (DSM)

We have already introduced DSM's and have seen that it is an interesting tool for modelling different aspect of the project. In this chapter we will focus two utilisations of the DSM: Team-based DSM and the activity-based DSM. In the first case, the different existing teams can be listed in the DSM and the different components can also be listed in order to study the exchanges between the existing teams and more accurately, which data, related to a component, has been exchanged. Early identification of these dependencies allows to stress the relationships between various teams and eventually to create a period where both teams will be collocated in order to define jointly the main architecture of each subsystem and in order to define accurately the interface.

In some cases, this exercise has been used to minimise the exchanges between team by reorganising the teams, through manipulation of matrixes. In Figure 29, McCord, K. and Eppinger, S. have used the DSM in order to reorganise an engine development OBS [McCORD and EPPINGER '93]. In order to perform this task, they have tracked the data exchanged between the former product development teams. For each data they have measured the subsystem that was concerned and the number of exchanges. The result is a matrix where one can realise that several exchanges occur between the former teams, which could be source of inefficient development. On Figure 30, a new organisation is established, following a predefined algorithm of DSM that forms clusters, gathering those subsystems that have mainly exchanged data between them. Finally, those subsystems that have exchanged with more or less all the subsystems will be clustered in a new team called “integration team”

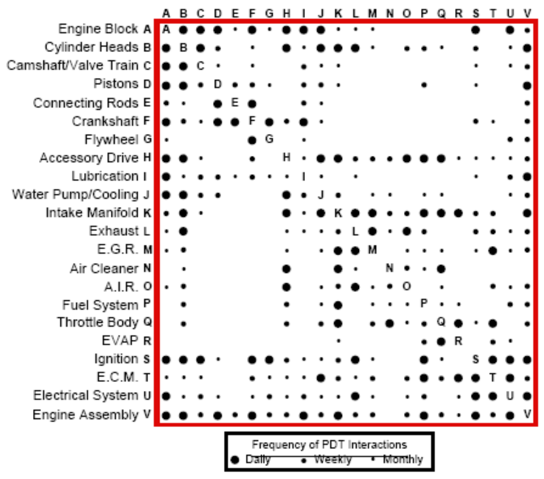


Figure 29: Design Structure Matrix of the Engine Development Project

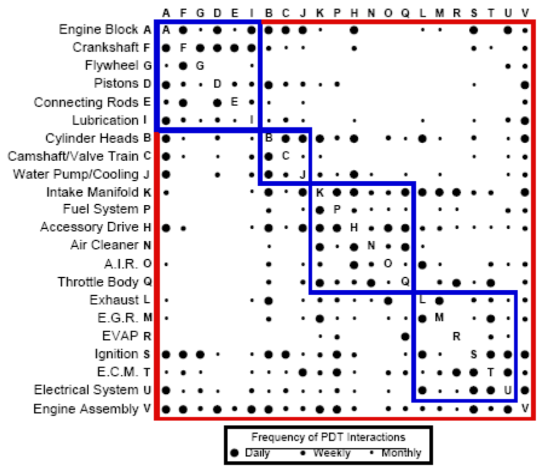


Figure 30: Ordered DSM Showing Existing System Team Structure

Instead of actors, teams or subsystems, the DSM can also be built listing the activities defined for the project. This is called an activity-based DSM. The advantage of the activity-based DSM is that it allows the identifications of loops and iterations. Indeed, activities that are completed might restart part of the work because of the fact that they depend on activities that are performed later. This type of dependencies can be very time consuming and in the worst case, they can make the project not to converge to the defined goals in the required time. Therefore, it is necessary to minimise this kind of dependencies.

3.5.2.3 Preliminary information and data maturity concept

Saint-Marc in [SAINT-MARC '06] investigates data dependencies networks, adjacent matrix and data maturity aspect in order provide decision making means for collaborations management.

A data dependencies networks can be built modelling as nodes the different teams and as arrows the data that has been exchanged between the teams. For small project, this can be a modelling method to track the exchanges between teams. Nevertheless for complex project it can be very difficult to understand. A first modification can add visibility to this representation adding a temporal aspect to the exchanges. Indeed, if key milestones are defined and data exchanges are modelled taking into account the phase when is performed, the model becomes more comprehensible. In this ordered dependency network, data represented on a same column is created at the same date. Therefore it is possible to study the dependencies that exist between data and which deliverables must be realised in order to deliver an accurate data. Nevertheless, this model does not allow studying iterations, or data that might be modified due to a loop involving a delivery realised in a later milestone.

In order to select relevant data two importance-indicators are used: Criticality and Potential Risk. While criticality enables the impact analyse of a data's descendants, Potential risk deals with the ancestors. If data created by a supplier is used by several customers, directly or indirectly in later deliverables (issued from the first data), this data is considered to be more critical than a data that is used only by few customers. In the same way, the potential risk will take into account the number of ancestors related to the data and will consider that a data that has several ancestors is a more risky data, meaning that the probability of receiving one of the data late and therefore the probability of delivering the data not on time is higher that the data which depends only on few ancestors. But Saint Marc has gone further in the representation and does not use this representation only for modelling dependencies between data. He also proposes the evolution of the data version using maturity levels.

Maturity concept is linked to a human perception of the performance linked to every characteristics of a data [SAINT-MARC, et al. '04]. Even if this concept is not new, there are few works that have developed it [EVERSHEIM, et al. '97]. Maturity level related to a data is calculated using the relative data maturity which is a ratio that represents the gap between the objective maturity of the data definition and the actual state of the data (absolute maturity).

$$M_{rel} = \frac{M_{abs}}{M_{obj}}$$

The objective maturity is the highest level of definition of the data. It is a reliable data that the customer is sure that will not be modified. It is also a range of value that has converged to an accurate solution.

Therefore, the relative data maturity can be tracked during the design process comparing it with the actual absolute maturity level. (See Figure 31)

This comparison is made in order to ensure that each customer will receive qualitatively and quantitatively the information it needs in order to perform the scheduled work.

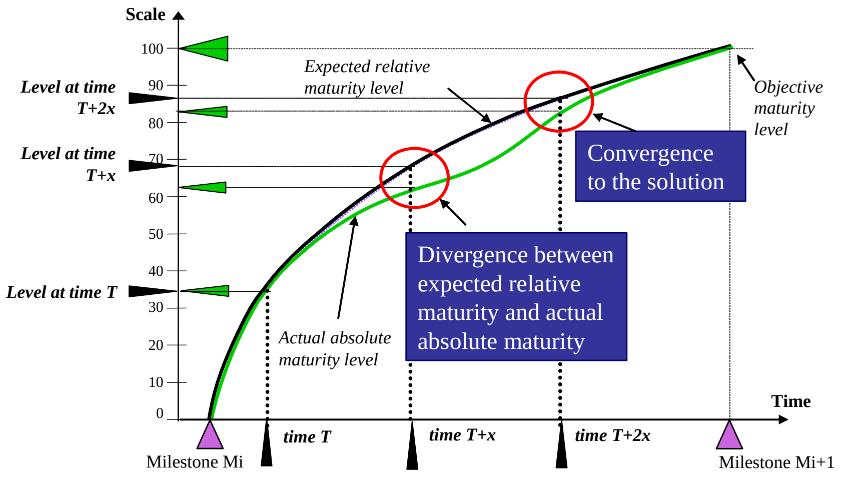


Figure 31: Relative Data Maturity Progress (from [SAINT-MARC '06])

In former work, the author has also dealt with the concept of maturity, focusing on the product data maturity progress definition taking into account exchanges definition with external design teams [LIZARRALDE '03].

Other authors have developed frameworks focusing computing systems like Product Data Management tools, in order to support collaborative activities of the design process. These frameworks focus on preliminary information exchanged between different design teams and deal also with the concept of maturity [BLANCO, et al. '06, GREBICI, et al. '05].

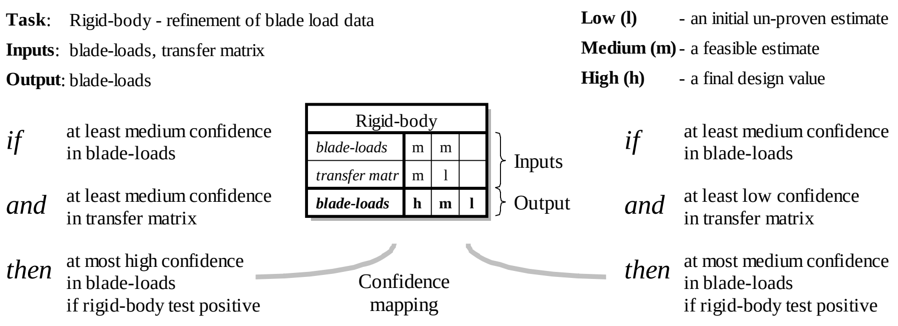
3.5.2.4 Signposting

Signposting is an approach to model the design process taking into account the dynamic aspects of this process. This approach was developed by Clarkson and Hamilton from EDC at Cambridge University [CLARKSON and HAMILTON '00]. The Signposting model is based on an activity-based network. Activities' connectivity is ensured by parameters and tracking these parameters allows a representation of the design process that not only takes into account the finish-end relations between activities but maturity evolution of these parameters. Indeed, the maturity of these parameters can be measured during the design process, this in realised using subjective confidence that the designer has in the parameter refinement.

In Signposting, confidence is an abstract quality which is linked to the actors judgement of the design‘s maturity. Performing the design activities will improve the maturity and therefore the confidence.

For each activity's context is specified as a level of confidence in input and output parameters. Input parameters are required at a specified level of confidence to begin each task. These levels can be defined as none, low, medium, or high level of confidence (Figure 32). When a task is completed, confidence in output parameters is usually increased.

An advantage of the signposting model is that evaluation activities can be defined in the schedule in order to re-evaluate the confidence level. If the confidence level is reduced, some of the tasks which have already been completed might have to be restarted.



Task: Rigid-body - refinement of blade load data
Inputs: blade-loads, transfer matrix
Output: blade-loads
Low (l) - an initial un-proven estimate
Medium (m) - a feasible estimate
High (h) - a final design value
if at least medium confidence in blade-loads
and at least medium confidence in transfer matrix
then at most high confidence in blade-loads if rigid-body test positive
Rigid-body
blade-loads m m
transfer matr m l
blade-loads h m l
Inputs
Output
Confidence mapping
if at least medium confidence in blade-loads
and at least low confidence in transfer matrix
then at most medium confidence in blade-loads if rigid-body test positive

Figure 32. A typical signposting confidence mapping

Applications of Signposting have included design process navigation or project simulation [WYNN, et al. '06]. Current researches based on Signposting models have been focused on flowchart models and probabilistic methods. This approach aims to model the design process taking into account resources requirements as well as uncertainty information [WYNN, et al. '05].

3.6 Conclusion

We have focused on three aspects that are key elements for the scheduling of design activities during the development of a new aircraft.

Uncertainties related to time and resources allocation of design activities can be managed using proactive or reactive techniques. While the first one focuses on defining baseline schedules aiming to stay valid even if a disturbance occurs, the second one focuses efficient techniques to redefine the schedule if it id needed. Moreover, we have investigated how risk analysis techniques can support our approach mainly in order to deal with unforeseen events that impact time and resources allocation.

Then we have briefly introduced different levels that handle the time and resources allocation problem with different views. Our approach aims to focus scheduling problems on the tactical level. This level needs to deal with exchanges between design teams and between different managerial levels. We have mainly focus on managing dependencies between different design teams and techniques that aim to enhance the collaboration between these teams.

Our aim is to integrate the three aspects into the same framework that will be supported by a common project scheduling management model. The resulting framework will be considered as a decision support system since it shall enable team managers to check consistency of its decisions with the different project constraints.

In the next chapter we will describe the project scheduling management model that we have developed and how the three aspects that have been investigated in this chapter are supported by this model and reused some of the reviewed concepts.

Chapter 4: A proposal based on Constraints Satisfaction

4.1 Introduction

In the former chapter we have defined three aspects of activities scheduling on which we want to base the functionalities of a DSS for team managers.

One possible way of integrating various sources of knowledge that must be taken into account in a problem solving model is to express them as constraints.

This chapter defines the constraint satisfaction problem (CSP) model to support decisions in activities scheduling.

First, we propose to characterize activities by the energy required, in order to manage both time and resources constraints.

Then, we describe our model, defining the set of variables and constraints, then the constraint propagation mechanisms used for the solving procedure.

Finally we explore some decision support functionalities that this model can bring at the tactical level scheduling management, the treatment of some uncertainties in design activities scheduling and interdependencies management.

4.2 A model based on the energy allocation problem

Energy has been generally considered as a synonymous of the "power”, "force" or “activity”. Energy has an accurate definition in physics: it is defined as the capacity to do work or the amount of work a physical system can do on another.

A work W is equal to the integral (along a certain path) of a force F.

$$W = \int F.ds$$

In contrast, the power P is the rate at which a given work is performed, or at which the energy is transferred during time. In the IS system of measurement, power is measured in watts (W)

$$P = \frac{W}{t}$$

where t denotes time (duration).

Conversely, energy can also be defined as:

W = P * t where P is a constant power and t denotes the duration of the utilization time. In fact the most general definition enables the power to be a function over time, as we will show later.

Considering work allocation problems in which one must decide how some shared resources may be allocated to activities, the energy concept enable to express energy conservation equalities or inequalities by comparing the energy required by the work activities and the maximal available energy that resources can provide [LOPEZ '91].

In the case of design activities, resources are usually linked to human labour and therefore power takes integer values, (unit = persons), en energy is defined as manpower which is the product of human productive units by a duration. For example, an activity requires 3 men.months, and belongs to a project for which a maximum of 30 men.months have been allocated. Depending on the time granularity, men.weeks, men.hours or men.years may be more suitable.

In the literature, other terms such as *strength* or *intensity* are used instead of power.

In our work will use the term intensity, denoted $A(t)$, or $a(t)$ to describe respectively the available or needed resource quantities at time t. Manpower will be used as a generic term that includes the available resources during the project and how they are distributed.

Therefore in our work energy characterizes a quantity of work and is then proportional to time and to the intensity of the resource able to perform it. More formally, energy is expressed as the integration of a resource intensity over time (Figure 33):

$$e^{[t_1,t_2]} = \int_{t_1}^{t_2} a(t).dt$$

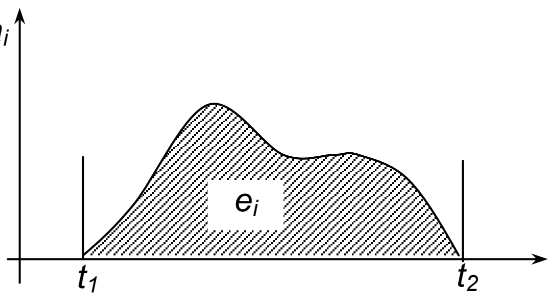


Figure 33: Energy represented in a resource-time diagram

If we make the assumption that the intensity allocated or required by the task *i* take discrete integer values (Figure 34), the curve representing it is generally made of one or several steps (see Figure 35).

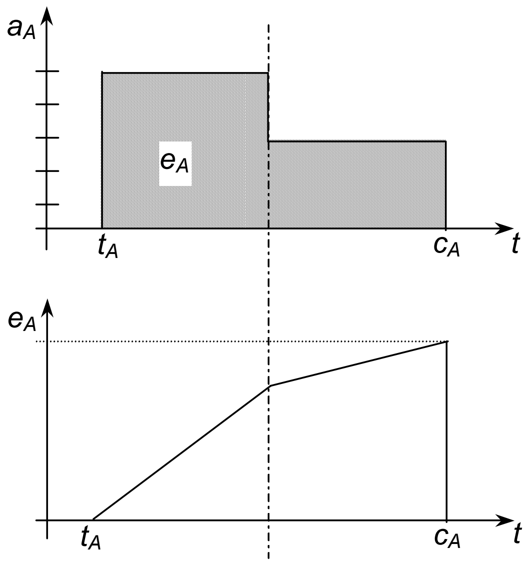


Figure 34: Resources allocation and energy consumption for task A.

Consequently, energy can be represented as a cumulative curve (never decreasing) showing the amount of work realised until a date.

In practice, energy (or a work quantity) is classically represented in a two-dimensions diagram by the area located under the resource consumption intensity curve, and between two dates. Under the assumption the problem is discretised into equal periods, the following drawing represents a possible performance of an activity i; notice that intensity may only vary from one period to another, but never within a period.

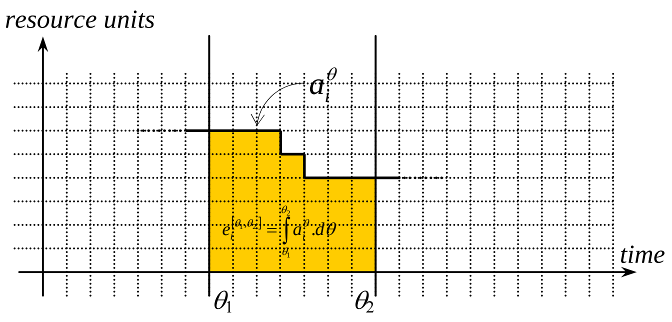


Figure 35: Energy represented in a discretised resource-time diagram

In the particular case where a_i^θ = a$_i$ $\forall \theta$ (intensity is constant), the definition of the intensity can be simplified: $e^{[\theta_1,\theta_2]} = (\theta_2 - \theta_1).a_i$

Energy is particularly interesting for tackling our scheduling problem in which work quantities that define the activities are well defined and can be considered as data, while durations and resource allocations are decision variables.

When evaluating engineering activities, most practices that we have identified in Airbus consider work quantity rather that activities duration. Nevertheless, scheduling activities often stress on activities duration, since some absolute limit times exist (earliest starting time and latest finishing time) that constraint the whole project and limit consequently any activity duration.

The utilisation of the energy concept allows a flexibility degree since it offers the possibility of varying duration and resources allocation. This flexibility fixes better the reality of the engineering activities. Nevertheless the utilisation of energy in a determinist framework supposes to have a good knowledge on work quantities associated to activities.

The energy concept enables us to build special constraint propagation algorithms (cf. for example, [BRUCKER '02, ESQUIROL, et al. '01, KUMAR '92]) that are useful both to characterize the problem consistency but also to improve the resolution process, by reducing dynamically the domain of remaining decision variables, after each decision step. The main idea of this so-called energy-based reasoning approach is to deduce restrictions on time and resource allocation for one activity by taking into account the resource availability and the minimal resource consumption of the remaining concurrent activities. This kind of reasoning has been successful in many scheduling problems [LOPEZ, et al. '92]. We will describe in the following sections how these ideas can be further reused in our model.

4.3 Problem Modelling

4.3.1 Constraint satisfaction problems (CSP) and constraint programming

Many scheduling problems can be represented as constraint satisfaction Problems (CSPs). A CSP is mainly characterised by a set of variables (decisions), a set of possible values for each variable (domains), and a set of constraints between the variables. A solution assigns a value to each variable, respecting that all the constraints are satisfied [BAPTISTE, et al. '01].

The CSP was firstly studied in the 70s by Huffman, Clowes and Waltz for solving line-labelling problems (from [DORNDORF '02]). Cohen in [COHEN '90] deals with CSP related to constraint programming environments which provide a framework for solving CSP models. The emergence of efficient constraint-based scheduling algorithms in the mid-90s [CASEAU and LABURTHE '94, COLOMBANI '96], and the diversity of scheduling problems have made constraint programming a useful approach for the resolution of complex industrial problems.

Constraint Programming enables to make a clear distinction between two knowledge types: the declarative definition of the constraints that defines the problem and the procedural methods (algorithms and heuristics) that exploit these constraints to solve the problem.

Moreover, constraint programming enables to distinguish constraint propagation and search algorithms. Constraint propagation consists in reducing the domains of the variables, eventually deducing new constraints from existing ones, and detecting inconsistencies. This deductive process is called constraints propagation. It can be embedded in the programming language, while search methods and heuristics are user-defined.

For example, from $y > x$ and $x > 3$, we can conclude that y is at least 5. This inference reduces the amount of computation needed to solve the problem. Moreover, following with the same example, if we add a new constraints $y < 5$, an inconsistency is detected. Without propagation, no inconsistency will have been detected until the instantiation of both x and y.

For some small problems with unique solutions, the solving can be performed only by propagating constraints [LOTTAZ, et al. '00]. Nevertheless, real industrial scheduling problems need a searching method to find a feasible solution, as propagating constraints are not sufficient.

Searching methods are characterized by the order in which variables are instantiated and by the order in which values are enumerated for each chosen variable. When all the variables have an assigned value and these values respect the array of constraints, a solution has been found. The search space is the set of all possible total assignments. It may be usually very large because it grows exponentially with the problem size: for instance, if all variables have initially the same discrete domain:

Search space size = (Domain size)$^{\text{Number of variables}}$

We can make two types of classifications regarding searching methods. First, we can separate complete and incomplete methods. Complete search means that the search space can be explored entirely. This method guarantees that the solving process is complete and that any solution is accessible. This is necessary when the optimal solution is needed (one has to prove that no better solution exists).

Incomplete search may be sufficient when just some solution or a relatively good solution is needed. This is often the case on large scheduling problems where a feasible solution, not necessarily optimal is needed in a short time.

Secondly, we can distinguish constructive and repair-based or improving methods.

With constructive searching methods, the solving process advances by incrementally constructing assignments (thereby reasoning about partial assignments which represent subsets of the search space).

On the opposite repair-based methods shift from one total assignment (not necessarily admissible) to another one until a correct solution is found. This change is usually done by modifying previously explored assignments. Local search methods as well as population-based methods are included in this category.

Separating constraint propagation and search has two main advantages. On the one hand, it allows the system developer to implement the constraints propagation code and the decision-making code independently of one another. The same constraint propagation code can then be used to propagate decisions made by a decision-making algorithm as well as decisions made by a human user.

In some problems, few decisions can lead to a solution of the problem thanks to the propagation mechanisms. On the second hand, the separation of constraint propagation and decision-making allows the developer of a constraint-based application to reuse constraints propagation techniques developed for other applications. Using constraint-solving tools marketed by software editors is a well known practice for application developers.

First software related to constraint logic programming where based on innovative extensions of the well-known logic programming Prolog language (developed by Alain Colmerauer and Philippe Roussel in the 70's); let us cite Prolog III [COLMERAUER '90] and CHIP [AGGOUN and BELDICEANU '93]. Nowadays the offer in the domain of constraint programming languages or dedicated constraint libraries for imperative languages (C++, Java) is quite large.

One of the main advantages of using constraint-solving tools marketed by software editors is that the tool providers have invested significant effort in applying robust constraint propagation algorithms. This is the case of two application developed by ILOG company[2] named ILOG SOLVER and ILOG SCHEDULER [LE PAPE '95].

Other implementations that are available nowadays in commercial or open source format are often based on Prolog language. Dozens of applications based on Prolog language are available nowadays for academic or industrial purposes. One of the most powerful systems is ECLIPSe[3].

ECLiPSe is a constraint logic programming system that includes Prolog and enriches it with constraint propagation mechanisms on a large variety of domains (integers, booleans, sets…). ECLiPSe was developed until 1995 at the European Computer Industry Research Centre (ECRC) in Munich and then until 2005 at the Centre for Planning and Resource Control at Imperial College London (IC-Parc). It is currently copyrighted by Cisco Systems. In September 2006, it was released as open source software under the Cisco Systems/Cisco-

[2] http://www.ilog.com
[3] http://eclipse.crosscoreop.com/

style MPL, equivalent to the Mozilla Public License. It is unrelated to the Eclipse software development framework. The constraint propagation mechanisms developed in the framework of this project where implemented using ECLiPSe.

4.3.2 Problem statement and modelling

The activity-scheduling problem that we consider is defined by the following assumptions.

4.3.2.1 Quality gates and time horizon

Let us note two consecutive quality gates *v-1* and *v* that determine the scheduling horizon devoted to the development of a given subsystem. The manager of a design team responsible of the development of this subsystem will define a set of activities noted I_v = {*i*=1..*n*} that are necessary in order to fulfil the requirements of the next quality gate v. We suppose the time horizon between these two quality gates is discretised into *H* time periods $\theta = 1..H$ [LIZARRALDE, et al. '07b].

Quality gate *v–1* will be performed at the beginning of the period $\theta = 1$ while quality gate v will be performed at the end of the period $\theta = H$. Periods are typically weeks, supposing that any activity requires at least one period to be achieved, even in the case of a maximal resource allocation.

4.3.2.2 Definition of Activities

As we saw before, activities are mainly defined by their energy: e_i denotes the energy required to perform *i*, between its starting date s_i and its finishing date f_i.

Due to this hypothesis, we consider full elastic preemptive activities [BAPTISTE, et al. '99, BUTTAZZO, et al. '02 , CHANTEM, et al. '06]. The duration of an activity *i* is not known in advance and its intensity a_i^θ can vary during the performance. Then, the number of resource units allocated to *i* may become null at some periods θ, excepted for s_i and f_i.

We also suppose this intensity to be integer, considering that elementary resource units are persons. Consequently, the intensities $\{a_i^\theta\}$ are the main variables of the problem, one per activity and per period. The scheduling problem is thus transformed into an allocation problem.

Activities may be submitted to individual time window constraints, defined by an earliest start period r_i and a latest ending period d_i, with $1 \leq r_i \leq d_i \leq H$. We will see further how to take these constraints into account in our model by setting null values for some a_i^θ variables.

4.3.2.3 Definition of resources

We define a cumulative problem, where maximum resource availability curve will be one of the most powerful constraints.

We made the assumption that the problem considered is a mono-resource problem. We will discuss later the more realistic case in which design teams have several distinct competencies, and how to take this feature into account with a multi-resource model.

The resources used in this model are mono-task, in other words, they can be allocated only to one task during a period. In other words, when we allocate an energy unit to a period, it means that we allocate a person. Other variants to this assumption will be discussed later.

As quality gates dates are given by senior management, the maximal resource availability A^θ is also supposed to be fixed at this decision level. A^θ is an integer number that represents the maximum number of persons in the team who may work concurrently at any period θ.

4.3.3 Constraints to be respected

The first three types of constraints of our model are easy to express:

4.3.3.1 Activity energy constraint:

As the energy e_i to be consumed for processing each activity *i* is a piece of data, any solution must respect:

$$\sum_{\theta=1}^{\theta=H} a_i^\theta = e_i \ \forall i = 1..|I_v|.$$

4.3.3.2 - Cumulative resource constraint:

The maximum resource availability curve A^θ is also a piece of data and we can state for each period:

$$\sum_{i=1}^{i=|I_v|} a_i^\theta \le A^\theta \ \forall \theta = 1..H.$$

To the maximum resource intensity, we can add individual resource intensity (one per activity), defined by a range of values denoted by two bounds: $[\underline{a}_i^\theta, \overline{a}_i^\theta]$. If no individual constraint is defined, domains $[\underline{a}_i^\theta, \overline{a}_i^\theta]$ are all equal and set to $[0, A^\theta]$. Nevertheless, individual intensity constraints can be defined independently of the maximum resource intensity. These constraints will be defined as follows.

$$\forall \theta = 1..H, \ \forall i = 1..n \ \ a_i^\theta \in [\underline{a}_i^\theta, \overline{a}_i^\theta]$$

In the examples we have developed in our work, we define in most of the cases these kinds of individual constraints. Therefore, the limits of a intensity a_i^θ will not be limited to A^θ but to a value $\overline{a}_i^\theta$ such that $\overline{a}_i^\theta < A^\theta$.

The rational behind this individual constraint is to avoid resources concentration in a particular task that could be identified as critical.

4.3.3.3 Time window constraints

If such constraints are needed, it is easy to initialize to zero any variable a_i^θ for each activity *i* in any period that does not belong to the time window of *i*:

$$a_i^\theta = 0 \text{ for } \theta \in [1, r_i - 1] \cup [d_i + 1, H]$$

The next two constraints are related to model interdependencies between two activities. The first one is an interdependency constraint that deals with a pair of activities belonging to the same design team schedule: the Energy-Precedence Constraint (EPC). The second deal with interdependencies between two design teams: Contract Dependency Constraints (CDC), which are usually formalised by contracts and are often designed as dependencies due to the fact that interactions are usually defined as a supplier/customer type relationship.

4.3.3.4 Energy-Precedence Constraints (EPCs):

Classically a scheduling precedence constraint between two activities {*i, j*} forces an activity *i* to be finished before an activity *j* begins. It is expressed as the potential inequality $t_j - t_i \geq p_i$ or, which is equivalent to : $t_j \geq c_i$.

In a concurrent engineering context, a full parallel execution of design and development activities is desired but not always possible since it could violate the resource availability constraint or because there may be interdependencies between some pairs of activities. In the latter case an activity *i* is forced to be in a state where it has already consumed a minimal energy e_{ij} (with $e_{ij} < e_i$) before activity *j* can start.

This energy corresponds to the minimal work that has to be done in activity *i* to produce reliable data that can be used to start activity *j*. For that reason we call it an Energy-Precedence Constraint (EPC): *EPC (i, j, e_{ij})*. Let us note that the traditional scheduling precedence constraint is a particular EPC in which $e_{ij} = e_i$.

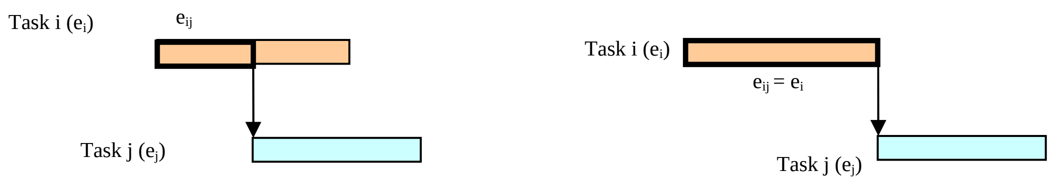


Figure 36: The Energy-Precedence Constraint and the traditional scheduling precedence constraint

The minimal work that has to be done in activity *i* to produce reliable data for *j*, is defined by the supplier and constrains the parallelism level that may exist between both tasks. From a time point of view this overlapping period will depend on the resources allocation. Furthermore, in order to define the parallel level degree, customer can also have some requirements. Indeed, the customer might ask a minimal work that has to be done once the task *i* is finished. This amount of energy corresponds to the minimal work that responsible of task *j* has defined to be done once the precedent task is finished and therefore it is certain that no other modifications will be realised. In order to make the difference between both types of EPC, we will name the former a “Precedes” EPC type (e_{ij}) and the later a “Follows” EPC type (e^*_{ij}).

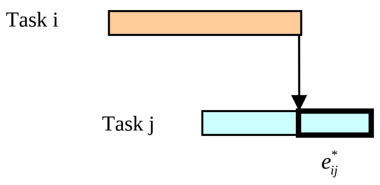


Figure 37: Follow type EPC

In Figure 38 both amount of energy are described in a precedence task relation. Nevertheless, we must insist that there is no relation between both quantities of work. While the first one is defined by the supplier independently, the second one is only related to the customer’s decision.

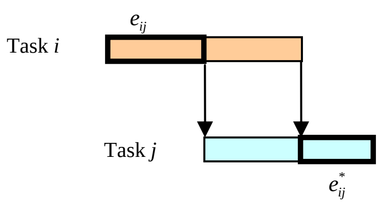


Figure 38: Precedes and Follows type EPC

In the case where constant amount of resources are used to define the tasks, we could define a formula to relate both concepts as follows:

$$\frac{e_i}{a_i} - \frac{e_{ij}}{a_i} = \frac{e_j}{a_j} - \frac{e^*_{ij}}{a_j}$$

where a_i and a_j are the resources intensities allocated to task *i* and *j* respectively.

Therefore, the fraction between intensities will be equal to the fraction between the energy allocated in the parallel period:

$$\frac{a_i}{a_j} = \frac{e_i - e_{ij}}{e_j - e^*_{ij}}$$

Nevertheless, this relationship is usable only in a very particular case since the fact of having e_{ij} before the start of task *j* at the same time of having exactly the e^*_{ij} after end date of task *i* is only a coincidence. In reality, in the case where constant amount of resources are used, one of the constraints will be more powerful that the other and only the former will be considered.

But, since our model allows the non constant allocation of resources, the case of Figure 38 can be feasible independently of the values of e_{ij} and e^*_{ij}.

EPCs are the most difficult constraints to express with allocation variables $\{a_i^\theta\}$ in place of the time variables $\{t_i, c_i, p_i\}$. We propose in the next part, some propagation routines dedicated to these constraints.

4.3.3.5 - Contract Dependency Constraints (CDCs)

Consider a dependency that involves two design teams and activities *i* and *j* for each team. These activities will have a new temporal constraint defined by a due date. It is a special temporal constraint since the due date is not related to the completion of the activity but to the carrying out of a certain amount of work, in other words a constraint related to a dependency forces a team to expend a certain amount of energy before a given date. Indeed, the Contract Dependency Constraint (*CDC_{ij}*) is defined by two pieces of data: $\{t_{ij}, e_{ij}\}$

For the activity *i* of the first design team, we have $\sum_{\theta=1}^{t_{ij}} a_i^\theta = e_{ij}$ and $\sum_{\theta=t_{ij}+1}^{H} a_i^\theta = e_i - e_{ij}$

while the earliest time of the activity *j* of the second design team is fixed and equal to t_{ij}. Therefore:

$$a_j^\theta = 0 \quad \forall \theta \in [1..t_{ij}]$$

In the chapter related to the contracts management we detail more accurately the logic behind this constraint and the relationship between the data delivery as well as the maturity level of this data.

4.3.4 Propagation of constraints and implementation in a CLP framework

We have three types of constraints in our model. Each of them may participate in some domain reductions that facilitate the problem solving procedure.

4.3.4.1 Propagating Activity energy constraint

Each activity must respect the constraint $\sum_{\theta=1}^{\theta=H} a_i^\theta = e_i$. Since allocation variables $\{a_i^\theta\}$ may have an initial domain $[\underline{a}_i^\theta, \overline{a}_i^\theta]$ we can deduce:

$\underline{a}_i^\theta = \max(0, e_i - \sum_{\theta'=1,\theta'\neq\theta}^{\theta'=H} \overline{a}_i^{\theta'})$ and $\overline{a}_i^\theta = \min(\overline{a}_i, e_i - \sum_{\theta'=1,\theta'\neq\theta}^{\theta'=H} \underline{a}_i^{\theta'})$ if $\overline{a}_i^\theta > A^\theta$ then we can updated $\overline{a}_i^\theta \leftarrow A^\theta$

Updating these bounds must be attempted initially, before any decision on variables $\{a_i^\theta\}$ has been taken; it has also to be re-considered as soon as a value is given to some a_i^θ variable (in that case the domain becomes a singleton value $\underline{a}_i^\theta = \overline{a}_i^\theta = a_i^\theta$). This behaviour is completely covered by the CLP language. Example:

Three activities {1,2,3} with a respective energy {5,8,12} require a resource available in 3 units on a horizon [1,10]. We want to schedule activity 1 and activity 2 as soon as possible while scheduling activity 3 as late as possible. We first try to search a solution in which activity 1 and activity 2 receive 1 unit of resource each time they are processed. This leads to the following partial solution:

Table 5: Example of activity energy constraint propagation

a_1^θ	1	1	1	1	1	0	0	0	0	0
a_2^θ	1	1	1	1	1	1	1	1	0	0
θ	1	2	3	4	5	6	7	8	9	10

Supposing we decide to allocate zero units at periods {1,2,3} for activity 3, the domains become:

Table 6: Domains after activity energy constraint propagation

$\overline{a}_3^\theta$	0	0	0	1	1	2	2	2	3	3
$\underline{a}_3^\theta$	0	0	0	0	0	0	0	0	1	1

We can notice that the value $\underline{a}_3^9$=1 is due to the energy constraint for activity 3.

