Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Project

Department of Automotive and Aeronautical Engineering

Cabin Refurbishing Supported by Knowledge Based Engineering Software

Author: Bianca Adina Szász

Examiner: Prof. Dr.-Ing. Dieter Scholz, MSME

Delivered: 30.06.2009

Abstract

The cabin related activities, especially refurbishing, are of interest in the present economical context. Airlines need to convert their fleet once the requirement change. A great number of configuration parameters are derived from the requirements (an example of such a requirement based parameter is the position of each cabin item, respecting the regulatory specifications). All these parameters need to be combined within the overall cabin layout and need to be optimized. A virtual medium sized engineering office is considered, having a Design Organizational Approval (DOA) for performing certified cabin conversions. In order to cope with the challenges coming from airliners or from VIP customers, engineering offices today have to make use of up to date software solutions based on Artificial Intelligence. Artificial Intelligence (AI) is “the study and design of intelligent agents”, where an intelligent agent is a system that perceives its environment and takes actions which maximize its chances of success. Such software systems allow the isolation of the knowledge behind a design problem and then run the problem solving component. This concept is especially required in cabin configuration. The paper investigates the use of a Knowledge Based Engineering (KBE) approach applied to a configuration system for aircraft cabins. The KBE approach is tested by using the Pacelab Cabin software. The regulatory specifications are implemented into the program by using the available rules engine of the software. The rules engine is then used to check the consistency of the cabin build-up in the program. If the task is about refurbishing, consistently replacing and updating of cabin items is likewise checked by rules. The paper summarizes the potential of using AI/KBE based configuration system in practical cabin refurbishing.

Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

DEPARTMENT OF AUTOMOTIVE AND AERONAUTICAL ENGINEERING

Cabin Refurbishing Supported by Knowledge Based Engineering Software

Project work towards a thesis at Universitatea Politehnica din Bucuresti (PUB)

Background

Cabin related activities, especially refurbishing, are of interest in the present economical context. Airlines need to convert their fleet once the requirements change. A great number of configuration parameters need to be handled. Some parameters are derived from certification requirements (e.g. the position of cabin items). An optimum value for all cabin parameters should be found. Knowledge Based Engineering (KBE) is one of the strategies that can be used. This thesis is part of CARISMA (Aircraft Cabin and Cabin System Refurbishing – Optimization of Technical Processes), a research project at HAW Hamburg in cooperation with industry.

Task

This thesis investigates in which way Knowledge Based Engineering can support Cabin Refurbishing. The task is broken down into these subtasks:

- Summarize Tasks in Cabin Refurbishing.
- Review the scientific field of Artificial Intelligent (AI), Knowledge Based Engineering (KBE) and Configuration Systems.
- Propose tasks in Cabin Refurbishing that would benefit from a solution with Knowledge Based Engineering.
- Present the program PaceLab Cabin.
- Describe one task in Cabin Refurbishing solved with PaceLab Cabin.
- Present the Rules Engine in PaceLab Cabin.
- Derive cabin-related design rules from CS 25.
- Demonstrate the application of existing rules and the definition of new rules for cabin-related design activities.
- List pros and cons of PaceLab Cabin and it's rules engine.

The report has to be written in English based on German or international standards on report writing.

Declaration

I declare that this diploma thesis is entirely my work. Where use has been made of the work of others, it has been fully acknowledged and referenced.

.....

Date Signature

Table of Content

List of Figures

List of Tables

List of Abbreviations

AI	Artificial Intelligence
KBE	Knowledge-Based Engineering
FAA	Federal Aviation Administration
DB	Database
CC	Cabin Conversion

1 Introduction

1.1 Motivation

For civil airplanes, the passenger cabin presents an important interest. According to **Seeckt 2006**, currently, an airplane is provided from four to five times with a new interior of cabin during its lifetime. That implies the fact that the cabin items are bought very often. In the future, this number is expected to grow. In addition, the Airlines always ask from the Aircraft Manufacturers for individual Cabin Design Models. The varieties of desires coming from costumers go from the extreme Low-Cost-Carriers to the maximum comfort and quality of the Airlines. The average level of comfort of the last ten years has been highly increasing and therefore the pretentions have increased too regarding the functionality, In-Flight-Entertainment etc. and this is what the cabin must fulfilled.

All these requirements imply efficient procedures of the Aircraft Manufacturer respectively Cabin Designers and these procedures should be cheap. The automation of this operations development and the possibility for use of the synergy-effects accelerate the time of development, reduce the staff requirements and support therefore decisively the economical success. This is a reason why the concept of Cabin Refurbishing is definitely important.

Pacelab Cabin is one of the most efficient tools for Cabin Design and for Cabin Refurbishing also and it develops the concept of Knowledge Based Engineering which has its roots in Artificial Intelligence.

1.2 Definitions

Artificial Intelligence
Artificial Intelligence (AI) is the intelligence of machines and the branch of computer science which aims to create it. Major AI textbooks define the field as “the study and design of intelligent agents” where an intelligent agent is a system that perceives its environment and takes actions which maximize its chances of success. Subfields of AI are organized around particular problems (**Wikipedia 2009b**).

Cabin

According to **Lexicon 2004**, the general definition of the cabin is :

> *A Cabin is often all what we call a closed room (respectively a box), in which people can sit or have some special activities, for example on a boat, in an airplane, in a Zeppelin or in a aerial lift. For this, it contains often glazed windows and doors.*

Regarding the airplanes, the cabin is the compartment and interior surrounding passengers and crew but also all systems, functions and services that ensure a safe and comfortable operation both in flight and on the ground (**Niţă 2009**).

Cabin Conversion

A cabin conversion is defined as the sum of the activities and processes necessary to transform the layout of a cabin from the original destination to a new one, having a new mission. Depending on the transformation scenario - Pax-to-Pax, Pax-to-Freighter, or Pax-to-VIP - the complexity of the activity changes as well as the certification requirements (**Scholz 2009**).

Cabin Layout

A cabin layout defines the passenger cabin of an aircraft by number, arrangement, type and position of the cabin interior components. Cabin interior components are objects like passenger seats, lavatories, galleys or cabin attendant seats which may be installed in the passenger cabin. The entire set of cabin interior components in the passenger cabin defines its capacity, its comfort level, its possible service etc. (**Kopisch 1992**).

Design

In the technical language use, the concept of “design” is used in terms of “figuration”, “definition”, “creation”. In contrast to literature where “design” is synonym with “interpretation”, in the technical language, it means an active process of creation to reach a certain purpose (here: the Cabin Layout) with the remark that it is used under some boundary conditions and requirements.

KBE

Knowledge-based engineering (KBE) is a discipline with roots in computer-aided design (CAD) and knowledge-based systems but has several definitions and roles. An early role was support tool for a design engineer generally within the context of product design (**Wikipedia 2009c**).

Layout

For the Layout term, the source **Lexicon 2004** says:

> *Layout is a word from English which has like synonym Plan (Lay-Out).*
> *The detailed and visible created items of the mental blackout drawings in the most part of the printing area. The visualisation communicates to the designer and to the client an impression over the form of the future results and uses for this some rules basement for the next designs.*

The word Layout is descended from the medium lexicon. However, its application in engineering is more related to 2D arrangement of the components. Also, it always contains the interpretation of the functionality's purposes and of the interaction of all components of the Layout. Therefore, the Airplane Cabin Layout contains the complete design in all the three dimensions.

PACE

The PACE AEROSPACE ENGINEERING AND INFORMATION TECHNOLOGY GMBH Company is in Berlin, Germany, which, as is written in **Pace 2003a**, „...develops knowledge-based Complete Solutions for CAD IT-Environment ...". „... The most important goals are Evolution - and Marketing Department development in the aviation, astronautics and the industry of transport"(according to **Pace 2003a**). The main product of the company is the KBE Pacelab Platform Design, based on the Pacelab Cabin. The program is characterised by its modular configuration, which enables customization of the speculative application delivering together with the implementation of the available concept.

Pacelab Cabin

Pacelab Cabin is "the central element of the whole software family Complete Solutions for the Cabin Environment" (as is it shown in **Pace 2003a**) of PACE GMBH. Therefore, *Pacelab Cabin* is a knowledge-based Software Technology (KBE). This denotes for the operator that: „...relevant properties of the design knowledge in the Application use and automate the actual condition of the configuration..."(according to **Pace 2003a**). In particular, this pushes to a multitude of elements in the Program database, which have to be administered.

1.3 Objectives

Primary Objectives:

- identify tasks in Cabin Refurbishing;
- explain the reasons why the fields of Artificial Intelligence and Knowledge Based Engineering (KBE) are important for the applications in Cabin Refurbishing;
- present a software solution based of KBE (Pacelab Cabin) for Cabin Refurbishing.

Secondary Objective:

- present how Knowledge Based Engineering relates with the software solution using the Rules Engine.

1.4 Report Structure

Chapter 2 contains the description regarding the concept of Cabin Refurbishing and some solution suggestions.

Chapter 3 contains the analysis over the fields of Artificial Intelligence and Knowledge Based Engineering and also describes how they relate to the applications in Cabin Refurbishing.

Chapter 4 contains the presentation of the program Pacelab Cabin and its Rules Engine.

Chapter 5 contains the summary of the work.

2 Cabin Refurbishing

2.1 Introduction to Cabin Refurbishing

It is foreseen – as shows a report from **La Rocca 2007a** – that in twenty years time, the aeronautic systems will differ from today's systems at least as much as the actual systems differ from those of 1930. Also, the number of people who use air transport has increased very much during the last decades. According to "ACARE 2020", **PACE 2003a**, especially in air travel, during the last 10 years, the number of passengers rose by about 5% per year and this growth is expected in future as well.

Also, because the comfort is a very important issue for the passengers, aircraft manufacturers will concentrate on improving the aircraft cabin interior. Due to the fact that airplanes can be used for different missions (passengers transport, cargo etc), it's more profitable to use the same airplane for different purposes. This is the reason why one of the most important task related to the cabin is *Cabin Refurbishing*.

Cabin Refurbishing is an important engineering issue and can be defined as a process by which various aerospace companies can operate modification in the cabin space or, as is defined by **Niţă 2009**, the Cabin Refurbishing is the sum of modifications taking place inside an aircraft, so to get from cabin type A to cabin type B. The process can mean – as it is said in **PACE 2003a** – upgrading the cabin interiors or leasing companies retrofitting aircrafts for new customers or transforming the airplane from a passenger airplane to a cargo airplane and the opposite.

A reason to look for an improvement is that market changes requires airlines to adapt. Additionally, adapting to a new market situation means the need to convert the fleet or, in other words, to make cabin refurbishing.

2.2 Tasks in Cabin Refurbishing

The domain of cabin conversion and refurbishing – as it was shown by **Niţă 2009** - is expected to grow and the airlines invest in optimizing their fleet even in a time of general economic downturn.

However, the field of cabin conversion has already expanded a lot in the last years. If in the past, classical assignments asked for design drawings, the future will see international big projects,

> *covering the entire process chain of a complete cabin conversion to be assigned to companies outside of the aircraft manufacturer* (**Niţă 2009**).

However, every refurbishment project takes into account both the scope of cabin changes and the resulting costs. The statuses of the current cabin layout ("Before Modification") and the future cabin design ("After Modification") need to be documented and compared – indicates **PACE 2003a**. The items that will be replaced and relocated must be identified. The following picture shows how this works:

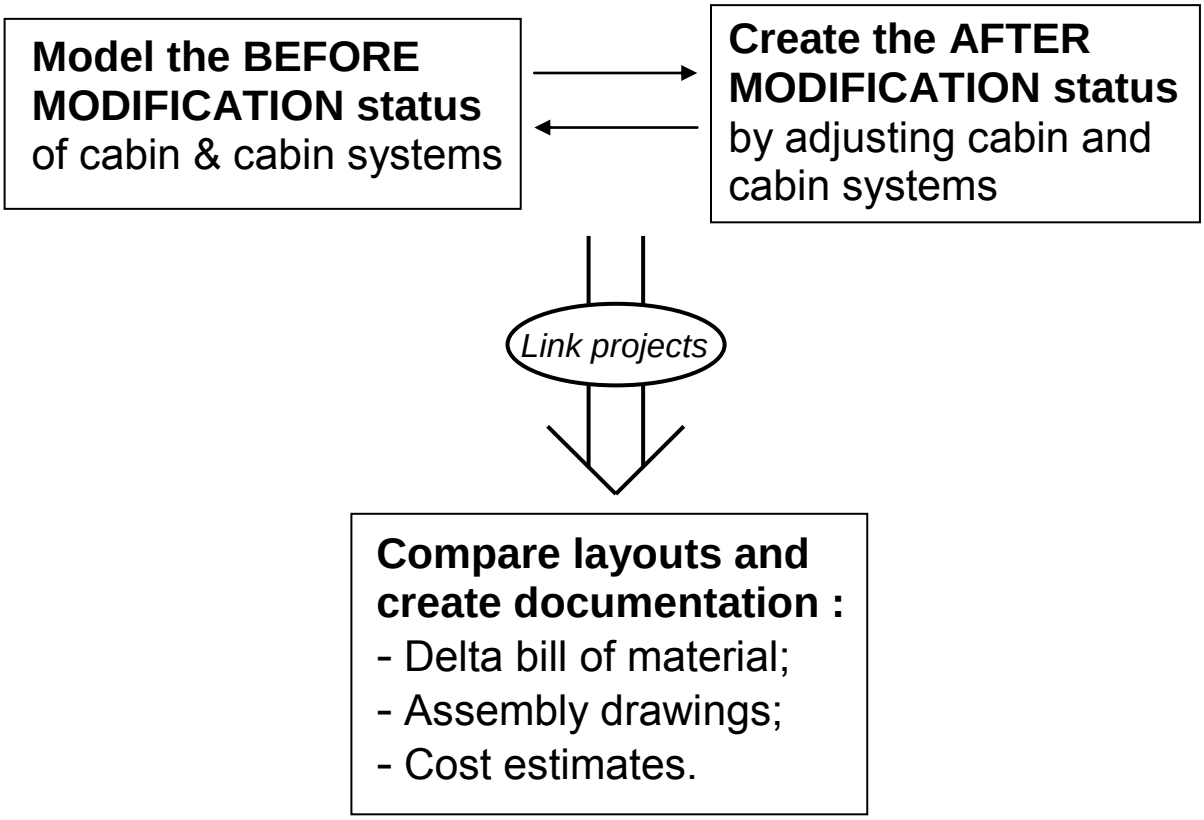


Fig. 2.1 The Refurbishment Project (**PACE 2003a**)

As a result, the intended cabin layout should be in the same time attractive for the passengers and technically feasible, certifiable and within the intended cost bracket.

To make the decisions regarding the refurbishing easier, the European Aviation Safety Agency (EASA) classifies the changes into minor and major changes (the last ones can only be approved by the Agency). No matter if the change is for transforming an aircraft

from pax to pax, from cargo to pax, or the other way around, the processes are the same, the personnel and the resources are the same too (**Niţă 2009**).

In Fig. 2.2, the way in which the changes are classified and some examples of major changes can be seen.

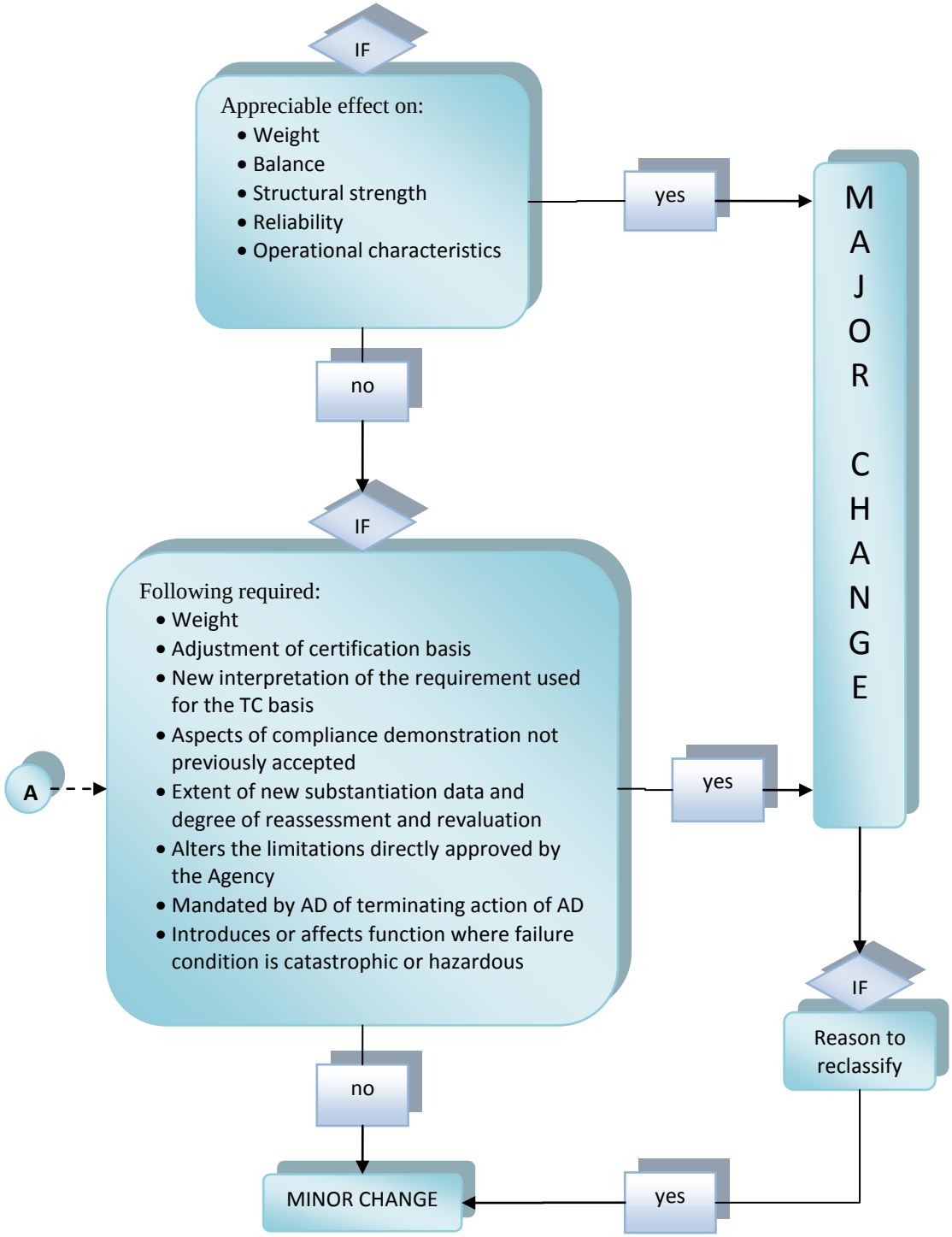


Fig 2.2 Classification Process of Minor and Major Changes (**EASA 2009**)

The “A” letter stands for some examples of major changes for cabin safety like:

- new cabin layout affecting pax & crew safety or requiring changes in emergency evacuation;
- introduction of dynamically tested seats;
- pitch between seat rows;
- distance between seat and adjacent obstacle like a divider;
- cabin layouts that affect evacuation path or access to exits;
- installation on new galleys, toilets, wardrobes, etc;
- installation of new type of electrically powered galley insert;
- pressurization control system

For a better insight in the processes regarding the new Cabin & Cargo design and development, it makes sense to have a look on how large manufacturer companies deal with this challenge. There are two big manufacturer companies: Boeing and Airbus. In the Fig. 2.3, the conversions for Corporate Jets (light nuances) and VIPs (dark nuances) at Boeing (red colour) and Airbus (blue colour) are compared.