4.3.4.2 Propagating Cumulative resource constraint

The resource consumption must respect the availability constraint $\sum_{i=1}^{i=|I_v|} a_i^\theta \le A^\theta \; \forall\theta$ = 1..*H*. This behaviour is also completely covered by the CLP language. Example:

Three activities {1,2,3} with a respective energy {4,7,10} require a resource available in 3 units on a horizon [1,10]. We want to schedule activity 1 and activity 2 as soon as possible while scheduling activity 3 as late as possible. We first try to search a solution in which activity 1 and activity 2 receive 1 unit of resource each time they are processed. This leads to the following partial solution:

Table 7: Example of cumulative resource constraint propagation

a_1^θ	1	1	1	1	0	0	0	0	0	0
a_2^θ	1	1	1	1	1	1	1	0	0	0
θ	1	2	3	4	5	6	7	8	9	10

Propagating the cumulative resource constraint lead to the remaining domains for variables { a_3^θ }:

Table 8: Domains after cumulative resource constraint propagation

$\overline{a}_3^\theta$	1	1	1	1	2	2	2	3	3	3
$\underline{a}_3^\theta$	0	0	0	0	0	0	0	0	0	0
θ	1	2	3	4	5	6	7	8	9	10

4.3.4.3 Propagating Time window constraint

Time windows constraints are easy to propagate at the beginning of the problem. Any variable a_i^θ for each activity *i* in any period that does not belong to the time window of *I* can be initialized to zero. This is usually the first bounds updating to be realised in the problem solving process because it allows reduce considerably the solution area. Example:

Two activities {1,2} with a respective energy {4,5} are constrained to the time window [2,6] for the first one and [3,9] for the second one on a horizon [1,10]. If we want to schedule activity 1 and activity 2 as soon as possible we first initialize to zero the intensity on the periods {1,7,8,9,10} for the first activity and the periods {1,2,10} for the second activity. This leads to the following partial solution:

Table 9: Example of time window constraint propagation

a_1^θ	0	1	1	1	1	1	0	0	0	0
a_2^θ	0	0	1	1	1	1	1	1	1	0
θ	1	2	3	4	5	6	7	8	9	10

For the first activity two possible solutions are available if pre-emption is not allowed and if activities receive 1 unit of resource each time they are processed:

Table 10: Domains of first activity after time window constraint propagation

a_1^θ Sol 1	0	1	1	1	1	0	0	0	0	0
a_1^θ Sol 2	0	0	1	1	1	1	0	0	0	0
θ	1	2	3	4	5	6	7	8	9	10

For the second activity the following possible solutions are available if pre-emption is not allowed and if activities receive 1 unit of resource each time they are processed:

Table 11: Domains of second activity after time window constraint propagation

a_2^θ Sol 1	0	0	1	1	1	1	1	0	0	0
a_2^θ Sol 2	0	0	0	1	1	1	1	1	0	0
a_2^θ Sol 3	0	0	0	0	1	1	1	1	1	0
θ	1	2	3	4	5	6	7	8	9	10

4.3.4.4 Propagating Energy-Precedence Constraints (EPCs)

Modelling energy-precedence constraints and propagating them need additional definitions.

s_i (starting-time of *i*) is the minimal date θ such that $a_i^\theta > 0$

f_i (finishing-time of *i*) is the maximal date θ such that $a_i^\theta > 0$

$e_{bef}(i,\theta)$, the energy consumed by *i* before time θ: $e_{bef}(i,\theta) = \sum_{u=s_i}^{u=\theta-1} a_i(u)$

$e_{aft}(i,\theta)$ the energy consumed by *i* after time θ: $e_{aft}(i,\theta) = \sum_{u=\theta+1}^{u=f_i} a_i(u)$

$\overline{e_{bef}}(i,\theta)$ the maximal energy that can be consumed by *i* before time θ: $\overline{e_{bef}}(i,\theta) = \sum_{u=\underline{s_i}}^{u=\theta-1} \overline{a_i}(u)$

$\overline{e_{aft}}(i,\theta)$ the maximal energy that can be consumed by *i* after time θ: $\overline{e_{aft}}(i,\theta) = \sum_{u=\theta+1}^{u=\overline{f_i}} \overline{a_i}^u$

An energy-precedence constraint can be represented by one or both of the following formulas:

Precedes (i,j,e_{ij}) which states that any schedule should verify: $e_{bef}(i,s_j) \geq e_{ij}$

Follows $(i,j,\ e^*_{ij})$ which states that any schedule should verify: $e_{aft}(j,f_i) \geq e^*_{ij}$

4.3.4.4.1 Precedes EPC type

We will refer to "Precedes" EPC type when dealing with the minimal work that has to be done in activity *i* to produce reliable data that can be used to start activity *j*; this mandatory preliminary work on *i* is defined by the provider of the data

During the problem solving, the main variables $\{a_i^\theta\}$ are progressively instantiated, one by one. Assigning a value to a_i^θ variable at a given solving stage is an event that triggers routines in charge of updating dynamically a time bound for some activities *i* which are possibly implied in a EPC constraint between *i* and some *j*. More precisely, it is possible to compute the earliest time t_{ij} at which activity *i* has certainly consumed the energy e_{ij} that enables *j* to start:

$$t_{ij} = \min \left(t \,/\, \sum_{\theta=1}^{\theta=t} \overline{a}_i^{\theta} \geq e_{ij} \right)$$

where $\overline{a}_i^{\theta}$ denotes the upper bound of the domain of a_i^θ (if a_i^θ has been assigned, the domain is reduced to the singleton value $a_i^\theta = \overline{a}_i^{\theta} = \underline{a}_i^{\theta}$). Now any time $\theta < t_{ij}$ is a forbidden value for processing activity *j*. We can then force $a_j^\theta = 0$ for all times $\theta < t_{ij}$.

In the following example (see Figure 33), we schedule an activity *i* consuming e_i = 15 resource units. Vertical black lines represent the time window bounds and horizontal ones the current maximal intensity of *i*. The minimal intensity is supposed to be zero. No decision has been taken for scheduling *i* and *j*. Activities *i* and *j* are linked by an Energy-Precedence Constraint *EPC(i,j,10)*. The dotted vertical line represent the earliest starting time of *j*. In order to show the effect of a EPC between *i* and *j*, we have represented the earliest scheduling of *i* (diagonal lined rectangles):

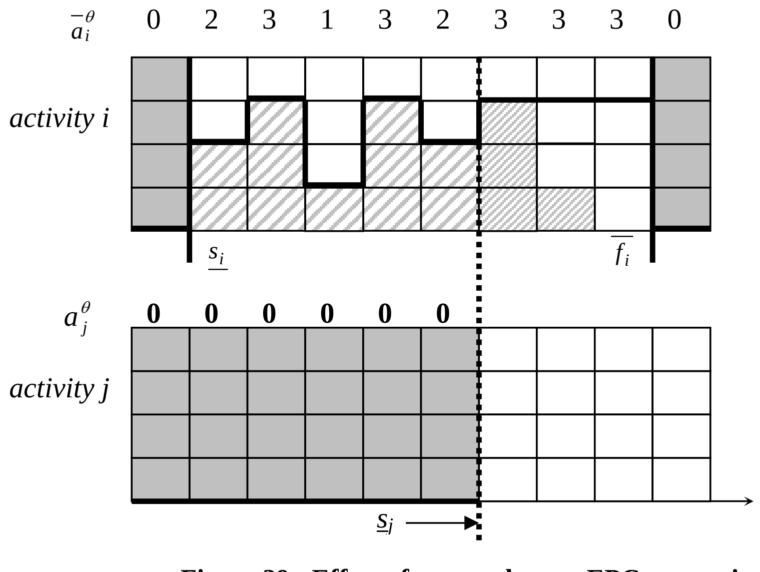


Figure 39 : Effect of a precede type EPC constraint on activities scheduling

The thick diagonal lined rectangles represent the resources units necessarily consumed before *j* starts. Let us notice that due to the discretised nature of time, in order to respect the Energy-Precedence Constraint, 11 resource units have been consumed by *i* before the period $\underline{s}_j$; the immediate previous period ($\underline{s}_j$-1) would have let only 9 resource units to be consumed by *i*, and do not satisfy the constraint. We can then state that a lower bound of s_j that respect the energy-precedence constraint is the period *t* such that:

$$\sum_{\theta=1}^{\theta=t-1} \overline{a}_i^{\theta} < e_{ij} \leq \sum_{\theta=1}^{\theta=t} \overline{a}_i^{\theta}$$

where $\overline{a}_i^{\theta}$ denotes the upper bound of the domain of a_i^{θ} (if a_i^{θ} has been assigned, the domain is reduced to the singleton value $a_i^{\theta} = \overline{a}_i^{\theta} = \underline{a}_i^{\theta}$).

This propagation is not obvious to implement, because no predefined built-in constraint of this type exists in a CLP language. To this end we use a so-called “suspended-goals” technique that links the EPC constraints to the domain reduction events for each constrained variable: each time a a_i^{θ} is instantiated or its upper bound reduced, we shall check the eventual EPC constraint (*i, j, e_{ij}*): first the bound *t* is updated and eventually some variables a_j^{θ} are forced to be null.

These properties suggest a first algorithm which computes the lower bound of s_j and propagate it on a_j^{θ}.

Algorithm 1 : Propagating the energy-precedence constraint *precedes*(i,j,e_{ij}) on *j*

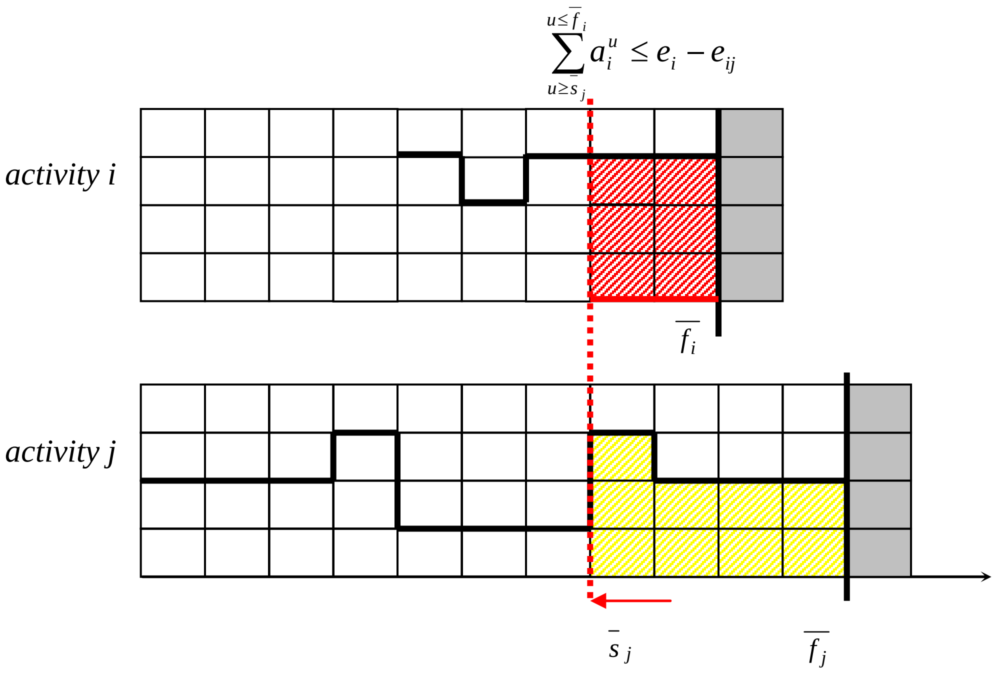
Input : $\underline{s_i}$, $\underline{s_j}$, $\{ \overline{a}_i^{\theta}\ \forall\theta = \underline{s_i} .. \overline{s_i} \}$, e_{ij}

Output : a new constraint on s_j, expressed as new constraints $a_j^{\theta} = 0$ for some periods *u*.

```
1 $e_{bef} \leftarrow 0$
2 $t \leftarrow \underline{s_i}$
3 while $e_{bef} < e_{ij}$ do
4       $e_{bef} \leftarrow e_{bef} + \overline{a}_i^{\theta}$
5       $t \leftarrow t + 1$
6 endwhile
7 if $t > \underline{s_j}$ then
8       for $u$ in $\underline{s_j} .. t$ do
9             $a_j^{\theta} \leftarrow 0$
```

```
10      endfor
12      $\overline{s_j} \leftarrow t$
13 endif
```

The same *precedes*(*i,j,e*$_{ij}$) constraint can also affect the scheduling of activity *i*. In order to ensure that at least e_{ij} units of resource must be consumed in processing *i* before *j* starts, at most (e_i- e_{ij}) remain available for activity *i* since *j* starts. It implies that if $\overline{s_j}$ is the latest starting time of *j*, at most (e_i- e_{ij}) units of resource are available for *i* during the interval [$\overline{s_j}$, $\overline{f_i}$] :



$\sum_{u \ge \overline{s_j}}^{u \le \overline{f_i}} a_i^u \le e_i - e_{ij}$
activity i
$\overline{f_i}$
activity j
$\overline{s_j}$
$\overline{f_j}$

Figure 40 : Effect of a follows type EPC constraint on activities scheduling

These properties suggest now a second algorithm which computes an upper bound *t* of s_j and propagate it as a new constraint that limits the sum of the last consecutive terms $a_i(\theta)$.

```
Algorithm 2 : Propagating the energy-precedence constraint precede(i,j,e_ij) on i
        Input : $\overline{f_i}$ , $\overline{f_j}$ , { $\overline{a}_j^{\theta}$  $\forall \theta = \overline{s}_j .. \overline{f_j}$ }, $e_{ij}$
        Output : new constraints on some terms $a_i(u)$ expressed as $\sum_{u=s_j}^{\overline{f_i}} a_i(u) \le e_i - e_{ij}$
 1 $t \leftarrow \overline{f_j}$
```

```
2 e ← $\overline{a}_j(t)$
3 while e < $e_j$ do
4       $t \leftarrow t - 1$
5       e ← e + $\overline{a}_j(t)$
7 endwhile
8 if $t \leq \overline{f_i}$ then
9       set the newconstraint $\sum_{u=t}^{\overline{f}_i} a_i(u) \leq e_i - e_{ij}$
10 endif
```

4.3.4.4.2 Follows EPC type

We will refer to "Follows" type EPC when dealing with the minimal work that has to be done on activity *j* once activity *i* is finished; this mandatory remaining work is defined by the customer of the data.

In this case we seek to compute the latest time t_{ij} from which activity *j* can still consume the energy e^*_{ij}, which is the minimum energy allocated after *i* ends.

$t_{ij} = t$ such that : $\sum_{\theta=t+1}^{\theta=f_j} \overline{a}_i^{\theta} < e^*_{ij}$ and $\sum_{\theta=t}^{\theta=f_j} \overline{a}_i^{\theta} \geq e^*_{ij}$

Consequently, any time $\theta > t_{ij}$ is a forbidden value for processing activity *i*. We can then force $a_i^{\theta} = 0$ for all times $\theta > t_{ij}$.

These properties suggests a third algorithm which computes the upper bound of f_i and propagate it on a_i^{θ}.

Algorithm 3 : Propagating the energy-precedence constraint *follows*$(i,j,\ e^*_{ij})$ on *i*

Input : $\overline{f_i}$, $\overline{f_j}$, $\{\overline{a}_j^{\theta}\ \forall \theta = \underline{f_j} .. \overline{f_j}\}$, e^*_{ij}

Output : a new constraint on f_i, expressed as new constraints $a_i^u = 0$ for some periods *u*.

```
1 $e_{aft}$ ← 0
2 $t \leftarrow \overline{f_j}$
3 while $e_{bef}$ < $e_{ij}$ do
4       $e_{aft}$ ← $e_{aft} + \overline{a}_j^{\theta}$
5       $t \leftarrow t - 1$
6 endwhile
7 if $t < \overline{f_i}$ then
8       for $u$ in $t..\overline{f_i}$ do
```

```
 9            $a_i^u \leftarrow 0$
10      endfor
12      $\overline{f_j} \leftarrow t$
13 endif
```

Lastly, the *follows*(*i,j,* e_{ij}^*) constraint can also affect the scheduling of activity *j*. In order to ensure that at least e_{ij} units of resource must be consumed in processing *j* after *i* ends, at most (e_j- e_{ij}) remain available for activity *j* before *i* finishes. It implies that if $\underline{f_i}$ is the earliest end time of *i*, at most (e_i- e_{ij}) units of resource are available for *j* during the interval [$\underline{s_j}$, $\underline{f_i}$]:

Which suggest the fourth and last algorithm related to EPC which computes a lower bound *t* of f_i and propagates it as a new constraint that limits the sum of the first consecutive terms $a_j(\theta)$.

Algorithm 4 : Propagating the energy-precedence constraint *follows*(*i,j,* e_{ij}^*) on *j*

Input : $\underline{s}_j$, $\underline{s}_i$, {$\overline{a}_i^\theta$ $\forall\theta=\underline{s}_i .. \underline{f}_i$}, e_{ij}^*

Output : new constraints on some terms $a_j(u)$ expressed as $\sum_{u=\underline{s_j}}^{f_i} a_j^u \le e_i - e_{ij}$

```
1 $t \leftarrow \underline{s}_i$
2 e $\leftarrow \overline{a}_i(t)$
3 while e < $e_i$ do
4       $t \leftarrow t + 1$
5       e $\leftarrow$ e + $\overline{a}_i^t$
7 endwhile
8 if $t \ge \underline{s}_j$ then
9       set the newconstraint $\sum_{u=\underline{s}_j}^{f_i} a_j^u \le e_i - e_{ij}$
10 endif
```

4.3.4.5 Propagating Contract Dependencies Constraints (CDCs)

We define a Contract Dependency Constraint (*CDC*$_{ij}$) with the data {t_{ij}, e_{ij}} is a constraint stating that a given part of an activity *i* (team *i*) must have been processed before a fixed date t_{12} at which the dependant activity *j* (team *j*) can start.

For the activity *i* we have $\sum_{\theta=1}^{\theta=t_{ij}} a_i^\theta = e_{ij}$ and $\sum_{\theta=t_{ij}+1}^{\theta=H} a_i^\theta = e_i - e_{ij}$

For the activity *j* the dependency is translated into an earliest starting-time constraint:

$a_j^\theta = 0 \;\; \forall \theta \in [1..t_{ij}]$

Example:

Two activities {1,2} from two different design teams, with a respective energy {7,3} are constrained by the delivery of a data during the period θ=6. This data requires 3 units of energy to be expended by activity 1 in order to achieve the maturity level required by activity 2. To restrict the complexity of the example, we suppose the maximal intensity of both tasks is fixed to 1.

Two type of actions are performed in the problem solving process related to this constraints:

On the one hand the periods {1,2,3,4,5,6} of activity 2 are initialised to zero, since this activity can not begin without the data that will be delivered in period θ=6.

Table 12; Example of CDC propagation

a_2^θ	0	0	0	0	0	0	?	?	?	?
θ	1	2	3	4	5	6	7	8	9	10

Therefore two possible solutions are available for activity 2 if pre-emption is not allowed:

Table 13: Solutions for the CDC example

a_2^θ (sol1)	0	0	0	0	0	0	1	1	1	0
a_2^θ (sol2)	0	0	0	0	0	0	0	1	1	1
θ	1	2	3	4	5	6	7	8	9	10

On the other hand activity 1 shall perform 7 units of energy, with at least 3 units before period θ=6.

If the maximal intensity is relaxed to 2 units of resource and if intensity for periods {1,2,3,4} are zero (due to some other constraints), then propagating the Contract Dependency Constraint (*CDC₁₂*) lead to the remaining domains for variables $\{a_1^\theta\}$:

Table 14: Solution for the CDC example after relaxation of maximal intensity

$\overline{a}_1^\theta$	0	0	0	0	2	2	2	2	2	2
$\underline{a}_1^\theta$	0	0	0	0	1	1	0	0	0	0
θ	1	2	3	4	5	6	7	8	9	10

4.4 Tactical level scheduling with quality gates

Based on the model introduced in the precedent chapter, we have implemented the different constraints using the Constraint Logic Programming (CLP) environment ECLiPSe. The result is a tool enabling the allocation of resources based on the allocation of energy packages. This tool is dedicated to managerial levels where workload needs to be managed in a dynamic environment taking into account internal and external constraints.

The project management is generally based on a quality gates development. From a scheduling point of view, the scope of problem is reduced significantly since sub-problems are smaller. Quality gates allow taking into account interfaces with interdependent subsystems by defining some common quality gates related to several subsystems. This kind of common quality gates allow the correct interrelation between several subsystems and the fulfilling of the constraints linked to the interface. Typically phases of 20-30 weeks long are created with the duty of releasing a defined deliverable at the end of this phase. Fixing quality gates and time horizon are considered as external constraints for activities, since they can not be relaxed by team managers.

The second external constraint is related to the maximum resource capacity, which is given for each phase. This global resource allocation let the responsible some degrees of freedom to locate tasks in time.

Therefore, since there do not exist activity progress reviews, managers are asked to anticipate the lack of maturity of the deliverable related to the quality gate and to promptly confirm without delays the quality gate review and if assigned resources are enough for the completion of the deliverable. For this reason the consistency of the whole set of constraints must be evaluated as soon as possible. The main advantage of a constraint-based solver is to characterize the problem consistency by taking into account the most complete set of constraints (internal and external constraints, time and resource constraints).

Before each phase starts, the described tool can be used to make a first estimation of required resources. This is a task usually performed using information and knowledge extracted from former programs experiences. Nevertheless, it usually takes into account only resources needed to perform internal activities without dealing with the contracts that will need to be respected between two quality gates.

The relationship between workload and contract time position has been studied by the author in a preliminary work [LIZARRALDE, et al. '06a]. Once a first estimation is done and a first set of resources are allocated, the main capability of this tool is to support the manager of these teams in order to reallocate resources in a dynamic environment, anticipating the commitment linked to the quality gate and to the contracts between two quality gates. This includes a support to trade offs between the demand of new resources and the deterioration of the quality of deliverables linked to the quality gate or to the contracts. This is the reason why we consider this tool as a Decision Support System.

In the literature of standard resource-constrained project scheduling problems, the objective is usually to find an optimal schedule that minimizes the makespan or the maximum lateness, with the help of a black-box one-step solving algorithm. In our case we prefer to provide a decision making support that helps the user to build iteratively a feasible solution that satisfies all the constraints, and negotiate some of them if they can not all be satisfied. This solution will stress the available time margins and tight periods. We know that in many cases the problem is over-constrained (no schedule can satisfy the whole set of constraints), and there is a need for a customisable solving strategy in which the expertise of the decider may be exploited by taking into account some hierarchy of constraints to be relaxed (e.g. outsourcing or hiring new personnel, renegotiating contracts or modifying quality gates).

Considering that the hierarchy of constraint is built taking into account four types of constraints (Manpower, QG, Task precedence and Interdependence), the user can order them in ascending order of importance. Relaxing constraints must be in coherence with the actions that shall be taken afterwards. Moreover, the validation of the relaxation does not depend only on the head of the team but on managerial higher levels as partners involved in a contract. The simulations that can be done relaxing different constraints allow the user define different possible solutions and define accurate demands avoiding the classical claim “I need more resources”.

For a clearer understanding, this process can be shown as follows:

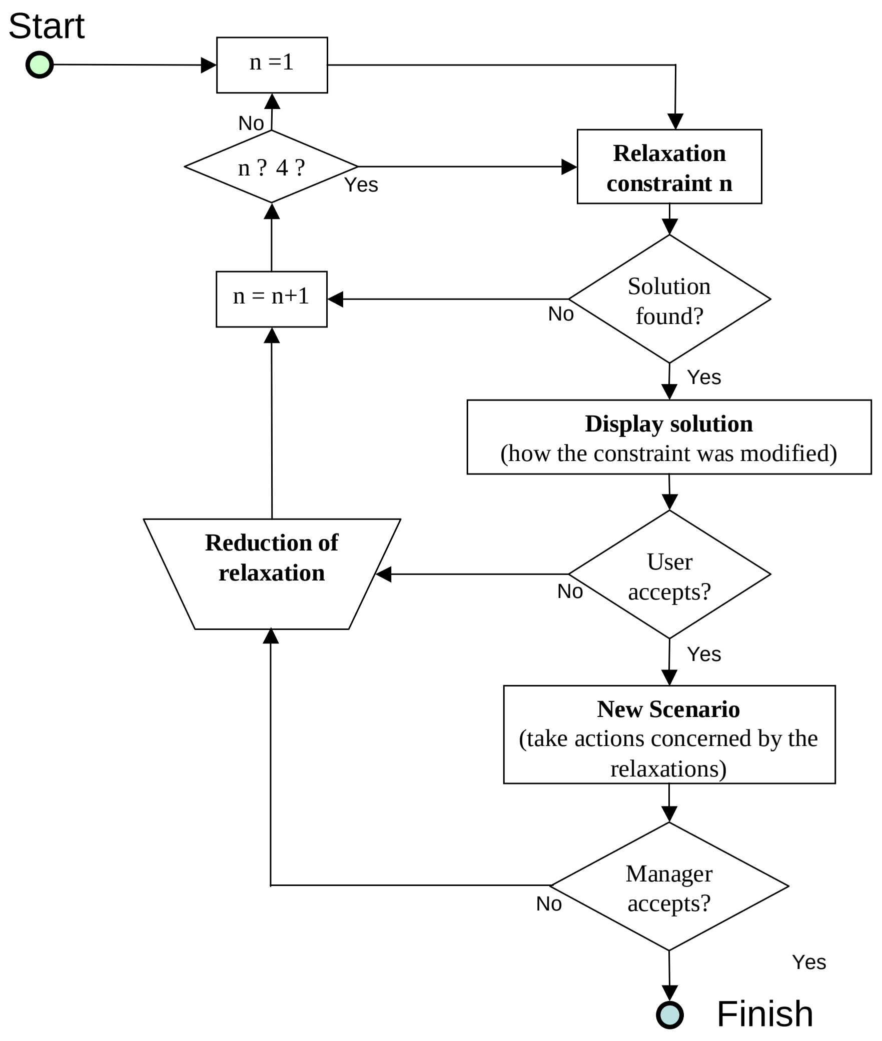


Figure 41: Constraints relaxation process

Relaxing first constraint in the list until a consistent solution can bring to non realistic solutions that must be accepted by the user before going on with the process. Once a feasible solution is defined, the head of the team will undertake the actions concerning the relaxed constraints and in the same way he or she will make an accurate demand to the concerned manager. If this manager agrees the modifications, the new schedule will be set up as the new baseline schedule. If not, the foreseen relaxation has to be reduced until the acceptance level and restart the process by relaxing the next constraint in the list until a feasible schedule arises.

The aim is to find a worthy balance between time constraints and resource constraints from an internal point of view but also from a systemic point of view (e.g. respecting contracts with external teams).

Accurate examples dealing with this trade offs in over-constrained context and customisable solving strategy examples will be provided in the next part.

4.5 Scenario based scheduling for uncertainties management

As we have explained in the 3.4.2 chapter, there are basically two methods to deal with the design activities scheduling uncertainties . On the one hand, the reactive methods and on the other hand the proactive methods. In this chapter we will develop a proactive method based on the scenario management.

Our model is efficient from a reactive point of view. If an unforeseen event happens, our model is not defined for including automatically a new task in the schedule. Indeed, we estimated that focusing on the constraints of the design process is more efficient for this dynamic environment [REPENNING '01]. Therefore, if an unforeseen event occurs, the main task will be to check if the new schedule remains consistent and respect all constraints. Reactive procedures need to asses in a short period different scenarii. The following methodology is then also usable for reactive scheduling processes.

In order to exploit efficiently this proactive method it is necessary to define a process to manage the scenarii. This process includes the scenarii generation that can be realised at the beginning of the project. This exercise will enable choosing the baseline schedule in order to steer the progress of the project. Moreover, other scenarii are saved. Later, if important modifications occur in the project running and if the baseline needs to be adjusted, scenarri saved at the beginning can be used to replace the existing baseline. Choosing among the saved scenarii can be realised comparing real progress with energy units allocated in each of the scenarii.

4.5.1 Definition of a scenario

Schoemaker [SCHOEMAKER '95] defines scenario planning as a “disciplined method for imagining possible futures”. Usually, “scenario planning” methods are defined for strategic level utilisation. They can include an important variety of quantitative and qualitative information [GOODWIN and WRIGHT '01]. But their use at tactical and operational levels is avoided due to the complexity of these types of methods [AHMED, et al. '03]. A scenario is not just a kind of story describing different futures. In order to define, understand and anticipate risk for the project, scenario management must be structured in a rigorous framework. This framework includes a model to generate different scenario and to evaluate them. It can also support the possibility of aggregating two or more scenarios. Lastly, it can include capabilities such as saving, updating and deleting the scenarios.

4.5.2 Scenario management process

4.5.2.1 Scenario generation

In our case, we focus on scenarios built for the tactical level. In our model scenarii are basically schedules with several ways to allocate resources, which leads to different time slots assigned to activities.

We can distinguish three methods to generate scenarios based on our model.

The first one is creating schedules as described in the 4.4 chapter. Based on solutions built iteratively with the user or heuristic solutions, each scenario is built from scratch.

The second method is based on the modification of one or several constraints. We have named this method the “what if” method since the user reviews each constraint usually imagining worst futures. In other words, “what if” method begins from an existing scenario and realises modification on constraints that can include the reduction of available resources or the violation of an interdependency

In the third method, the user inverts the process in order to create new scenarios focusing on a desired solution. Indeed, the user defines a value for each variable relaxing some constraints. This method is called a goal-seek analysis due to the fact that the user can build a schedule respecting all its goals but without taking into account some external constraints. This type of analysis is usually used when no resources are allocated to a design team. In this case, design team manager defines an accurate schedule and defines the amount of resources necessary for the completion of activities.

Finally, new scenario can be generated aggregating existing scenarii. Most used aggregations concern two or more scenarii built following the “what if” method.

4.5.2.2 Scenario evaluation

Using the same model as for different scenarii generation, including identical variables and constraints, implies that scenario can be compared to each other. The comparison can be done by contrasting the resources allocation graphes and the activities positioning. Moreover, two or more scenarii can be compared regarding how each constraint is respected. This point includes the comparison of margins related to a constraint, or the relaxation level if one or some constraints have been relaxed. In other words, the comparison depends on the evaluation method used to assess each scenario.

The satisfaction level of each constraint is an efficient way to evaluate a scenario. For example, if a solution is built with a given resource constraint relaxation, the variables that define this relaxation include:

- The overall extra energy quantity that is used by the scenario.
- The number of time periods in which the constraint is violated.
- The maximum quantity of resources added during a period.

Therefore, when comparing two scenarii, one could estimate that a scenario that uses less extra energy is a better scenario. Nevertheless, the evaluation of the other variables can demonstrate that it is also a less realistic scenario since it uses a high number of extra resources over a short period of time.

This type of evaluation can be considered as an explicit assessment. Other way to evaluate scenario includes risk analysis which is realised following the estimation of the user.

Risk associated to each scenario includes occurrence likelihood as well as an impact factor. A combination of both factors allows the user evaluate each scenario and make comparisons between different scenarios as well as to order hierarchically a set of scenarios.

Lastly, scenario can be evaluated through sensitivity analysis. The goal of this assessment method is to modify one or several variables (allocated energy in each period) and analyse the impact on different constraints. This type of analysis is realised in order to measure the robustness of each scenario. Robust scenario does not violate any constraint if minor modifications affect variables.

4.6 Contracts management

4.6.1 Definitions

Contracts between design teams can be observed through deliverables exchanges, which are often subject to negotiation. Therefore, contract management can be also defined as interfaces management, deliverables management, dependencies management or interdependencies management. A contract between two or more teams includes a data description and a date of delivering.

As we have already seen, data can be related to models, drawings, mock-ups, requirements specification document, calculation results, sketches, test results, etc. Data is generated by the supplier in order to answer to a specific requirement of a customer. Other cases include data produced by a specific domain that will be used by the customer as a design constraints to be respected. This is the case of the loads that will be a necessary input for the person in charge to calculate the size of the components. Hence, this size will be also a data for the designer that virtually assembles different parts.

Data is provided by the supplier as an output of a design activity and will be used by the customer as an input for its own design activity as described in Figure 42:

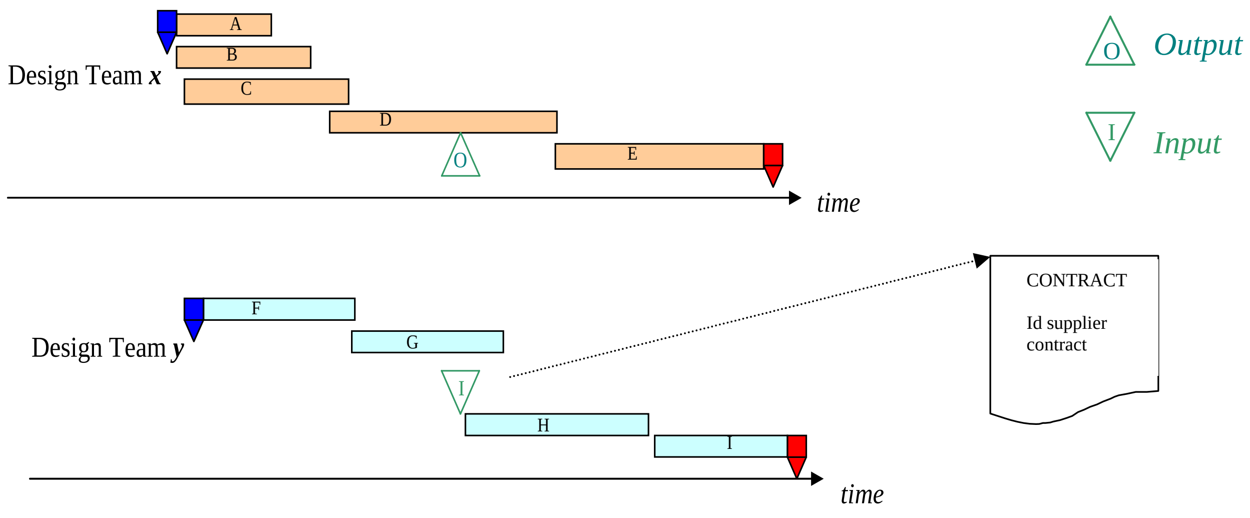


Figure 42: Contract definition: Input and output

Design teams and dependencies form a network (nodes = teams, edges = information flows). Traversing such a network enables to understand the constraints/contracts established by a design team on another.

Constraints propagation related to contracts becomes crucial in schedules modifications management. Each time a design team modifies its own schedule, especially after an unforeseen event, the information should be transferred to other teams it has some dependencies with. In order to anticipate changes in dependencies, a close link between dependencies management methods and schedules should be established. In Figure 43, we describe the process of a contract negotiation. Description of the data includes the type of the data, the maturity level and the needed date:

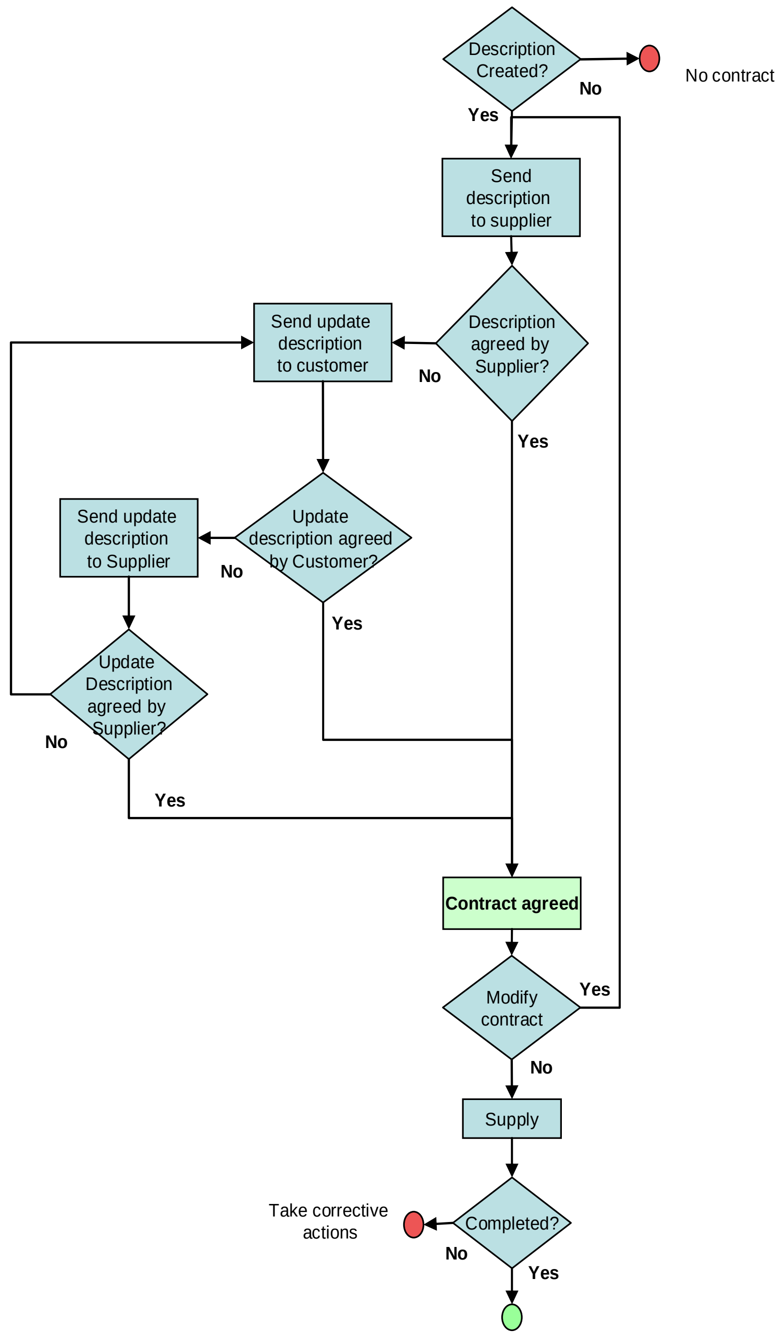

Description Created?
No
No contract
Yes
Send description to supplier
Description agreed by Supplier?
No
Send update description to customer
Yes
Update description agreed by Customer?
No
Send update description to Supplier
Yes
Update Description agreed by Supplier?
No
Yes
Contract agreed
Modify contract
Yes
No
Supply
Completed?
No
Take corrective actions
Yes

Figure 43: The process of a contract negotiation

First, the customer completes the description of the required deliverable, indicating the delivery date and requested maturity level. After having received this request, the supplier can either accept it or decide to renegotiate some aspects (usually, the delivery date or the maturity). If the supplier renegotiates the contract, his alternative proposal will then be sent to the customer, who can also choose to accept or renegotiate them. This process continues until the contract is accepted by both parties. Once the contract is accepted and signed, if no problem is detected, the supplier will deliver the requested data on the negotiated date. Nevertheless, due to numerous unexpected events or simple due to a wrong estimation of the work needed to perform the data, the supplier is very often not able to accomplish the contract. In this case the best picture is when supplier anticipates and he or she asks to renegotiate the contract. Later on we will describe the case when the contract needs to be negotiated.

In the worse case, the customer will no be notified and he or she will realize that the data will not be delivered on time, the day when it was supposed to be delivered. Lastly, we have also identified the case when data is delivered but the customer considers that it is not complete. These cases are usually source of conflicts since the contract can be interpreted in different ways if it has not been written correctly.

4.6.2 Energy and maturity

4.6.2.1 The concept of maturity of data

The concept of maturity of data is used in this work to represent the evolution of the data version, from a rough draft until the absolute maturity level where data is not supposed to be modified anymore.

In concurrent engineering, the customer accepts to begin its task with preliminary information, which corresponds to a data with an intermediary maturity level. From a supplier point of view, characteristics of data evolve and get closer to final data. From a customer point of view, reliability of provided data will increase at the same time as supplier's design activity progress. Generally, the likelihood of modifying a data decreases and its maturity increases until the completion of the design activity.

4.6.2.2 Relation between energy and maturity

For a given maturity level, energy needed to reach this level can be calculated, which corresponds to the work needed to be done so that the data gets the suitable maturity level. We can therefore define a function that relates the maturity level of the data and the energy expended:

$$e_u = f(u_d)$$

This relationship is not necessarily linear (case IV in Figure 44). Even if in many cases the maturity will increase at the same rate of the expended work; in other cases a very few work is necessary to deliver a data close to the absolute maturity (case I in Figure 44). In the opposed case, even if much work has been done, supplier can be in the case where no confident data can be delivered (case II in Figure 44). These differences depend basically on the nature of the design activity. Indeed, a design team that re-uses concepts from former project is able to supply data will high level of maturity without expending much energy. On the contrary a very innovative development needs high quantity of energy to fix the main concept of the subsystem. Until this point, very little information and not very confident can be delivered by the supplier.

In the different cases we have studied, the most frequent case is the so called "s curve" (case III in Figure 44) where maturity of the data evolves slowly for the first expended work units and suddenly it increases significantly with a little amount of energy; later, it goes on increasing smoothly until the absolute maturity level. This case reflects the lack of available data at the beginning of the project when energy is mostly used to correctly identify the design requirements, list the possible solutions, etc. The maturity increase period represents the period where design trade offs are concluded and even if it is not exactly the last accurate solutions, a concept is selected for the definition of the subsystem. These decisions will allow the supplier to be holder of data that is much more useful for the dependent sub-systems. Indeed, these subsystems will be able to begin the definition of the interface zone, being sure that even if some details might change, the global concept will remain stable.

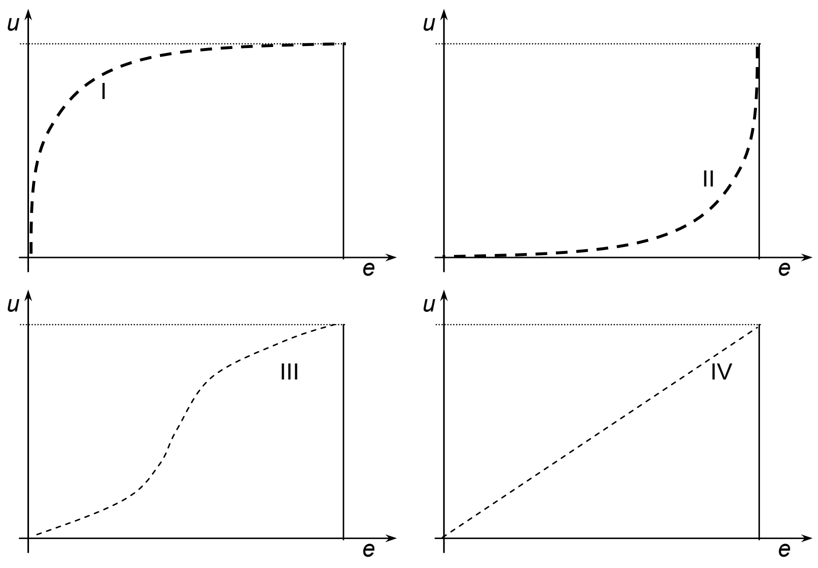


Figure 44: Functions relating energy and maturity level.

Basically, the supplier needs to define which type of function represents better the relation between the work he is doing and the maturity level progress. Moreover, he needs to estimate the work he will need to reach the maturity level asked by the customer. This amount of energy, as well as the delivery date agreed between both parts, is the foundations of the Contract Dependency Constraint (CDC_{ij}) described in former chapter: {t_{ij}, e_{ij}}

4.6.2.3 Consequences in scheduling

We consider two design teams which have performed a first schedule with respect to external constraints coming from the upper managerial level. The Figure 45 illustrates the case where a task *C* assigned to the design team *x* must deliver a data *d* with a minimal maturity level u_d to the design team *y* in order to start the task *F*.

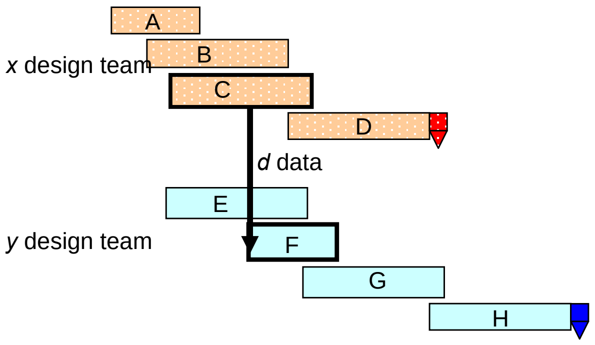


Figure 45: Dependency between x and y design teams.

We will note as follows:

C *is a task performed by* x:	$C \in Tasks(x)$
F *is a task performed by* y:	$F \in Tasks(y)$
d *is the output of* C:	$d = output(C)$
and is used by F:	$d \in Input(F)$

In the example of Figure 46, the assumption is made that first schedule realised by the design team x, assigned the following resources allocation to task *C* with a two-steps shape:

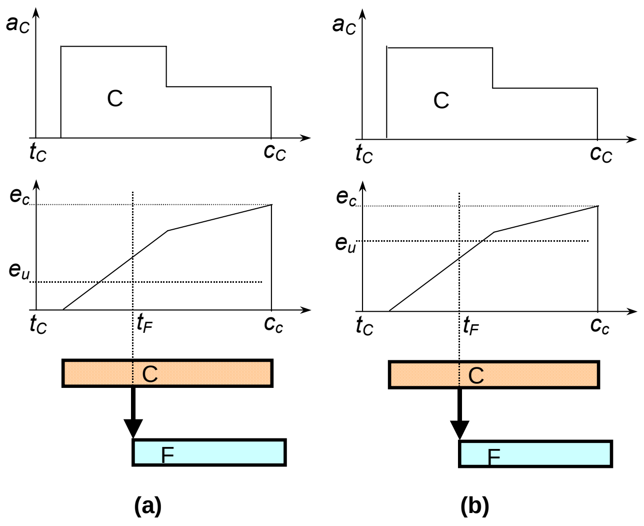


Figure 46: Energy needed to reach maturity level to begin F task.

Following the first schedule realised by x, two possible answers can be given by x to y after the demand of y.

Case *a*: the maturity level asked by y is lower than the maturity level forecasted by x at t_F:

$u_d^y(t_F) \leq u_d^x(t_F)$

Case *b*: the maturity level asked by y is higher than the maturity level forecasted by x at t_F:

$u_d^y(t_F) > u_d^x(t_F)$

In the last case, some constraints shall be relaxed by *x* in order to respect the demand of *y*.

Several ways of constraint relaxation may be explored in order to restore a schedule consistent with both *x* and *y*:

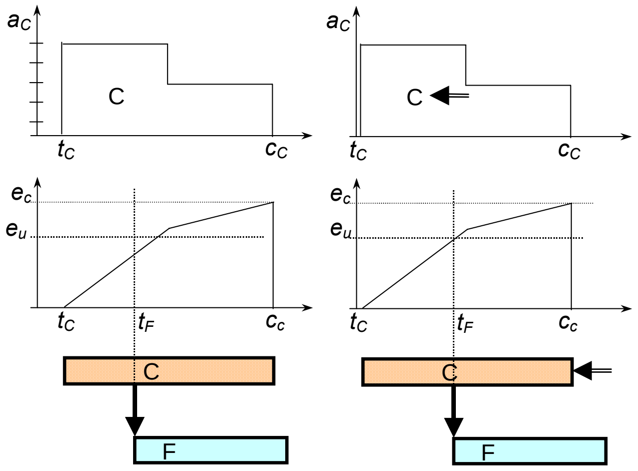


Figure 47: Reschedule C to respect maturity level to start task F

Reschedule *C* at time $t_C^{'} < t_C$ such that $u_d^x(t_F^{'}) = u_d^y(t_F)$: this will be possible if some temporal margin is available for *C*, which means that *C* is not a critical task.

Re-schedule tasks that constrain the starting time of *C*: this might cause new negotiations about the maturity of data they require.

Increase the manpower at the beginning of *C*.

Negotiate $u_d(t_F)$ with team *y*.

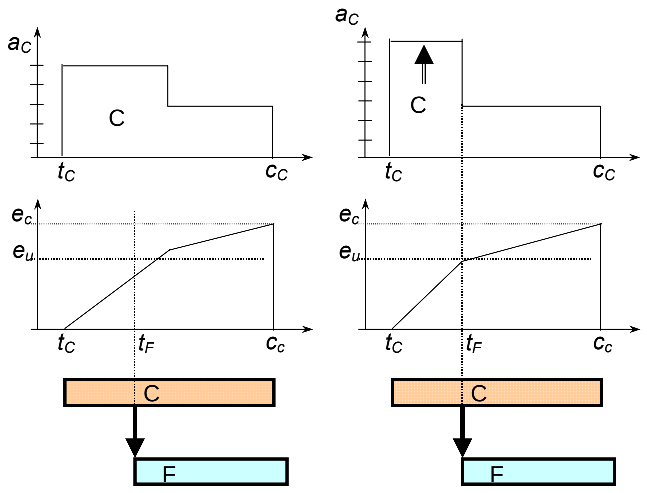


Figure 48: Increase the manpower at the beginning of C to respect maturity level needed to start task F.

Before focusing on manpower increase or target milestone relaxing, the dependency with *y* is negotiated and a trade-off should be performed.

In the former example we have considered only one demand realised to *x* design team and rescheduling is realised based on modifications needed by the concerned task.

In fact, several demands may be sent to the design team *x*, and if all these constraints can not be respected, it is necessary to define a strategy in order to identify the constraints that should be relaxed and the relaxing level.

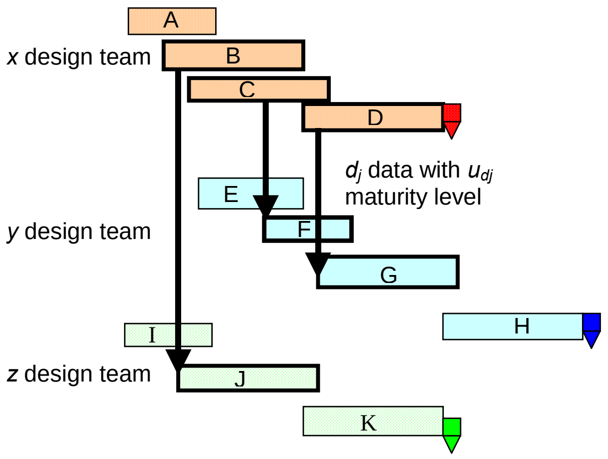


Figure 49 : Dependencies between *x* and *y* and *z* design teams

Several data can be asked to *x* team:d_j for j = 1,…, n

In the example of Figure 49 : Dependencies between *x* and *y* and *z* design teams, three tasks of *x* design team's schedule are asked to supply data: makes (d_j) = [B,C,D] for j = 1, 2, 3

For rescheduling problem we focus on the tasks that produce those data:

$$makes(d_j) \in Tasks(x)$$

A new variable is calculated taking into consideration the difference between the demanded maturity level and available maturity level.

$$M_j = max(0, u^y_{dj}(t_j) - u^x_{dj}(t_j))$$

Rescheduling of the tasks of team *x* can focus on minimising the sum of variables M_j (i.e. constraints related to the demands will be violated equally).

$$min \sum_{j=1}^{n} M_j$$

Design team *x* can define different priorities for each demand. In this case, the variable M_j can be weighted. Therefore, dependencies with other designs teams will need to be negotiated.

4.6.2.4 Energy to finish the supplier's task

Until this point, we have considered the energy needed by the supplier in order to achieve the maturity level asked by the customer. Furthermore, another amount of energy will support the scheduling process, modelling a fact that we have identified in current data exchanges practices: the amount of energy related to the work that needs to do the customer from the reception of the data needed to end the task *F* until the finish date of the concerned task.

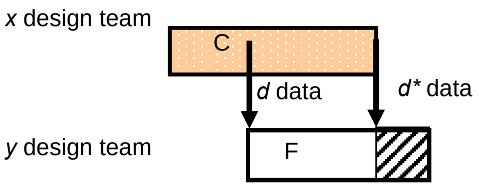


Figure 50: Energy to finish the supplier's task

On Figure 50 we can see a classical example, where a data *d* is necessary to begin the task *F* where $d \in Input(F)$, and a data *d** which is the final result of the task *C* where $d^* = output(C)$, and therefore data with absolute maturity level. Diagonal lined zone corresponds to the amount of energy necessary to finish task *F* from the reception of the final version of the data. Furthermore the necessary data in order to finalise the customer's task is not always the data with absolute maturity. Indeed, in many cases a lower maturity level will be enough in order to finish the customer's task. Moreover, it can also be possible that the absolute maturity of the data is achieved before the end of the task *C*. In this case, responsible of the task *C* might have identified actions related to data release procedure or capitalisation actions related to the data *d**. Therefore, on the contrary of the Energy-Precedence Constraint (EPC) case; the *d** data delivery does not need to be at the end of the supplier's task. The delivery of this data will be negotiated between both parts depending on the needs of the customer and the developing strategy of the supplier. The decided date will be a fixed date and not a relative milestone like in the case of the EPC.

4.6.3 Renegotiation of a contract

Renegotiation of a contract is the process where the customer or the supplier makes a special demand in order to modify a contract already signed. First negotiation of the contract is a common practice but is not treated with formal supports. To improve a collaboration based development, particularly in a dynamic environment, each partner should determine in advance that several deliveries will not be performed following the description of the first contract and inform the other as soon as possible in order to take corrective actions

In a renegotiation process the first contract is usually converted into two contracts. Considering the example of the Figure 51, the contract specifies that the data *d* shall be delivered the date t_F. If the design team *x* is not able to respect this contract; both parts can agree to convert the former contract into two contracts that will be less constrained for design team *x* ..

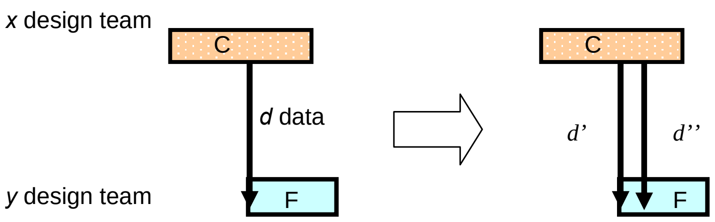


Figure 51: Renegotiation of a contract

Former contract includes the delivery of the *d* data with maturity level $u_d(t_F)$ and delivery date t_F, while new contracts include data *d'* and *d''*. Data d' will be characterised by maturity level $u_d^*(t_F)$ and delivery date t_F, while d'' will be characterised by maturity level $u_d(t_F^*)$ and delivery date t_F^*. Consequently:

$$u_d^* < u_d$$

$$t_F^* > t_F$$

This is a current Concurrent Engineering practice where data will be delivered in more that one step giving the possibility to the customer to follow the work with preliminary information [TERWIESCH, et al. '02].

Taking into account these new contracts, the constraints for our scheduling problem will be modified as follows. For a Contract Dependency Constraint *(CDC$_{ij}$)* defined by {t_{ij}, e_{ij}}, we consider e_{ij} is a function of the maturity demanded by the customer: $e_{ij} = f(u_d)$. Taking into account the resource allocation variable, we have defined the following formula for the activity *i* of the first design team: $\sum_1^{t_{ij}} a_i^\theta = e_{ij}$. Concerning the activity *j* of the second design team, we have initialised to zero all intensity variables linked to the activity *j* before the date t_{ij}.

Renegotiating this contract and defining two alternative deliveries will create two new CDCs that will substitute the former one:

CDC_{ij}^1 defined by {t_{ij}, e_{ij}^*}

CDC_{ij}^{2} defined by $\{t_{ij}^{*}, e_{ij}\}$

Nevertheless the alternative CDCs have not the same meaning and will not be interpreted in the same way in the scheduling tool.

Concerning CDC_{ij}^{1}, activity *i* of the first design team will now be constrained by $\sum_{1}^{t_{ij}} a_i^{\theta} = e_{ij}^{*}$, which will allow the user allocate less energy before this period. Concerning the activity *j* of the second design team, the procedure that initialises to zero all variables linked to the activity *j* before the date t_{ij} remains unchanged.

Concerning CDC_{ij}^{2}, the formula will be modified so that: $\sum_{1}^{t_{ij}^{*}} a_i^{\theta} = e_{ij}$, for activity *i* of the first design team. Moreover, no initialisation to zero will be performed for the activity *j* of the second design team as the activity has already begun.

Going further in the allocation of energy units to the activity *j*, the customer can estimate the amount of work that can be done between the first delivery date until the second one. Indeed, the first data delivery will allow beginning the task but after performing some work, the customer will be unable to go on with the work due to the necessity of more mature data. This amount of work is defined as $e_j^{**} = e_j^{t_{ij}..t_{ij}^{*}}$. This amount of energy will include a new constraint in our problem:

$$\sum_{t_{ij}}^{t_{ij}^{*}} a_j^{\theta} = e_{ij}^{**}$$

When the customer is not able to make a rescheduling in order to accept a renegotiation of a contract, it means that there is not any solution to respect the identified constraint from the supplier side and the customer side. In this case assumptions can be made in order to keep the schedule without any modification. The assumptions management process has been identified as a possible future research topic and it is described in the Annexe 1.

4.7 Conclusion

We have investigated the scheduling problems at the design stage using fully elastic activities with a defined energy quantity and considering the duration of activities and the allocation of resources as decision variables. Therefore our scheduling problem is transformed into an allocation problem.

The originality of this chapter resides in the description of the constraint propagation mechanisms that will be used for the resolution procedure. The implementation of this model is based on a Constraint Logic Programming (CLP) environment. CLP extends Logic Programming and provides a flexible and rigorous framework for solving CSP models. Beyond classical constraint propagation mechanisms, we have introduced the Energy-Precedence Constraints (EPCs), which is a new constraint type that models a partial precedence between activities based on the work quantity needed to define preliminary information. We have also proposed taking into account dependencies between design teams through Contract Dependency Constraints (CDCs). Indeed, our approach aims to facilitate cooperation in a complex managerial framework by enabling the propagation of scheduling constraints through different design team schedules. These two types of constraints reflect some practices that we have identified during the operational development of a new aircraft.

Based on simulations that are validated by our constraints propagation devices, our proposal supports the scenario creation for a proactive management of the uncertainties in the design process. Furthermore, these simulations support the steering process of engineering activities since it validates the consistency of the decisions offering a reference for the renegotiation of constraints when such renegotiation becomes necessary.

These new capabilities are the foundation of a Decision Support System which has been demonstrated through a prototype, described in the next chapter.

Chapter 5: The decision support prototype

5.1 Introduction

The former chapter has presented a CSP model of our design activities scheduling problem. This chapter describes now prototype based on this model and some functionalities it can offer to a design team manager

Firstly, we list the objectives of the prototype, then the functional requirements that have been specified, mainly together with actors involved in on-going aircraft development projects.

Then, we illustrate its architecture and we describe the capabilities of the different features of the prototype.

Finally, we explain the actions that have been performed in order to evaluate the model and solutions proposed during this research project. It includes a demonstration performed on a real use case and a demonstration of the constraints relaxing process based on an ad-hoc example.

5.2 Objectives of the prototype

The development of a prototype pursues the following goals:

- Illustrate the proposed concepts.
- Assess their foundation and their application on a real industrial case.
- Identify new concepts and required refinements to guarantee the development of a future application to be deployed operationally.

Beyond the scheduling tool, our goal has been to develop a global decision support system, dealing not only with scheduling and resources allocation aspects, but also with the collaboration aspects between different design teams in a horizontal level, as well as vertical collaboration between different managerial levels.

More precisely we have distinguished the functionalities that help a decision maker to solve its local design activities scheduling problem, and functionalities that help to negotiate or renegotiate constraint relaxations and trade offs with other development teams or with managers of upper decision levels.

5.2.1 Gathering functional and technical requirements

Requirements for the future application have been gathered in three types of environment during the course of the project:

- The Project Steering committee: This committee has been met each five months. The members of this committee includes project management skill responsible for three of the on-going Airbus projects under-development (A380, A400M and A350), Engineering and project management functions member in Airbus Central Entity and

Engineering experts form EADS-IW Research centre. The main goal of this committee has been to asses the project progress and to give some orientations. Most of the requirements have been gathered during the first meetings.

- Advisory committee: This committee has been met on request, in order to provide some expertise on specific issues. The members of this committee include experts on scheduling and product development.
- Operational contributors: In the first part of the project, different actors of the Airbus new aircraft development programs have been interviewed. These interviews have been one of the richest requirement sources at the beginning. Moreover, after the first illustrator presentation, some requirements update has been necessary.

We have identified 19 top functional requirements. The detailed list is given in Annexe 2.

Finally, 14 technical requirements have been specified after derivation from the functional requirements. These requirements concern the application to be developed. The detailed list of the technical requirements is given in Annexe 3.

5.2.2 Functional analysis for the prototype development

The main goal of the prototype is to support decision-making process related to engineering activities scheduling and resources allocation at each organisational and management levels of Airbus development programs.

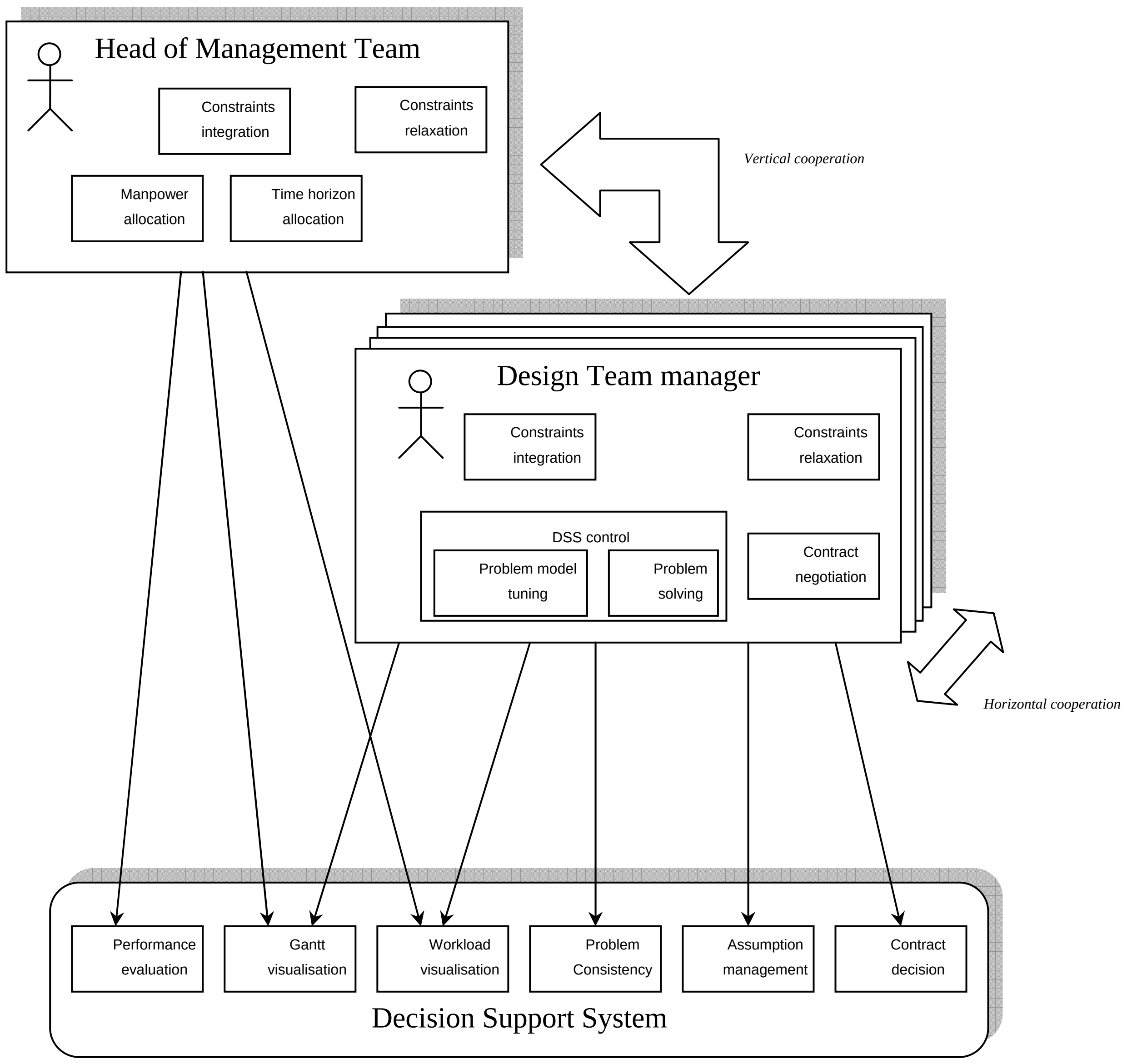


Figure 52: Mains functions of human actors and the DSS

Figure 52 summarises the main functions assumed by each actor (human or DSS).

This functional analysis has been done taking into account not only general functional requirements issued from the requirements gathering process, but also the problem approach that we have chosen.

The six major functions that shall support the future DSS have been the foundation of the prototype development project.

As we will see in the next section, we have not developed an integrated prototype but several parts of it and each of them has been used to evaluate one or more functions describes in this chapter.

We have begun our work focusing on functions related to the design team manager. Four major functions have been defined for this actor: **Constraints integrations, DSS control, constraints relaxation negotiations and contract negotiations.**

Constraints integration deals with the incorporation of the resources and deadlines that are defined by the head of the management team. Other constraints will be issued from the contract negotiation function. In order to respect these constraints, the design team manager allocates time and resources with through the DSS.

This is performed based on the DSS control function. This function includes two sub-functions. The first one is the problem model tuning which deals mainly with the constraints hierarchy settings. The second one is the problem solving control; this function guides the solutions research by setting some variables to user decided values.

The constraints relaxation negotiation function will deal with the renegotiation process with higher managerial level or other design teams.

Each of these functions will be supported by one or several DSS functions. For example, the DSS will not negotiate contracts, this is a function in the scope of the design team manager, but it will support this function based on contract decision management function as well as the problem consistency checking function.

5.3 Architecture and main capabilities of the realised prototype

We will present here above different parts of the prototype and additional tools developed in order to study the feasibility of the decision support tool.

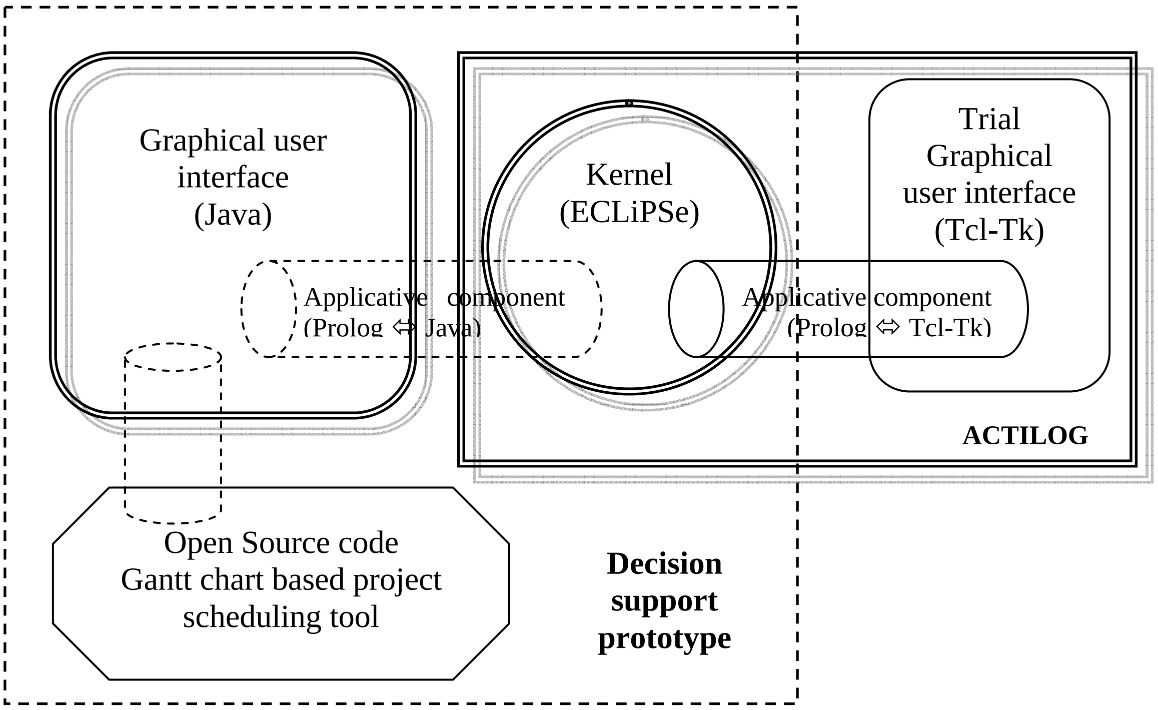

Graphical user interface (Java)
Kernel (ECLiPSe)
Trial Graphical user interface (Tcl-Tk)
Applicative component (Prolog ⇔ Java)
Applicative component (Prolog ⇔ Tcl-Tk)
ACTILOG
Open Source code Gantt chart based project scheduling tool
Decision support prototype

Figure 53: Prototype Architecture and its environment

The Kernel of the prototype has been developed in ECLiPSe. Several reasons has led to this choice. ECLiPSe is an extension of Prolog, a high-level logic programming language. Moreover ECLiPSe embeds constraint libraries that provide constraint propagation mechanisms. It is thus well-suited for a declarative programming of constraint satisfaction problems. Thanks to the usual Prolog programming advantages, the code is readable, concise and flexible. Prolog is known to be an efficient programming language for prototyping, because a large amount of produced code dedicated to the problem statement does not evolve frequently. Only the code part that implements particular solving strategies and/or scenarii needs to be updated.