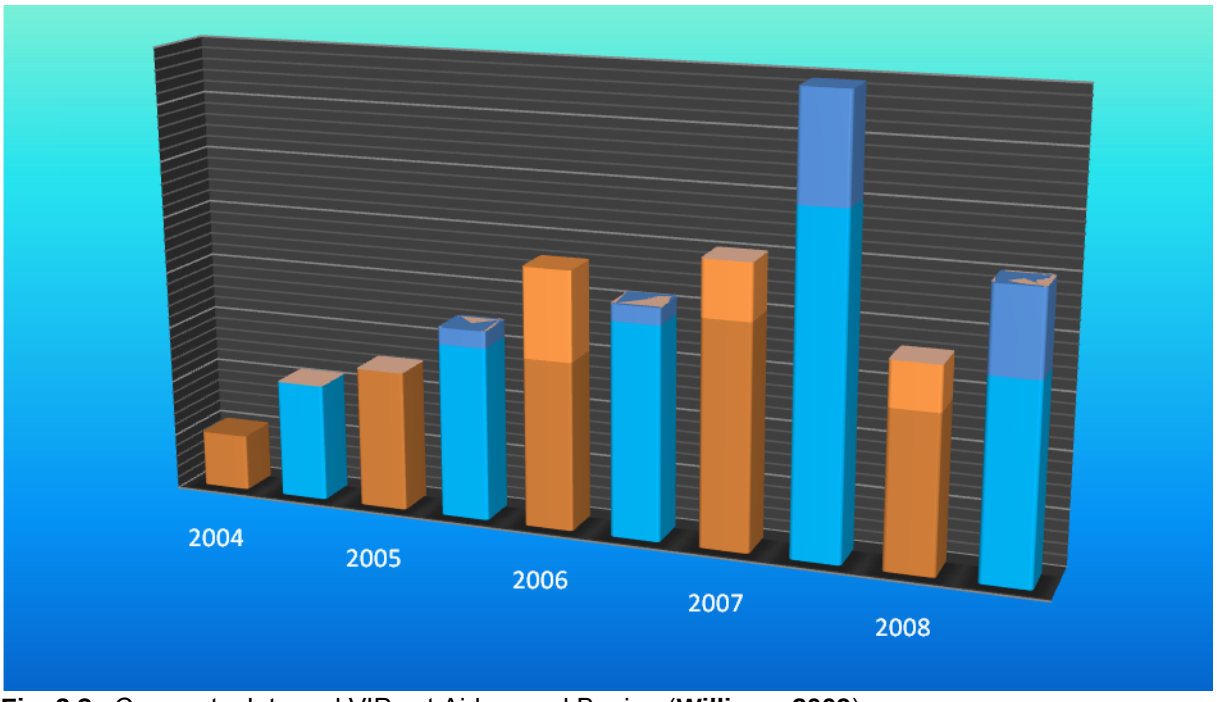


Fig. 2.3 Corporate Jets and VIPs at Airbus and Boeing (**Williams 2009**)

Airbus is the leader for Corporate Jet Sales with the following aircrafts:

Fig. 2.4 Airbus aircrafts conversions delivered by the Corporate Jet Center (**Williams 2009**)

The cabin upgrades market has doubled between 2005 and 2009 and the actual trend is to increase by 10% per year as shown in Fig. 2.5:

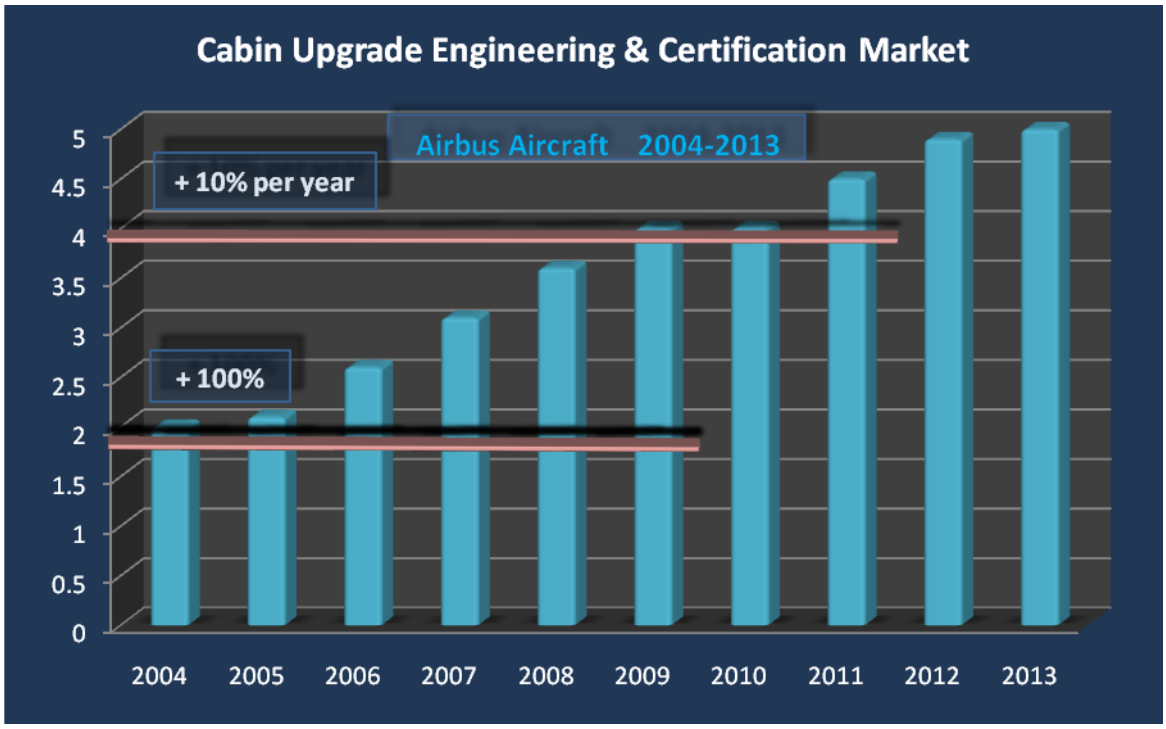


Fig. 2.5 The evolution of the Cabin Upgrade market (**Williams 2009**)

Another market segment is represented by the old aircrafts; there is a huge replacement market allowing conversions from pax-to-freighter (see Fig. 2.6).

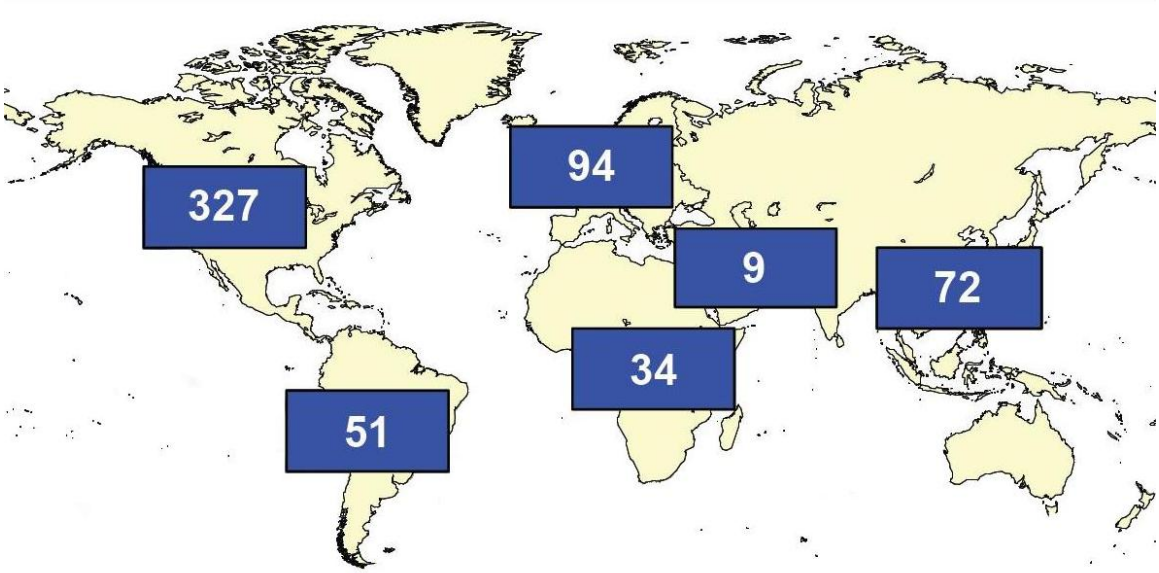


Fig. 2.6 The aircraft replacement market (pax-to-freighter) in number of units (**Williams 2009**)

In Fig. 2.6 one can see that in North America is strongly represented.

Making a step further in the conversion process, the Fig. 2.7 describes what components of the aircraft can be converted in the case of Pax-to-Freighter:

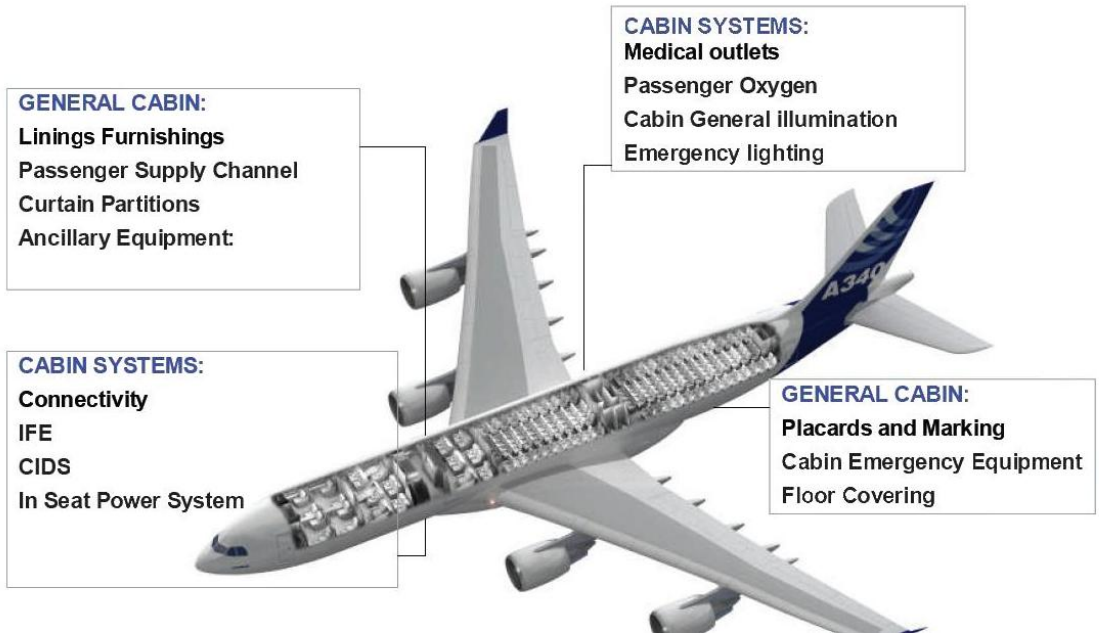


Fig. 2.7 The components that can be converted in the case of Pax-to-Freighter conversions (**Williams 2009**)

For the conversion of these components, the design phases of the development process are shown below (**AP2289**):

- Concept Phase;
- Architecture Phase;
- Definition Phase;
- Design Phase;
- MCA Preparation Phase;
- FAL Preparation Phase;
- Manufacturing & Testing Phase;
- Adjustment Phase;
- Final Project Phase.

Furthermore, each Phase is divided in what Airbus calls "swim lines" (**AP2289**):

- Project Management;
- Industrial Design;
- Engineering Vendor Management;
- Cabin & Cargo Integration;
- Electrical Systems;
- Mechanical Systems;
- Cabin & Cargo Furnishing;
- Structure Design;
- Manufacturing & Assembly.

For a complete cabin retrofit design at Airbus the procedure is adapted by the Upgrade Services organization and is based on the phases for the design of a new cabin (as shown earlier). The Upgrade Services department at Airbus is the one providing *retrofitting solutions* for the customers, starting from producing Service Bulletins and ending with full embodiment of aircraft upgrades (both for small – CS 23 – and large – CS 25 – aircrafts) (**AP2289**).

According to **Williams 2009**, the entire cabin can be modified at Airbus Upgrade Services:

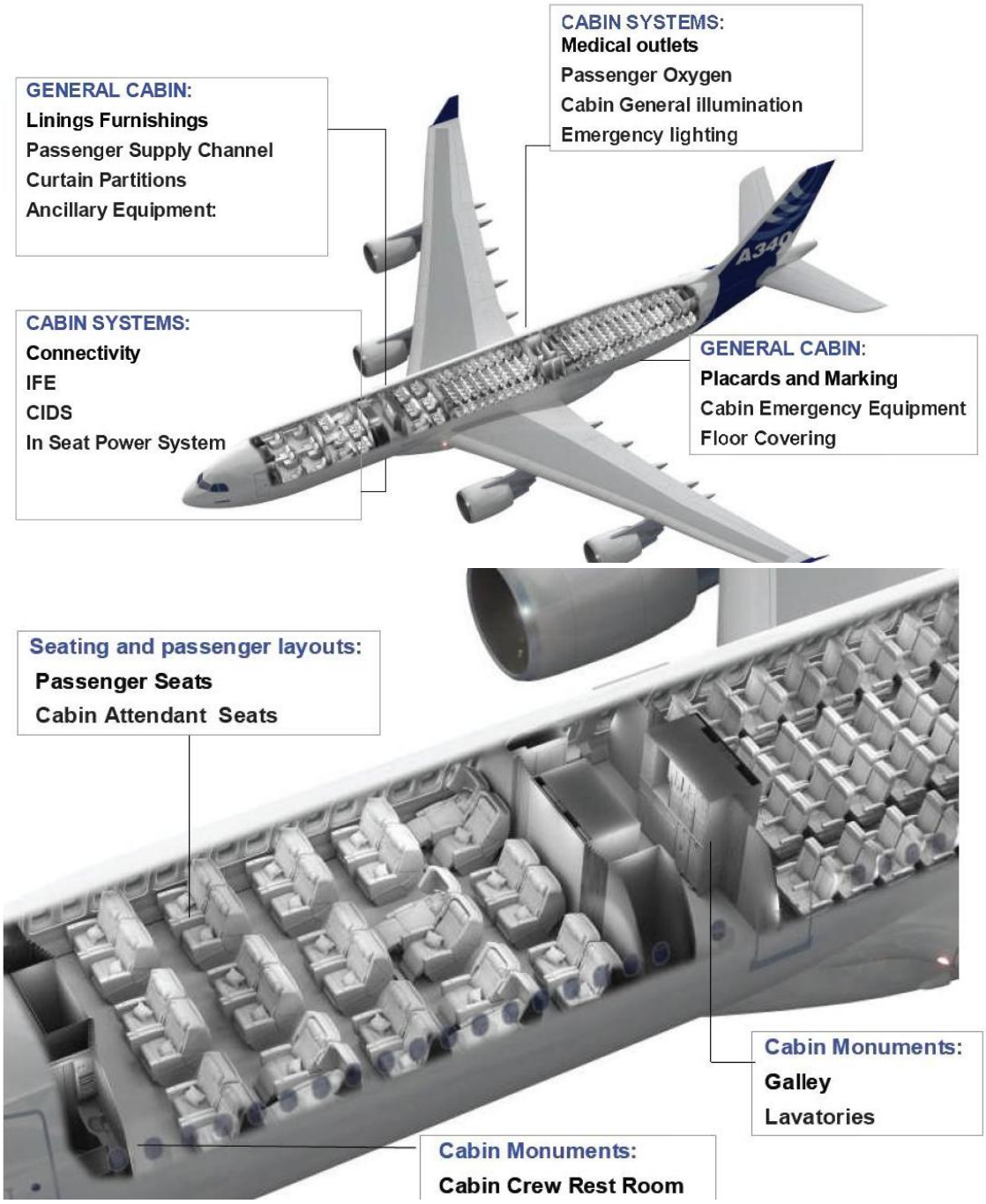


Fig. 2.8 Cabin conversion at Airbus Upgrade Services Department (**Williams 2009**)

The Completion Center is an organization with the specific task of developing the work for a complete conversion. Inside a Completion Centre all the activities related to the design, certification and monitoring are carried out, starting from the customer's request and up to the delivery. There must be noted that the customer is at the core of all activities built up in the Completion Centre.

Therefore, the first phase in the Completion Centre is the Offer Phase. If the offer is accepted by both sides, then the technical document, describing it and the technical implications, heads towards the *conversion processing*. The output, summarized altogether in the *Hand Over Phase* comes back to the customer, and a circle closes. The natural consequences of the correct functioning of this system are feedback output from customer and the update of the virtual catalogue. The Fig. 2.9 gives a better understanding:

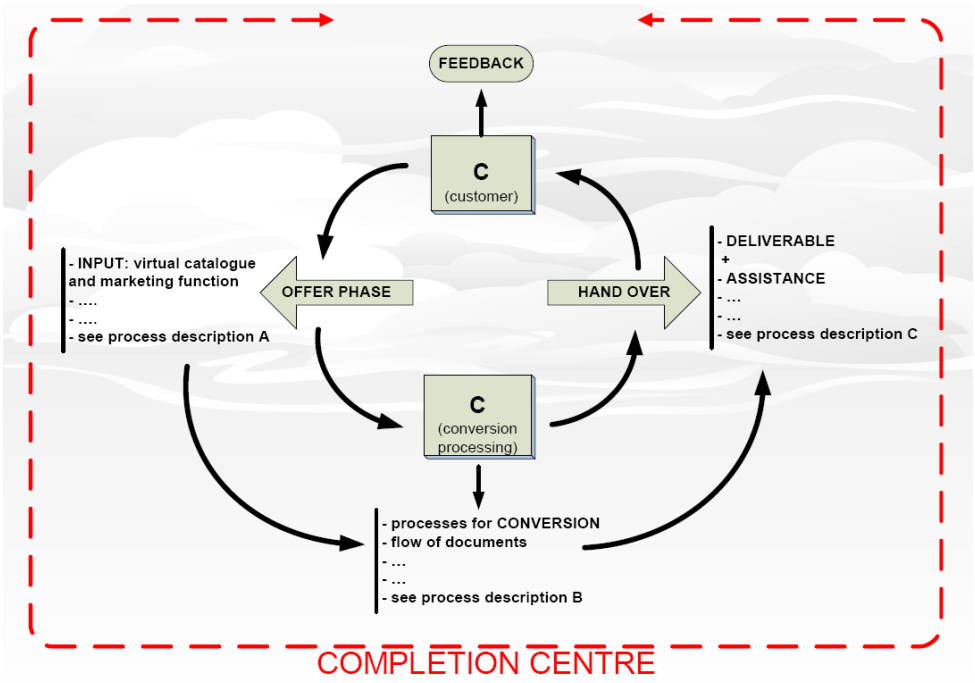


Fig. 2.9 Completion Centre concept suitable for a medium sized engineering office (**Niță 2009**)

Therefore, the process Chain description can be divided into three parts:

- Part A, referring to the offer phase description;
- Part B, referring to the description of the processes for completing the conversion;
- Part C, describing the end processes and the outputs received from the customer.

A thing that must be noticed is that the feedback coming from customers is used to improve the efficiency of the completion centre, functioning as a system.

Also, an important phase during the conversion processes is the *certification*. The certification requirements must be transformed into technical rules. The procedures for implementing the rules must be developed together with the design engineers. The path to conduct a certified conversion is shown by the Certification Agencies: it can only be accomplished by approved Design Organizations (DO) awarded a Design Organizational Approval (DOA). Within the DO the processes for conducting the design must be set up according to the rules of the Certification Authority (**Niţă 2009**).

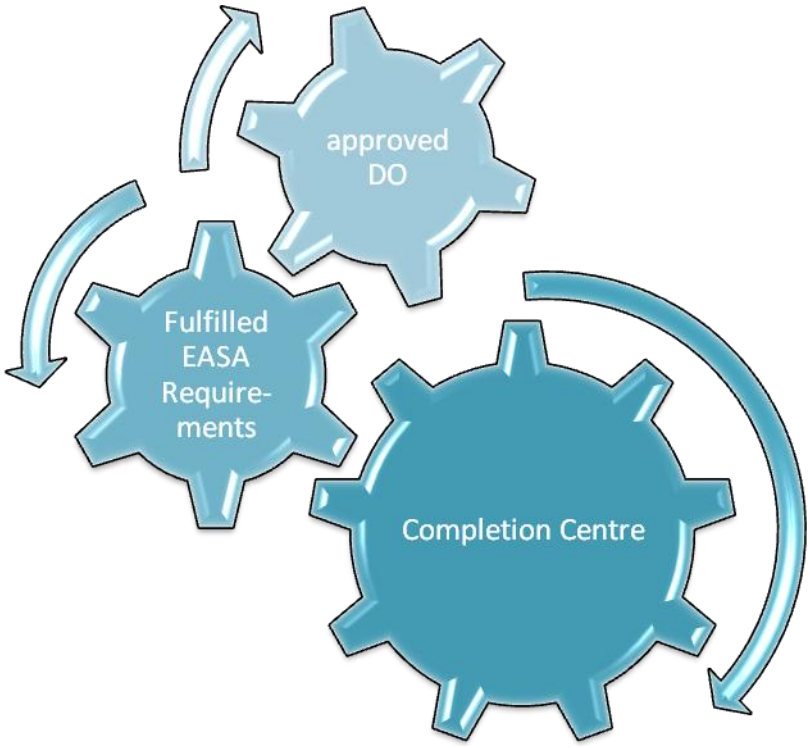


Fig. 2.10 The mechanism behind the conversion within a Completion Centre (**Niță 2009**)

Being a highly customized part of the aircraft, the cabin, respectively cabin design represents a complex work field to be managed, including phases like conception, definition, validation, testing, delivery, and after sales support.

As a direct interface with the passengers, the cabin plays a major role in fulfilling the customer satisfaction. Therefore the cabin becomes an essential tool for the airlines to differentiate from competitors (**Giesecke 2005**).

2.3 Summary

The present economical context shows a growing market for cabin related activities. This initiates the idea of an investigation into certification requirements with respect to cabin design and conversion (**Niţă 2009**).

Also, planning the refurbishment has to be achieved without the aircraft being present, because it is still in operation during that time and, as a consequence, the datasets are the only means for the engineer to prepare the refurbishment. In addition, computer tools which are designed so that they will support the generation of cabin layouts, are applied during early aircraft design (for the generation of the standard layout), or during negotiations with customers (for generation of an individual cabin layout) (**Niţă 2009**).

Therefore, the software tools play an important role. According to **Felfernig 2000**, product configuration creates big challenges on software development like:

- the complexity of the task requires the sophisticated knowledge of technical experts;
- the configuration knowledge base has to be adapted continuously because of changing-components and configuration constraints;
- configurator development time and maintenance time are short and strictly limited.
- development of the product and the product configuration system has to be done concurrently.

However, at the Airbus Upgrade Service Center, both the Engineering and the Fulfil Customer Order teams are looking for ways to improve the process of conversion. The aims are:

- reducing the number of iterations that can appear and cutting down lead times;
- speeding up the layout development;
- improving the quality of drafts and eliminating the potential for errors;
- making customer communication more efficient;
- increasing data exchange.

Because there are many repetitive steps, Airbus is looking into increasing the level of automation and one step in this direction is finding a software solution which is based on the concept of Knowledge Based Engineering with roots in Artificial Intelligence.

3 Knowledge Models

3.1 Overview of tools and concepts

In designing processes, there is need of automation. According to **Schut 2008**, a designer loses too much time in adapting old solutions to encompass for new requirements. Furthermore, for integration of new solutions, a method should exist that should reuse known solutions and to be able to efficiently and effectively integrate new design options and tools.

In this purpose, the knowledge based engineering methodology can support an implementation of such a method by automating repetitive non-creative processes and making more time available to exploit their creativity and engineering skills.