A graphical user interface devoted only to the demonstration for potential users has been written in Java, in order to reuse the graphical libraries devoted to the development of advanced user interfaces. This interface features what could be the final graphical user interface of the decision support tool. It has been mainly used to explain the future prototype capabilities to the Airbus potential end users and to gather new requirements for the future application.

In order to study the feasibility of a prototype based on this architecture, we have developed a trial prototype that deals only with the scheduling part. With this prototype, named ACTILOG, the goal was not to develop all the foreseen capabilities but to make tests with an applicative component between the kernel and the graphical user interface. ACTILOG has been developed in ECLiPSe and Tcl-Tk. It links a kernel and a graphical user interface which acts as a problem modeller but also provides a control on the solution search. Actilog is currently operational and enables the user to launch problem solving sessions without being an expert of constraint programming. In the following chapter, we will describe more deeply each part of the prototypes and illustrator developed in of this research project.

The ECLIPSe-PROLOG kernel code implements the model described in the previous chapter. It is composed of several modules and separates the problem description and the solution searching strategy. Figure 54 describes the different files developed:

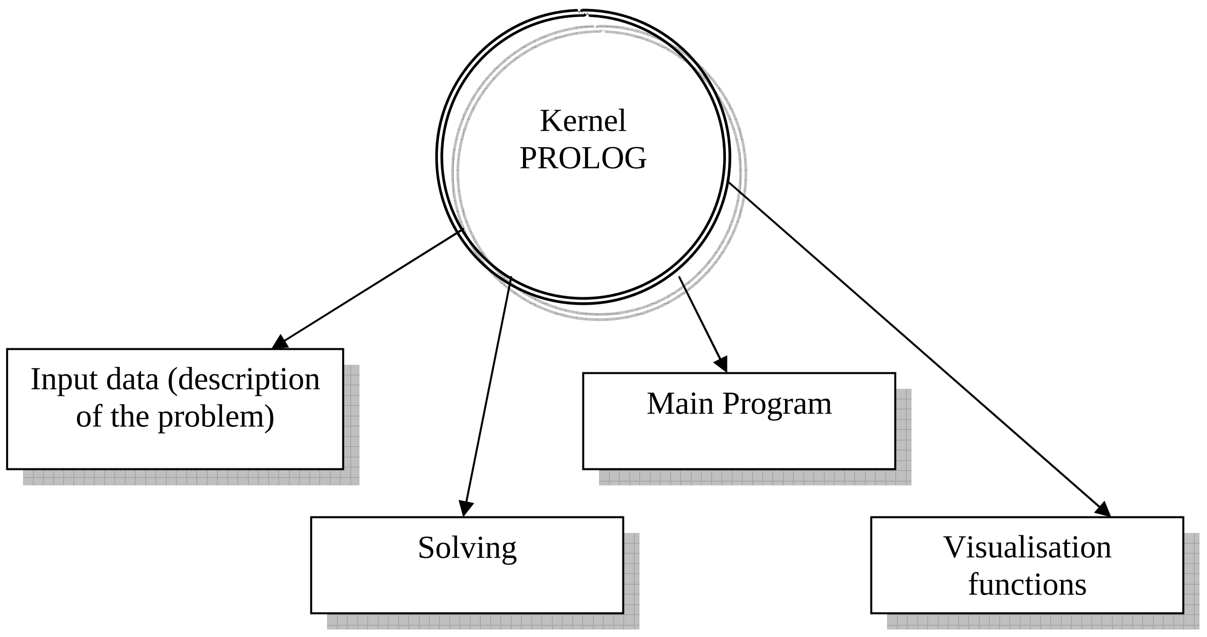


Figure 54: The ECLIPSe-PROLOG Kernel

The input data files describe several scheduling problems to be solved. The development of the resolution procedures was validated with simple scheduling problems for which one know he solution in advance..

It contains the description of the tasks, which includes the energies needed for task completion, the earliest start dates and the latest end dates. It also includes the available resources value for each period as well as the contracts and the description of the dependencies between tasks.

The core file of the system implemented the constraints propagation techniques and resolution strategy. The developed code that details these functions is presented in the Annexe 4.

Elementary visualisation functions were dedicated to trace the solving. They include procedures that display the domain of each decision variable as soon as propagation occurs. Then, the user can visualise the backtracks realised during the resolution process.

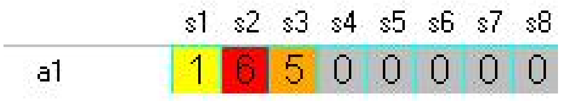
Finally, a main procedure controls the loading of input files and displays a solution.

5.3.1 ACTILOG

This prototype is not foreseen to be included in the future decision support tool; nevertheless it has allowed us to refine some technical requirements. The main goal of this prototype has been to develop a system that includes the different parts of the future application and therefore check the feasibility of this architecture. Therefore, the main requirements included the development of a part of the kernel, a simple graphical user interface and an application component between them.

ECLiPSe kernel of this prototype does not include all the constraint propagation techniques developed in the frame of this project but it deals with the main constraints.

Thanks to the link between ECLiPSe and Tcl-Tk, the applicative component between the kernel and the graphical user interface was easy to develop. Let us notice that the graphical user interface of ACTILOG includes an original way to present a task. It stresses the energy based approach developed in the frame of this project, as shown in the Figure 55.



	s1	s2	s3	s4	s5	s6	s7	s8
a1	1	6	5	0	0	0	0	0

Figure 55: Energy base approach in ACTILOG

The periods (*s*) of the scheduling problem are supposed to be weeks and for each activity (*a*) the allocated resources is displayed with a colour code. Therefore, we define a non rectangular task, stressing the periods with high quantity of resources allocated and maintaining the identification of the start and end dates.

Finally, the development of the application component has allowed us to define the data that shall be communicated between both applications at each step of the process (see Figure 56).

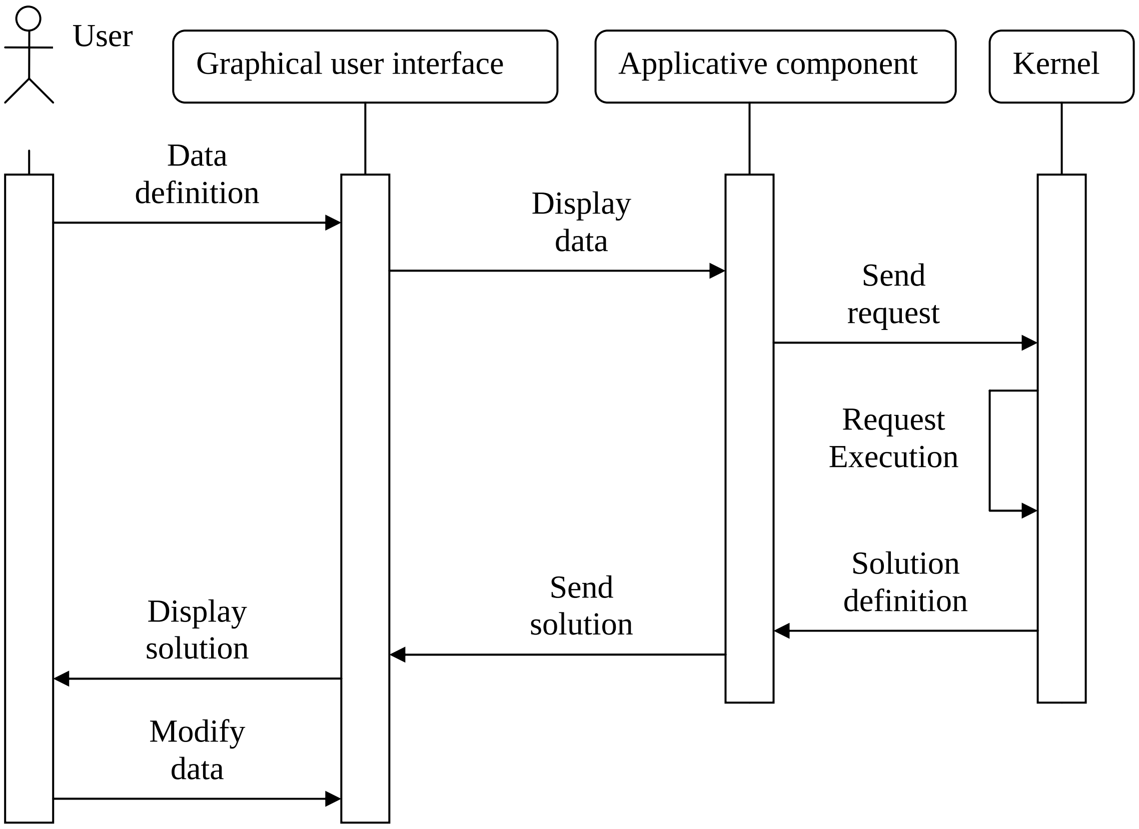
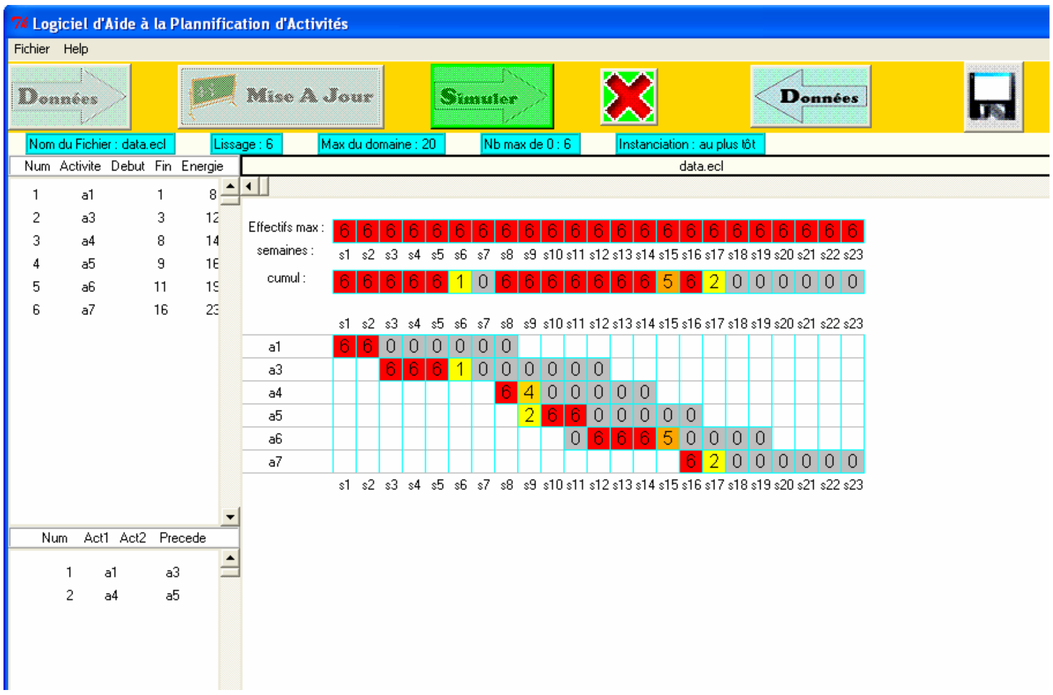


Figure 56: Communication between the different components of ACTILOG

The tool allows the user define a problem and obtain the results with the graphical user interface, without modifying the ECLIPSe-PROLOG code. On Figure 57, we show a screenshot of this application where a simple schedule is displayed.



Nom du Fichier : data.ecl | Lissage : 6 | Max du domaine : 20 | Nb max de 0 : 6 | Instanciation : au plus tôt

Num	Activite	Debut	Fin	Energie
1	a1	1	8	
2	a3	3	12	
3	a4	8	14	
4	a5	9	16	
5	a6	11	19	
6	a7	16	23	

Num	Act1	Act2	Precede
1	a1	a3	
2	a4	a5	

data.ecl

	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10	s11	s12	s13	s14	s15	s16	s17	s18	s19	s20	s21	s22	s23
Effectifs max :	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6	6
semaines :	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10	s11	s12	s13	s14	s15	s16	s17	s18	s19	s20	s21	s22	s23
cumul :	6	6	6	6	6	1	0	6	6	6	6	6	6	6	5	6	2	0	0	0	0	0	0

	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10	s11	s12	s13	s14	s15	s16	s17	s18	s19	s20	s21	s22	s23
a1	6	6	0	0	0	0	0	0															
a3			6	6	6	1	0	0	0	0	0	0											
a4								6	4	0	0	0	0	0									
a5									2	6	6	0	0	0	0	0							
a6											0	6	6	6	5	0	0	0	0				
a7																6	2	0	0	0	0	0	0
	s1	s2	s3	s4	s5	s6	s7	s8	s9	s10	s11	s12	s13	s14	s15	s16	s17	s18	s19	s20	s21	s22	s23

Figure 57 : Schedule displayed in ACTILOG.

The solution board is formed by a time window between two quality gates and the list of the activities to be scheduled. None coloured boxes are higher than latest end date or lower that earliest start date. On the top on the display we can notice the maximal capacity constraint for each week, as well as the allocated resources.

Once the prototype developed, we have not updated the entire system. Indeed, last functions developed in the ECLiPSe kernel are not available in this prototype. First, we have considered that the goals of this prototype were reached and secondly maintaining the overall system (Kernel+ Graphical user interface+ Applicative component) is time consuming. For that reason we decided to focus on the kernel development. Lastly, Tcl-Tk based graphical user interface has limited graphical capabilities and the final prototype is foreseen to be developed using JAVA. Therefore, the maintenance of this prototype was not our priority.

5.3.2 The decision support tool's illustrator

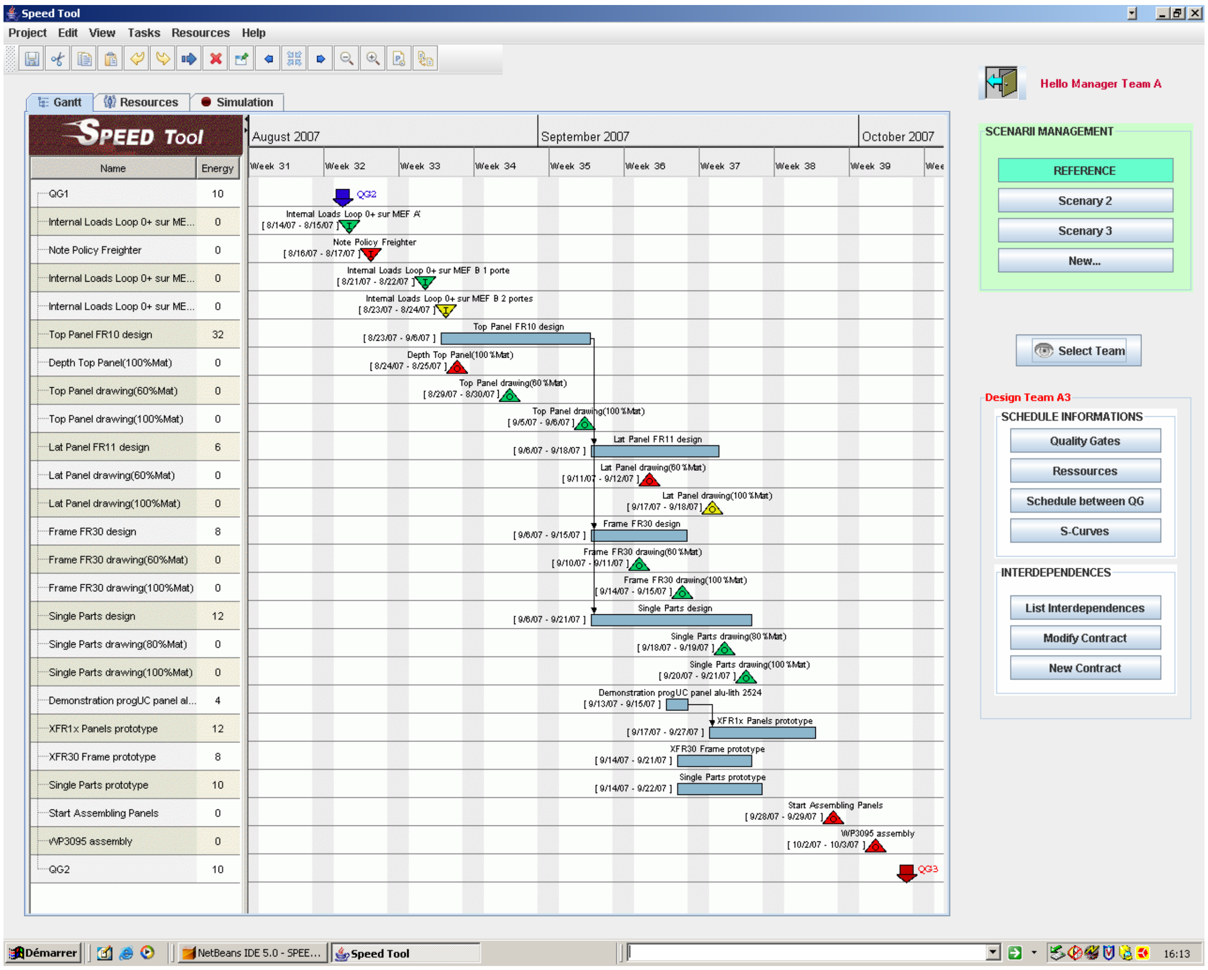
This illustrator presents the capabilities and the new possible practices available with the future application. The aim was to develop a friendly interface application to discuss with Airbus product development actors without showing mathematical background of the kernel. The illustrator has evolved taking into account the different suggestions of the actors that have been interviewed. It supports five different use cases:

- Scheduling
- Contract management
- Quality gates management
- Scheduling simulation based on constraints management
- Key Performance Indicators management
-

Moreover, it deals with the interfaces between these use cases. For example, contracts that have been negotiated between two actors are visible in both schedules.

5.3.2.1 Scheduling

Scheduling definition and their displays has not been defined as a priority in the development of this illustrator. On the one hand there are already existing tools that have tackled these aspects and we found that there was not added value developing a new interface based on a Gantt chart. On the other hand, the illustration of a graphical interface able to represent the resource allocation solution was already assigned to the ACTILOG prototype.


Speed Tool
Project Edit View Tasks Resources Help
Gantt Resources Simulation
SPEED Tool
August 2007
September 2007
October 2007
Week 31 Week 32 Week 33 Week 34 Week 35 Week 36 Week 37 Week 38 Week 39
QG2
Internal Loads Loop 0+ sur MEF A' [8/14/07 - 8/15/07]
Note Policy Freighter [8/16/07 - 8/17/07]
Internal Loads Loop 0+ sur MEF B 1 porte [8/21/07 - 8/22/07]
Internal Loads Loop 0+ sur MEF B 2 portes [8/23/07 - 8/24/07]
Top Panel FR10 design [8/23/07 - 9/6/07]
Depth Top Panel(100%Mat) [8/24/07 - 8/25/07]
Top Panel drawing(60%Mat) [8/29/07 - 8/30/07]
Top Panel drawing(100%Mat) [9/5/07 - 9/6/07]
Lat Panel FR11 design [9/6/07 - 9/18/07]
Lat Panel drawing(60%Mat) [9/11/07 - 9/12/07]
Lat Panel drawing(100%Mat) [9/17/07 - 9/18/07]
Frame FR30 design [9/6/07 - 9/15/07]
Frame FR30 drawing(60%Mat) [9/10/07 - 9/11/07]
Frame FR30 drawing(100%Mat) [9/14/07 - 9/15/07]
Single Parts design [9/6/07 - 9/21/07]
Single Parts drawing(80%Mat) [9/18/07 - 9/19/07]
Single Parts drawing(100%Mat) [9/20/07 - 9/21/07]
Demonstration progUC panel alu-lith 2524 [9/13/07 - 9/15/07]
XFR1x Panels prototype [9/17/07 - 9/27/07]
XFR30 Frame prototype [9/14/07 - 9/21/07]
Single Parts prototype [9/14/07 - 9/22/07]
Start Assembling Panels [9/28/07 - 9/29/07]
WP3095 assembly [10/2/07 - 10/3/07]
QG3
Hello Manager Team A
SCENARII MANAGEMENT
REFERENCE
Scenary 2
Scenary 3
New...
Select Team
Design Team A3
SCHEDULE INFORMATIONS
Quality Gates
Ressources
Schedule between QG
S-Curves
INTERDEPENDENCES
List Interdependences
Modify Contract
New Contract
Démarrer
NetBeans IDE 5.0 - SPEE...
Speed Tool
16:13

Name	Energy
QG1	10
Internal Loads Loop 0+ sur ME...	0
Note Policy Freighter	0
Internal Loads Loop 0+ sur ME...	0
Internal Loads Loop 0+ sur ME...	0
Top Panel FR10 design	32
Depth Top Panel(100%Mat)	0
Top Panel drawing(60%Mat)	0
Top Panel drawing(100%Mat)	0
Lat Panel FR11 design	6
Lat Panel drawing(60%Mat)	0
Lat Panel drawing(100%Mat)	0
Frame FR30 design	8
Frame FR30 drawing(60%Mat)	0
Frame FR30 drawing(100%Mat)	0
Single Parts design	12
Single Parts drawing(80%Mat)	0
Single Parts drawing(100%Mat)	0
Demonstration progUC panel al...	4
XFR1x Panels prototype	12
XFR30 Frame prototype	8
Single Parts prototype	10
Start Assembling Panels	0
WP3095 assembly	0
QG2	10

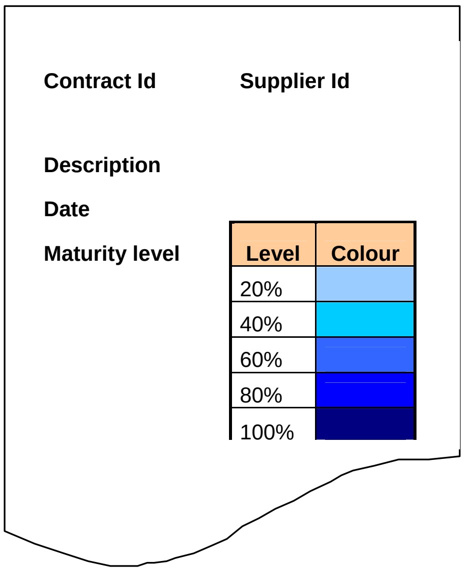
Figure 58: Prototype Main Interface

Figure 58 shows a screenshot of the schedule of a subsystem during eight weeks. Notice that the Gantt Chart displays not only the energy based activities to be performed, but also the quality gates milestones as well as the different input and outputs related to the contracts linked to the development of the subsystem. The colour code for each contract is the result of the identified risk for each of the contract as shown hereafter.

5.3.2.2 Contract management

A Contract management use case supports the negotiation process between two or more teams, the risk management process related to contracts and the contract renegotiation process.

A Contract definition begins by a data demand application (see Figure 59) completed by the customer.



Contract Id Supplier Id

Description

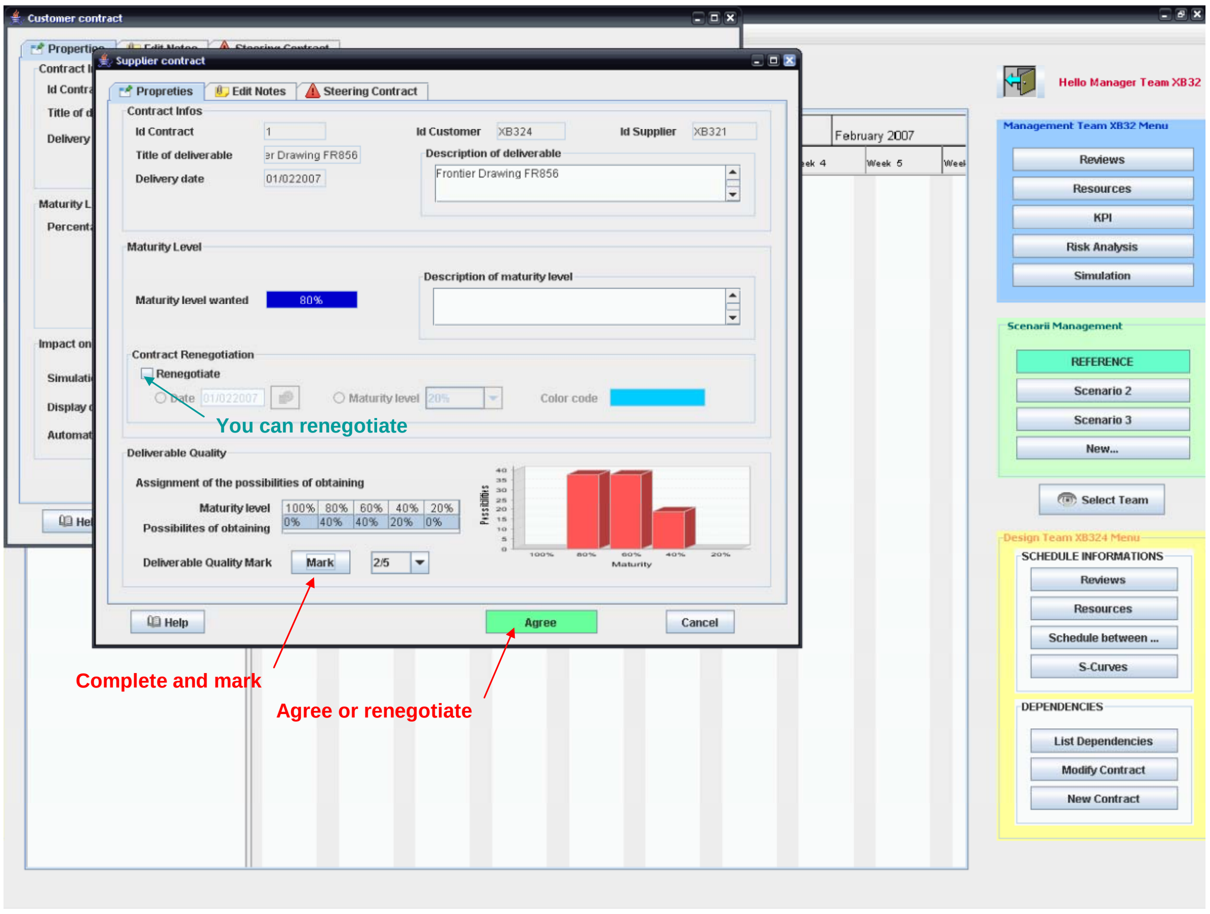
Date

Maturity level

Level	Colour
20%	🟦
40%	🟦
60%	🟦
80%	🟦
100%	🟦

Figure 59: Contract Definition

Contract will be defined by a supplier's definition, the data description, the date when delivery is required and the maturity level of this data supported by a colour code.



Supplier contract

Contract Infos

Id Contract	1	Id Customer	XB324	Id Supplier	XB321
Title of deliverable	er Drawing FR856				
Delivery date	01/022007				

Description of deliverable: Frontier Drawing FR856

Maturity Level

Maturity level wanted: 80%

Description of maturity level

Contract Renegotiation

Renegotiate

Date 01/022007 Maturity level 20% Color code

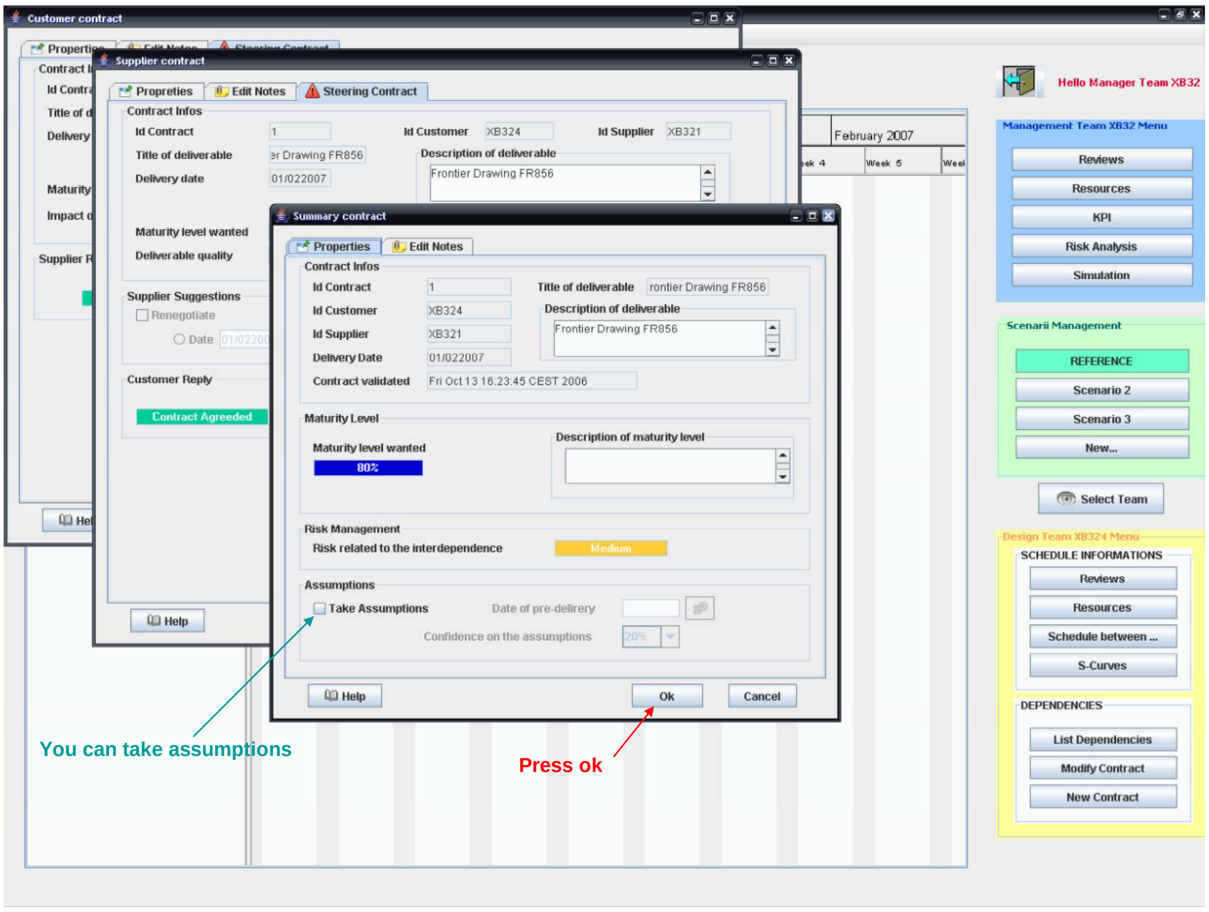
Deliverable Quality

Assignment of the possibilities of obtaining

Maturity level	100%	80%	60%	40%	20%
Possibilites of obtaining	0%	40%	40%	20%	0%

Deliverable Quality Mark: 2/5

Figure 60: Means for Contract Negotiation



Supplier contract

Contract Infos

Id Contract	1	Id Customer	XB324	Id Supplier	XB321
Title of deliverable	er Drawing FR856	Description of deliverable	Frontier Drawing FR856		
Delivery date	01/022007				

Maturity level wanted

Deliverable quality

Supplier Suggestions

Renegotiate

Date 01/02200

Customer Reply

Contract Agreeded

Summary contract

Contract Infos

Id Contract	1	Title of deliverable	rontier Drawing FR856
Id Customer	XB324	Description of deliverable	Frontier Drawing FR856
Id Supplier	XB321		
Delivery Date	01/022007		
Contract validated	Fri Oct 13 16:23:45 CEST 2006		

Maturity Level

Maturity level wanted: 80%

Description of maturity level

Risk Management

Risk related to the interdependence: Medium

Assumptions

Take Assumptions

Date of pre-delirery

Confidence on the assumptions: 20%

Figure 61: Last step for Contract Negotiation

Once the contract is negotiated between two actors, there is a milestone related to the contract that appears automatically in both schedules.

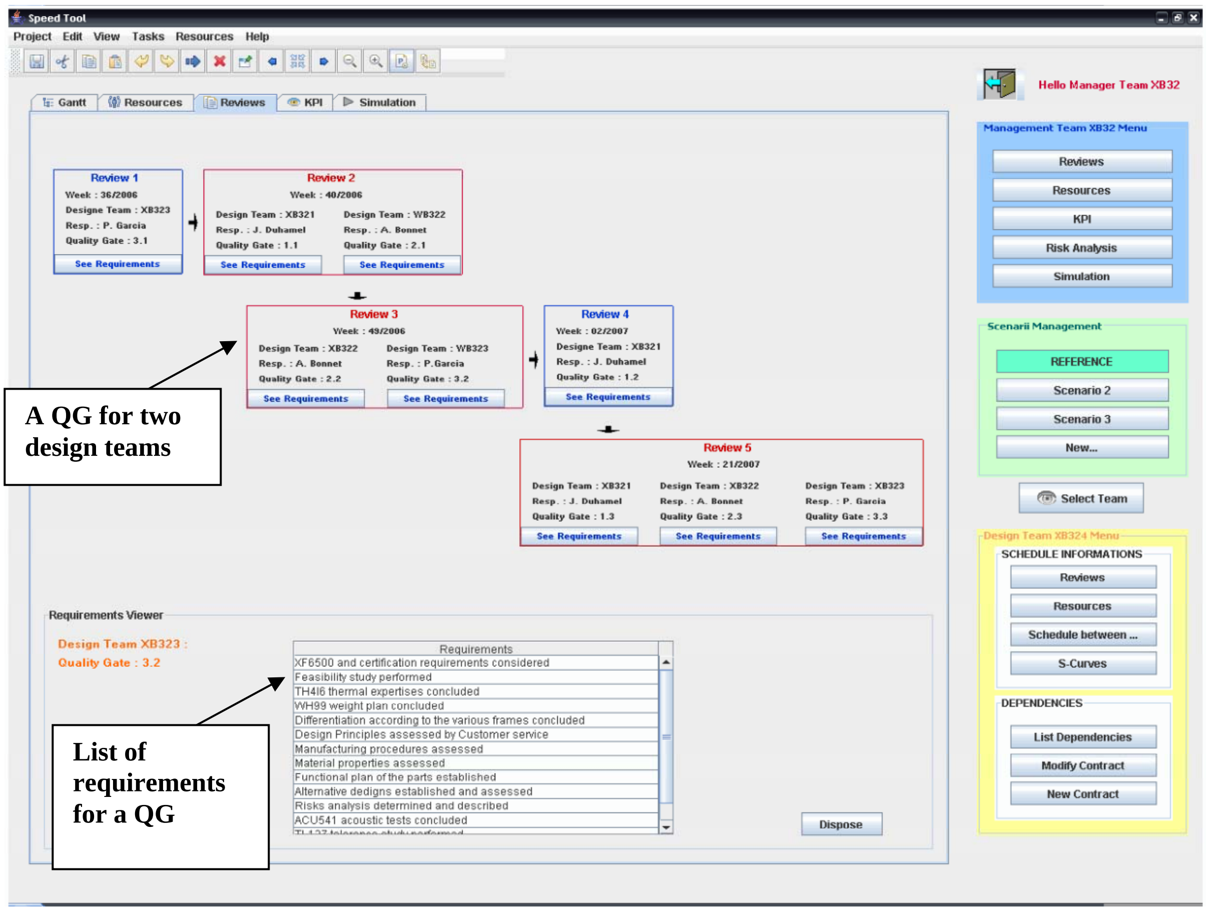
Moreover, a criticality of a contract will be defined following the next steps:

- The customer fills in the contract request, mentioning the impact on his or her schedule in the event that it is not respected. He or she will do this using a criticality scale.
- The Supplier receives the contract request with the information filled in by the customer. In return, he or she notes the quality of the deliverable in the event that the delivery date is respected. He does this using a calculation based on assumptions. Once he has filled it in, he returns the request.
- The application calculates the criticality level using a table and the information provided by the customer and the supplier. This table gives a colour code showing the criticality level. The deliverable will be displayed on the users' schedules, with the same criticality colour code.

Furthermore, over time, both the customer and the supplier can modify their answers concerning the impact and the quality of the deliverable. The application then recalculates the criticality level of the deliverable. If the level changes, the colour of the deliverable on the schedules also changes. Both users are warned as soon as the criticality level changes.

5.3.2.3 Quality Gates (QG) management

The tool enables to fix a series of Quality Gates (QG) over time for each of the design teams he leads. Within each QG, it notes a list of requirements which the team has to meet in order to move on to the next QG (See Figure 62). Some QGs are related to two or more teams.



Review 1
Week : 36/2006
Designe Team : XB323
Resp. : P. Garcia
Quality Gate : 3.1
See Requirements

Review 2
Week : 40/2006
Design Team : XB321
Resp. : J. Duhamel
Quality Gate : 1.1
See Requirements
Design Team : WB322
Resp. : A. Bonnet
Quality Gate : 2.1
See Requirements

Review 3
Week : 49/2006
Design Team : XB322
Resp. : A. Bonnet
Quality Gate : 2.2
See Requirements
Design Team : WB323
Resp. : P.Garcia
Quality Gate : 3.2
See Requirements

Review 4
Week : 02/2007
Designe Team : XB321
Resp. : J. Duhamel
Quality Gate : 1.2
See Requirements

Review 5
Week : 21/2007
Design Team : XB321
Resp. : J. Duhamel
Quality Gate : 1.3
See Requirements
Design Team : XB322
Resp. : A. Bonnet
Quality Gate : 2.3
See Requirements
Design Team : XB323
Resp. : P. Garcia
Quality Gate : 3.3
See Requirements

Requirements Viewer

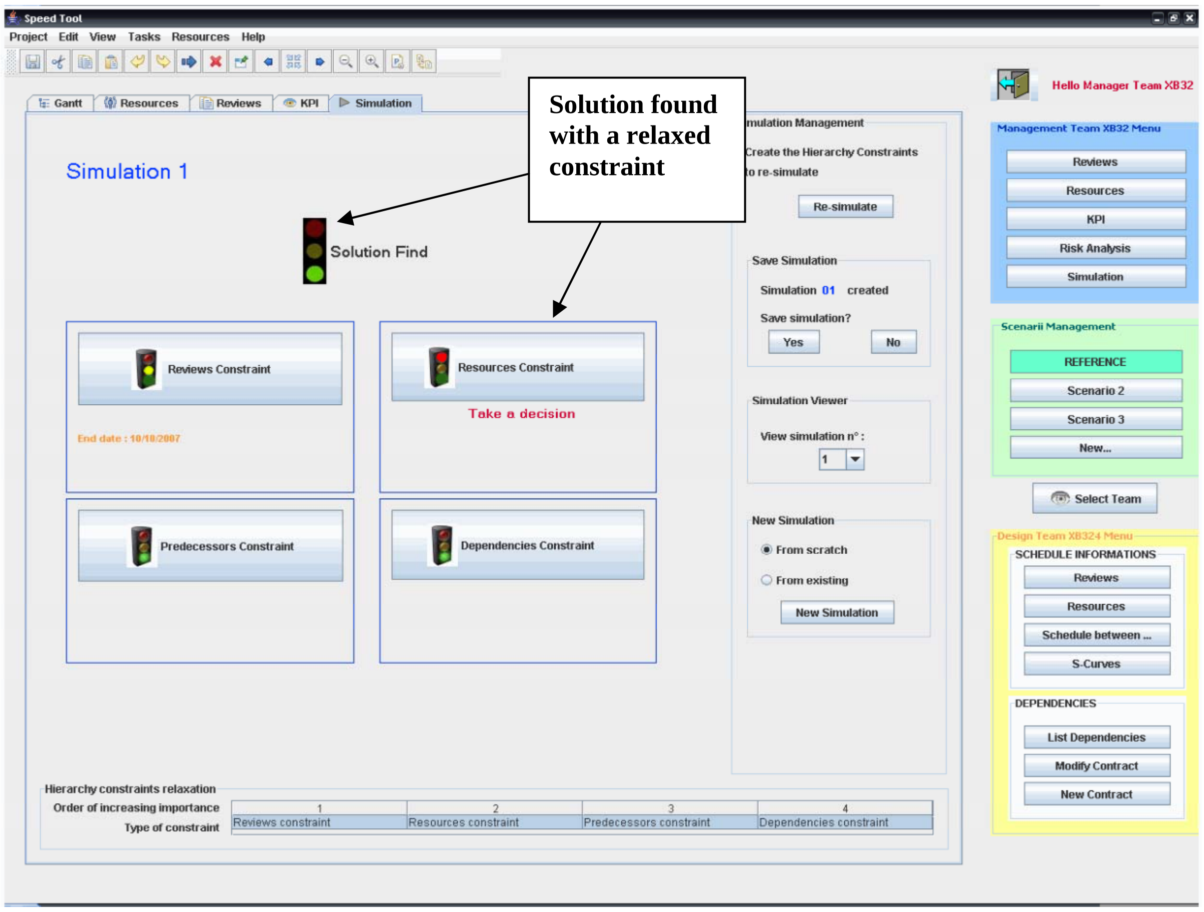
Design Team XB323 :
Quality Gate : 3.2

Requirements
XF6500 and certification requirements considered
Feasibility study performed
TH4I6 thermal expertises concluded
WH99 weight plan concluded
Differentiation according to the various frames concluded
Design Principles assessed by Customer service
Manufacturing procedures assessed
Material properties assessed
Functional plan of the parts established
Alternative dedigns established and assessed
Risks analysis determined and described
ACU541 acoustic tests concluded
TL127 tolerance study performed

Figure 62: Quality Gates

5.3.2.4 Scheduling simulation based on constraints management

Figure 63: Interface for launching scheduling simulations shows the graphical interface that we have proposed in order to launch scheduling simulations based on constraints management.



Hierarchy constraints relaxation

Order of increasing importance	1	2	3	4
Type of constraint	Reviews constraint	Resources constraint	Predecessors constraint	Dependencies constraint

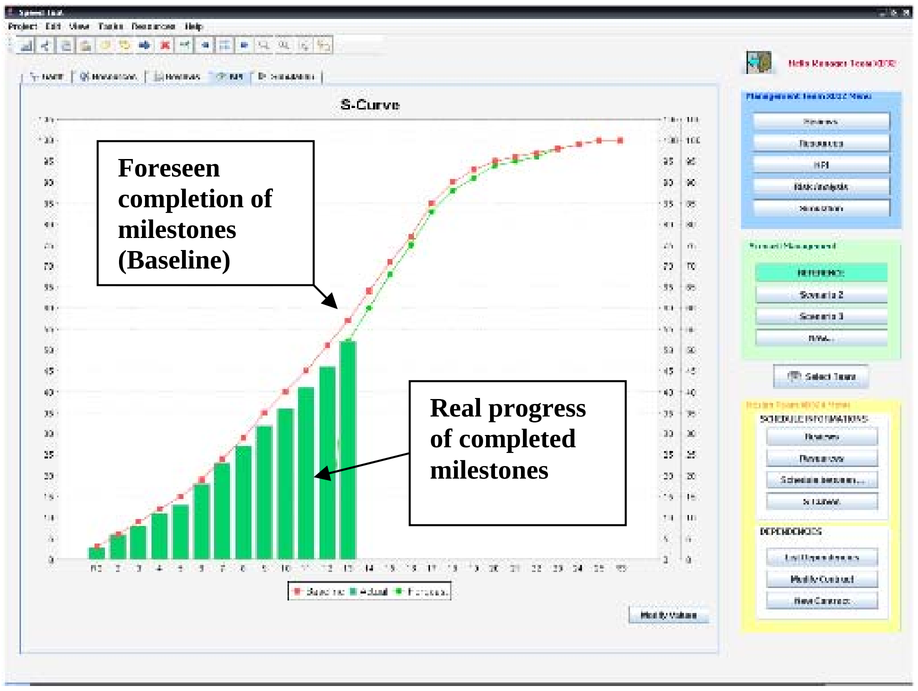
Figure 63: Interface for launching scheduling simulations

Constraints satisfaction is featured using a traffic lights metaphor. The main traffic light shows the status of each simulation at each moment. Furthermore, under the traffic light, the user can define the hierarchy of constraint to be relaxed. For example, an over constrained problem will be represented by a main red light. In order to find a solution we can launch a new calculation relaxing the first constraint in the list. This constraint will be relaxed until a solution is found. Therefore, the main traffic light will become green, while the traffic light related to the relaxed constraint will become red. The later traffic light will remain red until the user verifies if it will be possible to perform this relaxation in reality. Once verified that this is possible the traffic light will become yellow.

This graphical interface allows the user to launch several scheduling simulations dealing through a user friendly interface.

5.3.2.5 Key Performance Indicators Management

Figure 64 shows some of the KPIs that can be defined with the decision support Illustrator.



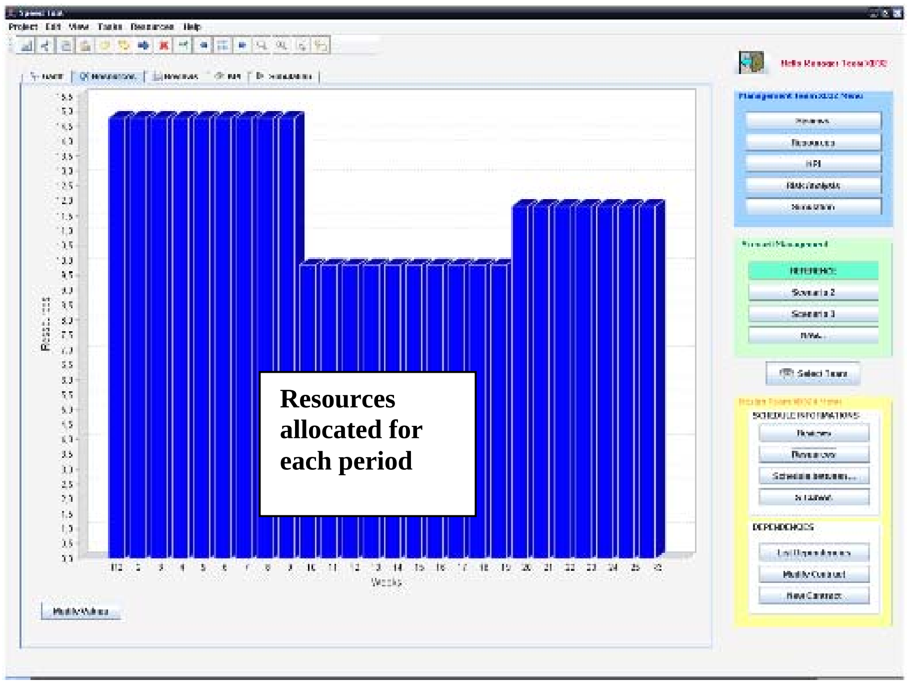


Figure 64: Key Performance Indicators Management

The KPIs are mainly based on the foreseen workload and resources as well as the real progress concerning the allocated resources and the realised work. Energy and resources from different teams can be aggregated in order to have an overall picture of the design progress of the teams under the same management level.

5.4 Evaluation of the prototype

In order to evaluate the model and solutions proposed in the framework of this research project, we have solicited the support of a new aircraft program currently under development. This is the A350 program which is a new bi-reactor aircraft mainly characterised by a mach

0.85 cruise speed and between 270 and 350 seats depending on the family model (A350-800, A350-900 or A350-1000), it is also foreseen to develop a freighter version. The team that has been chosen to implement the demonstration is the Section 11-12 which is the nose fuselage of the aircraft.

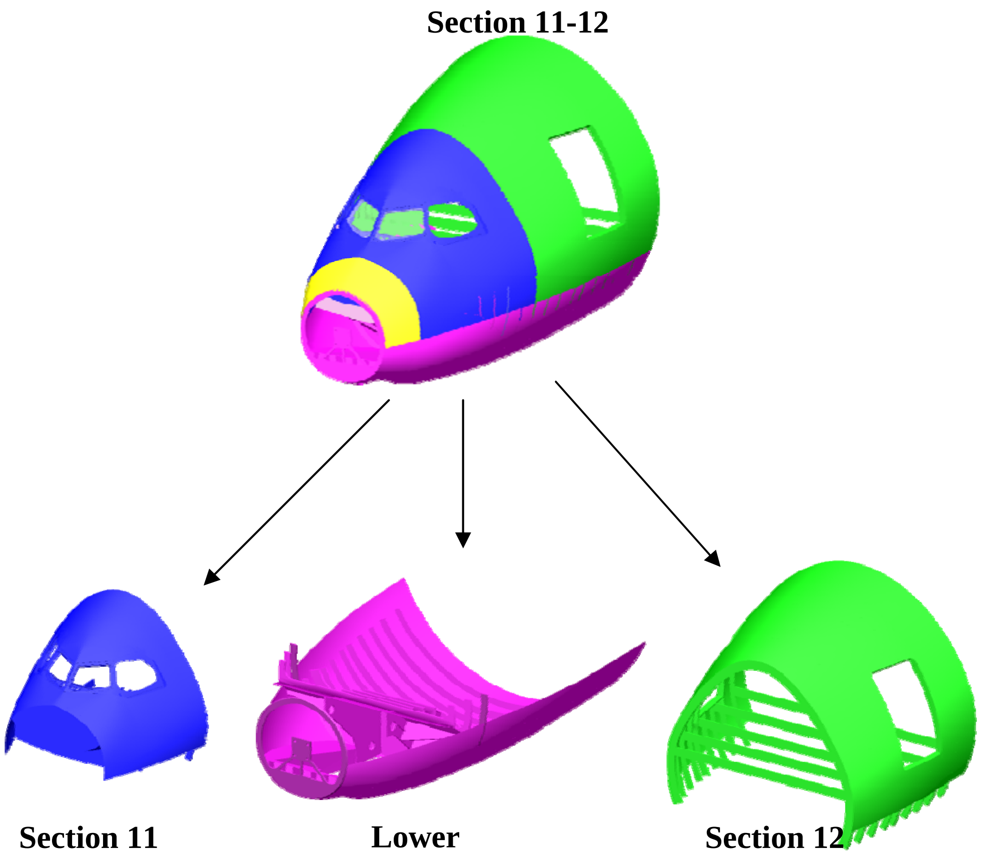


Figure 65: Section 11-12

5.4.1 Teams structures

Figure 65 shows the structure part of the section 11-12, which is composed by three subsystems (Section 11, Lower unit and Section 12). Each of these subsystems is developed by a design team responsible of delivering the entire subsystem with the required quality level a defined date. Two additional design teams are responsible of the installation of mechanical systems and the installation of electrical systems. In order to manage these five design teams and to coordinate the global development of this section, a management team has been created.

5.4.2 Deadlines

The first A350 will enter in service in mid 2013. In order to respect this deadline, the final assembly line is foreseen to start on September 2010. Therefore, most of the aircraft parts will be manufactured during the time horizon 2008-2009. In order to begin manufacturing, the definition phase corresponding to design activities will be performed during 2007-2008.

5.4.3 Actions performed at the section 11-12

Concerning the Section 11-12, the definition phase of 80% of the structural parts is foreseen to be performed between April 2007 and April 2008.

The first step of the definition phase consists mainly on defining a predefined number of design principles, essentially design drawings for a zone as described in 2.4.1 chapter.

Different levels of maturities are defined for each design principle, which are mainly linked to the agreement given by different skills. Once the absolute maturity level is achieved for a particular design principle, the pieces being part of this design principle will be represented accurately in a geometric reference mock-up (seen in 2.4.5 chapter).

At the moment, the different teams of the section 11-12 are not yet formed. Actually the first demand of the head of this section was to support the resource estimation process in order to perform all the design principles. At the moment, the three teams where not separated and there where 12 people working on them. The question that the head of the section shall answer is, taking into account the work to be done related to the design principles definition, how many people should integrate the team.

376 design principles have been identified, 141 for the section 11, 85 for the section 12, 130 for the lower unit and 20 related to junctions between three subsystems. They have been divided into groups in order to simplify the schedule.

The first task has been to estimate a latest end date for the completion of each design principle group and the energy needed to perform each one. Based on these characteristics, different schedules have been defined modifying the amount of allocated resource for each period and the starting date for each design principle definition activity.

We have tried to allocated resource avoiding differences between the numbers of allocated resources in two sequential periods.

Table 15: Time and resources allocation for each design principle

	DP num	E	2007																											
Week			25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52
Section 11																														
Window frames, windshields	45	175	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
Window frames, windshields …Stress	45	32																	1	2	2	3	3	3	3	3	3	3	3	3
S11 floor	15	14															1	1	1	1	1	1	1	1	1	1	1	1	1	1
Front lateral panels, lateral frames 5&6	15	25				1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Rear Upper panels, frames 7&8	7	9																				1	1	1	1	1	1	1	1	1
Rear lateral panels	7	9																				1	1	1	1	1	1	1	1	1
S11 EREBUS junctions	25	20									1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Sliding mechanism + lat window	20	56	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
Sliding mechanism + lat window Stress	20	19																		1	1	1	2	2	2	2	2	2	2	2
Section 12																														
Lateral panels, door fr, frames 9 to 18	35	99	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
S12 floor grid : cross beam & seat rails	15	10																			1	1	1	1	1	1	1	1	1	1
Upper panels, upper fr 9 to 18	20	58	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
Upper panels, upper fr 9 to 18 Stress	20	20																		1	1	2	2	2	2	2	2	2	2	2
S12 EREBUS junctions	15	10																			1	1	1	1	1	1	1	1	1	1
Lower Unit																														
NLG bay	40	90	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
NLG bay Stress	40	14																						1	1	1	2	3	3	3
Lower shell pan, lower fr 1 to 18	25	78			3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Lower shell pan, lower fr 1 to 18 Stress	25	36															1	2	2	2	2	3	3	3	3	3	3	3	3	3
NLG doors, articulations	20	7																						1	1	1	1	1	1	1
Junctions		0																												
Interface drawings	20	70	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2

Table 15 shows one of the schedules that have been defined. Each line corresponds to a group of design principles.

This exercise has allowed define not only the amount of resources that will be needed at the end of the year but also how the ramp up should be realised (see Figure 66).

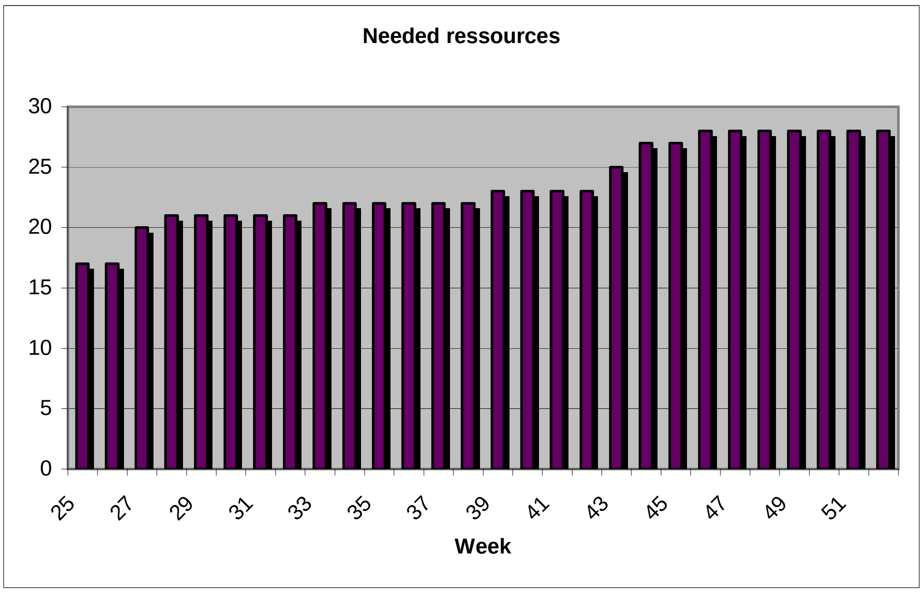


Figure 66 : Needed Ressources

In conclusion, we can say that the work that has been carried out until now has focused the preparation of schedules for next years and the definition of the ramp up of the resources. No comparison with real progress of the development has been measured and therefore we have not evaluated our model and solutions in a progress on going dynamic environment.

The next step is to detail the interfaces between the three design teams as well as between the teams responsible for systems installation and the management teams responsible for other sections.

Table 16: Interfaces between the different sections and systems

INTERFACES	SECTION 11							SECTION 12		LOWER UNIT					INSTALATION MECANICAL SYSTEMS						INSTAL ELECTR SYSTEMS		
	Windshield Structure	Floor S11	Floor S12	Frame 1	Sliding Widows	lateral panels	Upper Shell	Pax Door Frame	Lateral panels	NLG Bay	Lower Shell	NLG Doors	E-bay Doors	Fairing	FCRC	Cockpit furnishing	Flight Controls	Air conditionning	Commercial systems	Technical systems	Electrical bay	Control panels	Harnesses
LANDING GEAR										X		X								X	X		X
FAF (S13)			X					X			X							X	X	X	X		X
EY (systems)		X	X												X	X	X	X	X	X	X	X	X
TLE (elec)																			X	X	X	X	X
EYYAC (flight contr)																				X	X	X	X
EYYAR (radome)		X		X							X									X	X		X
EYT (tests)	X	X		X	X	X	X								X	X				X	X	X	X
EYAK (Cabin)		X	X												X				X	X	X	X	X
DOOR 1								X												X	X		X
SECTION 11	▨	▨	▨	▨	▨	▨	▨		X		X				X	X	X	X	X	X	X	X	X
SECTION 12		X	X			X	X	▨	▨		X				X			X	X	X	X		X
LOWER UNIT		X	X	X		X		X	X	▨	▨	▨	▨	▨	X					X	X		X
ISM	X	X	X			X	X	X	X	X	X				▨	▨	▨	▨	▨	▨	X	X	X
ISE	X	X	X			X	X	X	X	X	X				X	X	X		X	X	▨	▨	▨

Table 16 shows the possible interfaces between different teams. Currently, bilateral meetings are been held in order to define accurate contracts due to these interfaces.

The three teams we focus on have to ask for the data they need in order to begin some of the tasks. Furthermore, other teams will ask them some data that might not be scheduled as the request. Taking into account the inputs negotiated with other teams, some design principles definition activities might be started later. Moreover, in order to deliver a data to a design team that needs earlier that the foreseen data, some design principles beginning date might be advanced or more resources will have to be allocated at the beginning.

These modifications will be realised at the same time as contract will be signed.

Later on, once a fixed amount of resources are allocated to each design team, additional resources request will have to be proved with accurate schedules and recovery plans. We expect that interesting feed back will be gathered during these dynamic periods that will complete the first feed backs we have collected.

5.4.4 First feed backs

Activities realised for the evaluation of the prototype at the first stage of the definition phase, have allowed us to collect some feed backs about our approach and the prototype.

The scheduling activities performed have taken into account the cumulative resource constraint, time window constraint and the task energy constraint. The constraint related to the contracts will be added in the next step of the evaluation. Concerning the Energy-Precedence Constraints, the definition of the parallelism period between two tasks has been based on the results of our tool but it has also been necessary to make some modifications manually. The reason of this manual action is that the tasks to be scheduled do not deal with only one design principle but to groups of design principles. Concurrent tasks include for the same design principles, one task related to the design activities and the other one related to structural calculation activities. In other words, the main problem was to begin the structural calculation activities after the energy amount necessary to achieve a maturity level that has been defined by this skill. The results of our prototype included a beginning period of the second task that stated with maximal resources allocation. Nevertheless, not all the design principles of the group have achieved the requested maturity level; therefore we have modified manually the second task so that the resources allocated at the beginning of the task are enough but not too much in order to deal with only the available design principles.

One of the needs that have arisen from the users was the possibility of creating different levels of the maximum allowed intensity. Indeed, our prototype includes the possibility of limiting the value of the resources allocated to a period, but this value shall be applied to all the tasks. In reality, actors know that they will never allocate more than an amount of resources to the definition of a group of design principles. This new constraint could limit even more the scheduling problem.

Concerning the contract management capabilities, the negotiation support has been accepted positively, nevertheless the criticality definition process has been considered as too complex. Due to the fact that the contract to be negotiated can be numerous, actors expect to negotiate each one as fast as possible. It has been proposed to keep the risk management process related to the contract only to critical data exchanges based on former programs experience. Furthermore the relation between contract definition and the contract milestone display in the schedule has been considered as a powerful capability.

Quality gates definition linked to the schedules and the possibility of defining the requirement linked to each quality gate has been considered as a useful capability. It has been suggested to add a predefined list of requirements since this kind of requirements can be redeployed from one program to the next one.

Lastly, the constraints relaxing interface has not been evaluated using real cases. The demonstrations that have been realised where based on add-hoc examples prepared to explain the overall process. It has been requested to include the possibility of modifying the amount of energy related to a task in order to launch other simulations. Indeed, the prototype considers the energy of the task as a non modifiable data. In reality new evaluation of the work needed to perform the task are very frequent and this possibility shall be easily accessible when performing the simulations.

In the next chapter we describe one of the examples used in the demonstration of the constraints relaxing interface.

5.4.5 Demonstration of the constraints relaxing interface

The following example illustrates how the demonstration of the constraints relaxing interface could be used in order to support the decision making process of the managers of different hierarchical levels. Considering a part of the PBS composed by two design teams and a management team as described by Figure 67 :

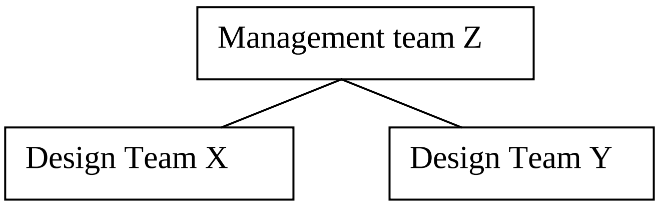


Figure 67 : PBS composed by two design teams and a management team

If we focus on the scheduling of the design team *X*, we define 8 tasks. The amount of work for the design team *X* is defined in the table below.

Table 17: Definition of Engergy Amounts

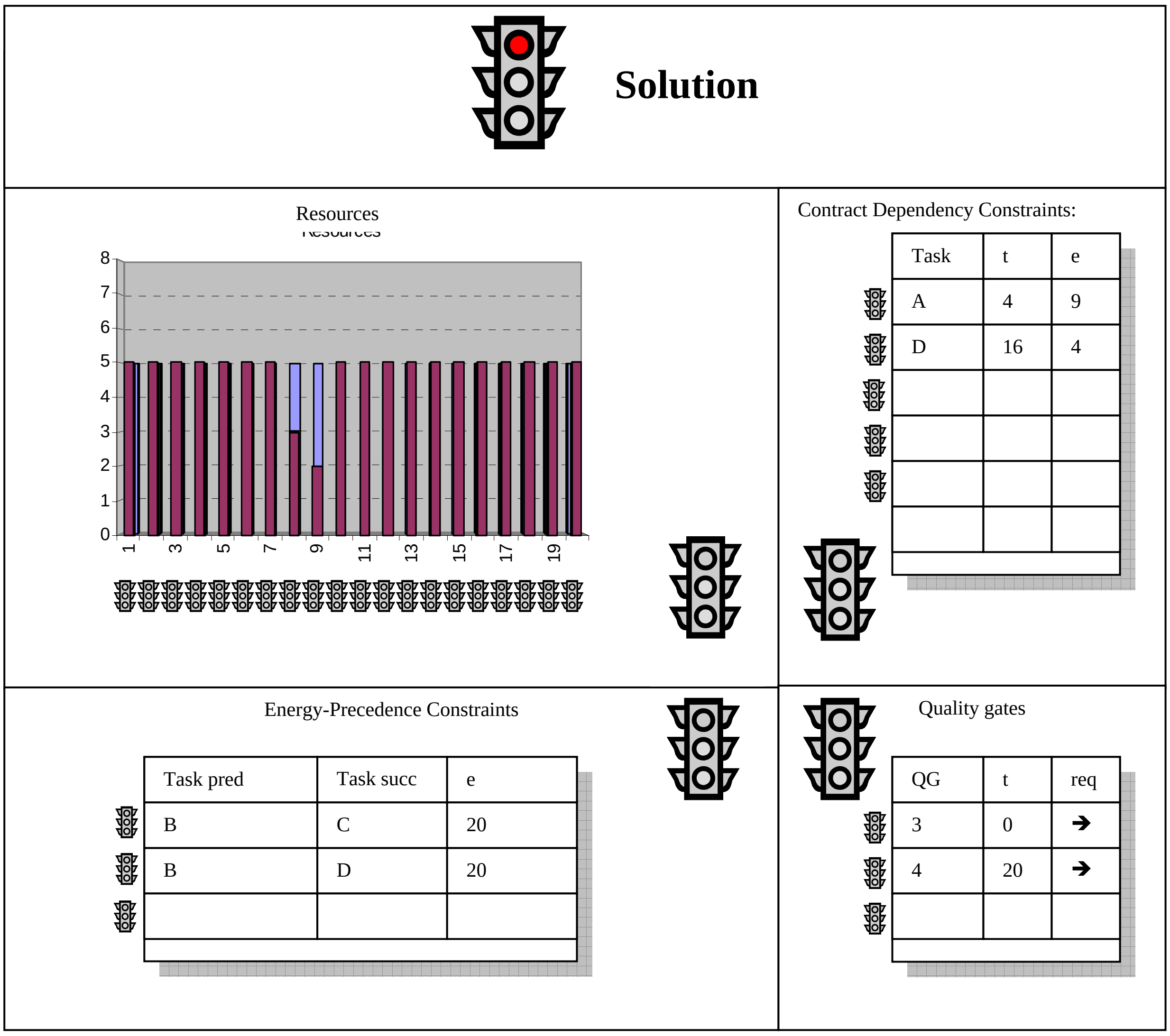
Task	e
A	20
B	20
C	19
D	14
E	11
F	8
G	6
H	4

Other constraints include a contract dependency constraint between the design team *X* and design team *Y*. It establishes that design team *X* shall deliver a data called u_{ij} to the design team *Y* before the period 4 with a maturity level of 50%. Design team *X* has identified the task *A* as the task that will define this data and it has identified that the energy that should be expended in order to achieve the maturity level of 50% is 9 units. Therefore, these 9 units shall be performed during the period 1, 2 and 3. Therefore we can establish: $CDC_{Aj} = \{t_{Aj}, e_{Aj}\} = \{4, 9\}$. Another contract with a third team incorporates a second CDC: $CDC_{Dj} = \{t_{Dj}, e_{Dj}\} = \{16, 4\}$.

Moreover an EPC exist between the tasks B and C/D:

EPC (B, C, 20j) and *EPC (B, D, 20)*. Let us note that both constraints are equivalent to the traditional scheduling precedence constraint since $e_{ij} = e_i$ *in both cases.*

Figure 68 shows how the problem will be represented by the constraints relaxing interface. Red light symbolizes the fact that there is not any solution for this problem that satisfies all the constraints.