La Rocca 2007 defines KBE as a technology that is based on the use of dedicated software tools (i.e. KBE systems) that are able to capture and reuse product and process engineering knowledge. As is written in **Schut 2008**, the main objective of KBE is reducing time and cost of product development by means of the following:

- automation of repetitive and non-creative design tasks;
- support of multidisciplinary integration from the conceptual phase of the design process.

However, KBE has its roots in knowledge-based systems (KBS) applied in the field of engineering, hence the name. Also, KBS is based on methods and techniques from artificial intelligence (AI) which aims at creating intelligent entities.

3.2 Artificial Intelligence

3.2.1 Short History and Definition

The field of modern AI research was founded at a conference on the campus of Darthmouth College in the summer of 1956 by some brilliant scientists like John McCarthy, Marvin Minsky, Allen Newell and Herbert Simon. They were very optimistic about the future of the new field, but their predictions like: “machines will be capable, within twenty years, of doing any work a man can do” (**Simon 1956**) or “within a

generation… the problem of creating 'artificial intelligence' will substantially be solved" (**Minsky 1972**) would not come true.

After a difficult period called "AI winter", in the early 80s, a form of AI – *expert systems* – that simulated the knowledge and analytical skills of one or more human experts appeared and now, in the early 21st century, AI achieved its greatest successes. These successes were determined - in **Russel 1994**'s opinion - by several factors like:

- the incredible power of computers today;
- a greater emphasis on solving specific sub-problems;
- the creation of new ties between AI and other fields working on similar problems;
- a new commitment by researchers towards solid mathematical methods and rigorous scientific standards.

Consequently, AI has become highly specialized and technical.

For a better understanding of what actually Artificial Intelligence (abbreviated AI) means, there is need to look at the meaning of the two terms that composed the expression. The first term, Artificial, has the following definition in dictionary:

> *made by humans; produced rather than natural; made in imitation of something natural; simulated: artificial teeth; not genuine or natural: an artificial smile* (**Wikipedia 2009**).

The second word, Intelligence, is defined as

> *the computational part of the ability to achieve goals in the world. Varying kinds and degrees of intelligence occur in people, many animals and some machines* (**McCarthy 1955**).

Consequently, Artificial Intelligence, or AI for short, is a combination of computer science, physiology and philosophy. AI is a broad topic, consisting of different fields, from machine vision to expert systems.

In this work, the technical aspect of AI will be highlighted. Therefore, technically speaking, Artificial Intelligence is the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence.

Artificial Intelligence can also be defined as "a car that behaves in a manner that could be considered intelligent if it be a man*"* (in **McCarthy 1955**' opinion) or "the study and design of intelligent agents" (as in **Poole 1998**) where an intelligent agent is "a system

that observes its environment and takes actions which maximize its chances of success" (as it was indicated by **Russel 1994**).

Another definition – given by **Brachman 1991** - can be "the study of intelligent behavior achieved through computational means". Or, Artificial Intelligence is

> *...that part of computer science concerned with designing intelligent computer systems, that is, systems that exhibit the characteristics we associate with intelligence in human behaviour – understanding language, learning, reasoning, solving problems, and so on* (**Barr 1981***).*

However, there can be said that AI aim at human-level intelligence, because the ultimate goal is to make computer programs capable of solving problems and achieve goals in the world as well as humans do (**Barr 1981**).

3.2.2 Concerns

Artificial Intelligence has two major concerns: *Knowledge Representation* and *Search*. Deryn Graham and Anthony Barrett define these terms like following: Knowledge Representation

> *addresses the problem of capturing the full range of knowledge required for intelligent behaviour in a formal language i.e. one suitable for computer manipulation* (**Green 1986**).

and Search is

> *a problem-solving technique that systematically explores a space of problem states, namely, successive and alternative stages in the problem-solving process* (**Green 1986**).

Also, **Graham 1997** says that "early AI placed great emphasis on search, modern AI emphasizes representation and knowledge". The importance of knowledge and its key role gives enough reason to highlight this concept. The definition of this concept offered by **Webster 2009** is:

> *a) the fact or condition of knowing something with familiarity gained through experience or association; acquaintance with or understanding of a science, art, or technique;*
> *b) 1) the fact or condition of being aware of something;*
> *2) the range of one's information or understanding;*
> *c) the circumstance or condition of apprehending truth or fact through reasoning: cognition;*
> *d) the fact or condition of having information or of being learned;*
> *e) the sum of what is known: the body of truth, information, and principles acquired by humankind.*

Many different definitions and interpretations exist for data, information and knowledge. To avoid any confusion between these three terms, they are defined below.

Data is understood as discrete, atomistic, tiny packets that have no inherent structure or necessary relationship between them. In contrast to data, information is data that is structured and put into context, so that it is transferable, but the immediate value of information depends on the potential of the user to sort, interpret and integrate it with their own experience (**Nawijin 2009**).

As it is written in **Nawijn 2009**, knowledge goes one step further and implies the combination of information with the user's own experiences to create a capacity for action.

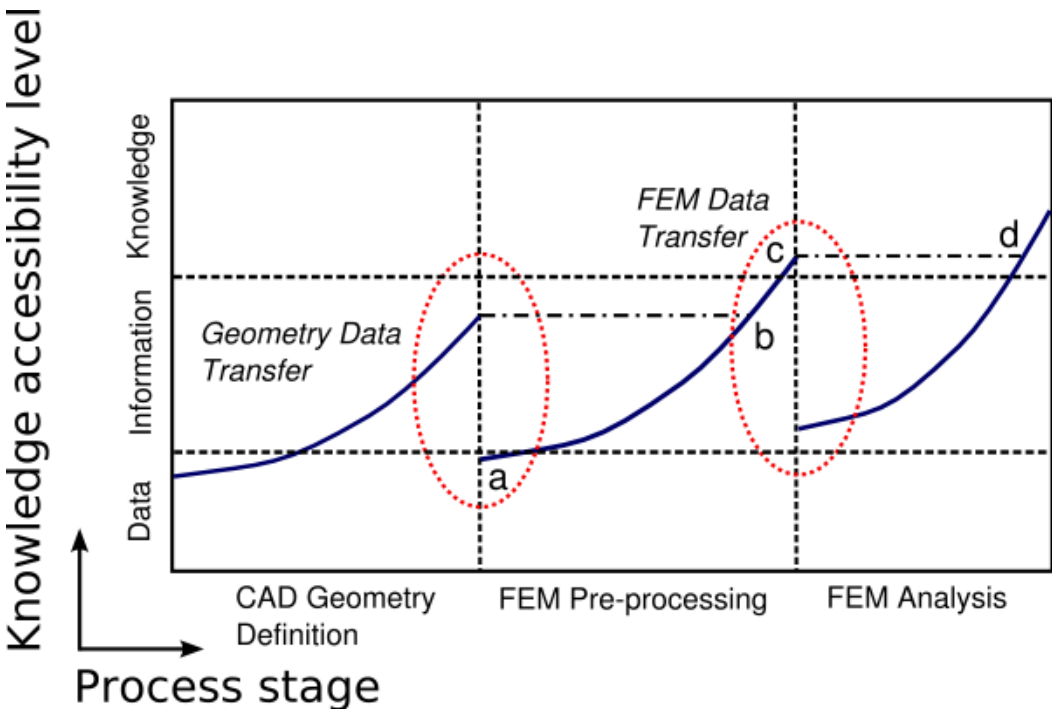


Fig. 3.1 Data, Information and Knowledge (**Nawijn 2009**)

Additionally, knowledge is generally structured in terms of specific relationships such as object/property, class/subclass, and agent/verb/object. There are *individual* knowledge of persons and the collective knowledge of the company, the so-called *organizational* knowledge.

However, the definition of knowledge is a matter of on-going debate among philosophers, but this work will focus on the scientific importance of knowledge. In the scientific field, by knowledge, one can understand 'the management of the resources which "has high importance from both strategic and operational points of view" (according to **Fischer 2002**). Knowledge refers to the context-related bringing together of information (as is shown by **Probst 1999**).

Also, it is suggested – by **Graham 1997** - that all you need in order to make a computer program intelligent is to provide some general rules and lots of very specific knowledge. Making a step further, Knowledge Representation is that part of AI that is concerned with how an agent uses what it knows in deciding what to do. It is the study of thinking as a computational process.

3.2.3 Applications

Today, Artificial Intelligence has become an essential part of the technology industry, providing "the heavy lifting for many of the most difficult problems in computer science" (according to **Russel 1994**).

In addition, there are many subfields of the Artificial Intelligence and they are organized around particular problems, but what they have in common are such traits as reasoning, knowledge, planning, learning, communication, perception and the ability to move and manipulate objects.

Also, Artificial Intelligence has many applications like: expert systems, fuzzy logic systems, genetic algorithms, neural networks, intelligent agents, hybrid intelligent systems etc. Artificial Intelligence can be decomposed into a number of sub-disciplines like Game Playing, Machine Learning, Natural Language Processing, Vision, Robotics, Neural Networks and Parallel Distributed Processing (PDP), as well as Expert or Knowledge-Based Systems.

According to **Luger 2004**, the sub-disciplines have some features in common:

- computers are used to perform symbolic reasoning;
- an aim to capture and manipulate the significant qualitative features of a situation rather than relying on numeric (quantitative) methods;
- issues of semantic meaning as well as syntactic form addressed;

- "sufficient" answers, neither exact or optimal, resulting from the essential reliance on heuristic problem-solving methods. Such methods employed in situations where optimal or exact results are either too expensive or not possible;
- application of meta-level knowledge to carry out more sophisticated control problem-solving strategies; meta-level knowledge means knowledge about knowledge and it refers to structure and strategy, when or how rules should be fired.

The Fig. 3.2 gives a better clue about the major concerns and the sub-disciplines of the Artificial Intelligence:

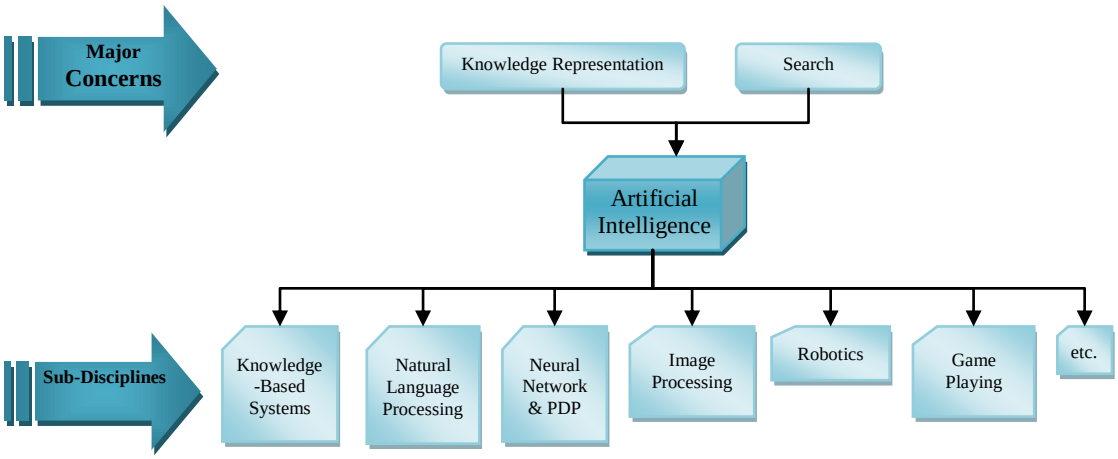


Fig. 3.2 The major concerns and the sub-disciplines of the Artificial Intelligence (**Graham 1997**)

One of the sub-disciplines is Knowledge-Based Systems which will be detailed in the next subchapter and the start point would be to make an insight in the concept of the based-knowledge.

3.3 Knowledge-Based Systems and Expert Systems

There are two types of approaches regarding the Artificial Intelligence: a classical manner or implicit and representation based on knowledge, or explicit.

For the classical approaches, knowledge is implicitly incorporated in the algorithms or in the programming language. One example can be a simulation of electronic circuits containing the form of formal physical laws (**Felfernig 2000**).

The principle of the one based on knowledge is shown in the next figure. First, a conceptual model of the configurable product is designed using a modelling language. After syntactic checks of the correct usage of the concepts, this model is then non-ambiguously transformed to logical sentences which are exploited by a general configuration engine for computing configurations of products. Consequently,

> *the configurator is based on a declarative, logic based, explicit representation of the configuration knowledge. Finally, the resulting knowledge base is validated by the domain expert using test runs on examples. In the case of unexpected results, the product model can be revised on the conceptual level. If the knowledge base is correct, it can be employed in productive use* (**Felfernig 2000**).

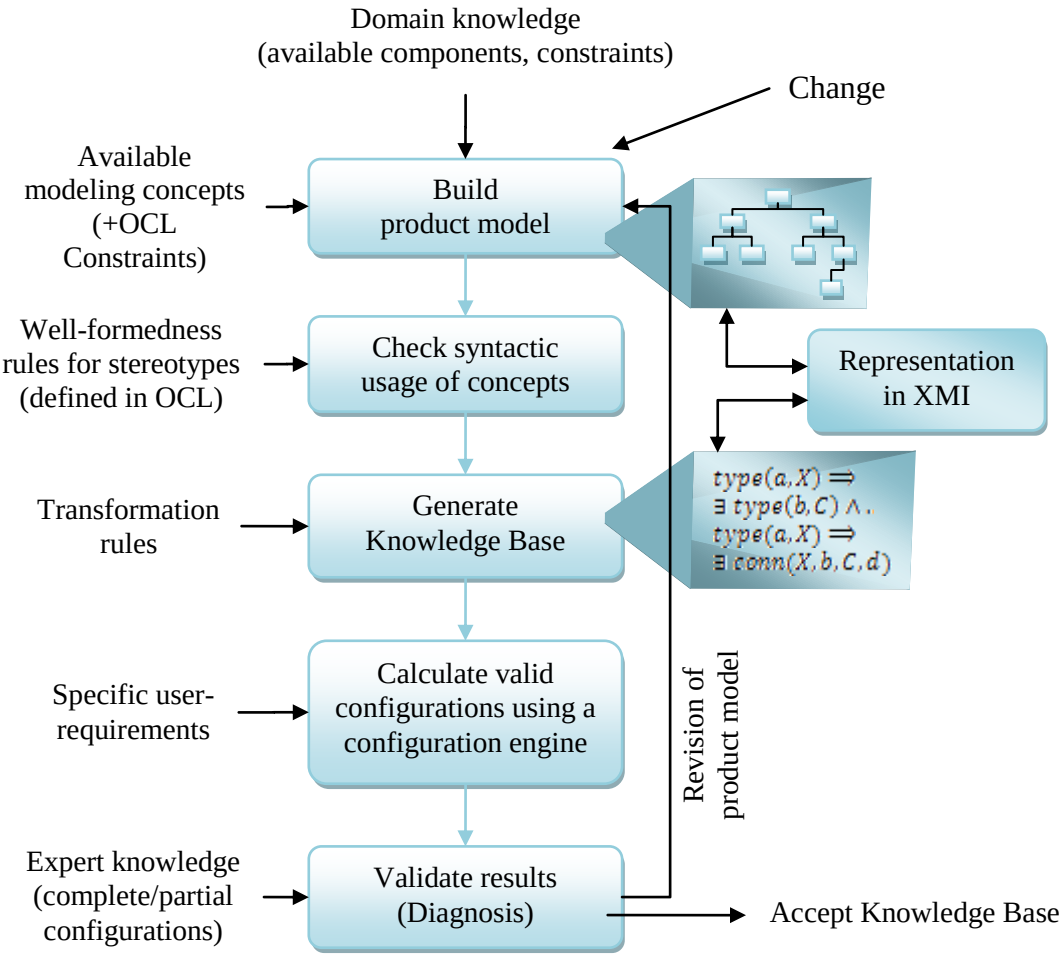


Fig. 3.3 Constructing a valid configuration Knowledge Base (**Felfernig 2000**)

The representation based on knowledge can be defined as a set of rules describing relations between elements in the domain of knowledge. The rules for deriving an outcome from a set of conditions are always formulated externally by an expert usually

aided by a knowledge engineer, i.e. a specialist in transforming the expert's information into statements suitable for a knowledge base. According to **Fischer 2002**, the knowledge engineer stands to the expert as anthropologists do to their informants.

Making a step farther, there are systems which use based-knowledge and which are developed as a result of work in Artificial Intelligence or, in other words, they have their roots in Artificial Intelligence.

However, shorter product cycles, lower prices of products, and higher customer demands have created big challenges for the product development process and at these challenges answer the Knowledge-Based Systems:

> *A successful approach to master these challenges is to employ knowledge-based systems with domain specific, high level, formal description languages which allow a clear separation between domain knowledge and inference knowledge. However, these techniques can be exploited to (partially) automate the generation of software solutions. Their core components are the knowledge base and the inference mechanisms* (**Felfernig 2000**).

It is important to notice that KBS technology is being investigated as a powerful tool rather than a problem solver: this means that there may, in fact, be no real problem at all, but there may be a perception that the power of KBS technology could in some way improve performance or increase productivity, and that may be the motivation for looking into its use. Here is the Basic Architecture of a Knowledge-Based System:

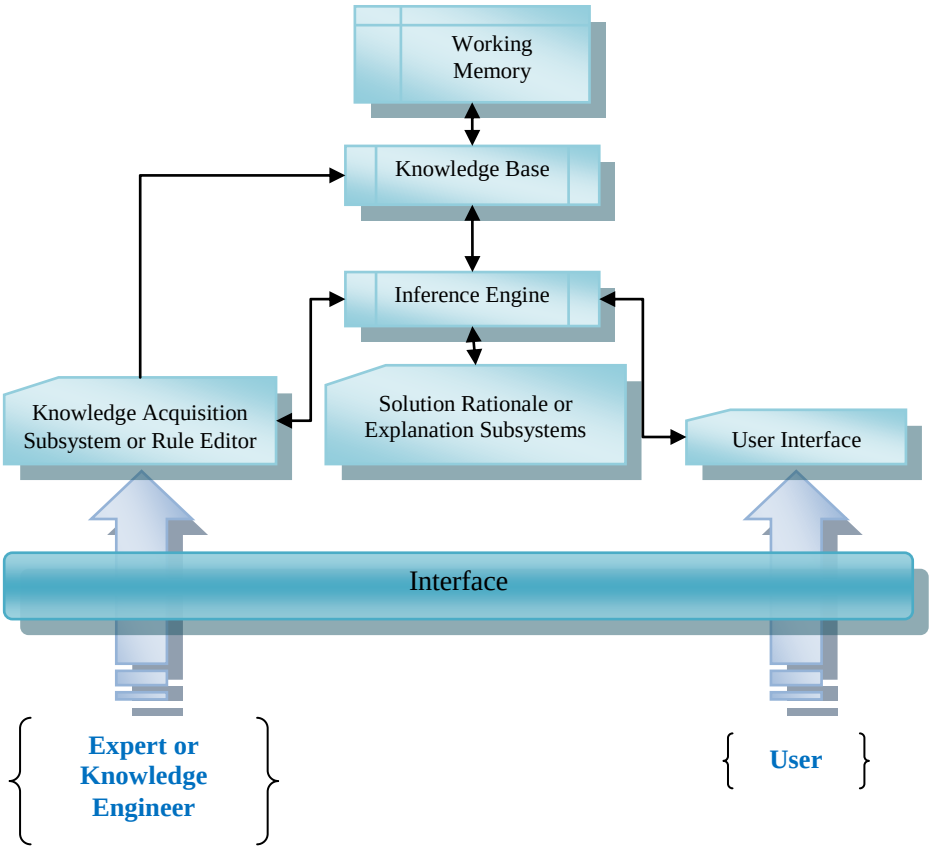


Fig. 3.4 The basic architecture of a Knowledge-Based System (**Graham 1997**)

Knowledge-based programs are intelligently solving complex problems by "providing a special explicit representation and processing techniques of knowledge involved in resolving the matter" (**Trăuşan 1998**).

According to **Graham 1997**, in a problem solving program, usually the following components can be found:

- a representation scheme in which the problem, and the knowledge required for its solution, can be expressed;
- a Knowledge-Base, constructed using the representation scheme – what is known about the problem domain;
- a Knowledge about the particular problem, encoded in a similar way;
- an "Inference Engine" or Search method which allows us to map onto.

There must be noticed that the knowledge which is selected for inclusion in the knowledge base can have a variety of forms, depending on the form of the inference engine.

On that account, an Inference Engine is a method of using the rules in the knowledge base to derive a conclusion. This might take the form: *if* condition *then add* outcome *to* context. Where *outcome* is the conclusion if *condition* is true, and *context* is an area where knowledge is recorded if conditions are true. An *outcome* is often part of another *condition* that matches another rule.

In other words, the inference engine takes the rules provided by the knowledge base, and uses internal rules of inference to draw a conclusion. As was suggested by **Fischer 2002**, the claim is that the internal rules are general to all inference. So the inference engine is a set of rules which are applied to the rules in the knowledge base. The inference mechanism is thus critical to the outcome: it is responsible for any interrelation of elements beyond the rules in the knowledge base.

Most designers consider it important that the rules can be easily inserted, modified, or deleted from the knowledge base, in any order. They usually consider the rules to be weakly connected: there is no sequencing information about the order in which they can apply, and the only connections between them are the use of common terms of reference (**Fischer 2002**).

Making a step farther, there is an evaluated type of knowledge-based system which can solve many rules issues and can be described as follows:

> *a knowledge-based system with an evaluated level of performance close to that of an expert* (**Graham 1997**).

Expert systems have started to be used since the eighties in aeronautical engineering. One of the well-known systems for the configuration of the layout of passenger cabins (according to **Kopisch 1991**) was XKL and was based on PLAKON which is a special kind of expert system called configuration system.

> *Configuration System is any one of the fields in expert system technology in which the application of AI-methods has advanced a great deal over the past few years (***Kopisch 1992***).*

In order to meet the technical challenges set for the future of aviation, new design systems are required to increase engineers' productivity. In the current design approach of a complex product like an aircraft, too much time is wasted in lengthy and repetitive activities; not enough available time remains in order to investigate more product configurations and to better exploit designer's skills and creativity (**Fischer 2002**).

For all of these, KBE is proposed as a suitable technology to help designers reducing time and cost for engineering applications by automating repetitive design tasks and supporting the systematic application of design best practices.

3.4 Knowledge-Based Engineering (KBE)

3.4.1 Definition and Concerns

Knowledge Based Engineering is a term introduced by Feigenbaum in 1977 and it describes the technology of the knowledge-based expert system.

Another definition for Knowledge Based Engineering can be: "the science of identifying, recording and re-using engineering knowledge" (**van Tooren 2009b**).

One of the most important characteristics of Knowledge Based Engineering is that it is aimed towards improving designers' productivity and to give them free time for creativity and innovation. According to **La Rocca 2007**, it can be used in many domains like the designing of high-tech automotive components and very large scale wind turbines. In this paper, we will discuss how it can be used in aeronautical industry. KBE technology can be considered as arising from the fusion of Artificial Intelligence and Computer Aided Design.

> *Indeed, Rule-based design, object-oriented modelling and parametric CAD represent the cornerstones of KBE technology* (**La Rocca 2007**).

Also, Knowledge Based Engineering (KBE) with its inherent capability of combining the design flexibility of a CAD system with the versatility and control power offered by the programming approach represents a promising solution for several of the designers needs.

The next figure shows the design paradigm according to the implementation of knowledge based engineering principles. The product (or generative) model represents the central repository of the design knowledge and plays a pivotal role in the design process. It represents the formalization of the design team relevant knowledge (engineering rules and reasoning mechanisms) by means of a scripting language (**La Rocca 2009**).

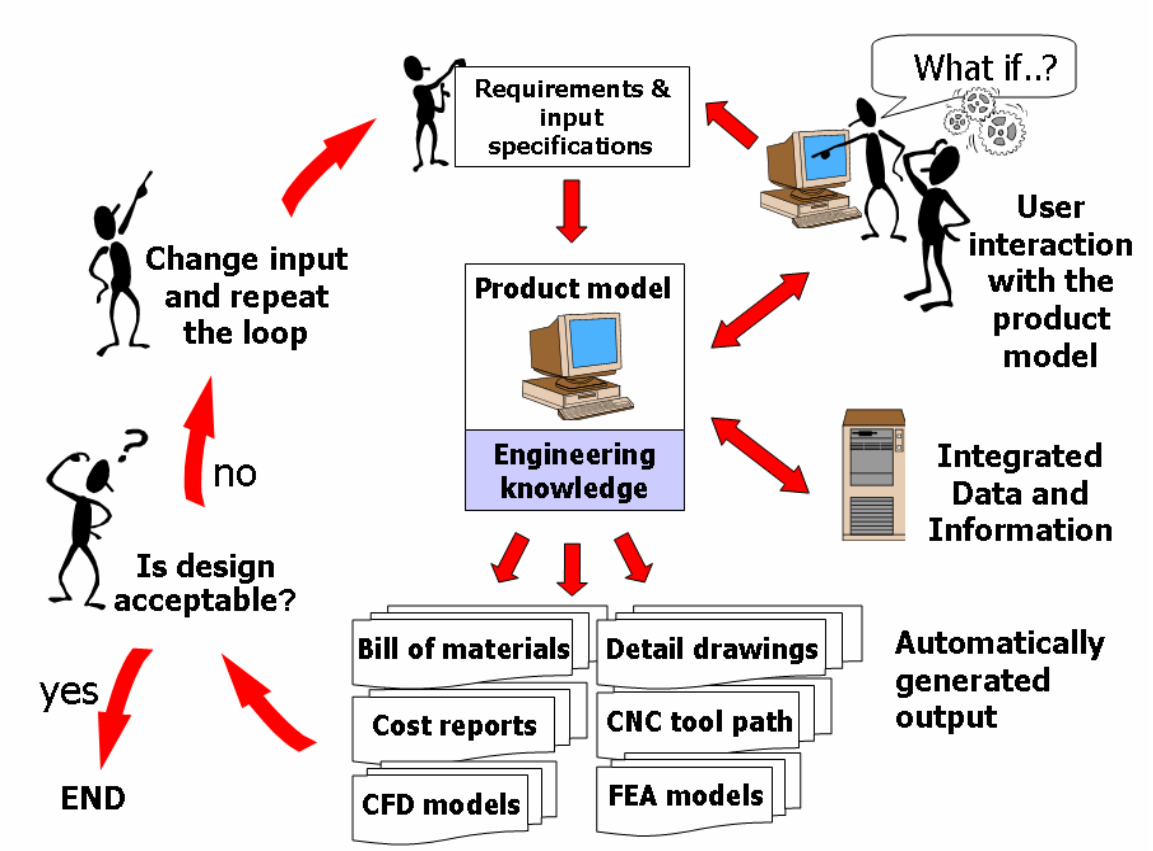


Fig. 3.5 The Knowledge Based Engineering Design Process (**La Rocca 2009a**)

This integrated environment, where Artificial Intelligence meets CAD, represents the most relevant feature of KBE systems. As is indicated by **La Rocca 2009a,** the functionality of the product model (or generative model) can be described by the simplified representation of the next figure: a set of input values is assigned to the parameters used by the engineering rules and reasoning mechanisms encapsulated in the product model, the KBE system brings these rules to bear in a systematic way and the engineered design is automatically generated as output.

As sketched in the last figure, the designer is fully in charge of the creative part of the design process: just by editing the input values for the product model, he can exploit the generative capabilities of the product model to automatically produce a potentially ready to be verified with (external) analysis tools. The designer can focus on what-if design, without repetitive involvement in activities associated with the generation of the data and information actually needed to evaluate the various what-ifs (**La Rocca 2009a**).

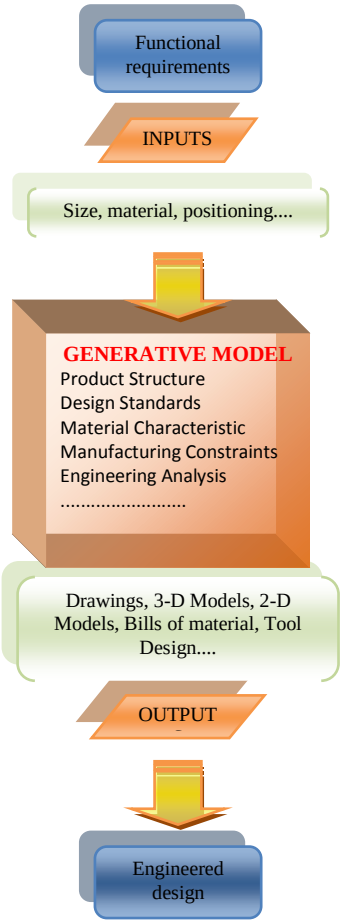


Fig. 3.6 The product model (**La Rocca 2009a**)

Moreover, the integration of rule-based, object-oriented design with parametric CAD, as provided by KBE, offers the possibility for flexible modeling of both product and process – according to **La Rocca 2009a**. Therefore, as stated earlier, the most relevant feature of KBE is its capability to merge *parametric CAD* with *rule based object-oriented design.*

Furthermore, rules combined in objects allow the effective manipulation of engineering knowledge. In order to support the design of aircraft, this feature has been implemented with the High Level Primitives: objects containing product and engineering knowledge that can be used and re-used in different aircraft configuration.

Summarizing, there are some advantages that must be highlighted:

- the expert does not have to assemble manually new analysis models when changes occur in the product configuration; in this way, the occurrence of human errors is reduced;

- more cases can be evaluated, more what-if scenarios can be investigated, more time can be dedicated to creative design;
- the consistency of the multidisciplinary analysis process can be guaranteed;
- rules and best practices can be systematically applied.

As it is highlighted by **La Rocca 2007**, though KBE has been *in service* since more than 20 years, the recent development of affordable KBE tools and the increased need of the industry to efficiently retain and exploit corporate knowledge, create a new momentum and set the conditions for an authentic KBE renaissance.

The contribution of KBE to the today industry is extremely significant especially regarding the time reduction. It is shown by **La Rocca 2009c** that Knowledge Based Engineering (KBE), which is a proper combination of object oriented programming, rule based instantiation of objects and a geometry engine, is very helpful for the designer in the preliminary design phase.

The object oriented approach allows the representation of the product and engineering process structure. The KBE environment gives access to a parametric geometric modeler and this allows the rule base to perform all geometric operations normally available in a CAD program (**La Rocca 2009c**).

However, the design process aims at finding a set of "optimal" product specifications (model and behaviour properties) to a certain set of requirements (functions, performances, and constraints):

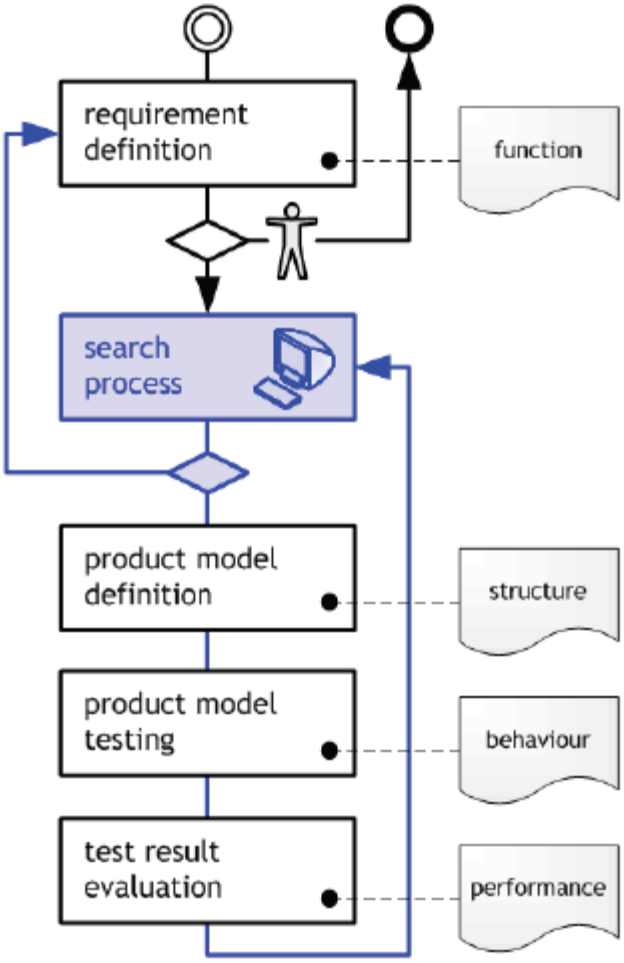


Fig. 3.7 KBE-enabled design process (**Schut 2000**)

Furthermore, Knowledge Based Engineering (KBE) can provide tools to harness and exploit engineering knowledge and design skills, and accelerate the transition of new concepts and technologies into operation. KBE allows designers to capture and reuse product and process multidisciplinary knowledge in an integrated way, in order to reduce time and cost for engineering applications via the automation of repetitive design tasks and a systematic application of design best practices.

According to **La Rocca 2009a**, object-oriented analysis and modelling can provide the analytical and structured approach to develop models of complex systems, which can be translated into KBE applications. In the next pictures some applications of KBE in the aeronautical field are described:

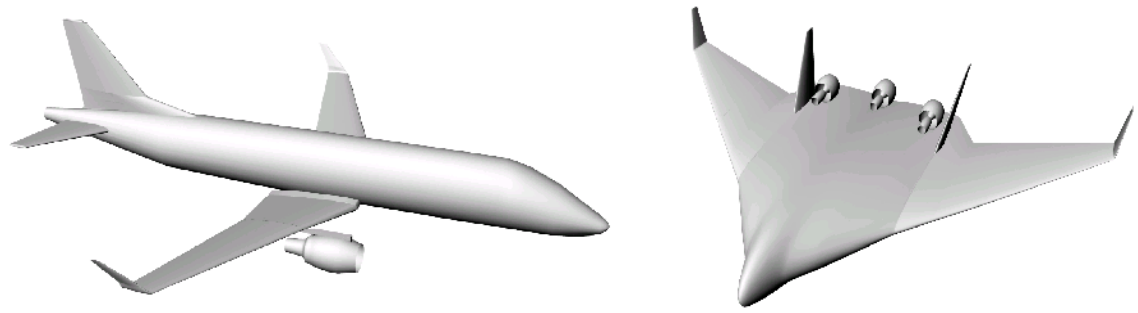


Fig. 3.8 Commonality between different aircraft configurations captured with KBE Based parametric modelling (**van Tooren 2009a**)

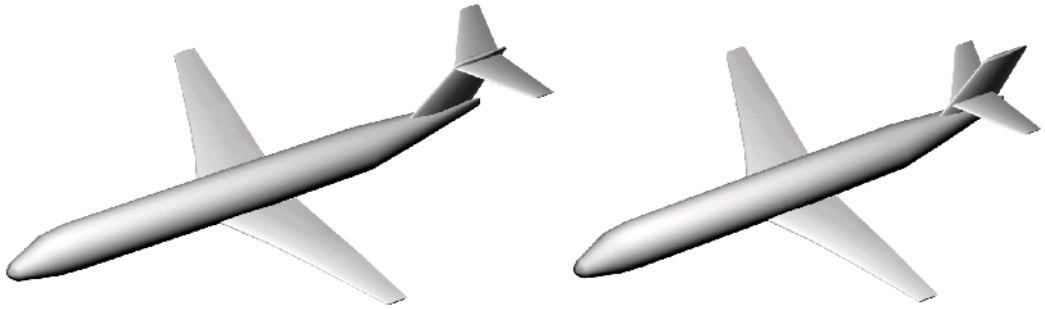


Fig. 3.9 Variation within an aircraft type captured with KBE Based parametric modelling (**van Tooren 2009a**)

The KBE technology helps to structure and record knowledge in such a way that (engineering) knowledge becomes reusable, transferable and expandable. The reuse and ease of extending of knowledge will help to meet the increasing knowledge demand (**van Tooren 2009a**).

A KBE application, basically operates as follows: first, the values of the parameters used in the definition of the product model rules must be assigned, either by reading them from a kind of input file (edited by the designer or automatically generated by some software tool) or by inserting them via some kind of user interface. Then, the product model is instantiated; the rules recorded in the rule base are systematically evaluated and applied by the KBE system. Finally, geometry, or some other model possibly not including any geometrical entity, is generated *with little or no human interaction*.

In conclusion, as is written in **La Rocca 2009a**, designers need more flexible and powerful tools that allow virtual access to their ideas, providing the base for an effective

multidisciplinary design, and increase the time and the freedom to investigate multiple what-ifs about their design.

3.4.2 Approach of the KBE to the Cabin Layout

A KBE system can be figured out as a sort of *Expert System* (as in **Russel 1994**) provided with the advanced geometry manipulation capabilities (as in **Chapman 2007**). Also, an expert system can have many applications and we will refer to its applications to the Cabin Layout.

As it was shown by **Kopisch 1991**, the difficulty in configuring a cabin layout is to consider all restrictions and requirements at the same time. Besides, one has to strive for optimality with regard to criteria like number of seats, comfort level, cost, delivery time etc., which are different for each order.

However, configuring a cabin layout is getting harder and this is due to the constantly increasing size of aircrafts permitting more and more cabin interior components to be placed and a rising number of certification rules.

> *Especially, the aircraft builder's representative is no longer able to consider every restriction and thus cannot assure correctness and optimality of his layouts* (**Kopisch 1991**).

For these challenges, there is the expert system which might provide assistance by relieving the representative from routine tasks. It could make the necessary tests for correctness and thereby guarantee the consistency of the layout developer's decisions. Moreover, such a system - like the expert system - should be able to carry out parts of the configuring and optimizing on its own. Expert system technology must provide suitable formalisms and mechanisms to handle typical problems of a configuration task (**Kopisch 1991**).

According to **Kopisch 1991,** a configuration task can be characterized through a set of components, a description of their properties, namely attributes and possible attribute values, connection points (ports), and constraints on legal configurations. Given some customer requirements, the result of computing a configuration is a set of components, corresponding attribute valuations, and connections satisfying all constraints and customer requirements.

The configuration process is divided in generally in three phases: passenger cabin decomposition, passenger class parameterization, specification of the cabin interior components (**Kopisch 1991**).

Also, a representation of the dependencies between objects is needed in addition to the representation of these objects and their properties and that is why the constraints are a very important issue.

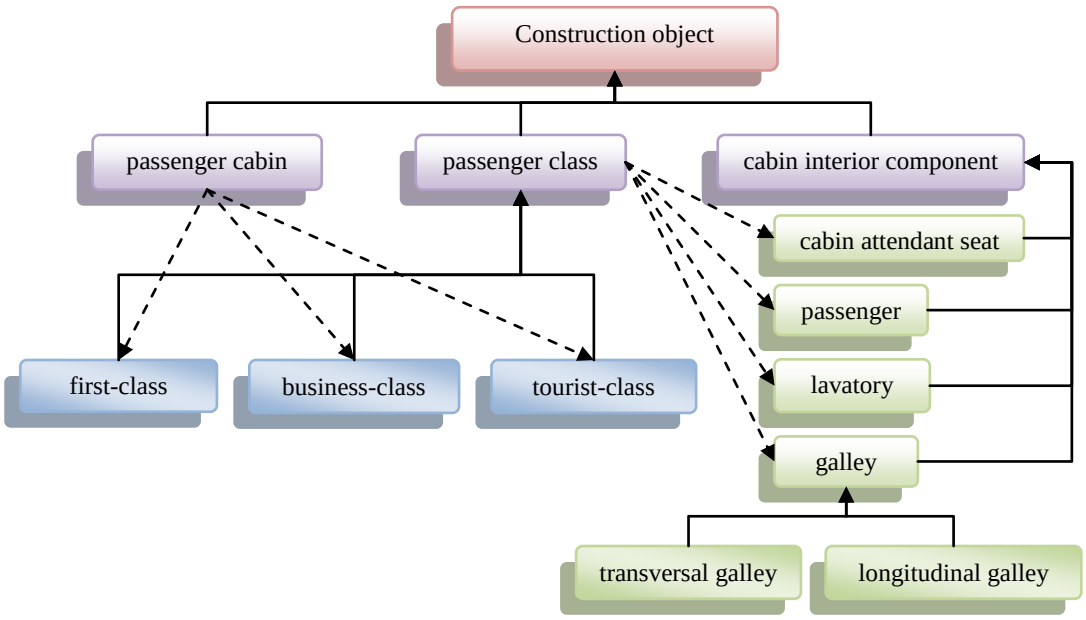


Fig. 3.10 Taxonomic and compositional hierarchy of the Cabin Layout domain (**Kopisch 1991**)

The taxonomic (defined as the science of classification) and the compositional hierarchy together describe all admissible configurations of a cabin layout.

Besides, the possibilities to choose cabin interior components for the cabin layout are restricted by technical requirements and certification rules. The values for their properties are also restricted. According to **Kopisch 1991**, there are several constraint-classes like *adder*, *multiplier*, *number-restriction*, *sum*, *equal* etc.

In the following example, dependencies between the properties of a passenger class are described declaratively in a conceptual constraint:

> The airline request for passengers and for the level of service provided by the cabin attendants results in the necessary number of cabin attendant seats in the concerned class.

There is a compound constraint by the use of two constraint-classes:

```
( Constrain     ( (#?C  ( a passenger class ) ) )
                ( number-restricition   ( #?C has-parts )
                                        ( a cabin attendant seat )
                                        ?CAS
                ( multiplier            ( #?C passenger-per-attendant )
                                        ?CAS
                                        ( #?C passengers ) ) )
```

Fig. 3.11 Constraints (**Kopisch 1991**)

In the first part of the constraint one or more concepts are specified to which the constraint will be attached whenever the concept is instantiated. In this example, the constraint will be attached to each passenger class. In the second part, the dependency is described by means of a constraint-class and the variables of the constraint.

> *The variables are either object variables or internal variables. Object variables refer to slots of objects in the partial construction, such as (#?C passengers), which refers to the slot passengers of an instance of the concept passenger class. Internal variables like "?CAS" are used where internal connections between constraints exist* (**Kopisch 1991**).

Obviously, using an expert system, the knowledge engineer is relieved from thinking about evaluation of a constraint. Even the dependencies between several restrictions are considered by a constraint net. This is a great advantage over rule-based systems.

Conclusioning, product configuration creates big challenges on software development (**McDermott 1980**):

- the complexity of the task requires the sophisticated knowledge of technical experts;
- the configuration knowledge base has to be adapted continuously because of changing components and configuration constraints;
- configurator development time and maintenance time are short and strictly limited. Development of the product and the product configuration system has to be done concurrently;

3.4.3 Advantages due to the use of Knowledge-Based Engineering in Cabin Refurbishing

As it was highlighted in the previous subchapters, during the conceptual and preliminary design phases of aircraft (or any other hardware of similar complexity), designers need tools to facilitate the instantiation of their ideas and creative insight and to analyze and evaluate the quality and performance of such ideas. Concerning the first need, CAD systems are by far the most widespread tools at date. However, despite their indisputable impact on the overall design process, they are not capable by their nature to support a true conceptual design approach (**La Rocca 2009c**).

Various design tools are used to support the different stages of the design process. Each tool can be considered a substantiation of design supporting knowledge. All tools together, added to the specific knowledge and craftsmanship of the design team complete the knowledge base for the design work. According to **van Tooren 2009b**, Knowledge-Based Engineering is a modern approach for the compilation of knowledge required in a product development process. It aims to the identification, record and re-use of engineering knowledge, by combining Artificial Intelligence techniques, IT tools and Object-Oriented methodologies.