Solution

Contract Dependency Constraints:

Task	t	e
A	4	9
D	16	4

Energy-Precedence Constraints

Task pred	Task succ	e
B	C	20
B	D	20

Quality gates

QG	t	req
3	0	➔
4	20	➔

Figure 68 : Interface shows that there is no solution for the problem

Therefore the manager of the design team *X* launches several simulations in order to find a feasible schedule. These simulations are launched relaxing one or more constraints. Once the different solutions are identified, he or she can choose a new schedule and perform the actions in order to relax the constraints. In this example the identified solution has been the schedule shown in Table 18:

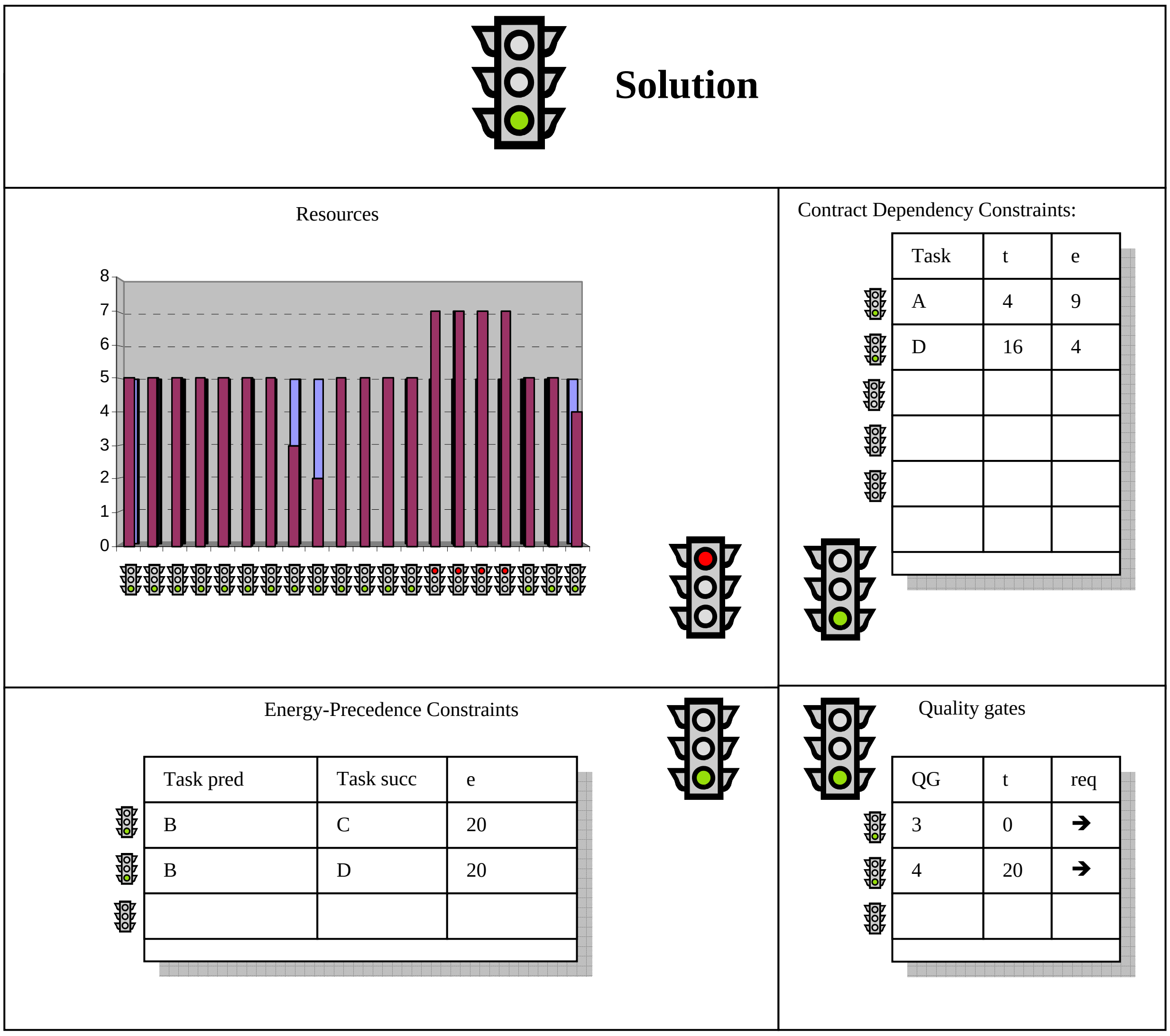
Table 18: Proposed Schedule after Resources Constraints Relaxation

A	3	3	3	3	3	3	2													
B	2	2	2	2	2	2	3	3	2											
C										3	3	3	3	3	3	1				
D										2	2	2	2	2	2	2				
E														2	2	3	3	1		
F																1	3	3	1	
G																	1	1	3	1
H																			1	3
Period	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

In order to perform this schedule the resource constraint has been relaxed during the periods {14,15,16,17}:

$$\sum_{i=A}^{i=H} a_i^{\theta=14} = 7$$

Now, the major traffic light that defines if a solution has been found is green. And the medium traffic lights related to each constraints show if every constraint is respected. Due to the relaxation of the resources constraint, and the fact that we still do not know if this relaxation will be possible in reality, the traffic light corresponding to the resource constraint is red. Nevertheless, the schedule of this simulation respect the CDCs, the EPCs and the quality gates, therefore their traffic light is green as shown in Figure 69.


Solution
Resources
0
1
2
3
4
5
6
7
8
Contract Dependency Constraints:
Energy-Precedence Constraints
Quality gates

Task	t	e
A	4	9
D	16	4

Task pred	Task succ	e
B	C	20
B	D	20

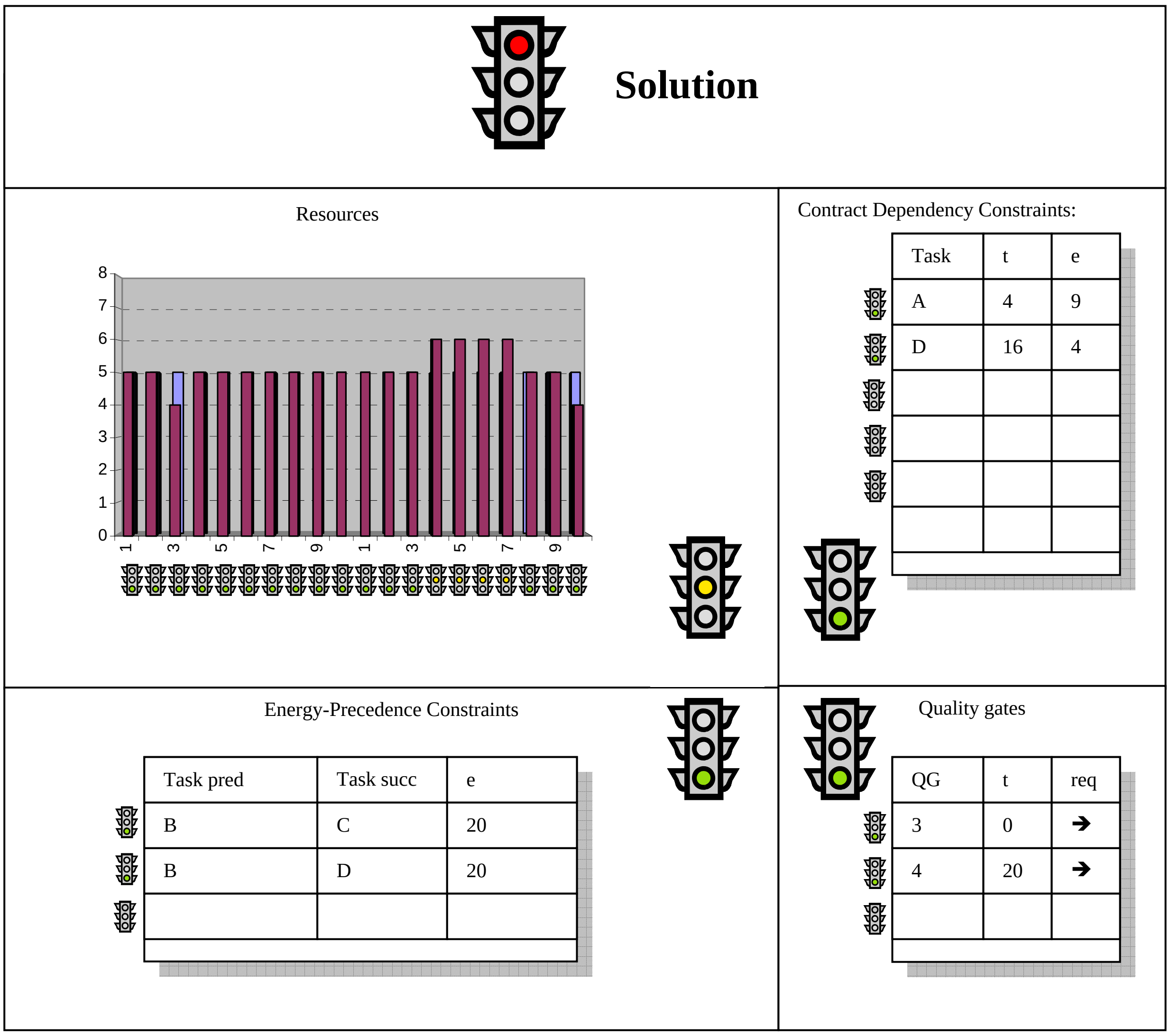
QG	t	req
3	0	➔
4	20	➔

Figure 69: There is a solution relaxing resources constraints

The actions linked to the implementation of this relaxation includes the validation of the head of the “Management Team *Z*” concerning the hiring of two more designers during four periods.

Consider that the head of the “management team *Z*” does not agree to allocate two more people in the design team *X* and accepts only one more resource.

Taking into account this fact, manager of design team *X* restarts the simulation process as the new scheduling problem with only one more resource allocated to the periods {14,15,16,17} has not been considered as a solution.


Solution
Resources
8
7
6
5
4
3
2
1
0
1
3
5
7
9
1
3
5
7
9
Contract Dependency Constraints:
Energy-Precedence Constraints
Quality gates

Contract Dependency Constraints:

Task	t	e
A	4	9
D	16	4

Energy-Precedence Constraints

Task pred	Task succ	e
B	C	20
B	D	20

Quality gates

QG	t	req
3	0	→
4	20	→

Figure 70: No solution with partial relaxation

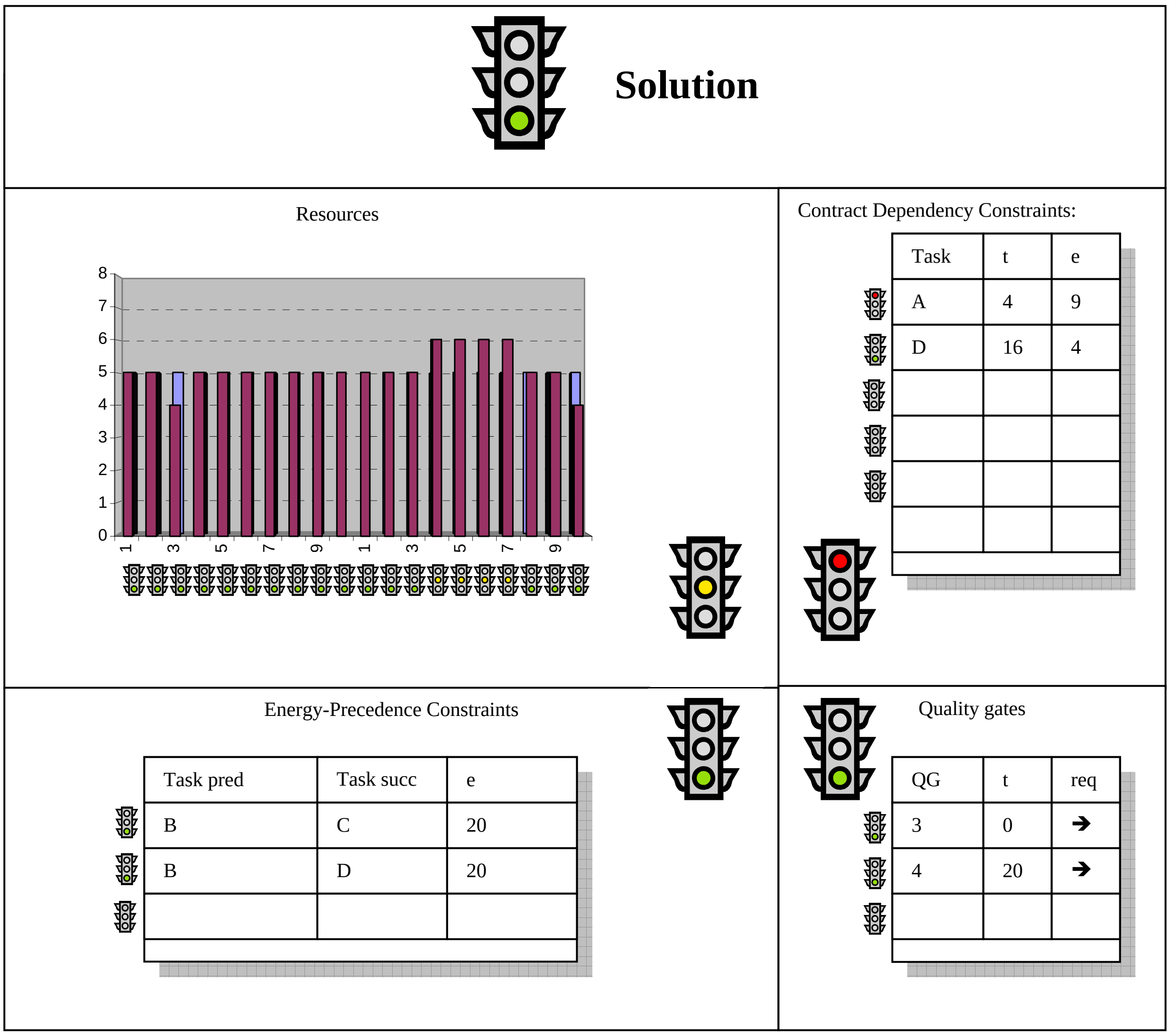
Now the resources constraint traffic light is yellow as shown in Figure 70 . This means that the relaxation is not only an assumption being part of the simulation but that it has been confirmed that in reality this relaxation will be possible.

In order to find a new solution, the following relaxation to be considered will be the CDC related to the task *A*. Table 19 shows the schedule found if this constraint is relaxed.

Table 19: New schedule relaxing the CDC

A	2	2	1	2	3	3	3	3	1	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨
B	3	3	3	3	2	2	2	2						▨	▨	▨	▨	▨	▨	▨
C	▨	▨	▨						3	3	3	3	3	3	1				▨	▨
D	▨	▨	▨	▨	▨				1	2	2	2	2	2	3			▨	▨	▨
E	▨	▨	▨	▨	▨									1	2	3	3	2		▨
F	▨	▨	▨	▨	▨	▨	▨	▨	▨							3	3	2		
G	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨							1	3	2
H	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨					2	2
Period	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20

In order to perform this schedule, a new person will need to be hired during periods {14,15,16,17} and the contract with design team *Y* will have to be renegotiated. The constraints relaxing interface for this new simulation is described by Figure 71 :



Contract Dependency Constraints:

Task	t	e
A	4	9
D	16	4

Energy-Precedence Constraints

Task pred	Task succ	e
B	C	20
B	D	20

Quality gates

QG	t	req
3	0	➔
4	20	➔

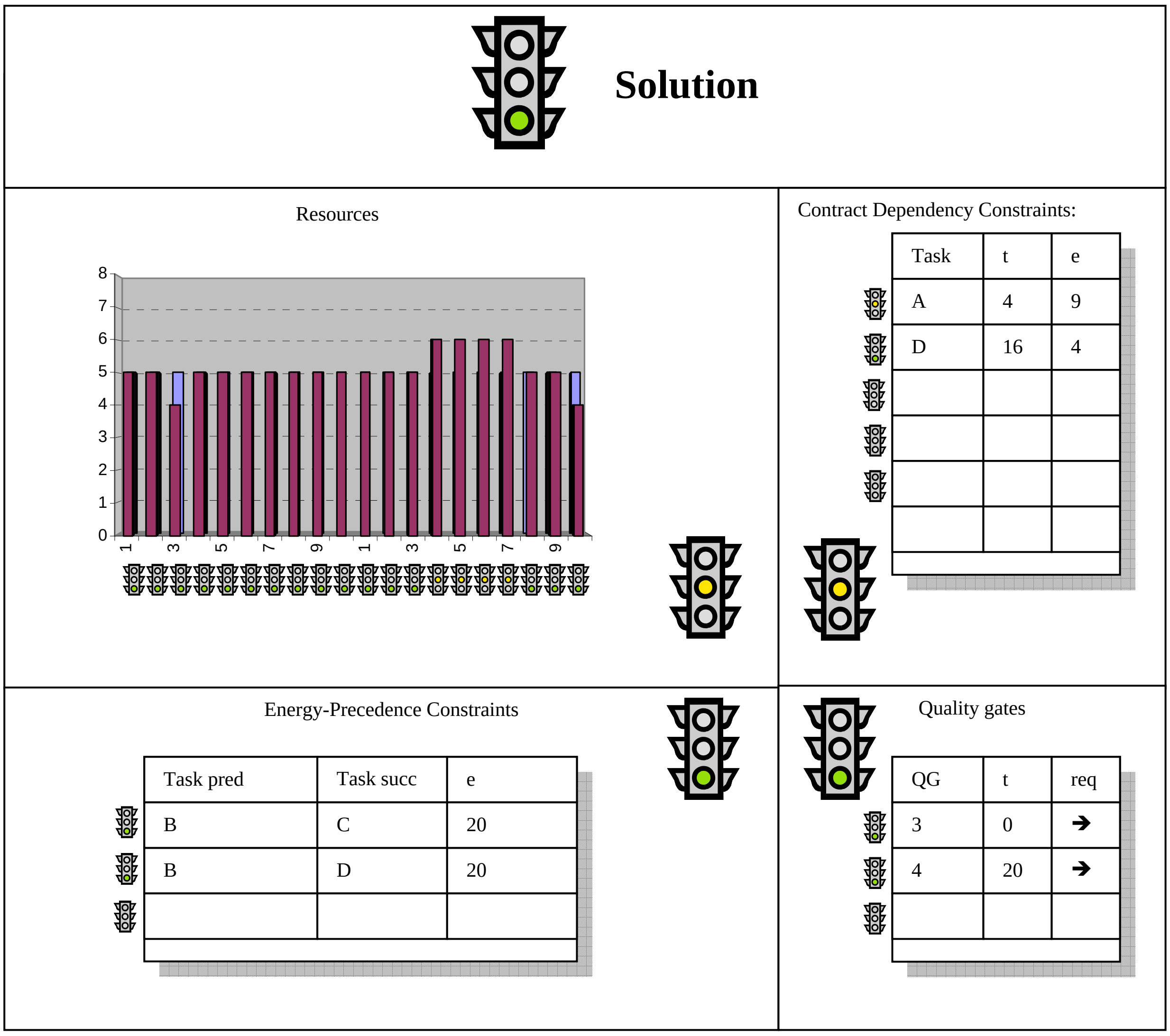
Figure 71 : Interface shows the relaxed constraints.

The manager of design team *X* announces to design team *Y* that even if the date of the contract will be respected, it will not be able to attain 50% of the maturity demanded by design team *Y*. Indeed only 5 units of energy will be expended for task *A* before period four. The maturity level expected by design team *Y* will be available during period 5. This is a current Concurrent Engineering practice where data will be delivered in more that one step giving the possibility to the customer to follow the work with preliminary information. These two deliveries correspond to the following new constraints:

$CDC_{ij} = \{t_{ij} = 4, e_{ij} = 5\}$

$CDC_{ij} = \{t_{ij} = 5, e_{ij} = 9\}$

If design team *Y* can rearrange its schedule in order to accept the data coming form design team *X* with a lower maturity level a new contract will be signed between both teams and two feasible solutions will be defined for the project.



Contract Dependency Constraints:

Task	t	e
A	4	9
D	16	4

Energy-Precedence Constraints

Task pred	Task succ	e
B	C	20
B	D	20

Quality gates

QG	t	req
3	0	➔
4	20	➔

Figure 72: Relaxed constraints can be performed in reality.

Figure 72: shows the simulation state if design team *Y* accepts the new conditions.

Nevertheless, if design team *Y* is not able to find a solution with the new imposed constraint, it could launch simulations relaxing constraints in its scope. Possible solutions can include the allocation of a new resource or the modification of the contract with another team. We can realise on the one hand how the problem can come back to the head of the "management team Z" who will receive the demand of a new resources this time from design team *Y*. On the other hand, we realize that a contract modification can be propagated from team to team with the risk of complicating the overall constraint satisfaction problem.

5.5 Conclusion

The prototype developed in order to demonstrate the solutions proposed in the framework of this research project, has three main components. Firstly, the ECLiPSe kernel is the central component of our system; it includes the constraints propagation devices described in the chapter four. Then, the ACTILOG includes the three elements of the future application: the kernel, a graphical user interface and an application component between them. Lastly, the decision support Illustrator is a proposal of graphical user interface which has allowed gathering new functional and technical requirements concerning the future application.

The evaluation actions performed in the framework of a new Airbus product development have only allowed us to validate some of the capabilities of our prototype. Indeed, the prototype needs to be evaluated during the whole definition of the aircraft and mainly during the phase when recovery plans need to de defined for the development of some subsystems. These recovery plans include the modification of the baseline schedules and the renegotiation of the project constraints. This evaluation will be performed in 2007.

General conclusion

1- Introduction

"*I need more resources*", this is a recurrent claim of design team managers in charge of the development of a subsystem being part of a complex product. At the same time, a frequent question occurs in a higher managerial level for the same project: "*I do have extra resources for this project, but where should I allocate them?*"

Deviations from the originally proposed schedule affect not only a team, but have also negative influence on the overall development of the project. Anticipating these deviations implies to tackle both time and resource allocation problems. Minor modifications of resources allocation during the project can be considered as normal practices. Nevertheless, one may observe in many case studies that a high number of resources can be allocated during the last period of the definition phase -with a high cost increase-, that denotes that time and resources allocation problem solving can still be improved considerably.

-"Mr. Smith, I am very sorry but I will not be able to send you the data you ask me for the next week"

-"What's that! We signed a contract, you must respect it"

This second example describes a common conversation between managers of two teams belonging to the same project. This disagreement will surely cascaded to a higher level managers' meeting and the decision to resolve it will surely be undertaken too late. Indeed, the development of a complex product based on distributed design teams supposes numerous interactions between these teams. Nevertheless, the rational of the product development organisation focuses on the establishment of autonomous teams based on cross-functional skills with tight time and cost commitments. The result is that interdependencies between teams are considered as secondary project constraints. Team managers are inclined to focus on the optimisation of the internal project constraints rather than to collaborate for an efficient integration of the overall system.

But, these two examples have common characteristics. Once admitted that disruptions in the foreseen schedule are unavoidable events for one or more teams being part of the development of a new aircraft, waiting until program level managers take drastic decisions might not be considered as the best strategy. But rather to argue on difficulties, each team can analyse its autonomy and the possible local corrective solutions through a collaboration with others teams in order to find the best balance from the overall project point of view. In the first example, this collaboration will happen between teams of different managerial levels. In the second case, it will happen between two teams evolving at the same level.

Analysing and explaining explicitly its own difficulties need a rigorous framework to check accurately the consistency of the ongoing situation with the project constraints. Moreover, in a collaboration process, each part shall be able to propose feasible accurate solutions and therefore be able to negotiate being aware of the consequences of each proposal.

It seems therefore necessary to define a framework able to take into account every project constraint and to support the steering of the design activities of each team, focusing on the overall project progress. The definition of this framework is basically the result of this work.

2- Report structure

These results can be subdivided into five topics related to the five chapters developed in this report.

In the first chapter, we show that a complex project management system has to deal with a high number of resources and numerous detailed activities to be scheduled. We stress the fact that interdependencies between design activities is unavoidable and that their interactions should be more supported,

The second chapter focuses on the design activities of the new aircraft development. It describes the concurrent engineering practices and especially the exchanges of information supports. The activities that are defined in order to develop these supports are scheduled following a process that includes different types of schedules. It concludes that dependency management process between design teams is a key element for an efficient product development. Thus, research questions are still open concerning the problem of supporting the collaboration process between design teams and between different managerial levels.

Three aspects that are key elements for the scheduling of design activities during the development of a new aircraft are investigated in the third chapter: the scheduling problem at the tactical level, the uncertainties related to time and resources allocation problem, and the management of dependencies between different design teams. We conclude on the necessity to integrate these three aspects into the same framework that should be supported by a common project scheduling management model.

To develop this integration, the fourth chapter describes a constraint satisfaction problem model, which has been considered to be the most suitable model to answer to the research question. Beyond classical constraint propagation mechanisms, it proposes two new constraints types for expressing a partial precedence relation between activities with variable duration. This model can be used for various simulations that support the steering process of engineering activities. It enables to validate the consistency of the decisions and offers a reference for the renegotiation of constraints.

Lastly, the fifth chapter presents the prototype developed in order to demonstrate the solutions proposed in the framework of this research project. It was used to partially validate the proposals and get the feed-back of potential en-users for future improvements.

3- Contributions

The estimation of the work to be performed during design activities is usually formalised by time and resources allocation but these two notions are very often not coupled. Therefore, modifications in design activities durations are not followed by an assessment of the needed amount of resources. Moreover, demands for more resources are rarely based on accurate evaluations of the work to be performed and managers have difficulties to define the most constrained teams where extra resources need to be allocated. Our first contribution has been to propose an energy-based characterisation of design and development activities, and the constraints that link them.

If classical project constraints, like available resources or deadlines, are unavoidable when assessing the steering decisions, the definition of additional constraints is necessary in order to include concurrent engineering practices as well as interdependencies between teams at the same level. Our second contribution has been to propose a way of modelling these interdependencies and the associated constraint propagation mechanisms. The first one is the Energy-Precedence Constraints (EPCs), which a partial precedence between activities based on the work quantity needed to define preliminary information. The second one takes into account dependencies between design teams through Contract Dependency Constraints (CDCs).

Based on classical project constraints and these two new project constraints, a constraint satisfaction problem (CSP) model has been defined for supporting activities scheduling decisions. The accurate definition of this model as well as the description of the capabilities it offers can be considered as the third contribution. This model supports the steering of the design activities for each team while validating the consistency of the steering decisions (activities brought forward or set back, allocation of additional resources) through a rigorous tool based on constraints programming. The constraints propagation approaches can be used to validate different simulations in order to be used as a reference for the renegotiation of constraints when such renegotiation becomes mandatory. Each simulation is performed using constraints propagation algorithm and it is considered as a scenario.

Managing these scenarii is one of the capabilities of the application prototype, our last contribution. The detailed specification of the operating procedure for a Decision Support System (DSS) has been the input for the development of a prototype acting as a proof of concept to be shown to end-users in order to get their feed-back. The evaluation actions performed in the framework of a new Airbus product development allowed us to validate only some of the capabilities of our prototype. Indeed, the prototype needs to be evaluated during the whole definition stage of an aircraft and especially during the phase in which recovery plans for the development of some subsystems are required. These recovery plans include the modification of baseline schedules and the renegotiation of the project constraints. This evaluation is part of the perspectives to be developed later on.

4- Perspectives

The investigations carried out in this research project have focused on design activities steering practices at the tactical level of a new aircraft development.

But there still some open issues that could be the initial specification for a future work, they are developed below.

- We must validate the proposed solutions during the complete aircraft definition and development phase. This assessment needs to be performed during critical phases of the aircraft development focusing on teams that are involved in a continuous fire-fighting process.
- We have also developed the case where no possible renegotiation can be performed by the supplier and the customer of a deliverable. Assumption management can be a process that supports practices that already happen in reality. The definition of a rigorous framework in order to deal with these assumptions could enhance even more the collaboration between different design teams.
- We have briefly described the relationship that could be defined between the concept of the design maturity and the energy. This issue needs to be investigated more deeply, focusing on the characterisation of the design maturity concept and the classification of different functions that link both concepts.
- The project scheduling management model described in this report can be enlarged adding other types of constraints. One of them is a generalisation of the EPC, we could call a “generalised EPC constraint”, which states that a given part of the work (energy) associated to activity *i* must be realised before a given part of the work (energy) of a customer activity *j.* We have seen that this constraint could be interesting in the case of second or further deliveries of preliminary information. Another constraint that could be modelled is the possibility of limiting the resource allocated to the same task on a given horizon. Indeed, the current model supports a cumulative resource constraint related to the overall available resources and it also includes the possibility to limit the amount of resources allocated to each period and each activity. A new constraint could focus on a unique activity taking into account more than one period. This constraint could be interesting for the activities that need more stability where important resources variations should be avoided.
- The model assumes a unique resource profile. It should be enlarged to include a multi-resources management.

- Aggregation of time and resources allocated to lower lever teams has not been developed in our work. It could be an interesting field to support even better the managerial levels situated on the top of the program.
- Several improvements on the CSP model can be performed in order to make the solving procedure more efficient. These improvements include the implementation of the best state-of-the-art algorithms in the domain of scheduling with flexible activities, and new constraints propagation techniques.
- Finally and in order to support the time and resources allocation problem solving strategies, accurate heuristics can be developed taking into account the resolutions strategies commonly used by the systems' users.

References

[ACHA, et al. '01] ACHA, V., BRUSONI, S. and PRENCIPE, A., 2001. *Exploring The Miracle: The Knowledge Base in the Aeronautics Industry*. TELL Project (Technological Knowledge and Localized Learning: What Perspectives for a European Policy, Contract HPSE-CT2001-00051.

[AFNOR '94] AFNOR, 1994. *Norme X 50-415: Management des Systèmes - Ingénierie Intégrée - Concepts Généraux et Introduction aux Méthodes d'Application*.

[AFNOR '02] AFNOR, 2002. *Norme X 50-127 : Outils de Management – Maîtrise du Processus de Conception et de Développement*.

[AGGOUN and BELDICEANU '93]AGGOUN, A. and BELDICEANU, N., 1993. *Extending CHIP in Order to Solve Complex Scheduling and Placement Problems*. Mathematical and Computer Modelling, 17, pp.57-73.

[AHMED, et al. '03] AHMED, M., D., SUNDARAM, D. and SRINIVASAN, A., 2003. *Scenario Driven Decision Systems: Concepts and Implementation,* Proceedings of the Eighth CAiSE/IFIP8.1, International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design EMMSAD'03,, June 16-17, 2003, Velden, Austria

[AIRBUS '01a] AIRBUS, 2001a. *ACMR – Airbus Configuration Management Rules*. Internal document.

[AIRBUS '01b] AIRBUS, 2001b. *AP1002, Aircraft Project Management*.

[AIRBUS '03] AIRBUS, 2003. *AIRBUS General Press Presentation*. Internal document.

[AIRBUS '05a] AIRBUS, 2005a. *Les revues dans la logique de déroulement d'un programme. Positionement et characteristiques.* RG.Aéro 000 67A.

[AIRBUS '05b] AIRBUS, 2005b. *Means and Methods AM 2450, Airbus Project Management*.

[AIRBUS '05c] AIRBUS, 2005c. *Means and Methods AM 2454, Schedule Management*.

[AIRBUS '05d] AIRBUS, 2005d. *Means and Methods AM 2457, Project Management and Risk Management*.

[ALLEN '83] ALLEN, J., 1983. *Maintaining Knowledge About Temporal Intervals*. Communications of the ACM, 26(832-843).

[ANDREASEN and HEIN '87] ANDREASEN, M.M. and HEIN, L., 1987. *Integrated Product Development*. Springer-Verlag, Berlin.

[ARTIGUES and ROUBELLAT '00] ARTIGUES, C. and ROUBELLAT, F., 2000. *A polynomial activity insertion algorithm in a multiresource schedule with cumulative constraints and multiple modes*. European Journal of Operational Research, 127(2), pp.297-316.

[ARTIGUES and ROUBELLAT '02] ARTIGUES, C. and ROUBELLAT, F., 2002. *An efficient algorithm for operation insertion in practical jobshop schedules*. Production Planning and Control, 13(2), pp.175-186.

[ATA '99] ATA, 1999. *Spec 100: Aircraft Systems*.

[AVEN '03] AVEN, T., 2003. *Foundations of Risk Analysis: A Knowledge and Decision-Oriented Perspective*. John Wiley & Sons, Chichester.

[BACELLI '93] BACELLI, F., 1993. *A survey on solution methods for Task Graph Models,* Proceedings of the QMIPS Workshop on Formalisms, Principles and State of Arts,

[BAPTISTE and LE PAPE '00] BAPTISTE, P. and LE PAPE, C., 2000. *Constraint Propagation and Decomposition Techniques for Highly Disjunctive and Highly Cumulative Project Scheduling Problems*. Constraints, 5, pp.119-139.

[BAPTISTE, et al. '99] BAPTISTE, P., LE PAPE, C. and NUIJTEN, W., 1999. *Satisfiability Tests and Time-Bound Adjustments for Cumulative Scheduling Problems*. Annals of Operations Research, 92, pp.305-333.

[BAPTISTE, et al. '01] BAPTISTE, P., PAPE, C.L. and NUIJTEN, W., 2001. *Constraint-based Scheduling*. Kluwer Academic Publishers, Norwell.

[BEDILLION and ORR '99] BEDILLION, A.R. and ORR, T.H., 1999. *How Risk Management Has Become a Way of Life in Project Management,* Proceedings of the 30th Annual Project Management Institute, Philadelphia, Pennsylvania

[BENABEN, et al. '04] BENABEN, F., GOURC, D., VILLARREAL, C., RAVALISON, B. and PINGAUD, H., 2004. *Une méthode d'identification des risques : application à un projet coopératif,* Congrès francophone du management de projet "Projet, Entreprise, Intégration", AFIS-AFITEP, AFAV, 6-7 December 2004, Paris, France

[BLANCO, et al. '06] BLANCO, E., CALVI, R., GREBICI, K. and LE DAIN, M.-A., 2006. *How to manage preliminary information exchanged with suppliers in new product development,* Proceedings of the 6th International Conference on Integrated Design and Manufacturing in Mechanical Engineering - IDMME'06, Grenoble, France

[BOSSARD, et al. '97] BOSSARD, P., CHANCHEVRIER, C. and LECLAIR, P., 1997. *Ingénierie Concourante : de la Technique au Social*. Editions ECONOMICA.

[BRUCKER '02] BRUCKER, P., 2002. *Scheduling and constraint propagation*. Discrete Applied Mathematics, 123(1-3), pp.227-256.

[BRUCKER, et al. '94] BRUCKER, P., JURISCH, B. and SIEVERS, B., 1994 *A branch and bound algorithm for the job-shop scheduling problem* Discrete Appl. Math., 49(1), pp.107-127

[BUTTAZZO, et al. '02] BUTTAZZO, G.C., LIPARI, G., CACCAMO, M. and ABENI, L., 2002 *Elastic scheduling for flexible workload management*. Computers IEEE Transactions, 51(3), pp.289-302

[CARRASCOSA, et al. '98] CARRASCOSA, M., EPPINGER, S.D. and WHITNEY, D.E., 1998. *Using the design structure matrix to estimate product development time,* ASME Design Engineering Technical Conferences, 1998, Atlanta, GA

[CASEAU and LABURTHE '94] CASEAU, Y. and LABURTHE, F., 1994. *Improved CLP Scheduling with Task Intervals,* Proceedings of the 11th International Conference on Logic Programming, ICLP'94, [CASEY '05] CASEY, R.J., 2005. *An Innovative Approach to Schedule Management on the F/A-22 Major Defense Acquisition Program (MDAP):Demonstration of Critical Chain Project Management,* Ph.D. Thesis, Faculty of Virginia Polytechnic Institute and State University, Falls Church, Virginia

[CHANTEM, et al. '06] CHANTEM, T., SHARON, X. and LEMMON, M.D., 2006. *Generalized Elastic Scheduling,* 27th IEEE International Real-Time Systems Symposium RTSS '06, Dec. 2006,

[CHAUVET, et al. '98] CHAUVET, F., LEVNER, E. and PROTH, J.M., 1998. *On PERT Networks with Alternatives*. Research Report 3583, INRIA, Le Chesnay.

[CHIANG and FOX '90] CHIANG, W. and FOX, M.S., 1990. *Protection Against Uncertainty In a Deterministic Schedule,* Proceedings of The Fourth International Conference on Expert systems in Production and Operations Management, Hilton Head Island

[CHLEBUS '98] CHLEBUS, E., 1998. *CAX application for process oriented concurrent design*. Journal of Materials Processing Technology, 76(176-181).