The main idea is to capture the engineering knowledge and formalise it in set of re-usable rules. When these rules are applied to different sets of data, new instantiations of the knowledge are generated in support of new products design. Moreover, as was mentioned in **Cooper 1999,** in this way, families of products (classes) can be defined in rules, and family members can be generated (instantiated) automatically, based on a given input data set. Many case studies of such application of the KBE methodology are now available for products of different complexity.

However, there are many reasons for implementing KBE in an engineering process. Although KBE is not suitable for replacing every phase in the process, KBE can be used to automate repetitive work, and thus should not be used to automate the creative part of the process **(Stokes 2001**).

Generally, the process of storing the knowledge has three steps according to **Vermeulen 2005:**

- creation of a knowledge database;
- definition of parametric model;
- implement design rule base (design principles).

The first step is to create a database containing the captured knowledge. This database consists of an input file containing the product information and design rules that apply to the product model. In the second step the product information and the rule base are used to define a parametric product model. In the third step the design rule base is implemented, using the product information already stored in the product model. Finally the properties of the product model are evaluated and the product information is adapted until the specifications are met (**Vermeulen 2005)**.

In this subchapter, layout problems will be presented. They are a special class of configuration problems with a main focus on spatial aspects. Configuring a cabin layout means to place a certain number of cabin interior components optimally in the restricted space of the passenger cabin. This depends on the number of seats, the lead-time or a mix of these factors. In this process, there are some restrictions like certification rules of the approving airworthiness authority or technical requirements. The division of the passenger cabin into passenger classes is taken into account (**Kopisch 1992**).

> *The developer of a cabin layout needs knowledge of the following fields:*
> *-available cabin interior components and their properties;*
> *-technical details of the aircraft (measures, capacity of supply for water and electricity, mounting - points for the cabin interior components etc.);*
> *-technical restrictions and technical dependencies between objects;*
> *-certification rules;*
> *-airline requirements;*
> *-proceeding during layout development*
> **(Kopisch 1992).**

The spatial arrangement of the components is of great importance for the configuration of a passenger cabin and also has consequences for the objects. Configuration tasks have the following characteristics (according to **Guenter 1990**):

- large solution space;
- objects are composed of components;
- dependencies between the objects;
- heuristic decisions;
- consequences of the decisions are not totally predictable.

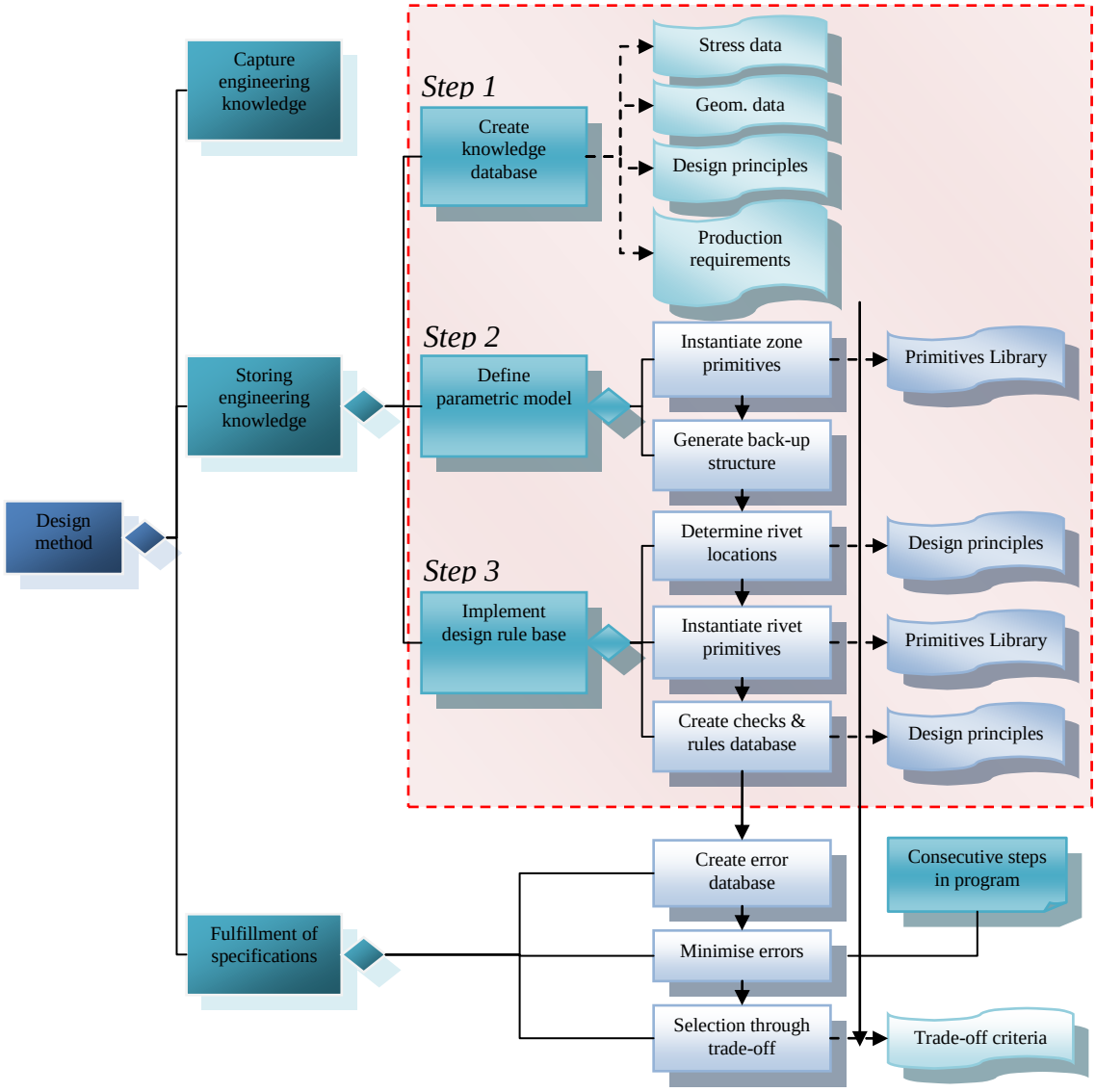


Fig. 3.12 General approach for the design tool (**Vermeulen 2005**)

Furthermore, a graphical representation of the developing layout is of crucial importance for the use of the expert system. There is need of a user interface, which enables the user to operate with the cabin interior components on the screen and to place them in the passenger cabin. The available cabin interior components are visualized in a way which shows all possible locations. The next figure suggests how an expert system can present all possible locations for longitudinal galleys for example:

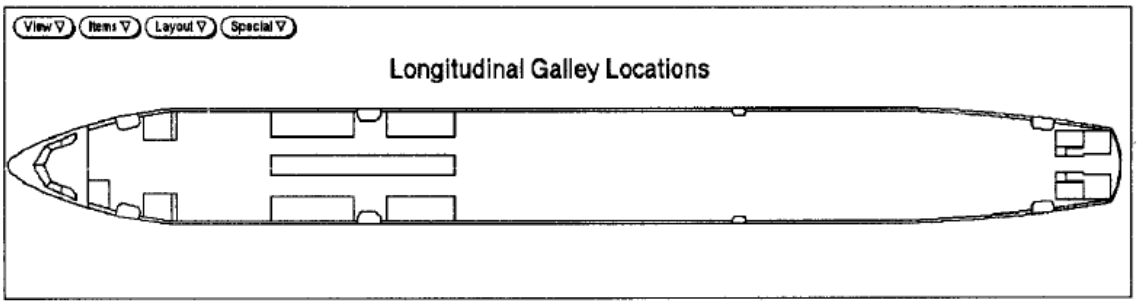


Fig. 3.13 All possible locations for a longitudinal galley (**Kopisch 1991**)

In addition, the user should be able to ask for information about the cabin interior components by clicking on an item.

Besides, one must handle restrictions and requirements at the same time and, with time, there are more and more interior components to be placed and a rising number of certification rules. Because of this, as it is shown by **Kopisch 1992,** there is a need of a program which could *provide assistance by relieving the representative from routine tasks*.

Moreover,

> *engineering knowledge can be recorded in a rule base. In an object-oriented environment the rule base can be used to create a product model using product information specified by the user. The rule base can be re-applied on new product information, instantiating different products within a product family. By changing the rule base different product families can be instantiated* (**van Tooren 2009b**).

What there must be noticed is that the principles of KBE can be used to generate and evaluate new designs quickly and easily by changing the product information. This frees the engineer from time-intensive, detailed engineering tasks such as repetitive calculations and allows more time for creative design work (**Vermeulen 2005**).

Using KBE the influence of design changes can be studied in more detail and different solutions fulfilling the engineering requirements can be generated. KBE can furthermore be used to store knowledge, making it available for a company at any given moment. Since knowledge is implemented automatically, KBE also provides a platform for less experienced designers to design a very complicated product, which would normally require long experience.

By using such platforms, Cabin Layout Configurations can be changed quickly by changing design parameters. This is advantageous for aircraft parts because a conceptual

design of the various aircraft parts can be made even while the final configuration of the aircraft is not fixed. Through parametric modelling the parts can be updated easily to design changes at a higher assembly level. Work done before the change can therefore be reused. As is it shown by **van der Laan 2004**, this also works the other way round, a lot of information about the different aircraft parts is available early on in the project, and this can result in a different, more optimized, final aircraft configuration.

KBE applications are being developed in an increasing scale, especially in large companies in the automotive and aerospace industry. As is it shown in **Vermeulen 2005**, competitiveness based on innovation combined with short time to market will be the key to continued success of aircraft manufacturers.

Knowledge-based software solutions from PACE are already well-known and widely used by AIRBUS Cabin Engineering. As is it highlighted by **PACE 2009b**, Airbus uses the Pacelab technology platform for the layout of various cabin systems.

4 Pacelab Cabin

4.1 Introduction to the Software

Pacelab Cabin of Pace Aerospace Engineering and Information Technology GmbH is a standard aircraft cabin configuration tool. It significantly streamlines the modelling and positioning of cabin interior items as well as the utilization of results by applying a consistent Knowledge Based Engineering (KBE) approach (**PACE 2009a**).

The use of Cabin interiors software by PACE at the Airbus Upgrade Services Centre – which is one of the five business areas of Airbus Costumer Services - for cabin refurbishment proves that this is a very good tool for aircraft cabin configuration. However, Mr. Rolf Kortig, Fulfil Costumer Order Manager Airbus Upgrade Services Centre Hamburg, Germany, said: "Using Pacelab-based software for retrofitting has enabled us to cut the average project acquisition cycle by 50 %".

The close collaboration between PACE's development team and the AIRBUS project team has been the key to the project's success. As is it said in **PACE 2009b**, beyond development expertise, the project benefited from PACE's deployment support, where training and consultancy services accompanied the introduction phase.

> *We are pleased to collaborate with PACE on the highly demanding project of automatic schematics generation. Both, technology as well as PACE's engineering expertise contributed to the success, and convinced us to further identify key process areas for future automation solutions based on Pacelab (says Manfred Gaida, Director of Technical Documentations Team at AIRBUS Germany – according to* **PACE 2009b***).*

Pacelab Cabin is applied at Embraer and at Sukhoi Civil Aircraft Company (SCAC) for the Sukhoi Superjet 100. Airbus has sponsored a tailored development of Pacelab Cabin to suit Airbus' needs (**Scholz 2009**).

Moreover, a reason that Pacelab Cabin is considered a very good tool in designing is that it provides parametric models of fuselage, cabin and cabin systems components. In applications, a template based approach is used which means that procedures such as changing parameters in the component database or in a given layout configuration, require no more time than modifying a single object.

Also, the way of creating layouts are very efficient (via drag & drop operations) and with intelligent positioning features. There is a high degree of automation and this *leads to the*

fact that complex procedures can be performed at the touch of a button – as it is written in **Pace 2009a** - and there are user-defined constraints and rules of arrangement.

Furthermore, the results can be displayed both in 2D and 3D visualizations with sophisticated 2D and 3D viewing features. As it is indicated in **Pace 2009a,** for obtaining the results with accuracy, there is a rule-based programming and this implies that if a rules is violated, there will be an announce automatically at the moment it occurs.

To make a long story short, Pacelab Cabin is one of the most powerful cabin design system which can simplify very much the way of making Layouts. In this chapter, the way of configuring a cabin layout and the rules implementation using the Pacelab Cabin Program is presented shortly.

4.2 Program Presentation

4.2.1 The Program Interface

After opening the program, at a first glance, there can be seen the Pacelab Tree, the Object Palette, the Browser Pane, the Graphics Pane as follows :

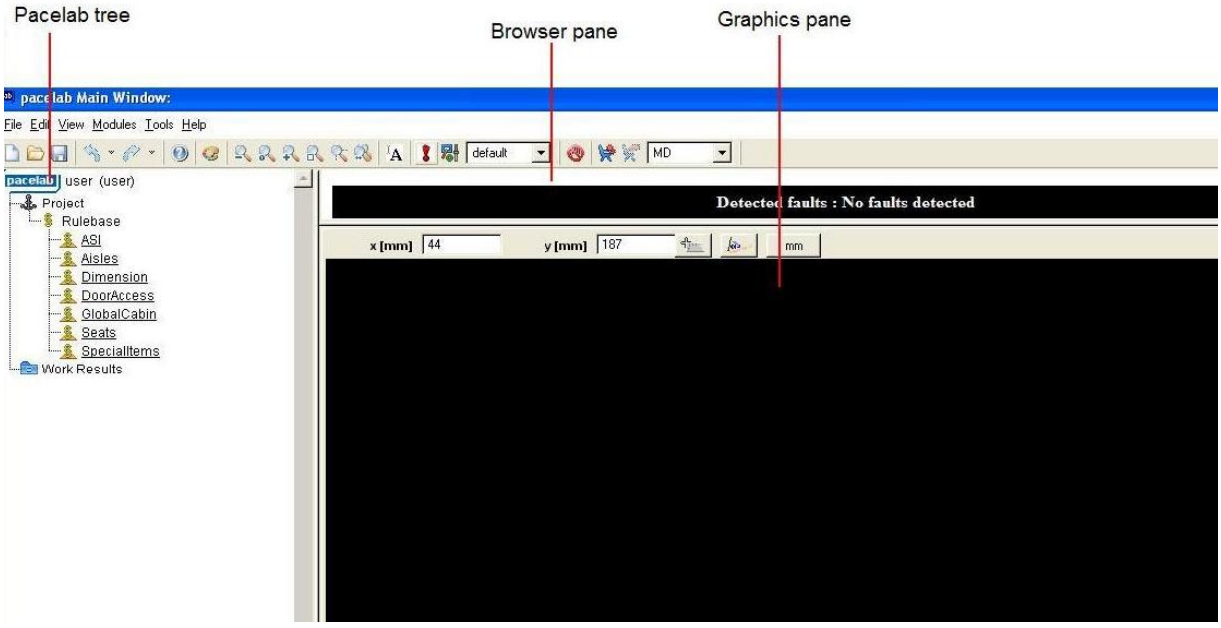


Fig. 4.1 The interface of the program

If something is changed in the Pacelab Tree, the Layout will change automatically and vice versa. If new items are added to the Pacelab Tree or some properties are changed, the Graphics Pane is updated immediately.

There are two principle menus: Configuration and Drawing Properties. The Configuration menu can be accessed from Tools like in the Fig. 4.2.

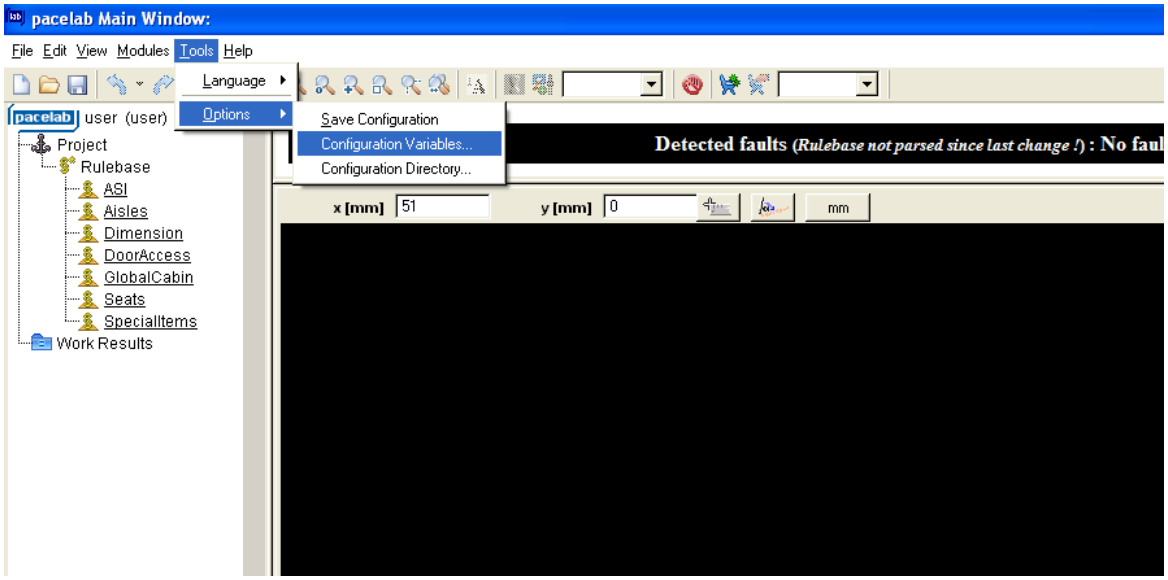


Fig. 4.2 Configuration menu

In the "Configuration" Dialog, there are several tabs like: Cabin Layout, User management, Output Generation, Pacelab Kernel, Pacelab Rules. In the Cabin Layout tab, the user can change for example the time in which the project is autosaved or he can specify which type of items he wants to be visible.

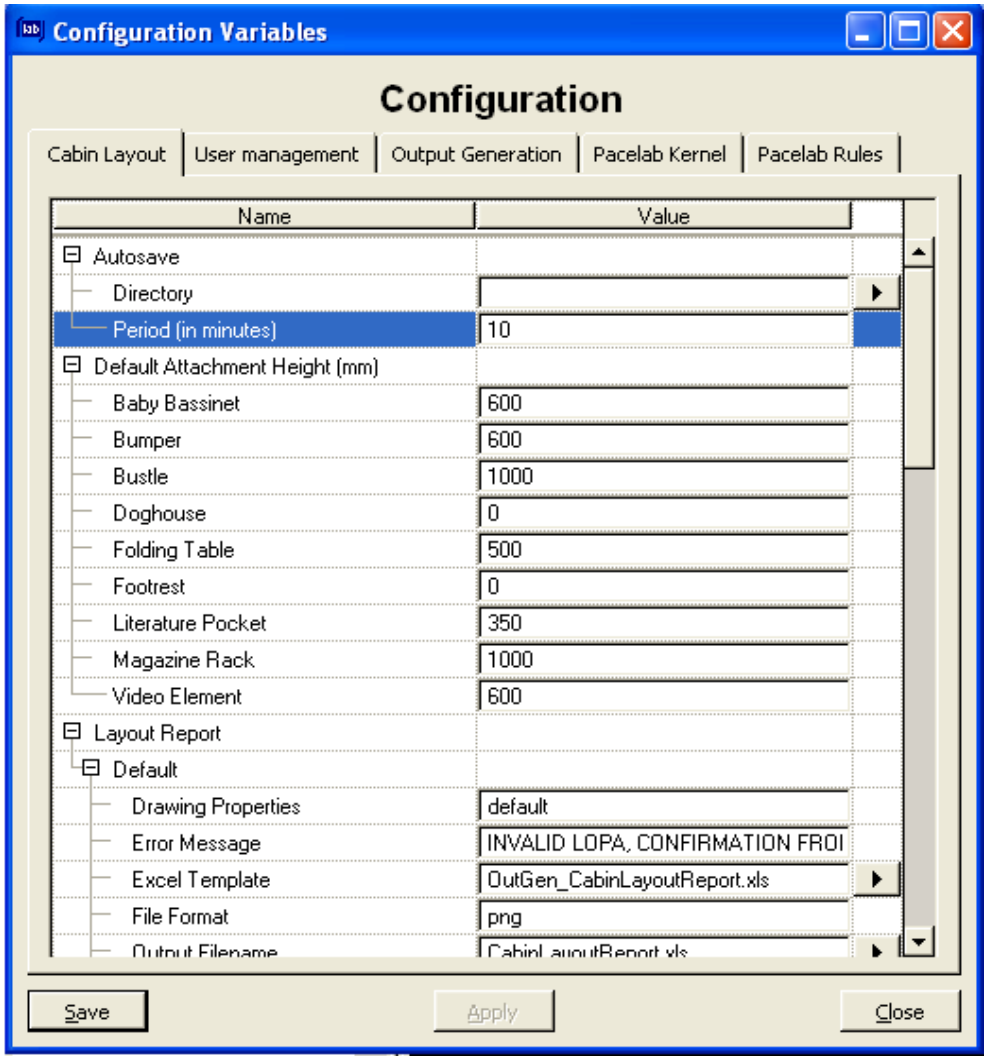


Fig. 4.3 The Cabin Layout tab

In the "User Management", there is the option for remembering the last login of the user, in the "Output Generation" there are some options regarding the saving operation, in the "Pacelab Kernel" the user can change colour settings and in the "Pacelab Rules", the user can decide if the warn on missing Rules licence will show or if the predefined rule base will be used instead of project rule bases:

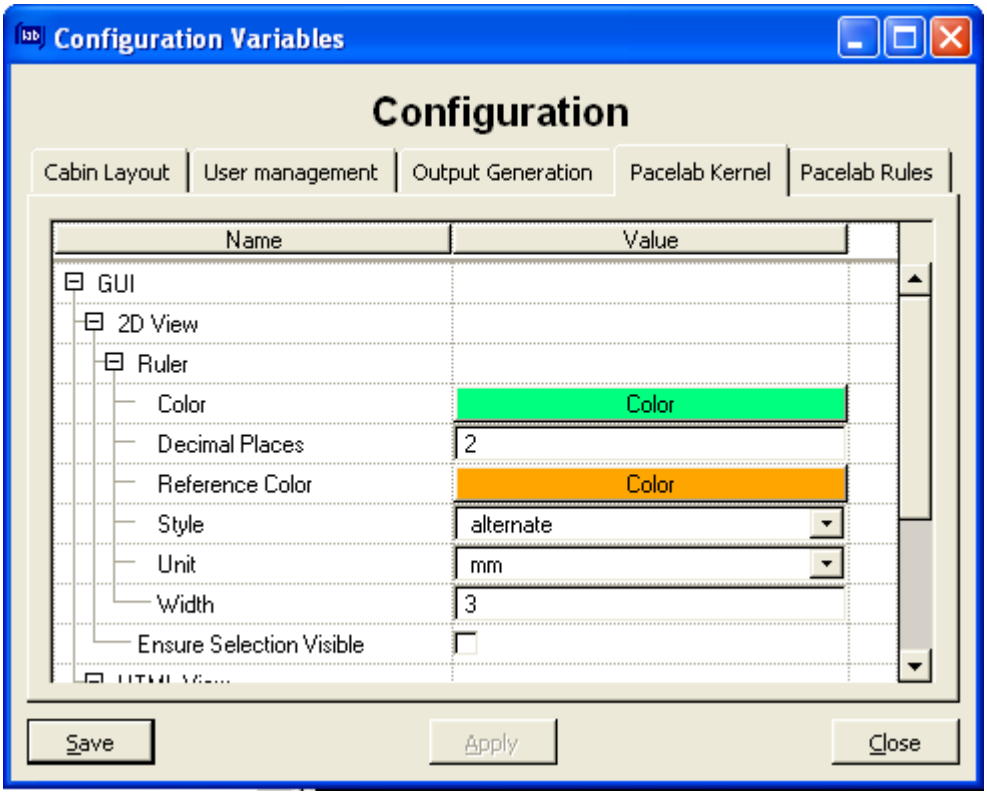


Fig. 4.4 Pacelab Kernel tab

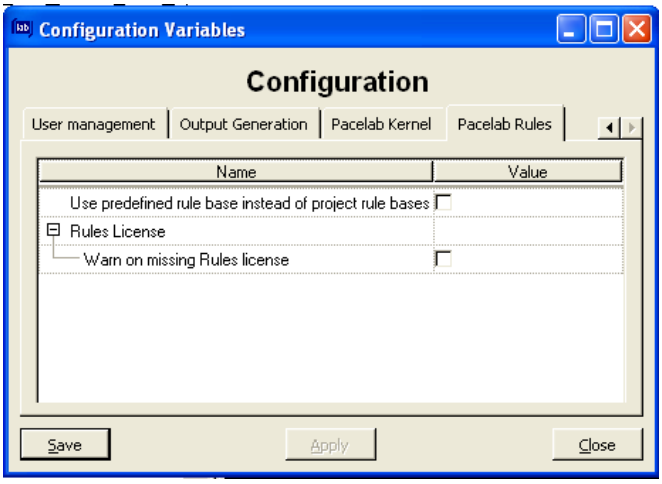


Fig. 4.5 Pacelab rules tab

The second important menu is “Drawing Properties” which can be use only after the Fuselage Prototype is installed. For this, after opening the File Menu and selecting from there ‘Open Prototype’, A340-300 Tutorial Prototype, for example, can be chosen from the list. Now, in the Graphic Pane, the fuselage contour can be seen.

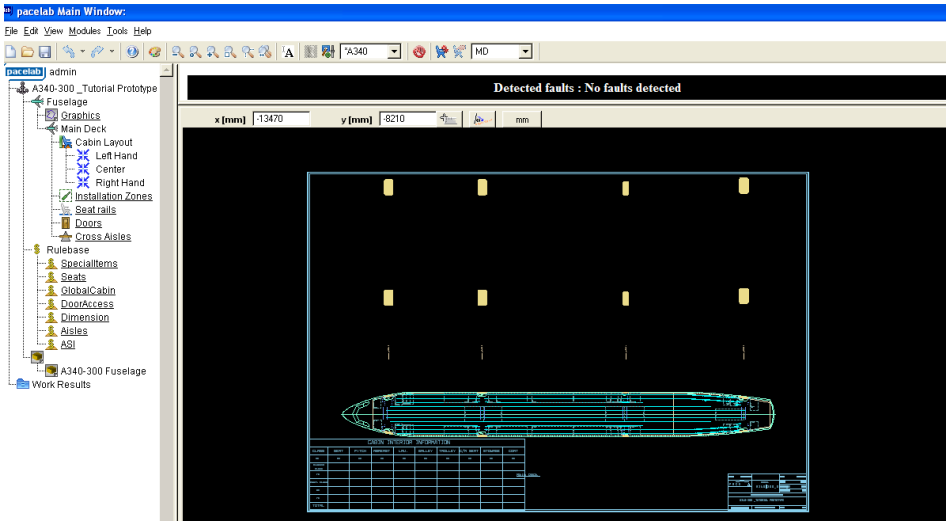


Fig. 4.6 Installing the prototype

The “Drawing Properties” Menu can be accessed by making click on his icon which looks like in the following picture:

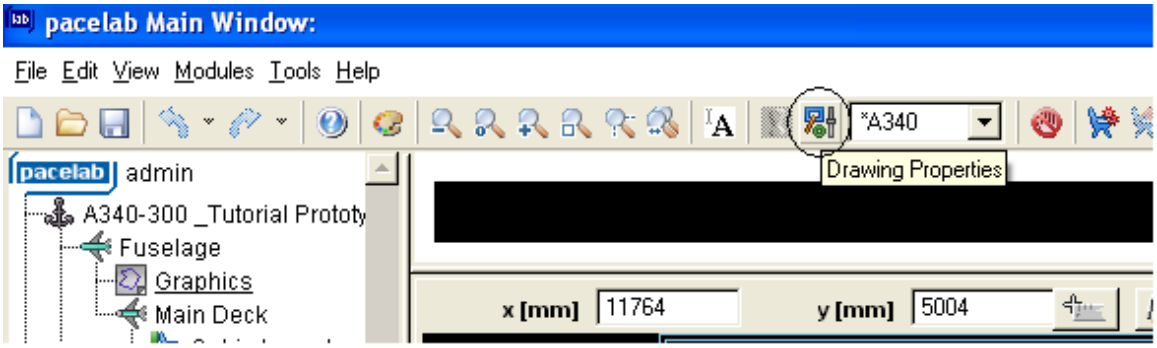


Fig. 4.7 Drawing Properties icon

The Menu contains several tabs: Cross Section Setup, Placement, Dimensioning, Printout Configuration and Fonts. The first one contains settings regarding the Cross Sections which appear in the drawing implicitly. Other tabs are regarding the automatic dimensions for example or the features which can be chosen to appear at the Printout.

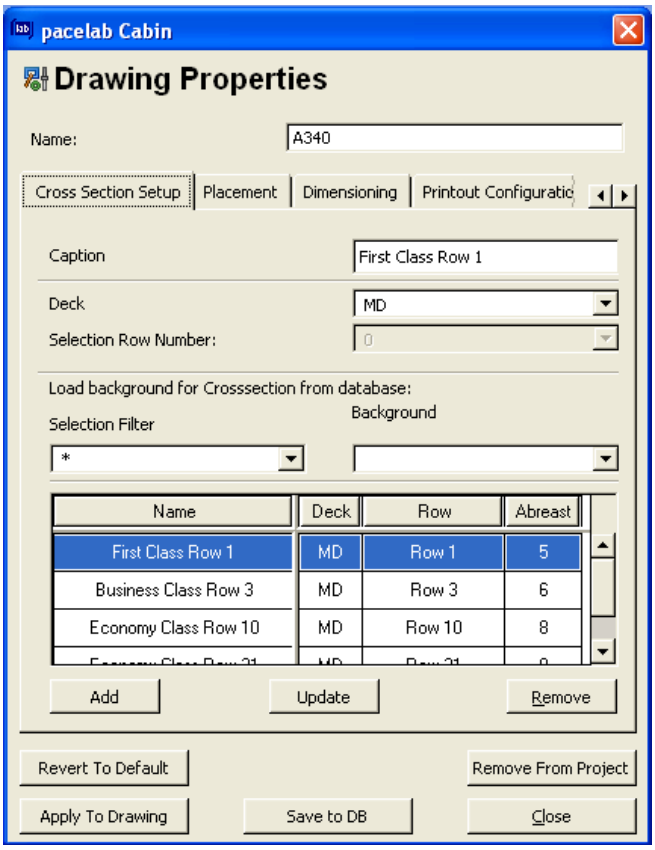


Fig. 4.8 Drawing Properties – cross section setup

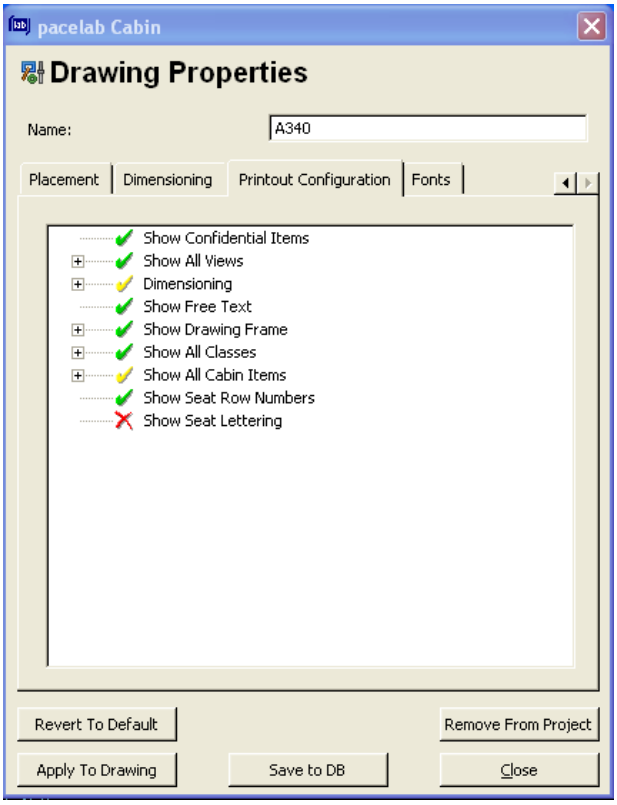


Fig. 4.9 Printout Configuration

Another important and indispensable tool is the Object Palette:

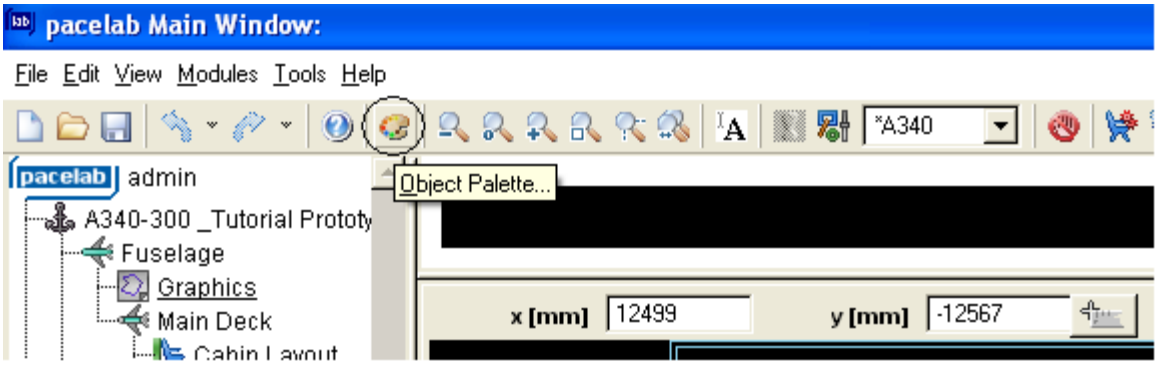


Fig. 4.10 The Object Palette icon

There are three options in the Object Palette Box: Rules, Structure and Cabin Layout. We will choose Cabin Layout to insert items to the cabin:

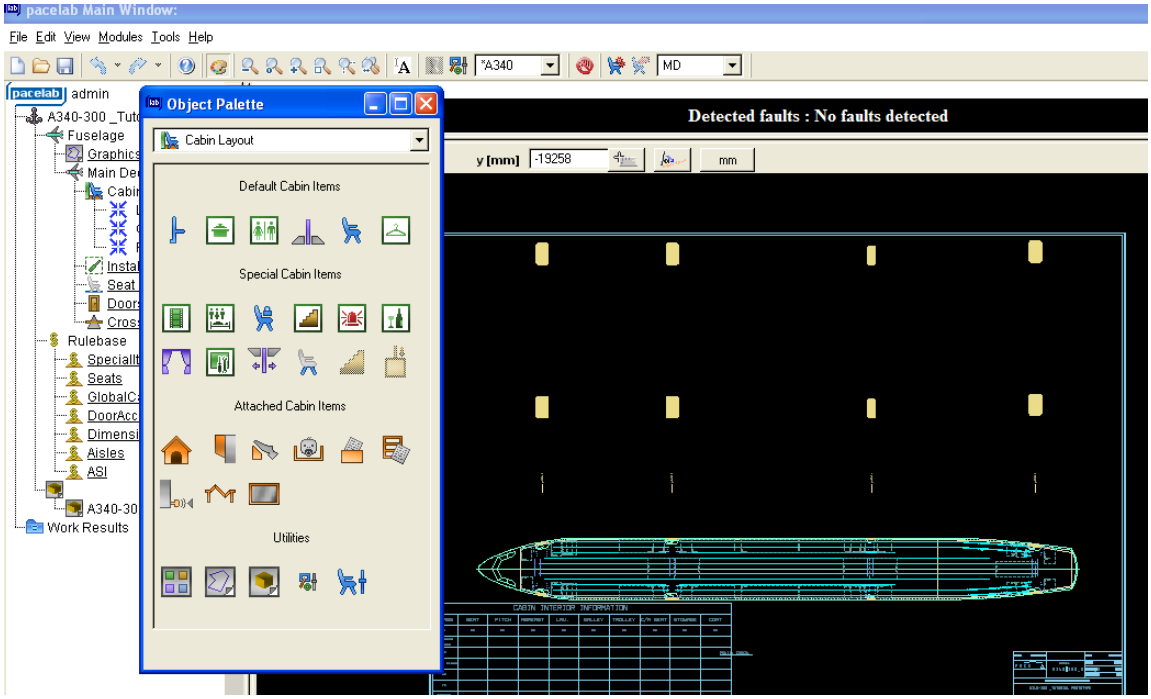


Fig. 4.11 The Object Palette

At the Cabin Layout option, there are: Default Cabin Items, Special Cabin Items, Attached Cabin Items and Utilities.

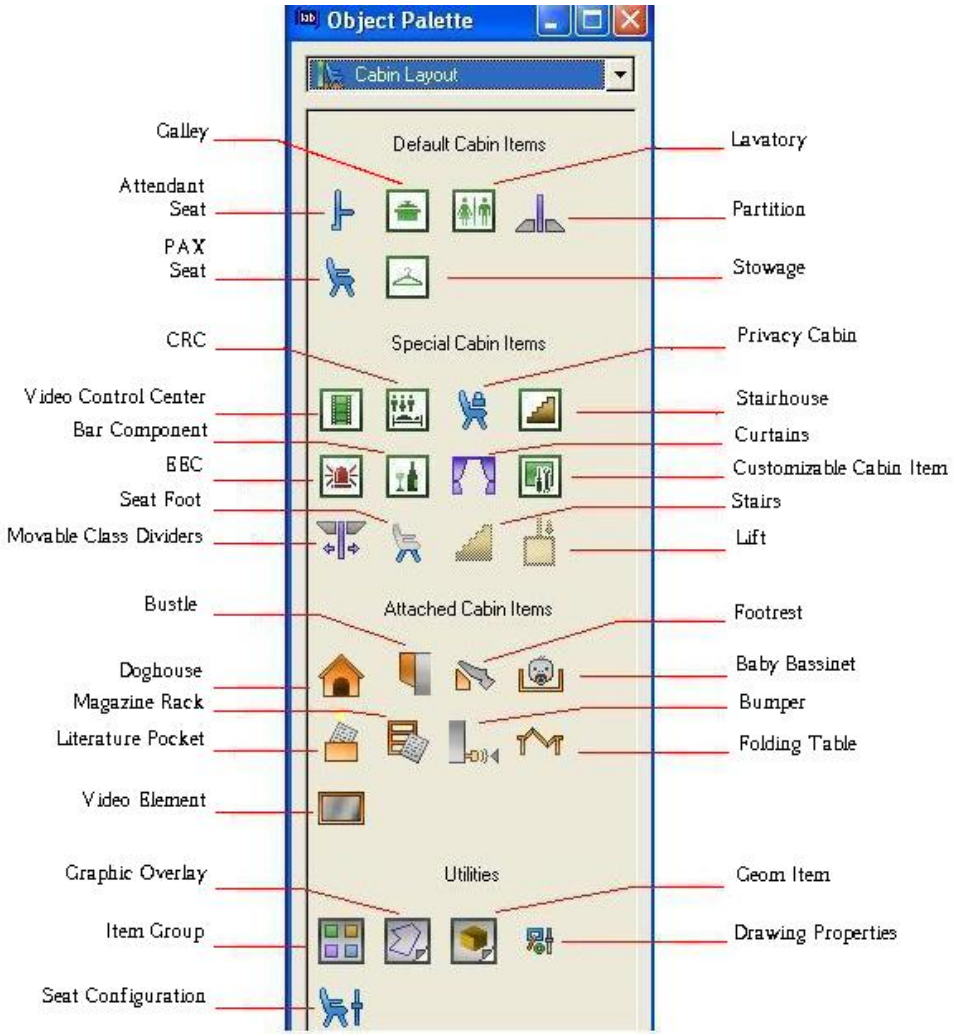


Fig. 4.12 The items from the Object Palette

For providing a suitable position and orientation for the items, the installation zones are used and they act as a guide for suitable installation areas. To make them visible on the layout, one can make right-click the “Installation Zones” folder in the Pacelab tree and select “Installation Zone Container – Items active”:

Installation Zones

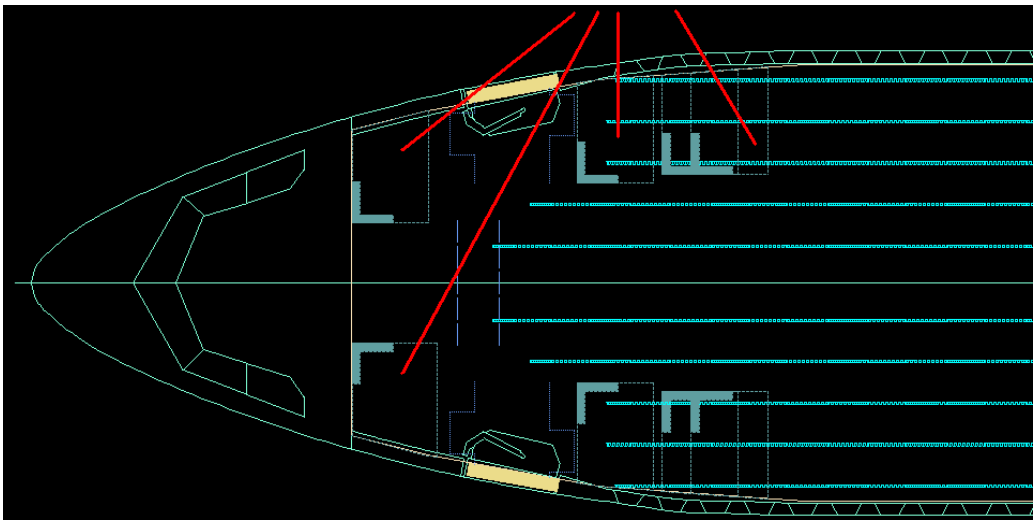


Fig. 4.13 The installation zones

The "Installation Zones" folder in the Pacelab tree has several subfolders for each kind of monument for which zones have been defined. Only one type of installation zones can be displayed at a time. In the next example, the 'Lavatory Installation Zones' are displayed while the other types are "forbidden":

Fig. 4.14 Pacelab Tree – installation zones

For displaying another installation zone type, one can make right-click the folder for the desired type, e.g. "Stowage Installation Zones", and select "Installation Zone Container – Items active" from the context menu.

In this chapter, the way of creating a layout for Airbus 340-200 will be presented. The numbers which are referred to the coordinates are taken from a trusted source, Pacelab Tutorial.

According to the Pacelab Tutorial, for every class, it's recommended, but not compulsory, to respect a workflow like follows:

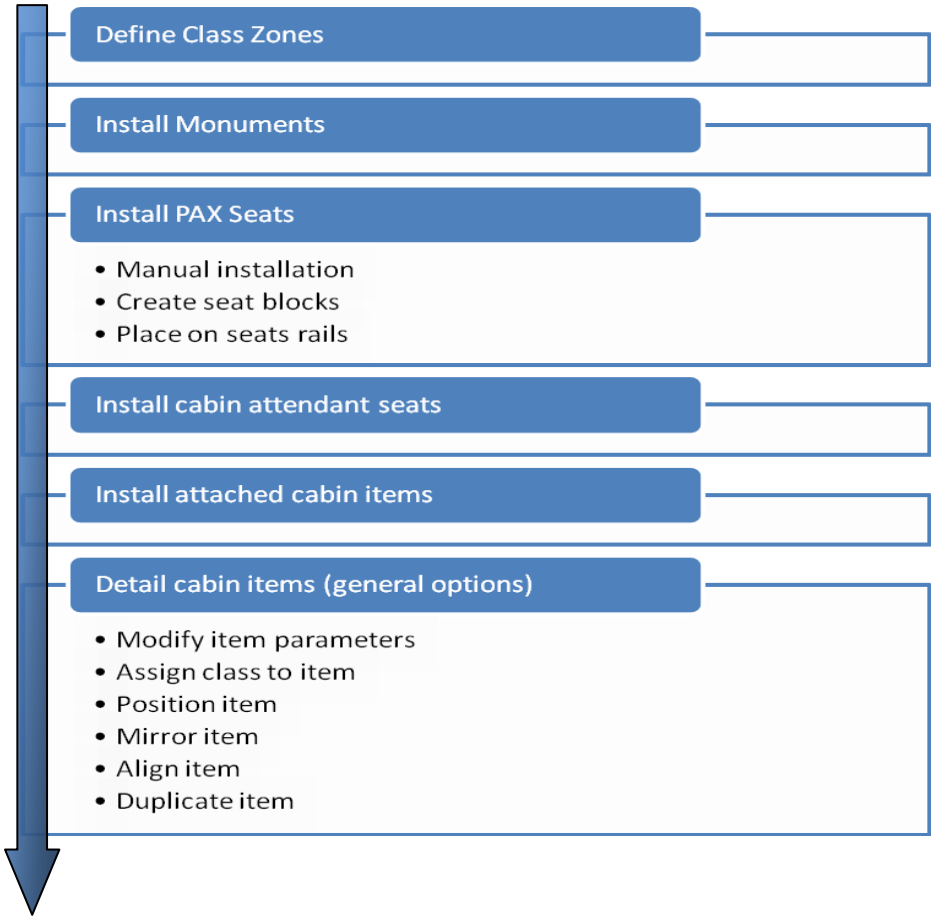


Fig. 4.15 The workflow in Pacelab Cabin

As there is shown, the first step is to define the Class Zones which divide the cabin into regions which are allocated to certain classes. A way to make that is to open "Class Zones Main Deck" dialog: - Select Modules – Cabin Layout – Class Zones – Main Deck from the menu.

Boundaries of class zones:

Start of first zone
Boundary between zones
End of last zone

X position of boundary at left, center and right installation side

Class for the zone after this boundary

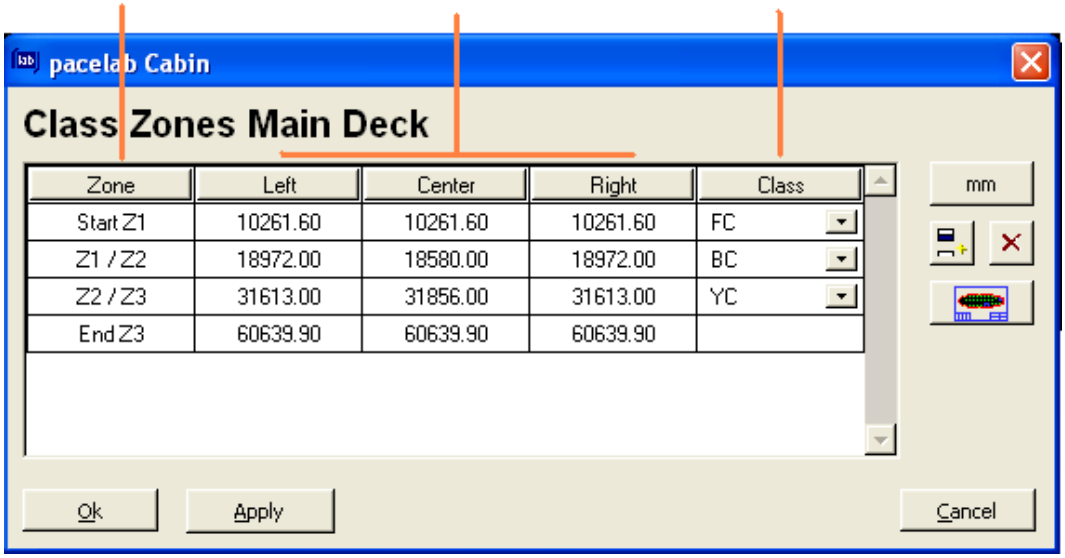
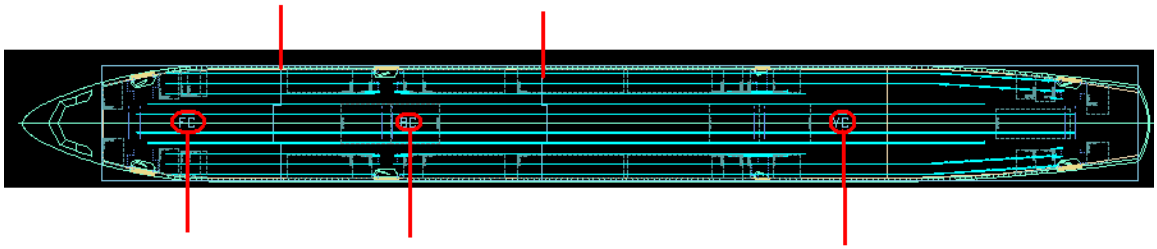


Fig. 4.16 The class zones Main Deck

While the dialog is open, there can be seen the class zones in the layout. Each zone is labelled to show its class :

Boundaries between class zones



Labels marking first class, business class and economy class zones

Fig. 4.17 The class zones shown in the Layout

Because the installation of first class, business class and the economy class have many similarities, in this paper, some steps of creation will be neglected.

4.2.2 The First Class

The second step is to install the monuments. In the First Class, the monuments will be added to the Layout via Drag & Drop. To install the first lavatory, the "Object Palette" dialog will be used. By selecting "Cabin Layout" from the drop-down list at the top of the Object Palette, clicking the Lavatory icon and holding the mouse button down, the Lavatory can be installed.

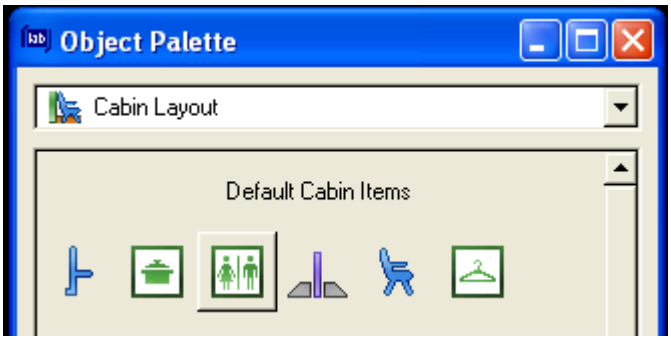


Fig. 4.18 Selecting an item from the Object Palette

Dragging the icon to the cabin layout and looking for the installation zones for lavatories in the layout, the lavatory can very easily install when it is dragged over the appropriate installation zone. After doing this, the "Lavatories" database dialog will open and show all lavatories stored in the database:

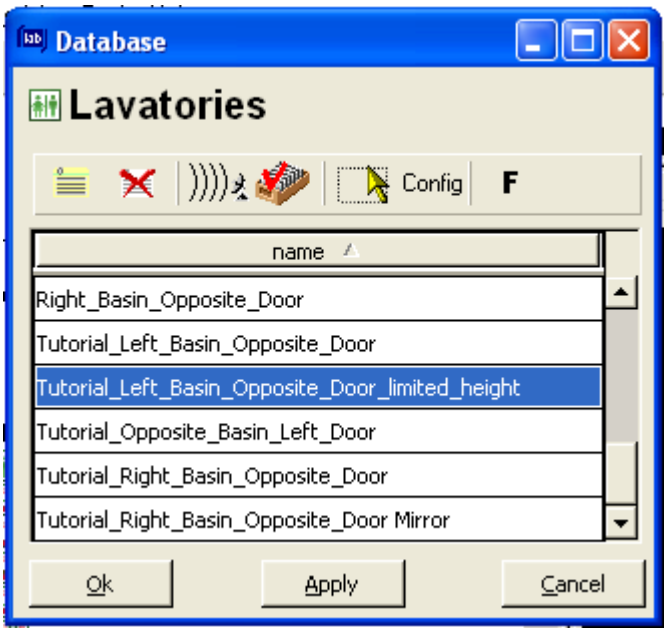


Fig. 4.19 The database for the lavatories

Selecting "Tutorial_Left_Basin_Opposite_Door_limited_height" Lavatory, and then clicking on Apply and after "Ok", the lavatory will be shown in the layout.

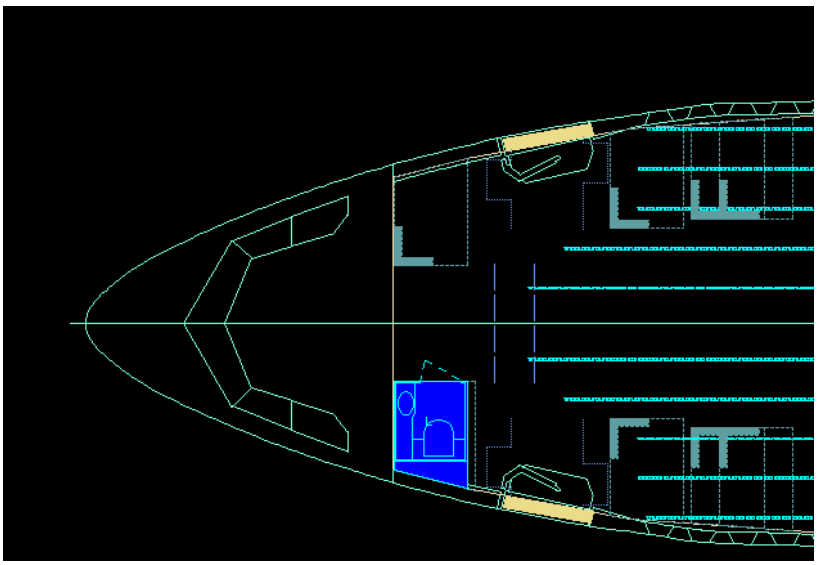


Fig. 4.20 The installed lavatory shown in the Layout

In the Pacelab tree, the new node "Tutorial_Left_Basin_Opposite_Door" has been added to the tree under the node 'Fuselage/Main Deck/Cabin Layout/Left Hand'. All items added to the layout are displayed in the Pacelab tree and in the Graphics pane. Depending on where they are installed, they are added to the node of the corresponding installation side (left, centre, right). In the same way, the other Lavatory at the Right Opposite Door can be added.

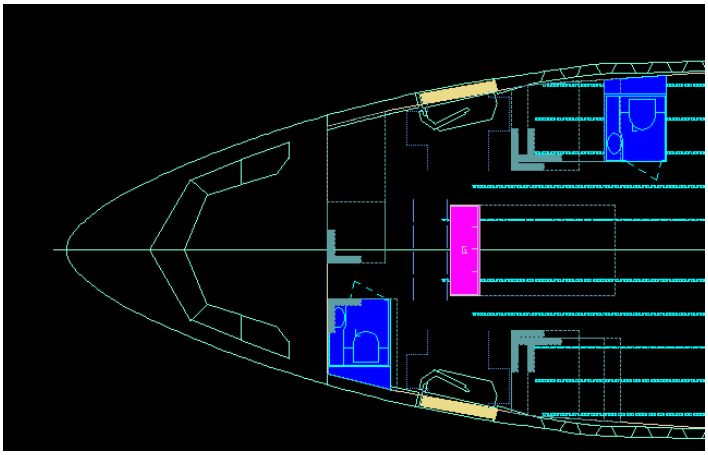


Fig. 4.21 Installing the first galley

After installing the galley in the centre zone, like below, the galley can be moved to the rear of the zone using the alignment feature, which helps in determining the precise position of the cabin item by snapping it to a point relative to another cabin item.

Furthermore, to align the galley to the rear of the installation zone, the user should: press and hold the ALT key, click the galley and hold the mouse button down, drag the mouse cursor to the installation zone and after one can see a red and white dotted line surrounding the galley first and then the installation zone to show that the installation zone is the target for the alignment. In the end, the mouse button can be release and the "Alignment" dialog will open.

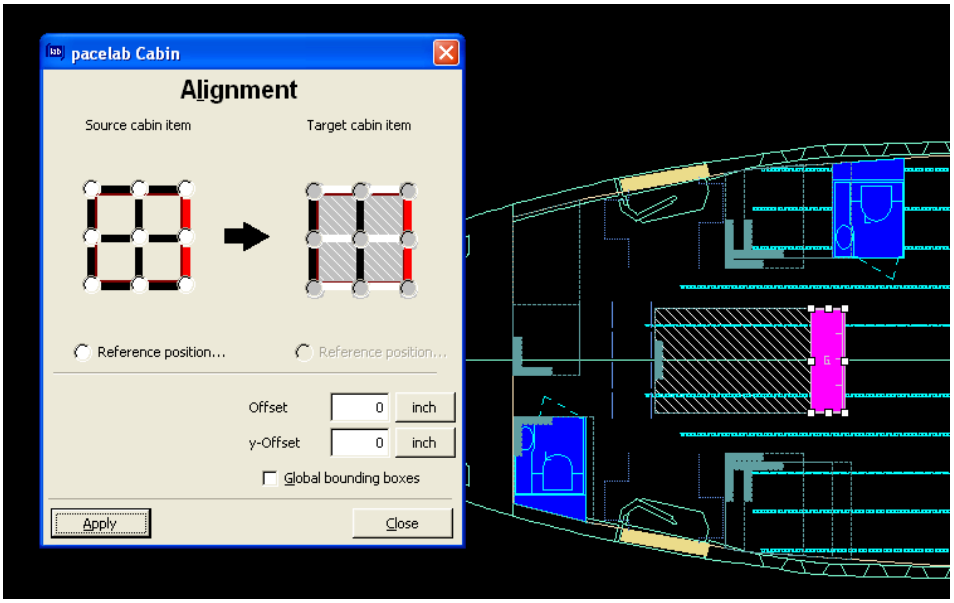


Fig. 4.22 The Alignment procedure

In the Alignment dialog, the grids reflect the orientation of the item in the layout. For example, the left vertical edge represents the front of the item; the top bar, the right side, etc.

The second galley, already installed in the figure below, can be mirrored, by right click on it and select from the list Mirror command.

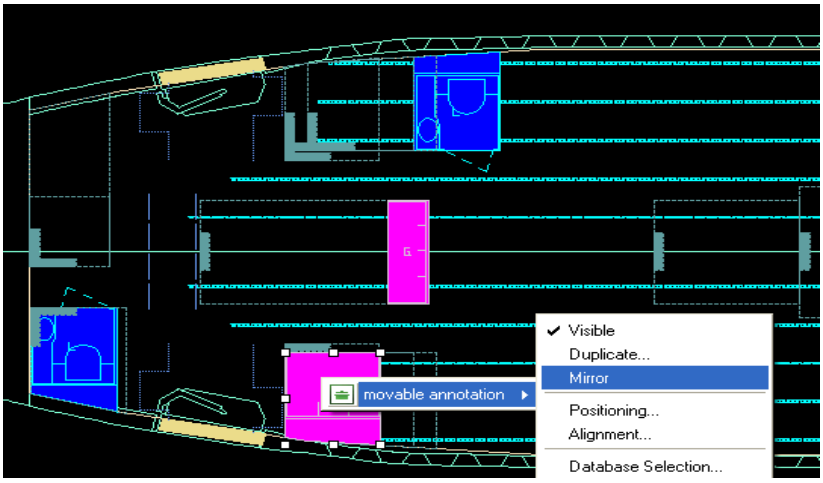


Fig. 4.23 Mirror command

After installing the Crew Rest Compartment at the front of the cabin, just behind the cockpit, there can be seen that the sidewall extension clashes with the assist space next to the door. To solve this problem, there is need to move the sidewall extension:

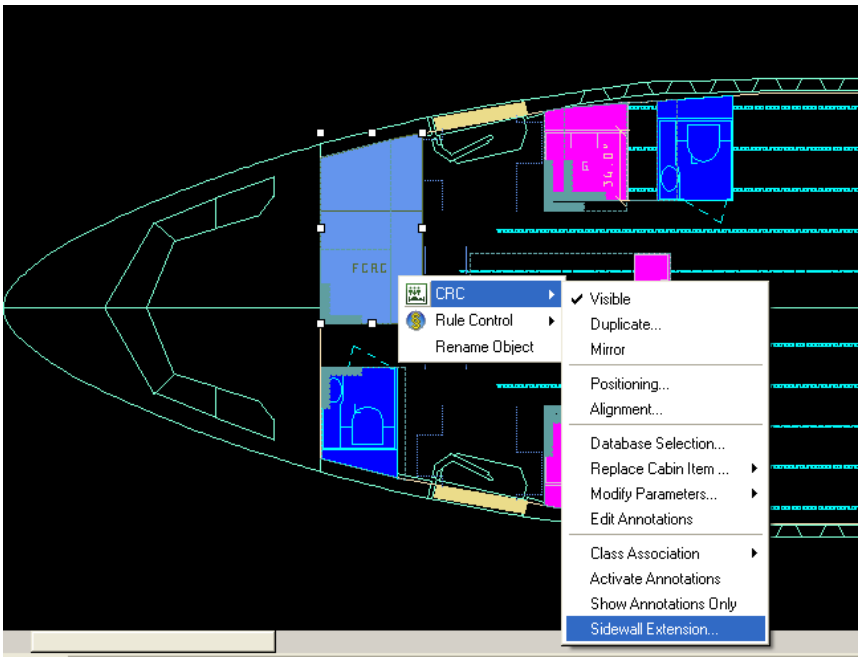


Fig. 4.24 Installing the Crew Rest Compartment

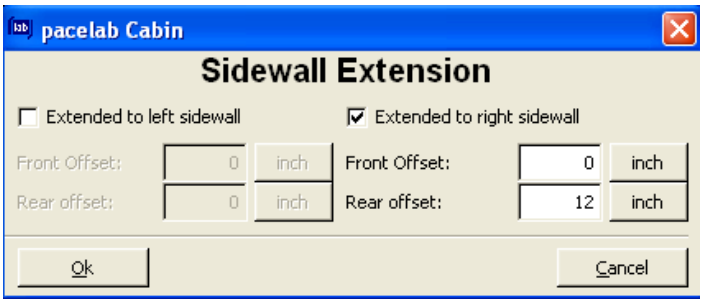


Fig. 4.25 The Sidewall Extension dialog

After doing this, the result can be seen on the layout:

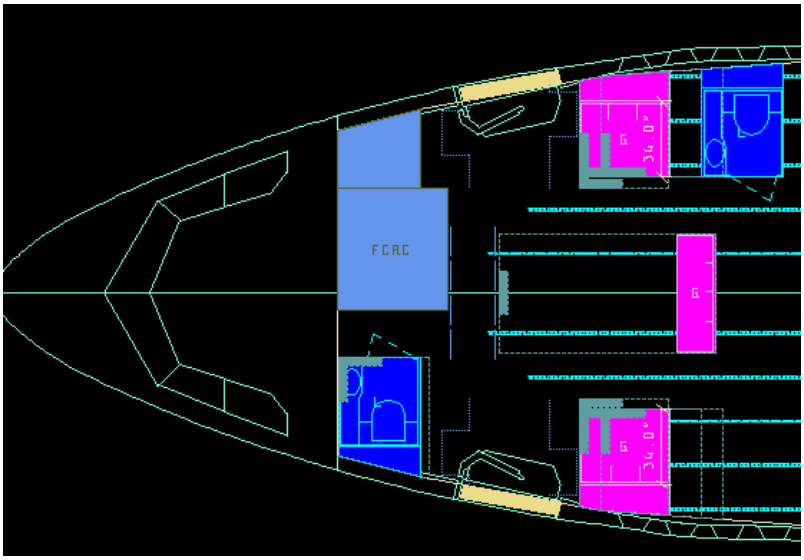


Fig. 4.26 The result after using Sidewall Extension

Afterwards, the sidewall extension is offset towards the front of the cabin so that it does not impede the assist space.

For installing the stowage in the centre zone, instead of using Alignment option, the "Move" icon will be activated to align the stowage to the front of the installation zone. By activating the move mode, a cabin item can be moved in a flexible installation zone in steps of whole inches with the mouse or with the arrow keys.

After installing the stowage on the left side, behind the galley, the red sign for rules violation will appear. Pacelab Cabin has a rules engine which constantly checks the layout for compliance with certification regulations or company standards. It takes into account all layout changes and updates the list of rule violations. This problem is fixed with the Alignment option. The reason is that the zone where the stowage was installed was a fixed one and in a fixed installation zone, the cabin item can't be moved inch-wise,

but only drag it from installation zone to installation zone with the move mode. Other positioning options such as “Alignment” permit locating a cabin item anywhere in the layout.

Fig. 4.27 Lifting up the rules violation

The next step for completing the First Class is to install PAX seats. The installation areas for seats in the cabin are defined by seat rails. First of all, before installing seats, the seat rails must be activated (by making right-click on the “Seat rails” node in the Pacelab tree and select “Seat Rail Container – Items active”).

The procedure is the same as at the other items (using drag & drop and, for a correct positioning, “Alignment” where the offset position from the stowage will be specified).