[CHURCH and UZSOY '92] CHURCH, L.K. and UZSOY, R., 1992. *Analysis of periodic and event-driven rescheduling policies in dynamic shops*. International Journal of Computer Integrated Manufacturing, 5, pp.153-163.

[CLARK, et al. '87] CLARK, K., CHEW, B. and FUJIMOTO, T., 1987. *Product Development in the World Auto Industry*. Brookings Paper on Economic Activity, 3, pp.729-771.

[CLARKSON and HAMILTON '00]CLARKSON, P.J. and HAMILTON, J.R., 2000. *Signposting: a parameter-driven task-based model of the design process*. Research in Engineering Design, 12(1), pp.18-38.

[COHEN '90] COHEN, J., 1990. *Constraint Logic Programming Languages*. Communications of the ACM, 33, pp.52-68.

[COLMERAUER '90] COLMERAUER, A., 1990. *An introduction to Prolog III*. Communications of the ACM, 33, pp.69-90.

[COLOMBANI '96] COLOMBANI, Y., 1996. *Constraint Programming: An Efficient and Practical Approach to Solving the Job-Shop Problem,* Proceedings 2nd International Conference on Principles and Practice of Constraint Programming,

[COOPER '80] COOPER, K.G., 1980. *Naval ship production: a claim settled and a framework built*. Interfaces, 10(6), pp.20-36.

[DARSES F. '01] DARSES F., F.P., 2001. *La Conception Collective: Une Approche de l'Ergonomie Cognitive*. In Octarès, ed. *Conception et Coopération*, pp. 123-135.

[DEPUY and WHITEHOUSE '01] DEPUY, G.W. and WHITEHOUSE, G.E., 2001. *A simple and effective heuristic for the resource constrained project scheduling problem*. International Journal of Production Research, 39(14), pp.3275-3287.

[DOD '01] DOD, 2001. *Systems Engineering Fundamentals*. Department of Defense Systems Management College, Defense Acquisition University Press, Virginia 22060-5565.

[DORNDORF '02] DORNDORF, U., 2002. *Project Scheduling with Time Windows*. New York.

[DRUMMOND, et al. '94] DRUMMOND, M., BRESINA, J. and SWANSON, K., 1994 *Just-in-case scheduling,* Proceedings of the twelfth national conference on Artificial intelligence, Seattle, Washington

[EISNER '02] EISNER, H., 2002. *Project and systems engineering management*. Wiley, .

[ELKINGTON and SMALLMAN '02] ELKINGTON, P. and SMALLMAN, C., 2002. *Managing Project Risks: A Case Study from the Utilities Sector*. International Journal of Project Management, 20, pp.49-57.

[ENHANCE '02] ENHANCE, 2002. *Codecymo sub-task repport of the Enhance project*.

[EPPEN, et al. '89] EPPEN, G.D., MARTIN, R.K. and SCHRAGE, L., 1989. *A scenario approach to capacity planning*. Operations Research, 37, pp.517-527.

[EPPINGER, et al. '94] EPPINGER, S., WHITNEY, D., SMITH, R.P. and GEBALA, D.A., 1994. *A model-based method for organizing tasks in product developmen*. Research in Engineering Design, 6, pp.1-13.

[EPPINGER '01] EPPINGER, S.D., 2001. *Innovation at the Speed of Information*. Harvard Business Review, 79(1), pp.149-158.

[ESCUDERO, et al. '93] ESCUDERO, L.F., KAMESAM, P.V., KING, A.J. and WES, R.J.-B., 1993. *Production planning via scenario modelling*. Annals of Operations Research, 43, pp.311-335.

[ESQUIROL, et al. '01] ESQUIROL, P., LOPEZ, P. and HUGUET, M.J., 2001. *Propagation de contraintes en ordonnancement*. In Roubellat, P.L.F., ed. *Ordonnancement de la Production*, pp. 131-167, Hermes.

[EVERSHEIM, et al. '97] EVERSHEIM, W., A., R., ZIMMERMANN, H.J. and DERICHS, T., 1997. *Information Management for concurrent engineering*. European Journal of Operational Research, 100, pp.253-265.

[FAULCONBRIDGE and RYAN '03] FAULCONBRIDGE, R. and RYAN, M., 2003. *Managing Complex Technical Projects: a Systems Engineering Approach*. Artech House.

[FORRESTER '61] FORRESTER, J.W., 1961. *Industrial Dynamics*. M.I.T. Press, Cambridge.

[GAO '95] GAO, H., 1995. *Building robust schedules using temporal protection: an empirical study of constraint-based scheduling under machine failure uncertainty.* Masters thesis, Department of Industrial Engineering, University of Toronto, Toronto, Canada.

[GEIA '03] GEIA, 2003. *ANSI/EIA 632 Processes for Engineering a System*.

[GIARD '97] GIARD, V., 1997. *Gestion de projets*.

[GINO '02] GINO, F., 2002. *Complexity Measures in Decomposable Product Structures,* The European Academy of Management, Innovative Research in Management (EURAM 2002), 9-11 May 2002, Stockholm

[GIRARD, et al. '03] GIRARD, P., DESCHAMPS, J.-C. and ZOLGHADRI, M., 2003. *Collaborative engineering design spaces*, 10th ISPE International Conference on Concurrent Engineering: Research and Applications CE’03, ASME. Invited paper, 26-30 July, Madeira, Portugal,

[GIRARD and DOUMEINGTS '04]GIRARD, P. and DOUMEINGTS, G., 2004 *Modelling the engineering design system to improve performance* Computers and Industrial Engineering, 46(1), pp.43-67.

[GOLDRATT '97] GOLDRATT, E., 1997. *Critical Chain*.

[GOLENKO-GINZBURG '88] GOLENKO-GINZBURG, D., 1988. *On the Distribution of Activity Time in PERT* The Journal of the Operational Research Society, 39(8), pp.767-771.

[GOODWIN and WRIGHT '01] GOODWIN, P. and WRIGHT, G., 2001. *Enhancing Strategy Evaluation in Scenario Planning: A Role for Decision Analysis*. Journal of Management Studies, 88, pp.1-16.

[GREBICI, et al. '05] GREBICI, K., BLANCO, E. and RIEU, D., 2005. *Framework for Managing Preliminary Information in Collaborative Design Processes,* Proceedings of the 2nd International Conference on Product Lifecycle Management PLM05, July 11-13, 2005, Lyon, France

[GRUNDER '98] GRUNDER, O., 1998. *Apport de la Modélisation Cognitive à la Planification de Projets,* Ph.D. Thesis, Université de Franche-Comté

[GUTHRIE '04] GUTHRIE, C., 2004. *La Coexistence des Medias de Communication dans le Processus de Conception. Un projet de recherche-action dans une entreprise industrielle* Université Paris 1 Sorbonne,

[HARTMANN and KOLISCH '00] HARTMANN, S. and KOLISCH, R., 2000. *Experimental evaluation of the state-of-the-art heuristics for the resource-constrained projects scheduling problem*. European Journal of Operational Research, 127(2), pp.394-407.

[HAUG '93] HAUG, E.J., 1993. *Concurrent Engineering: Tools and Technologies for Mechanical System Design*. NATO ASI Series, Series F: Computer and Systems Sciences, 108.

[HERROELEN, et al. '98] HERROELEN, W., DEMEULEMEESTER, E. and REYCK, B.D., 1998. *Resource- constrained project scheduling: A survey of recent developments*. Computers & Operations Research, 25(4), pp.279-302.

[HERROELEN and LEUS '01] HERROELEN, W. and LEUS, R., 2001. *On the merits and pitfalls of critical chain scheduling*. Journal of Operations Management, 19(5), pp.559-577.

[HERROELEN and LEUS '04] HERROELEN, W. and LEUS, R., 2004. *Project scheduling under uncertainty: Survey and research potentials.* European journal of operational research., 165(2005), pp.289-306.

[HERROELEN, et al. '02] HERROELEN, W., LEUS, R. and DEMEULEMEESTER, E., 2002. *Critical chain project scheduling: Do not oversimplify*. Project Management Journal, 33(4), pp.48-60.

[HOBDAY '98] HOBDAY, M., 1998. *Product Complexity, Innovation and Industrial Organisation*. Research Policy, Volume 26(Issue 1), pp.689-710.

[KIRKPATRICK, et al. '83] KIRKPATRICK, S., GELATT, C.D. and VECCHI, M.P., 1983. *Optimization by simulated annealing*. Science, 220.

[KLANSNIC and DITTENBERGER '81] KLANSNIC, J.E. and DITTENBERGER, R.J., 1981. *Boeing 757 / 767 Commonality Design Philosophy*. SAE Technical Paper Series, n°810845.

[KOLISCH and HARTMANN '99] KOLISCH, R. and HARTMANN, S., 1999. *Heuristic algorithms for solving the resource-constrained project scheduling problem: Classification and computational analysis*. In Weglarz, J., ed. *Project scheduling: Recent models, algorithms and applications,* pp. 147-178, Kluwer, Amsterdam, the Netherlands.

[KOLISCH and PADMAN '01] KOLISCH, R. and PADMAN, R., 2001. *An integrated survey of deterministic project scheduling*. Omega, 29(3), pp.249-272.

[KOTSIOPOULOS and CASSAIGNE '02] KOTSIOPOULOS, I. and CASSAIGNE, N., 2002. *A project resource management method for a single project with multiple participating organisations*. Systems, Man and Cybernetics, 6(6).

[KUMAR '92] KUMAR, V., 1992. *Algorithms for constraint-satisfaction problems : A survey*. AI Magazine, 13(1), pp.32-44.

[LABORIE '06] LABORIE, F., 2006. *Le concept de salle de décision collective et son application aux processus complexes EADS,* Spécialité Informatique, Univertsité Paul Sabatier, Toulouse

[LARSSON '05] LARSSON, A., 2005. *Engineering Know-Who: Why Social Connectedness Matters to Global Design Teams,* Ph.D. Thesis, Department of Applied Physics and Mechanical Engineering, Division of Computer Aided Design, Luleå University of Technology, Luleå, Sweden

[LASMIS, et al. '03] LASMIS, CRAN and LAP, 2003. *Modèle du processus d'ingénierie*. Intégration Produit, Processus, Organisation pour l'amélioration de la Performance en ingénierie (IPPOP).

[LE PAPE '95] LE PAPE, C., 1995. *Three Mecanisms for Managing Resources Constraints in a Library for Constraint-Based Scheduling,* Proceedings of the INRIA/IEEE Conference on Emerging Technologies and Factory Automation, Paris

[LEACH '99] LEACH, L.P., 1999. *Critical chain project management improves project performance.*. Project Management Journal, 30(2), pp.39-51.

[LEON, et al. '94] LEON, J., WU, S.D. and STORER, R.H., 1994. *Robustness measures and robust scheduling for job shops*. IIE Transactions, 26(5), pp.32-43.

[LI and AICKELIN '03] LI, J. and AICKELIN, U., 2003. *A Bayesian Optimization Algorithm for the Nurse Scheduling Problem,* Proceedings of 2003 Congress on Evolutionary Computation (CEC2003), Canberra, Australia

[LIZARRALDE '03] LIZARRALDE, I., 2003. *Développement distribué d'un système complexe : Le concept de maturité de conception*. Mémoire de DEA de Systèmes Industriels, Ecole doctoral Systèmes, INSAT and CNRS-LAAS.

[LIZARRALDE, et al. '06a] LIZARRALDE, I., ESQUIROL, P. and RIVIERE, A., 2006a. *Adapting project management to complex systems development reality: a maturity and energy constraints based approach,* Proceedings of the 16th CIRP International Design Seminar, July 16-19, 2006, Kananaskis, Alberta, Canada

[LIZARRALDE, et al. '06b] LIZARRALDE, I., ESQUIROL, P. and RIVIERE, A., 2006b. *Project management for the development of complex systems: maturity and energy constraints based approach,* 4ème Conférence Annuelle d'Ingénierie Système, 2006, Toulouse, France

[LIZARRALDE, et al. '07a] LIZARRALDE, I., ESQUIROL, P. and RIVIERE, A., 2007a. *A Decision Support System to schedule design activities in aircraft industry,* 17th International Council on Systems Engineering (INCOSE 2007), June 24 -28, San Diego (USA)

[LIZARRALDE, et al. '07b] LIZARRALDE, I., ESQUIROL, P. and RIVIERE, A., 2007b. *Scheduling the development of a civil aircraft,* International Conference on Industrial Engineering and Systems Management (IESM 2007), May 30-June 2, Beijing (China)

[LOPEZ '91] LOPEZ, P., 1991. *Approche énergétique pour l'ordonnancement de tâches sous contraintes de temps et de ressources.,* Université Paul Sabatier, Toulouse

[LOPEZ, et al. '92] LOPEZ, P., ESQUIROL, P. and ERSCHLER, J., 1992. *Ordonnancement de tâches sous contraintes : une approche énergétique*. RAIRO-APII, 26, pp.453-481.

[LOTTAZ, et al. '00] LOTTAZ, C., SMITH, I.F.C., ROBERT-NICOUD, Y. and FALTINGS, B.V., 2000. *Constraint-based support for negociation in collaborative design*. Artificial Intelligence in Engineering, 14, pp.261-280.

[LUO, et al. '03] LUO, S., WANG, C. and WANG, J., 2003 *Ant Colony Optimization for Resource-Constrained Project Scheduling with Generalized Precedence Relations* Proceedings of the 15th IEEE International Conference on Tools with Artificial Intelligence table of contents,

[LYNEIS, et al. '01] LYNEIS, J.M., COOPER, K.G. and ELS, S.A., 2001. *Strategic management of complex projects: a case study using system dynamics*. System Dynamics Review, 17(3), pp.237-260.

[MACGREGOR, et al. '01] MACGREGOR, S.P., THOMSON, A.I. and JUSTER, N.P., 2001. *A case study on distributed, collaborative design: investigating communication and information flow,* The Sixth International Conference on Computer Supported Cooperative Work in Design, 07/12/2001 - 07/14/2001, London, Ont., Canada

[MALCOLM, et al. '59] MALCOLM, D.G., ROSEBOOM, J.H., CLARK, C.E. and FAZAR, W., 1959. *Application of a technique for research end development program evaluation*. Operations Research, 7, pp.646-669.

[MARLE '02] MARLE, F., 2002. *Modèles d'informations et méthodes pour aider à la prise de décision en management de projet,* Ph.D. Thesis, Ecole Centrale Paris.

[MARTIN '01] MARTIN, G., 2001. *Intégration et Confrontation des Points de Vue dans le Cadre de la Conception en Ingénierie Concourante,* Ph.D. Thesis, Conservatoire National des Arts et Métiers,

[McCORD and EPPINGER '93] McCORD, K.R. and EPPINGER, S., 1993 *Managing the Integration Problem in Concurrent Engineering*. MIT Sloan School of Management Working Paper, no. 3594.

[MEINADIER '02] MEINADIER, J., 2002. *Le Metier d'Integration de Systemes*. HERMES Science, Coll. Management at Informatique.

[MERKLE, et al. '02] MERKLE, D., MIDDENDORF, M. and SCHMECK, H., 2002. *Ant Colony Optimization for Resource-Constrained Project Scheduling*. IEEE Transactions on Evolutionary Computation, 6(4).

[MEYER, et al. '02] MEYER, A.D., LOCH, C.H. and PICH, M.T., 2002. *Managing Project Uncertainty: From Variation to Chaos*. MIT Sloan Management Review, Winter, pp.60-67.

[MINGOZZI, et al. '98] MINGOZZI, A., MANIEZZO, V., RICCIARDELLI, S. and BIANCO, L., 1998. *An Exact Algorithm for the Resource-Constrained Project Scheduling Problem Based on a New Mathematical Formulation*. Source Management Science archive, 44 (5), pp.714-729.

[MODER, et al. '83] MODER, J.J., PHILLIPS, C.R. and DAVIS, E.W., 1983. *Project Management with CPM, PERT and Precedence Diagramming*. Van Nostrand Reinhold, New York.

[MURMAN, et al. '00] MURMAN, E.M., W., M. and E., R., 2000. *Challenges in the Better, Faster, Cheaper Era of Aeronautical Design, Engineering and Manufacturing, The Lean Aerospace Initiative,*. Report n°RP00-02, Massachusetts Institute of Technology.

[NAIM, et al. '04] NAIM, P., WUILLEMIN, P.-H., LERAY, P., POURRET, O. and BECKER, A., 2004. *Réseaux bayésiens*. Eyrolles, Paris.

[NOZIK, et al. '01] NOZIK, L.K., TURNQUIST, M.A. and LIST, G.F., 2001. *Project management under uncertainty with applications to new product development,* Change Management and the New Industrial Revolution; IEMC '01 Proceedings Ithaca, NY

[OZDAMAR and ULUSOY '95] OZDAMAR, L. and ULUSOY, G., 1995. *A Survey on the Resource-Constrained Project Scheduling Problem*. IIE. Transactions, 27, pp.574-586.

[PAPAZOGLOU '01] PAPAZOGLOU, J.Y., 2001. *La Démarche Concurrent Engineering pour le Développement des Nouveaux Airbus*, Conférence C5, MICADO, Paris

[PINSON, et al. '94] PINSON, E., PRINS, C. and RULLIER, F., 1994. *Using tabu search for solving the resource-constrained project scheduling problem,* Proceedings of the 4. International Workshop on Project Management and Scheduling, Leuven

[PMI '00] PMI, 2000. *A Guide to the Project Management Body of Knowledge*. Project Management Institute.

[PRASAD '95] PRASAD, B., 1995. *Concurrent Engineering Fundamentals : Integrated Product and Process Organization.* Prentice Hall.

[PRIEST and SANCHEZ '01] PRIEST, J.W. and SANCHEZ, J.M., 2001. *Product Development and Design for Manufacturing: A Collaborative Approach to Producibility and Reliability*. Marcel Dekker, Inc., New York, NY.

[RAMAT '97] RAMAT, E., 1997. *Modélisation et planification de projets complexes à contraintes de ressources: le modèle RAIH.,* Ph.D. Thesis, Université de Tours École d'Ingénieurs en Informatique pour l'Industrie,

[REPENNING '01] REPENNING, N.P., 2001. *Understanding fire fighting in new product development*. Journal of Product Innovation Management, 18(5), pp.285-300.

[RODDEN and BLAIR '91] RODDEN, T. and BLAIR, G.S., 1991. *CSCW and Distributed Systems: The Problem of Control.* Proceedings of the Second European Conference on Computer-Supported Cooperative Work (ECSCW'91), Amsterdam, Netherlands

[ROGERS '03] ROGERS, E.M., 2003. *Diffusion of innovation*. Simon and Schuster, New York.

[ROY '62] ROY, B., 1962. *Graphes et ordonnancement*. Revue Française de Recherche Opérationelle, 25, pp.323-333.

[SABBAGH '95] SABBAGH, K., 1995. *21st Century Jet, The Making of the Boeing 777*. London.

[SAINT-MARC '06] SAINT-MARC, L., 2006. *Methods for Managing Collaboration*. Internal Report EADS-CRC, 07051-1-DCR/STI.

[SAINT-MARC, et al. '04] SAINT-MARC, L., CALLOT, M., REYTEROU, C., MOLY, M., GIRARD, P. and DESCHAMPS, J.-C., 2004. *Toward a data maturity evaluation in collaborative design processes,* Proceedings of the 8th International Design Conference DESIGN04, 2004, Dubrovnik

[SCHEIBLE '02] SCHEIBLE, R.S., 2002. *Airbus Concurrent Engineering*. ENHANCE Project, Forum 3.

[SCHOEMAKER '95] SCHOEMAKER, P.J.H., 1995. *Scenario planning: A tool for strategic thinking*. Sloan Management Review, 36(2), pp.25-40.

[SMITH '95] SMITH, S., 1995. *Reactive Scheduling Systems*. In Scherer, B.a., ed. *Intelligent Scheduling Systems*, pp. 155-192, Kluwer Press.

[STEWARD '81] STEWARD, D.V., 1981. *The design structure matrix: A method for managing the design of complex systems*. IEEE TRANSACTIONS ON ENGINEERING MANAGEMENT, 6(1994), pp.1-13.

[STINSON, et al. '78] STINSON, J.P., DAVIS, E.W. and KHUMAWALA, B.M., 1978. *Multiple resource-constrained scheduling using branch and bound*. AIIE Transactions, 10, pp.252-259.

[TALBOT '01] TALBOT, D., 2001. *Proximités et Dynamiques des Relations de Sous-Traitance: Le Cas d'EADS Airbus à Toulouse,* Colloque de l'Association Française de Comptabilité, Mai 2001, Metz, France

[TAVARES '02] TAVARES, L.V., 2002. *A review of the contribution of Operational Research to Project Management*. European Journal of Operational Research, 136(1), pp.1-18.

[TERWIESCH and LOCH '97] TERWIESCH, C. and LOCH, C.H., 1997. *Management of Overlapping Development Activities: A Framework for Exchanging Preliminary Information,* Proceedings of the 4th EIASM Conference on Product Development,

[TERWIESCH, et al. '02] TERWIESCH, C., LOCH, C.H. and MEYER, A.D., 2002. *Exchanging Preliminary Information in Concurrent Engineering: Alternative Coordination Strategies*. Organization Science, 13(4), pp.402-419.

[THOMAS and SALH '98] THOMAS, P.R. and SALH, S., 1998. *A tabu search approach for the resource constrained project scheduling problem*. Journal of Heuristics, 4(2), pp.123-139.

[THOMSON, et al. '00a] THOMSON, A.I., MACGREGOR, S.P. and ION, W.J., 2000a. *An Evaluation and Comparison of the Industrial and Educational Usage of CSCW within the Design Process*, IEEE 9th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WET ICE'00),

[THOMSON, et al. '00b] THOMSON, A.I., MACGREGOR, S.P. and ION, W.J., 2000b. *An evaluation and comparison of the industrial and educational usage of CSCW within the design process,* Proeedings. IEEE 9th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, (WET ICE 2000), 14-16 June,

[TURNQUIST and NOZIK '03] TURNQUIST, M.A. and NOZIK, L.K., 2003. *Allocating time and resources in project management under uncertainty,* System Sciences; Proceedings of the 36th Annual Hawaii International Conference,

[ULRICH '95] ULRICH, K., 1995. *The Role of Product Architecture in the Manufacturing Firm*. Research Policy, 24, pp.419-440.

[VAN DE VONDER, et al. '05] VAN DE VONDER, E., DEMEULEMEESTER, E., HERROELEN, W. and LEUS, R., 2005. *The use of buffers in project management: The trade-off between stability and makespan*. Int. J. Production Economics, 97 pp.227-240

[VAN DE VONDER, et al. '06] VAN DE VONDER, E., HERROELEN, W. and LEUS, R., 2006. *The trade-off between stability and makespan in resource-constrained project scheduling*. International Journal of Production Research, 44(2), pp.125-236.

[VIDAL and FARGIER '97] VIDAL, T. and FARGIER, H., 1997. *Contingent durations in temporal CSP : from consistency to controllabilities*, 4th Int. Workshop on Temporal Representation and Reasoning (TIME97), May 10-11 Daytona Beach (Florida), USA

[VIEIRA and HERRMANN '03] VIEIRA, G.E. and HERRMANN, J.W., 2003. *Rescheduling Manufacturing Systems: A Framework of Strategies, Policies, and Methods* Journal of Scheduling, 6(1), pp.39-62.

[WANG, et al. '05] WANG, H., LIN, D. and LI, M., 2005 *A Genetic Algorithm for Solving Resource-Constrained Project Scheduling Problem. Lecture Notes in Computer Science* Springer Berlin / Heidelberg

[WEBER '05] WEBER, C., 2005. *What is "complexity"?*, International Conference on Engineering Design ICED, Melbourne

[WEGLARZ '81] WEGLARZ, J., 1981. *Project scheduling with continuously-divisible, double constrained resources.* Management Science, 27, pp.1040-1053.

[WINKLER '96] WINKLER, R.L., 1996. *Uncertainty in probabilistic risk assessment* Reliability Engineering and System Safety, 54(2), pp.127-132.

[WINNER, et al. '88] WINNER, R.I., PENNELL, J.P., BERTREND, H.E. and SLUSARSZUCK, M.M.G., 1988. *The Role of Concurrent Engineering in Weapons System Acquisition*. IDA Report R-338, Institute for Defense Analyses.

[WOMACK, et al. '91] WOMACK, J., JONES, D. and ROOS, D., 1991. *The Machine that Changed the World*. Harper Collins, New York.

[WU, et al. '99] WU, S., BYEON, D., EUI-SEOK and STORER, R.H., 1999. *A graph-theoretic decomposition of the job shop scheduling problem to achieve scheduling robustness*. Operations Research, 27(1), pp.113-124.

[WYNN, et al. '05] WYNN, D., ECKERT, C. and CLARKSON, P.J., 2005. *Modelling and simulating iterative development processes*, International Conference on Engineering Design ICED 05, Melbourne

[WYNN, et al. '06] WYNN, D.C., Eckert, C.M. and Clarkson, P.J., 2006. *Applied Signposting: A Modeling Framework to Support Design Process Improvement*, Proceedings of the 2006 ASME International Design Engineering Technical Conferences & Computers and Information In Engineering Conference, Philadelphia

Annexes

Annexe 1: Assumptions management

An assumption is a decision that is taken expecting that it will be true. Therefore, there is a risk associated to the assumption; this risk is related to the fact that real progress could be different of the choice settings. The process related to the contract negotiation can therefore be modified as described on Figure 73:

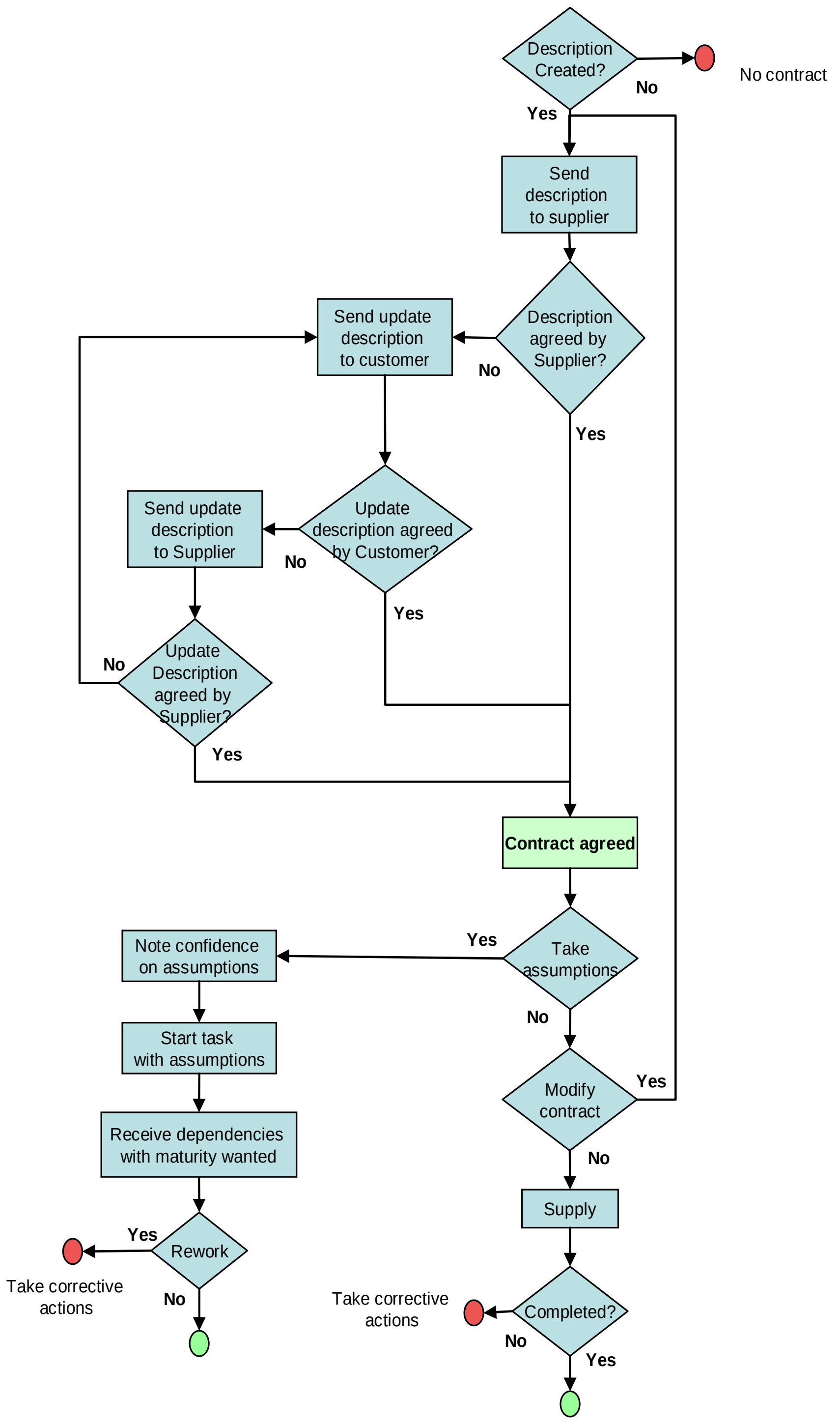

Description Created?
No
No contract
Yes
Send description to supplier
Description agreed by Supplier?
No
Send update description to customer
Yes
Update description agreed by Customer?
No
Send update description to Supplier
Yes
Update Description agreed by Supplier?
No
Yes
Contract agreed
Take assumptions
Yes
Note confidence on assumptions
Start task with assumptions
Receive dependencies with maturity wanted
Rework
Yes
Take corrective actions
No
No
Modify contract
Yes
No
Supply
Completed?
No
Take corrective actions
Yes

Figure 73: Contract management process with assumptions definition

Once the contract is agreed, there is no further modification or renegotiation. If supplier is not able to supply the data respecting the date and maturity level defined by the customer, an assumption has to be taken by the customer. Assumptions are usually defined taking into account former deliveries and former projects experience. Customer can ask the supplier participation in the assumption definition. Nevertheless the later is not accountable of the assistance provided to the customer.

Customer can measure the risk level of the assumption measuring the confidence level of the supplier. This process is very subjective and can have negative consequences which are usually reflected with a rework activity to be performed by the customer. Even if it is subjective, it is a well-known process in concurrent based product development. but only few investigation led to the formalisation of this process [TERWIESCH and LOCH '97, TERWIESCH, et al. '02].

If supplier is not able to deliver the data for the date t_{ij}, the customer can define an assumption related to the data that he or she was supposed to receive and begin the task as foreseen. When requested maturity level is delivered (t_{ij}^*), the customer will be able to determine is the assumption he or she defined was exact or not. In the first case, work progress will go on without being impacted by the delayed delivery. Nevertheless, in the case where assumption was not exact, the supplier will have to do a rework taking into account new data. This rework is an amount of energy that will be added to the scheduling problem, usually with serious effects on the new schedule definition.

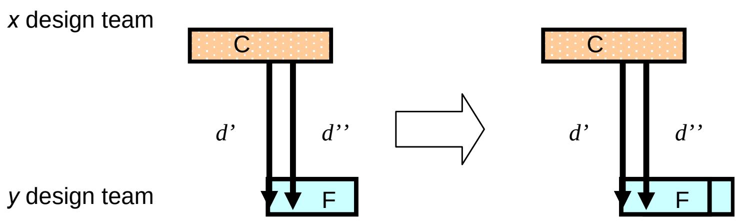


Figure 74: Rework due to assumption failure

Figure 74 represents the case where the assumption is not exact and task *F* remakes the work performed until the delivery of the *d''.* Therefore, the task *F* will expend more energy that foreseen at the beginning. This fact can affect the scheduling of tasks after *F.*

For each assumption a probability of doing rework is defined (P_{ij}). The user can foresee two schedules, one considering the assumption as correct and the other one anticipating the assumption failure. More than one assumption will lead to several schedules. Indeed, every time the user has to make an assumption, two possible schedules appear depending on whether the assumption is correct or not. This gives us 2^n schedules (with n = assumptions quantity). For each schedule, a probability can be calculated taking into account the combination of probabilities related to each assumption.

Consider P_{AC} as the probability of doing rework related to the delivery of data d_1. Being $d_1 \in Input(C)$ and $C \in Tasks(y)$.

Also, consider P_{BD} as the probability of doing rework related to the delivery of data d_2. Being $d_2 \in Input(D)$ and $D \in Tasks(y)$.

Since two assumptions are taken, four different schedules can be built. The probability related to each schedule will be calculated as follows:

Schedule 1: $P_{sch1} = P_{AC} * P_{BD}$

Schedule 2: $P_{sch2} = (1-P_{AC}) * P_{BD}$

Schedule 3: $P_{sch3} = P_{AC} * (1-P_{BD})$

Schedule 4: $P_{sch4} = (1-P_{AC}) * (1-P_{BD})$

First schedule is related to the most negative scenario, where both assumptions fail and an important amount of energy is added to the scheduling problem due to the rework of part of the tasks *j* and *l*.

Schedule 4 deals with the case where both assumptions are correct and therefore the first schedule remains valid. Therefore the probability of maintaining valid the first schedule is (P_v):

$$P_v = \prod_{j \in Tasks(y)} (1 - P_{ij})$$

While the catastrophic scenario, meaning that all assumptions fail, will be calculated as follows:

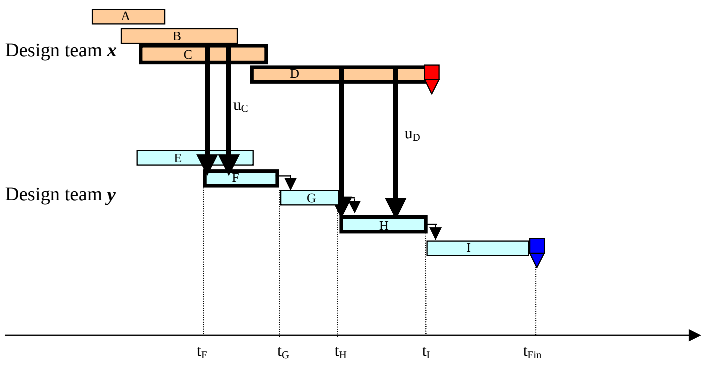
$$P_{neg} = \prod_{j \in Tasks(y)} P_{ij}$$

Depending on the consequences of each schedule, these practices allow the user to calculate the probabilities of respecting certain constraints. For example, the due date of the quality gate can be overstepped by certain schedules. Adding the probabilities related to each schedule, the user can announce the probability of respecting the quality gate date. Therefore, following the example of four possible schedules, if only schedule 1 and 2 respect the quality gate, we can conclude that taking into account the risk of the two assumptions the probability of non respecting the quality gate date is $P_{qgrisk} = P_{sch3} + P_{sch4}$.

This value is a very interesting value since it can be used in risk management exercises. The gravity of not respecting a quality gate can be measured taking into account the impact that it will have in the overall project development; nevertheless, the likelihood of non respecting is much more difficult to calculate. Without taking into account the unexpected events and only dealing with the assumptions, we have proposed a methodology to calculate accurately this probability value.

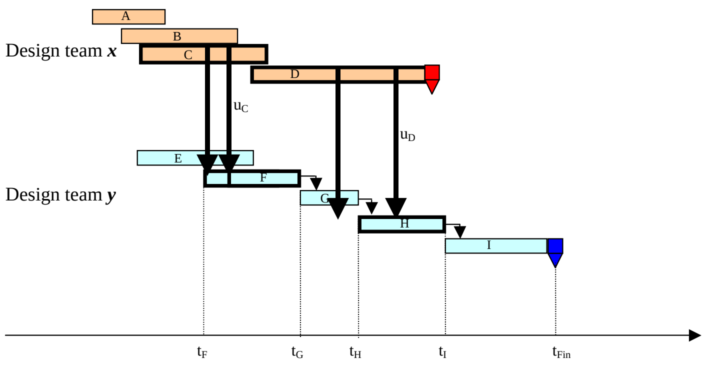
In the following example we describe a very simple schedule where rectangular tasks are considered. This example illustrates the methodology explained here above.

In this example the beginning of two tasks F and H of a design team *y* require deliverables from another design team *x*. The start dates t_F and t_H cannot be postponed. In addition, there are classical precedence constraints for tasks G, H and I. Therefore, the design team makes assumptions to enable the start of tasks F and H at the scheduled date. Note the difference between the 4 possible schedules (depending on the re-work phases) and the change to the end date of the last task.



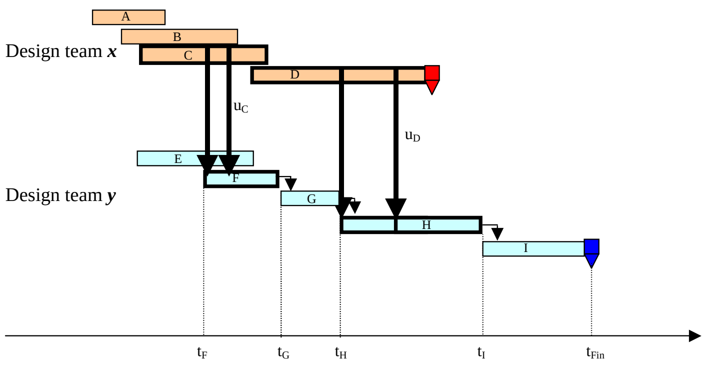
Schedule 1: no rework

In the 1st Schedule, the assumptions made to start tasks F and H are correct. There is no rework and Task I can be completed at the planned date.



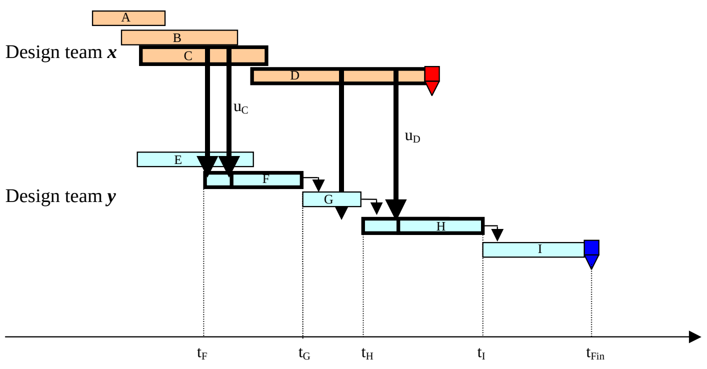
Schedule 2: rework at F

In the 2nd Schedule, the assumptions made to start Task H are correct but those made to start Task F are false. There is some rework for Task F. The constraints linked to this task are propagated and Task I finishes later.



Schedule 3: rework at H

In the 3rd Schedule, the assumptions made to start Task F are correct but those made to start Task H are false. There is some rework for Task H. The constraints linked to this Task are propagated and Task I finishes even later than in the previous case.



Schedule 4: rework at F and H

In the 4th Scenario, the assumptions made to start Tasks F and H are false. There is some re-work for Tasks F and H. The constraints linked to these tasks are propagated and Task I finishes later.

Remark: in Schedule 3 and 4 Task I finish at the same date because of the fact that resources are allocated constantly to each task. The rework period of Task H absorbs the lateness due to the rework phase of Task F.

In order to assess the likelihood that one scenario will occur rather than another, a probability has to be given to each scenario. The following probabilities are defined:

- o Rework probabilities for Task *F*: 90%
- o Rework probabilities for Task *H*: 20%
- o

Therefore we can calculate the probabilities for each schedule:

- o Schedule 1: all assumptions are correct

$$P_1 = 0.1 \times 0.8 = 8\%$$

- o Schedule 2: the assumptions for F are false, those for H are correct

$$P_2 = 0.9 \times 0.8 = 72\%$$

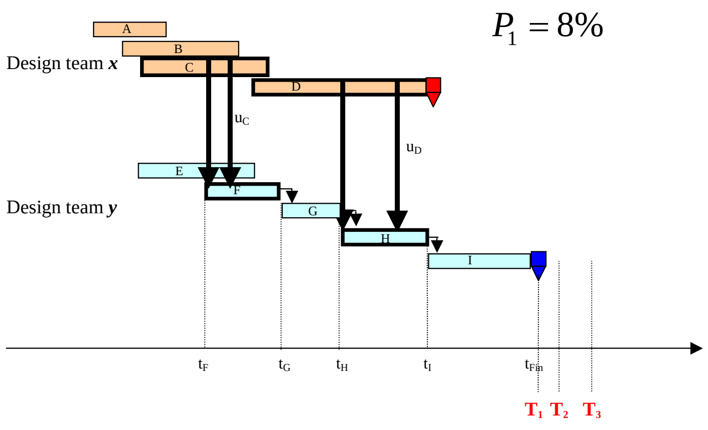
- o Schedule 3: the assumptions for F are correct, those for H are false

$$P_3 = 0.1 \times 0.2 = 2\%$$

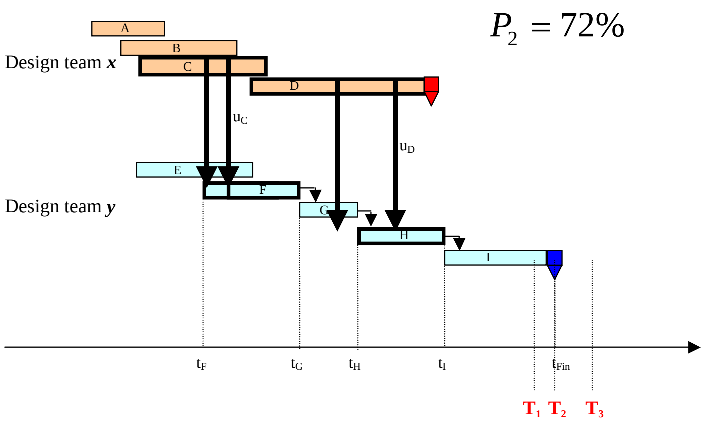
- o Schedule 4: all the assumptions are false

$$P_4 = 0.9 \times 0.2 = 18\%$$

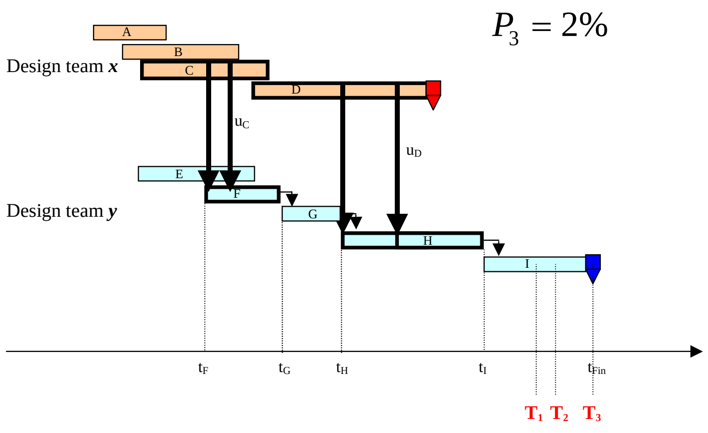
Taking into account these probabilities, we will focus on the completion date of the last task (Task *I*).



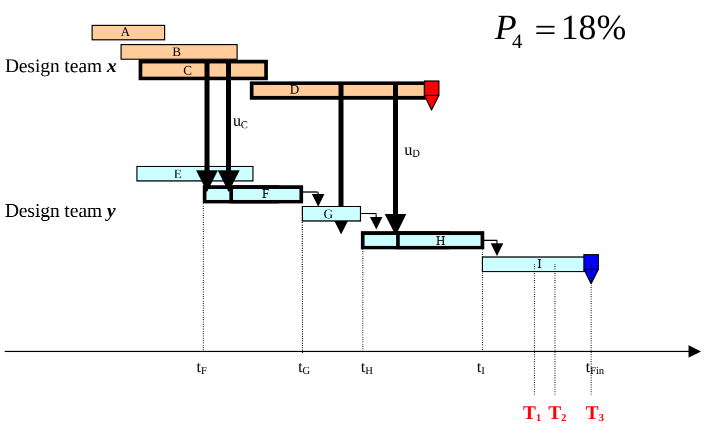
Schedule 1 : no rework



Schedule 2 : rework at F



Schedule 3 : rework at H



Schedule 4 : rework at F and H

Using the probabilities for each scenario, we can calculate the probability of finishing Task I at the latest at date *t*:

At the latest at T_1: $P_{T_1} = P_1 = 8\%$

At the latest at T_2: $P_{T_2} = P_1 + P_2 = 80\%$

At the latest at T_3: $P_{T_3} = P_1 + P_2 + P_3 + P_4 = 100\%$

These calculations therefore show us that in the worst-case scenario, task *I* will finish at latest at T_3 ($P_{T_3} = 100\%$). However, we note that there is a high probability that it will finish at the latest at T2 ($P_{T_2} = 80\%$).

The fact of dealing only with rectangular tasks without accepting a non constant resource allocation, make possible only three possible end dates and not four, since in the case where both assumptions fail, second rework task will need to perform less work due to the fact that the task has begin later. This is the consequence of dealing with time periods rather that energy amounts. In our model this is not the case since energy allocation is not constrained to be constant during the task execution.

Annexe 2: Functional requirements

In this annexe we will detail the functional requirements defined with Airbus for the SPEED project. SPEED is the acronym that has been used inside Airbus to name our research project. We have identified 19 functional requirements linked to two main roles as well as to three main use cases.

The two roles identified for this exercise are on the one hand the leader of a team in the lowest level of the organisation which is designed as the design team and the head of a management team which is at least one level above the design team.

Concerning the three main use cases that we have identified, the first one deals with the scheduling and resources allocation of a team (no matter the level). Secondly, we consider the collaboration process when data needs to be exchanged from a horizontal point of view as well as a vertical point of view. And thirdly we consider the uncertainties linked to task definition, resources capacity and inputs delivery.

Dealing with the scheduling and resources allocation aspects related to the design team leader user, we have identified the following functional requirements:

- FR-001: SPEED project shall develop resources allocation methods and tools that allow the leader of a design team organise the different Work Packages inside its team and manage the resources that will be allocated at each period to each Work Package.
    - Rationale: Work Packages include the main subsystems to be developed in the frame of a design team where resources will be allocated during each phase of the development.
- FR-002: SPEED project shall develop engineering activities scheduling methods and tools that allow the leader of a design team define the start and end dates for each activity.
    - Rationale: Scheduling includes defining the order in which the activities will be performed as well as fixing these activities in a time window so that the different actors know in advance the foreseen work.
- FR-003: SPEED project shall develop scheduling management methods and tools that support robustness in scheduling process.
    - Rationale: Very often, schedules are accurate and informative, but static and very sensitive to unforeseen events. Indeed, a design team might have to re-evaluate a new context and update the schedule each time an unforeseen event is detected. The update frequency can be high considering the dynamic characteristic of the design process.
- FR-004: SPEED project shall develop scheduling management methods and tools that allow the definition of quality gates as well as the requirements associated to each quality gate.
    - Rationale: Quality gates must be part of each schedule. Tasks defined between two quality gates must be defined taking into account requirements of each quality gate.

Dealing with the scheduling and resources allocation aspects related to the head of a management team user, we have identified the following functional requirements:

- FR-005: SPEED project shall develop scheduling management and resources allocation methods and tools that support the aggregation of resources and allocated workload considering all the teams included in a management team.

    - Rationale: Each management team contains different management teams of inferior level or design teams. In order to calculate overall resources or workload, allocation work that has already been done in this teams must be considered.

- FR-006: SPEED project shall develop engineering activities scheduling methods and tools that allow the definition of mayor phases decided at program level as well as the key program milestones.
    - Rationale: The definition of mayor phases decided at program level as well as the key program milestones allow to define internal tasks without been regardless of global development. The overall project will be delayed if only one of the subsystems is delayed.
- FR-007: SPEED project shall develop scheduling management methods and tools that allow the definition of quality gates at different managerial levels, identifying the ones that correspond to a “rendez-vous” between one or more teams.
    - Rationale: The identification of the “rendez-vous” milestones allow the head of the management team control the progress of each team been part of its scope based on a temporal reference.
- FR-008: SPEED project shall develop scheduling management methods and tools that allow each managerial levels control not recurrent cost.
    - Rationale: Not recurrent cost deals mainly with resources allocated to each level. Moreover, in engineering project, major part of the resources from a cost point of view is related to human resources.

Concerning horizontal and vertical collaboration aspects related to the design team leader user, we have identified the following functional requirements:

- FR-009: SPEED project shall develop contract management methods and tools that support the data delivery negotiation process between the distributed design teams during Airbus development programs.
    - Rationale: Data negotiation process is a collaborative process that needs a specific method in order to seek the more efficiently possible a compromise between both actors. Moreover a specific module of the tool will support this negotiation mainly based on a date and maturity level decision.
- FR-010: SPEED project shall develop project management methods and tools that allow to link scheduling and contracts management processes between them.
    - Rationale: A connexion between the scheduling module and the module that supports the contract negotiation will be necessary in order to propagate modifications either in the schedule or in the contract and will allow both processes be consistent between them.
- FR-011: SPEED project shall develop contract management methods and tools that support the definition of assumptions related to the data that is going to receive the customer.
    - Rationale: Assumptions definition in the tool, will allow to formalise a current concurrent engineering practice and to add flexibility to the product development process while performing a risk assessment.
- FR-012: SPEED project shall develop interdependencies management methods and tools that support criticality (the likelihood of not respecting the interdependency, related to the impact of that fact in the project) of each interdependency.

- Rationale: The aim of this type of categorization is to assist in the early identification of risks and their assessment and determine the level of management focus required.

Concerning horizontal and vertical collaboration aspects related to the head of a management team user, we have identified the following functional requirements:

- FR-013: SPEED project shall develop scheduling management methods and tools that can be deployed at different levels of the Airbus organisation.
  - Rationale: Schedules used in the design process need to be designed for and managed at different levels of the organisation. Links between different levels should be used to cascade project milestones to lower levels but also to escalate teams' constraints and progress to upper levels. Both cascading and escalation processes can be time consuming and sources of errors.
- FR-014: SPEED project shall develop scheduling management methods and tools that support schedules cascading process.
  - Rationale: The reality of Airbus development programs is that organisation structure has different organisational and management levels with different missions and responsibilities. Links between different levels shall take into account not only a top down approach but also a bottom up approach. Indeed, on one hand, top-level targets shall be cascaded effectively to different organisational levels, and on the other hand, a design team shall be able to measure work progress taking into account the progress of teams situated in the level below. These activities are time consuming and can be a source of errors.
- FR-015: SPEED project shall develop project management methods and tools that support management teams decision support process by making available Key Performance Indicators related to activities progress and contracts status.
  - Rationale: Head of different management levels need a at a glance vision of the activity progress of teams in its scope. Moreover status of the contracts between these teams will allow him or her identify possible source of inconsistency cases.

Dealing with the uncertainty aspects related to the design team leader user, we have identified the following functional requirements:

- FR-016: SPEED project shall develop methods and tools to link scheduling and interdependencies management processes with risk management processes.
  - Rationale: Risks are currently managed using specific methods and tools and are not directly supported by scheduling and contract management tools.
- FR-017: SPEED project shall develop scheduling management methods and tools that support alternatives management.
  - Rationale: At the design stage, different alternatives have to be managed simultaneously through different planning scenarii. Schedules should support the evaluation of the different scenarii and be used within the decision-making process. Schedules are currently considered as objects to be updated as a consequence of decisions rather than objects enabling the preparation and consolidation of decisions.
- FR-018: SPEED project shall develop scheduling management methods and tools that support buffers and margins management in scheduling process.
  - Rationale: In the case that the problem solving method finds a solution, available margin concerning an accurate constraint can be useful for leader of

the team in order to identify tight periods and periods where constraints are largely respected.

Dealing with the uncertainty aspects related to the head of a management team user, we have identified the following functional requirements:

- FR-019: SPEED project shall develop scheduling management and resources allocation methods and tools that support the head of a management team in the trade offs concerning critical resources allocation.
    - Rationale: Critical resources allocation includes supplementary resources hiring when tight periods arise. The allocation of these resources to the right teams needs to deal uncertainties concerning the real needs of each team.

Annexe 3: Technical requirements

Technical requirements are specific requirements issued from the functional requirements. These constraints concern the tool development.

- TR-001: SPEED Tool shall allow illustrating the task and milestone information varying task forms and date and titles position taking into account the specificities of different Airbus skills.
    - Rationale: The tool is going to be deployed into different development groups as well as different countries.
- TR-002: SPEED Tool shall be able to illustrate the activities of the development as well as the main milestones that could be common for different groups.
    - Rationale: Common milestones need to be modified in all the schedules where has been deployed. It is the case for example of "rendez voues" milestones.
- TR-003: SPEED Tool shall be able to illustrate the resources allocations that has been decided by the development group related to an activity.
    - Rationale: Resources allocation will be later a key point for the energy oriented method that will be utilised.
- TR-004: SPEED Tool shall be able to illustrate the milestones related to contracts defined between two or more teams.
    - Rationale: It is been discussed the accurate information that has to be illustrated relating the contract.
- TR-005: SPEED Tool shall be able to illustrate the constraints that have been defined for the scheduling model.
    - Rationale: This requirement is related to the model that has been proposed for scheduling design activities.
- TR-006: SPEED Tool shall be able to illustrate the solutions that satisfy all the constraints defined for the scheduling model.
    - Rationale: This requirement is related to the model that has been proposed for scheduling design activities.
- TR-007: SPEED Tool shall be able to illustrate the margins of the solutions if there are.
    - Rationale: This requirement is related to the model that has been proposed for scheduling design activities.
- TR-008: SPEED Tool shall be able to propose a way to define a hierarchy of constraints relaxing.
    - Rationale: This requirement is related to the model that has been proposed for scheduling design activities.
- TR-009: SPEED Tool shall be able to relax the first constraints defined in the hierarchy list in order to find a solution.
    - Rationale: The first constraint could be relaxed in a way that could not be implemented n reality.
- TR-010: SPEED Tool shall be able to illustrate the process for relaxing constraints

    - Rationale: The used will easily realise the modifications until a solution id found.

- TR-011: SPEED Tool shall be able to save a scenario that has been defined by the user.
    - Rationale: This requirement is related to the model that has been proposed for scheduling design activities.
- TR-012: SPEED Tool shall be able to compare different scenarios depending on the constraints that have been relaxed in order to choose the best one.
    - Rationale: This requirement is related to the model that has been proposed for scheduling design activities.
- TR-013: SPEED Tool shall be able to relax automatically some constraints in order to find a feasible and acceptable solution.
    - Rationale: This is an automatic mode with no interaction with the user.
- TR-014: SPEED Tool shall allow illustrating the task considering the energy needed to perform the task as well as earliest start date and latest end date.
    - Rationale: The tasks are not defined by the duration but the energy needed to perform it, nevertheless in order to constraint the task from a time point of view a time window is defined.

Annexe 4: ECLIPSe-Prolog Kernel Code

4.1- Visualisation

```
spy_Table(_M, [_Nb_Activites, _Nb_Semaines], no).

spy_Table(M, [Nb_Activites, Nb_Semaines], yes) :-
      (for(Num_Activite,1,Nb_Activites), param(M,Nb_Semaines)
      do
            (for(Num_Semaine,1,Nb_Semaines), param(M, Num_Activite)
            do
                  X is M[Num_Activite, Num_Semaine],
                  monitor(X,Num_Activite,Num_Semaine)
            )
      ).

monitor(X,A,S) :-
      suspend(report(X,A,S), 1, [X->constrained, X-> inst]).

report(X,A,S) :-
      var(X),
      dom(X,D),
      write('Domaine   activite'),write(A),
      printf(" semaine%3d = ", [S]),
      writeln(D),
      suspend(report(X,A,S), 1, [X->constrained, X-> inst]).

report(X,A,S) :-
      nonvar(X),
      write('Intensite activite'),write(A),
      printf(" semaine%3d = ", [S]),writeln(X).


      %*********************%
      %    Print solution   %
      %*********************%

print_Table(M,Nb_Activites,Nb_Semaines) :-
      nl,
      write('   \\ sem'),
      (for(N, 1, Nb_Semaines)
      do
            printf(" %2d ",[N])
      ),
      nl,
      write('act \\---'),
      (for(_N, 1, Nb_Semaines)
      do
            write('----')
      ),
      nl,
      (for(Num_Activite,1,Nb_Activites), param(M,Nb_Semaines)
      do
            printf("%4d     ",[Num_Activite]),
            (for(Num_Semaine,1,Nb_Semaines), param(M, Num_Activite)
            do
                  X is M[Num_Activite, Num_Semaine],
                  dvar_domain(X,D),
                  dom_range(D,Min,Max),
                  print_var(Min,Max)
            ),
            nl
      ).

print_var(Min,Max) :-
      (Min=Max -> (Min=0 -> write('    ')
                                    ;
                              write(' '), write(Max),write('  ')
                        )
                  ;
                  write(Min), write('~'), write(Max), write(' ')
      ).
```

4.2- Predicats

```
:- lib(fd_search).

      %*********************
      % Methode de résolution
      %*********************



instancierplustot([]).
instancierplustot(L):-
      deleteff(X,L,Xs),     % selection de la variable la plus contrainte
      indomain(X,max),            % X prend la valeur max de son domaine
      instancierplustot(Xs).

/*
instancierplustot([]).
instancierplustot([X|Xs]):-   % sélection simple (dans l'ordre gauche
droite)
            indomain(X,max),       % X prend la valeur max de son domaine
            instancierplustot(Xs).
*/
```

4.3- Data

```
      %*****************
      % FICHIER EXEMPLE *
      %*****************

      %***************************
      % DESCRIPTION DES ACTIVITES *
      %***************************
            % activite( ?Nom, ?Energie, ?debut_plus_tot, ?fin_plus_tard).

Activite(1,’A’, 20,  1, 9).
Activite(2,’B’, 20,  1, 13).
Activite(3,’C’, 16,  4, 18).
Activite(4,’D’, 10,  6, 17).
Activite(5,’E’,  8,  6, 18).
Activite(6,’F’,  8, 10, 19).
Activite(7,’G’,  4, 12, 20).
Activite(8,’H’,  3, 15, 20).


Intensite_maximum(_Activite, 3).
```

```
            % energie totale requise : = 90


      %****************************
      % DESCRIPTION DES RESSOURCES *
      %****************************
            % horizon( ?debut_horizon, ?fin_horizon)
            % disponibilite( ?numero_semaine, ?intentisité_maximale).

Horizon(1,20).

Disponibilite(_NS, 5).
% :-            % 5 par exemple …
%     horizon(Inf,Sup),
%     NS :: Inf..Sup,
%     indomain(NS).

            % energie totale disponible : 5x(20) =100
```

4.4- Constraints

```
      %***************************
      % CONTRAINTES DE PRECEDENCE *
      %***************************
            % precede( ?activite1, ?activite_2,
energie_minimale_1_avant_debut2)

precede('B','A',6).
%precede('A','C',20).
%precede('B','E',8).
%precede('B','F',8).

            % succede( ?activite1, ?activite_2,
energie_minimale_2_apres_fin1)


contraintes_dates_limites(M, Liste_activites, Nb_activites, Nb_semaines) :-
      (foreach(A,Liste_activites),for(Num_activite,1,Nb_activites),
param(M,Nb_semaines)
      do
            activite(_,A,_,R,D),

                  % contrainte de debut au plus tot
            R2 is R-1,
            (for(Num_semaine,1,R2), param(M,Num_activite)
            do
                  X is M[Num_activite,Num_semaine],
                  X #= 0
            ),

                  % contrainte de fin au plus tard
            D2 is D+1,
      (for(Num_semaine,D2,Nb_semaines), param(M,Num_activite)
          do
            X is M[Num_activite,Num_semaine],
                  X #= 0
          )

      ).
```

```
Contraintes_cumul(M, Nb_activites, Nb_semaines) :-
      (for(Num_semaine,1,Nb_semaines),param(M,Nb_activites)
      do
            L is M[1..Nb_activites, Num_semaine],
            flatten(L, List_resources_pour_semaine),
            disponibilite(Num_semaine, Max_Domain),        %défini dans le
fichier de donnees
            (foreach(Intensite, List_resources_pour_semaine),
fromto(0,In,Out,Cumul)
                                                             do Out =
Intensite+In
            ),
            Cumul #<= Max_Domain
      ).


Contraintes_energie(M, Liste_activites, Nb_Activites, Nb_Semaines) :-
      (foreach(A,Liste_activites),for(Num_activite,1,Nb_Activites),
param(M,Nb_Semaines)
      do
            activite(_,A,Energie,_,_),
            L is M[Num_activite, 1..Nb_Semaines],
            flatten(L, Distribution_intensite_tache),
            (foreach(Intensite, Distribution_intensite_tache),
fromto(0,In,Out,Energie_tache)
                                                             do Out =
In+Intensite
            ),
            Energie_tache #= Energie
      ).

Contraintes_prec_ener(M, Nb_Semaines) :-
      findall([A,B,Eab], precede(A,B,Eab), Liste_Precedences),
      (foreach([I,J,Eij],Liste_Precedences), param(M,Nb_Semaines)
      do
            activite(Num_I,I,_Ei,_Ri,_Di),
            activite(Num_J,J,_Ej,_Rj,_Dj),

            Liste_Ai is M[Num_I, 1..Nb_Semaines],
            Liste_Aj is M[Num_J, 1..Nb_Semaines],
                  % Propagation uniquement sur l’activité I

            suspend(algorithme1(Liste_Ai,Liste_Aj,Eij), 2, Liste_Ai ->
constrained)

            %,

                  % Propagation uniquement sur l’activité J
            %Ereste is Ei-Eij,
            %algorithme2(Liste_Ai,Liste_Aj,Ereste)
      ).


Algorithme1(Liste_Ai,Liste_Aj,Eij) :-
      recherche_debut_plus_tot(Liste_Ai,Eij,1,Tij,0),
      Tpred is Tij -1,
      annuler_intensites(Tpred,Liste_Aj),
      suspend(algorithme1(Liste_Ai,Liste_Aj,Eij), 2, Liste_Ai ->
constrained).

Recherche_debut_plus_tot([As|Liste_Ai],Eij,Semaine,Tij,Somme) :-
```

```
      Somme < Eij,
      dvar_domain(As,D),
      dom_range(D,_Min,Max),
      Somme_suiv   is Somme  + Max,
      Semaine_suiv is Semaine+ 1,
      recherche_debut_plus_tot(Liste_Ai,Eij,Semaine_suiv,Tij,Somme_suiv).

recherche_debut_plus_tot(_,Eij,Semaine,Semaine,Somme) :-
      Somme >= Eij.

Annuler_intensites(0,_Liste_Aj).
Annuler_intensites(N,[Aj|Liste_Aj]) :-
      N>0,
      N1 is N-1,
      Aj #= 0,
      annuler_intensites(N1,Liste_Aj).

/*

pour chaque contrainte precede(i,j,Eij)
      algo1(i,j,Eij
      calculer t tel que E=Ebef(i,t)>= Eij
      annuler tous les aj(u) pour u=t..est(j)
      si E < Eij, redéclencher algo1
      */
```

4.5- Main program

```
      %****************************************
      % Chargement des librairies nécessaires *
      %****************************************

:- lib(fd).
:- lib(fd_search).


:- [contraintes].
:- [visualisation].
:- [predicats].        %predicats autres que contraintes ou affichage

      %*********************
      % PROGRAMME PRINCIPAL *
      %*********************


main(Fichier_exemple,M, Spy_or_not) :-

            %************************
            % CHARGEMENT D'UN EXEMPLE
            %************************
      compile(Fichier_exemple),

      findall(A, activite(_,A,_E,_R,_D), Liste_Activites),
length(Liste_Activites,Nb_Activites),

      write(Nb_Activites),write(' activites :'), write(Liste_Activites),nl,

      horizon(A,B),
      write('Horizon : '), write([A,B]),nl,

      Nb_Semaines is B-A+1,
            %**********************************************
            % creation du Table des variables intensites
            %**********************************************
      dim(M, [Nb_Activites, Nb_Semaines]),

            %**************************************
            % creation des variables debuts et fins
            %**************************************
      writeln('creation des variables deb et fin'),
      dim(Debuts_activites, [Nb_Activites]),
      dim(Fins_activites,   [Nb_Activites]),

      (foreach(A, Liste_Activites), for(NA, 1, Nb_Activites),
       param(Nb_Semaines,Debuts_activites,Fins_activites)
      do
            activite(_,A, _E, R,D),
            DebA is Debuts_activites[NA], % acces à la NAieme variable du
Table des debuts
            FinA is Fins_activites[NA],   % acces à la NAieme variable du
Table des fins
            DebA :: R..Nb_Semaines,        % définition des
            FinA :: 0..D,                 % domaines
            DebA #<= FinA                     % contrainte minimale (on peut
mieux faire...)
      ),


            %*******************************************************
            % surveillance des variations de domaines des variables
            %*******************************************************
      spy_Table(M, [Nb_Activites, Nb_Semaines], Spy_or_not),


      %**********************************************
      % Pose des contraintes sur le Table principal
      %**********************************************
      writeln("Pause des contraintes ...."),

      % de domaine
      writeln("...de domaines"),
      L is M[1..Nb_Activites, 1..Nb_Semaines],
      flatten(L, V),
      intensite_maximum(_Ac, IMax),
      V :: 0..IMax,

      % de dates de debut + tot et de fin + tard
      writeln("...de dates initiales"),
      contraintes_dates_limites(M, Liste_Activites, Nb_Activites,
Nb_Semaines),

      % de cumul maximum par semaine
      writeln("...cumulatives"),
      contraintes_cumul(M, Nb_Activites, Nb_Semaines),

      % energie de chaque activite = cumul des intensités sur l'horizon
      writeln("...d'energie"),
      contraintes_energie(M, Liste_Activites, Nb_Activites, Nb_Semaines),

      % de precedence energetique
      writeln("...de precedence energetique"),
      contraintes_prec_ener(M, Nb_Semaines),

      % affichage du Table (avant résolution)
      print_Table(M,Nb_Activites,Nb_Semaines),

      %****************
      %   instanciation
      %****************
      !,
      instancierplustot(V),

      % affichage du Table (après résolution)
      print_Table(M,Nb_Activites,Nb_Semaines)

      .
```

Aide au pilotage d’activités d’ingénierie pour le développement distribué d’un système complexe

Résumé : De nos jours, pour maîtriser la complexité structurelle et fonctionnelle associées à la conception et au développement d’un système complexe tel qu’un avion, les entreprises mettent en place des organisations elles aussi complexes, à la fois hiérarchisées et distribuées. Ainsi le développement du système est confié à différentes équipes provenant d’entreprises aux métiers différents mais complémentaires. Ces équipes fonctionnent en ingénierie concourante et doivent se coordonner lors de la conception (échanges de résultats intermédiaires concernant des sous-systèmes à différents niveaux de maturité) et lors de l’intégration (travail en « plateaux »).

Ce travail se focalise plus particulièrement sur le pilotage des activités d’ingénierie au sein d’une équipe, compte tenu de contraintes globales sur les ressources (nombres de personnes allouées) et sur les délais (fenêtres temporelles des activités), mais aussi compte tenu des contraintes de synchronisation que traduisent l’interdépendance des équipes.

L’originalité de ce travail est de proposer une caractérisation énergétique des activités et des contraintes qui les lient et de valider la cohérence des décisions de pilotage (avance ou retard des activités, allocation de ressources supplémentaires) par l’utilisation d’un outil rigoureux basé sur la programmation par contraintes. Les mécanismes de propagation de contraintes peuvent être utilisés pour valider différentes simulations afin de servir de références pour la renégociation de contraintes lorsque celle-ci devient obligatoire. Une première spécification des modes d’utilisation d’un outil d’aide à la décision est également proposée. Nous concluons sur les extensions du modèle et sur les travaux d'expérimentation et de validation qui doivent prolonger ce travail afin de parvenir à un outil opérationnel diffusable à l’ensemble des équipes partenaires d’un projet de développement d’un système complexe.

Mots clés : *activités d’ingénierie, ordonnancement, aide à la décision, coopération, gestion de projet, simulation de scenario, équipes de conception distribuées, allocation de ressources, contraintes.*

Steering engineering activities for the distributed development of a complex system

Summary: At the present time, in order to manage the functional and structural complexity associated with the design and development of a complex system such as an aircraft, companies put in place organisations that are themselves highly complex – both hierarchical and distributed. Thus, system development is outsourced to different teams from companies that are specialised in different, complementary areas. These teams work according to the principle of concurrent engineering and must coordinate their activities during the design phase (exchange of intermediary results concerning sub-systems at various levels of maturity) and the integration phase (working together on “plateaux”).

The present study focuses primarily on the steering of engineering activities within a team, taking into account general constraints related to resources (number of people allocated) and lead times (time slots assigned to activities), as well as the synchronisation constraints inherent to interdependency between the teams.

The originality of this study lies in the energetic characterisation it offers of the activities and the constraints that link them, and the fact that it validates the consistency of the steering decisions (activities brought forward or set back, allocation of additional resources) through a rigorous tool based on constraints programming. The constraints propagation devices can be used to validate different simulations in order to be used as a reference for the renegotiation of constraints when such renegotiation becomes mandatory. We also offer an initial specification of the operating procedure for a decision support system tool. We conclude by studying possible extensions of the model as well as the experimentation and validation work that must accompany this study in order to obtain an operational tool that can be circulated among all the teams that are partners in a complex system development project.

Keywords: *engineering activities, scheduling, decision support system, cooperation, project management, scenarios simulation, distributed design teams, resources allocation, constraint satisfaction problem.*