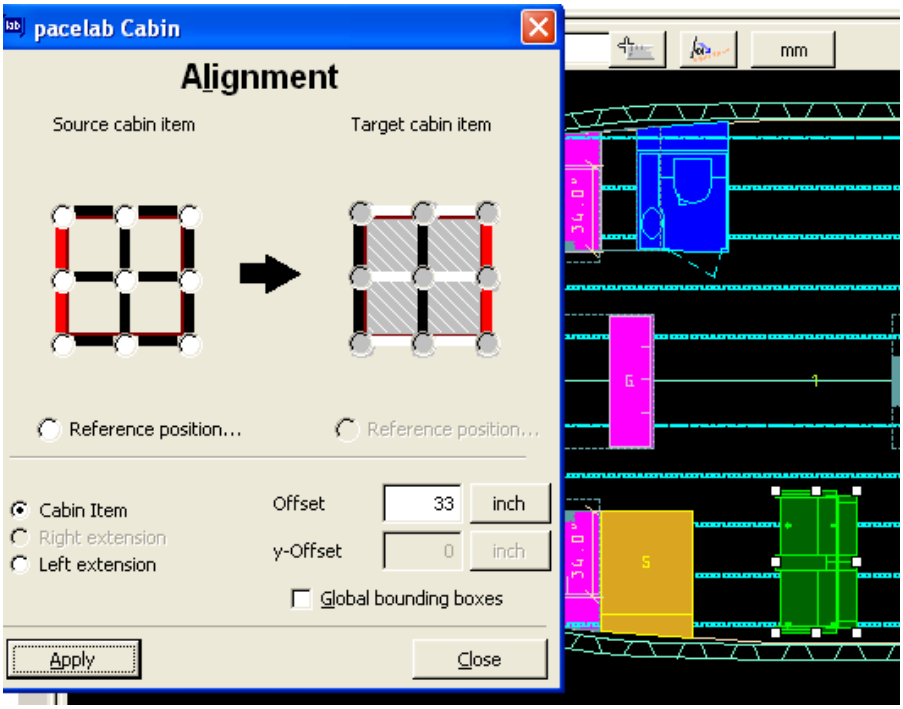


Fig. 4.28 The Alignment command

With this first seat, a block seat can be made by right-click on the seat and choosing "Build Seat Block".

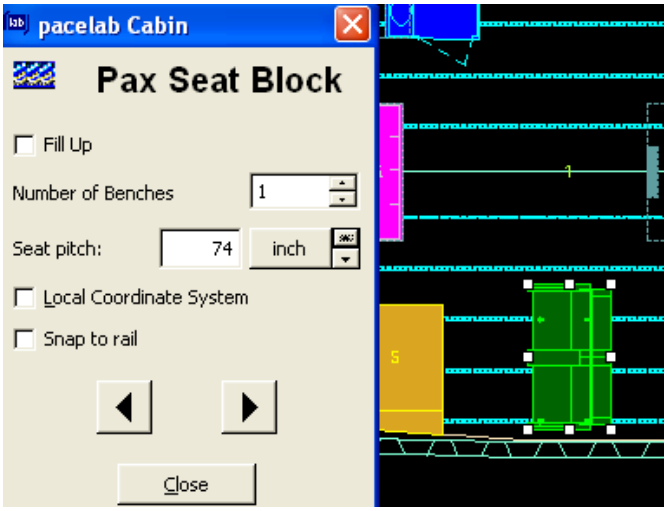


Fig. 4.29 Installing the Pax seat block

:

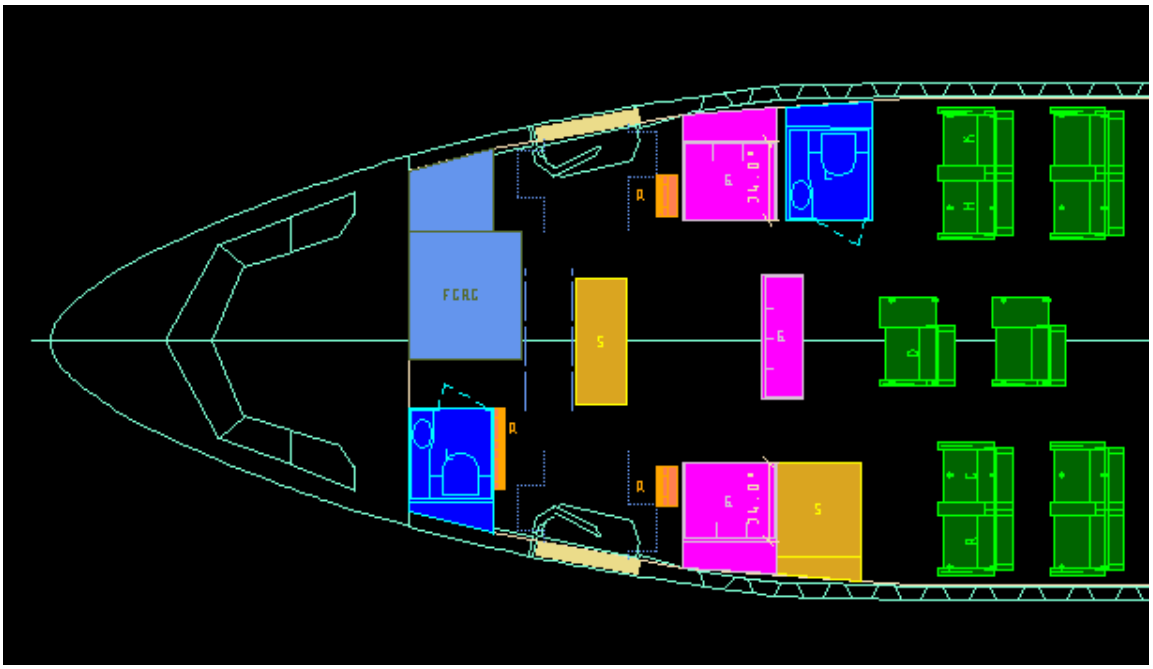


Fig. 4.30 The Cabin Attendant seats

As is shown in **Tutorial 2009** the type of the seat can be changed anytime by double-click on the seat. To complete the First Class, the Cabin Attendant Seats are remained. There are two types of cabin attendant seats:

- Floor-mounted – They need to be mounted either on seat rails or on hard points shown in the layout as installation zones for cabin attendant seats;
- Wall-mounted – These are attached to other items of cabin equipment.

The major difference is that a wall-mounted item needs to know which item it will be attached to and this means that you need to select an item in the cabin before dragging the item to the layout (one example can be the wall-mounted cabin attendant seat attached to the lavatory). After doing this, the "Attachment" dialog will open:

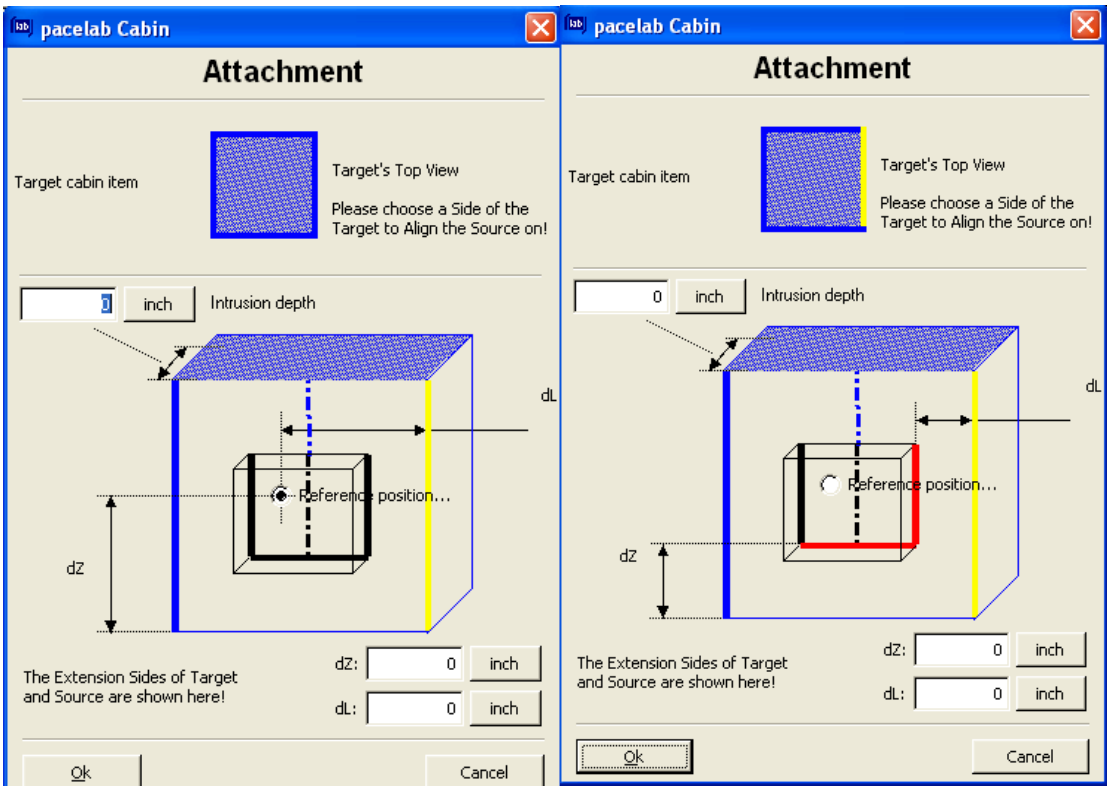


Fig. 4.31 The Attachment dialog

The top-section square represents the lavatory and is oriented the same way as in the cabin, e.g. its left side is the front side in the cabin. The attendant seat will be installed on the rear side of the lavatory. If the right side of the square is turned to yellow, that means that the attendant seat will be attached to this side.

However, the diagram in the middle section of the dialog represents the side you selected in the top section. This diagram allows you to determine the exact position of the attendant seat on the side of the lavatory. The target (lavatory) is represented by the blue framework while the source (attendant seat) is black (**Tutorial 2009)**.

There should be notice that, in the example above, for a correct positioning, the attendant seat is installed at the same level with the edge of the lavatory ("0" into the "dL" field to set the horizontal offset). Clicking the black line at the bottom of the source allows to set the height of the attendant seat from the floor "dZ" and typing "0" into the "dZ" field allows setting the height of the attendant seat, i.e. it is installed at floor level.

Also, the partitions are added with drag & drop too.

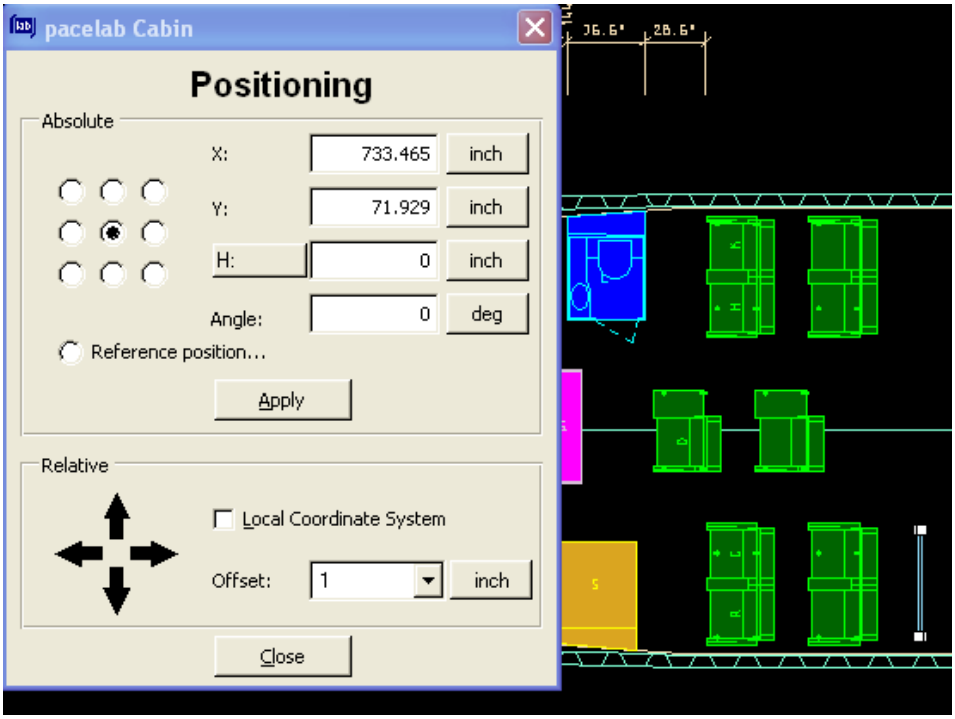
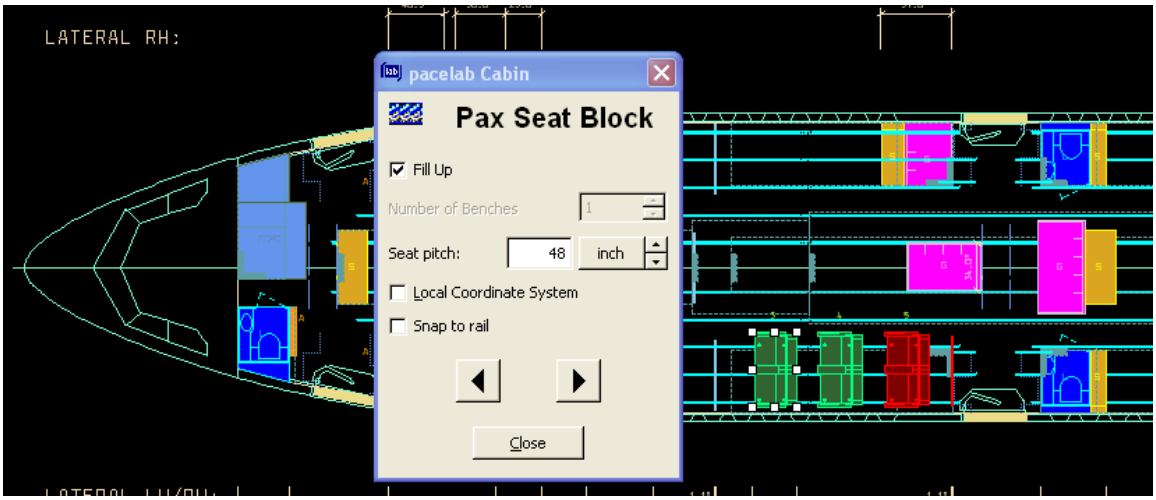


Fig. 4.32 The “Positioning” dialog

However, for a correct positioning, the “Positioning” menu will be used (by right-click on the partition). In the dialog, the right values for X and Y can be typed. Also, this option can be used for positioning (absolute, relative or changing the orientation) other items in the cabin.

4.2.3 The Business Class

In the Business Class, the monuments can be installed similarly to those in the First Class. When building “Block Seats” using the option “Fill up”, a rules violation can be seen.



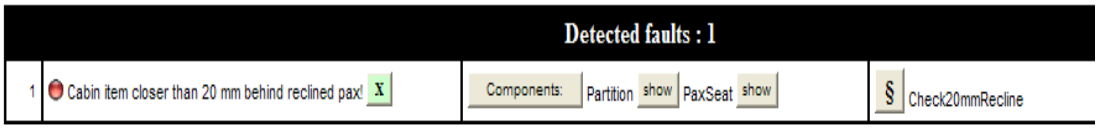


Fig. 4.33 The rules violation

There can be seen that the last seat and the partition zone are too close to each other. A way to solve this problem is to move the whole seat block forward. However, we already know that the front seat causes a head strike radius violation if it is moved forward. To avoid this, the head strike radius can be reduced by installing a different type of seatbelt on the front seat. By right-click the front seat in the block and selecting "Pax Seat – Modify Parameters – Project Parameters", the "Pax Seat" project parameters dialog will open and from there, selecting the "Mechanics" tab in the dialog. In this example, the "Inflatable Belt" type will be used.

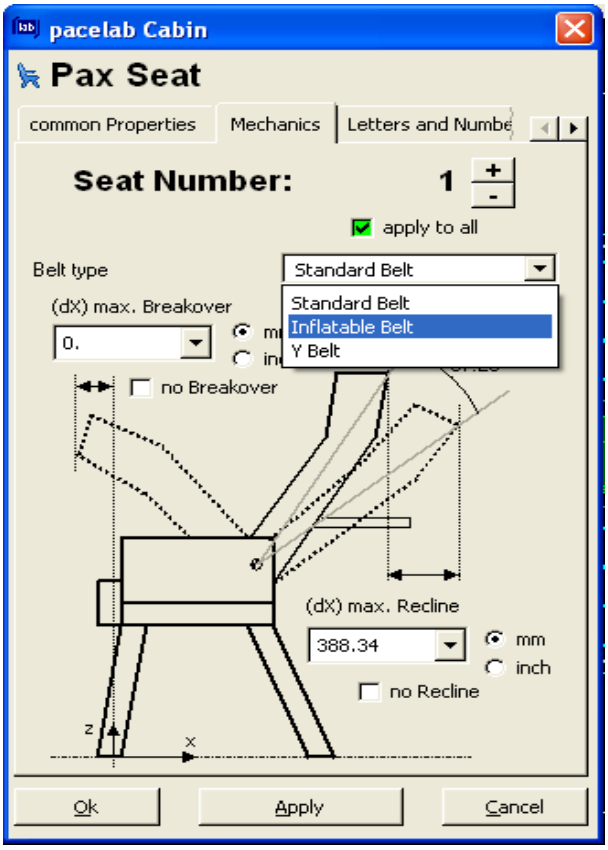


Fig. 4.34 Changing the seat belt

After this procedure is done, the all seats can be moved in the same time by selecting all and using the left arrow key until the rules violation is lifted.

Also, the seats behind the monuments can be installed similarly. Another option that can be used to remove the rules violation is the command ‘Snap to next rail position’. For example, the front seat in the centre is positioned at 26 inches from the stowage and the seat is not on a valid seat rail position, but, by selecting “Pax Seat – Snap to next rail position” from the context menu (right-click on the seat), the seat will jump to the next valid seat rail position :

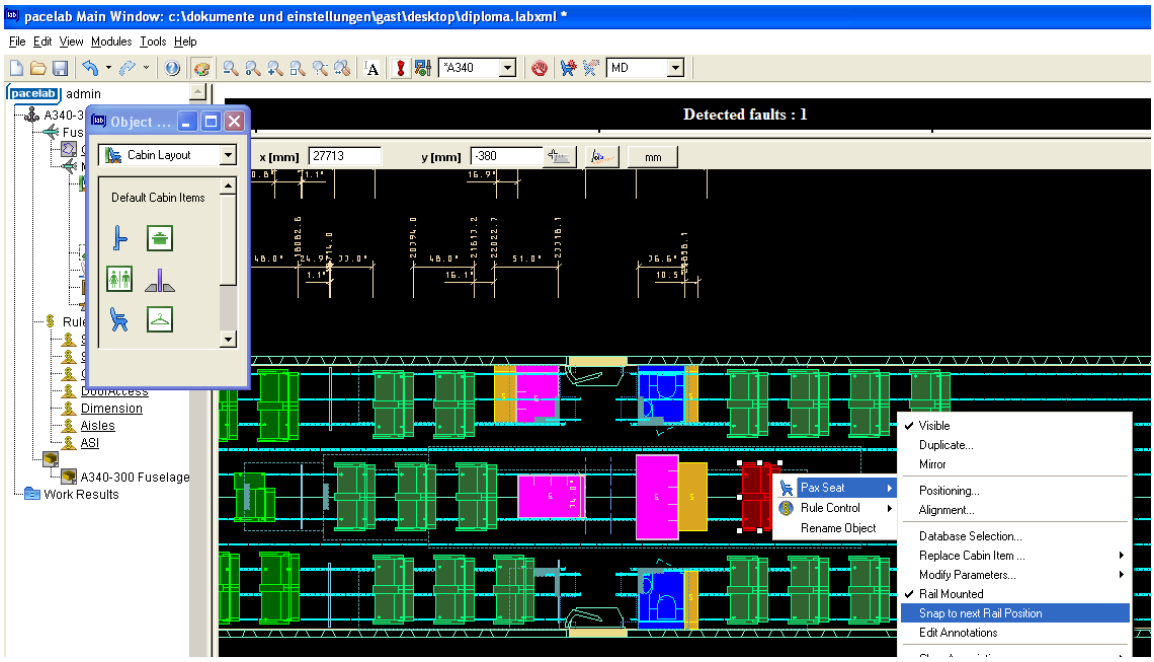


Fig. 4.35 Snap to next rail position

After building the seat rows, the Business Class can be considered done.

4.2.4 The Economy Class

The next Class to install is the Economy Class. The partitions between the Business Class and the Economy Class are needed. In the next picture, the way in which changes the database selection for the right class partition built by mirroring is presented:

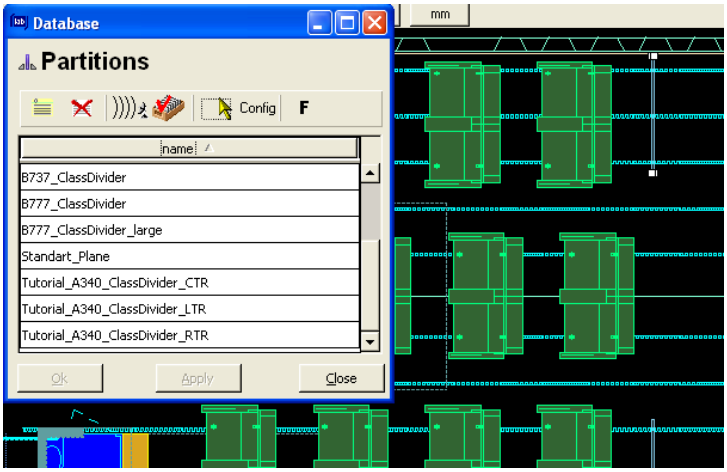


Fig. 4.36 Partitions

The shown dialog can be open by right-click on the partition and selecting “Database Selection”. This operation is necessary because, otherwise, the new partition on the right side, built by mirroring, would be considered as included in the “Tutorial_340_ClassDivider_RTR”. Furthermore, the class dividers can be included in Business Class or in the Economy Class and that depends on how close are to one class or another. The class association can be changed by right-clicking on the partitions, selecting “Class Association” and from there selecting the Class desired.

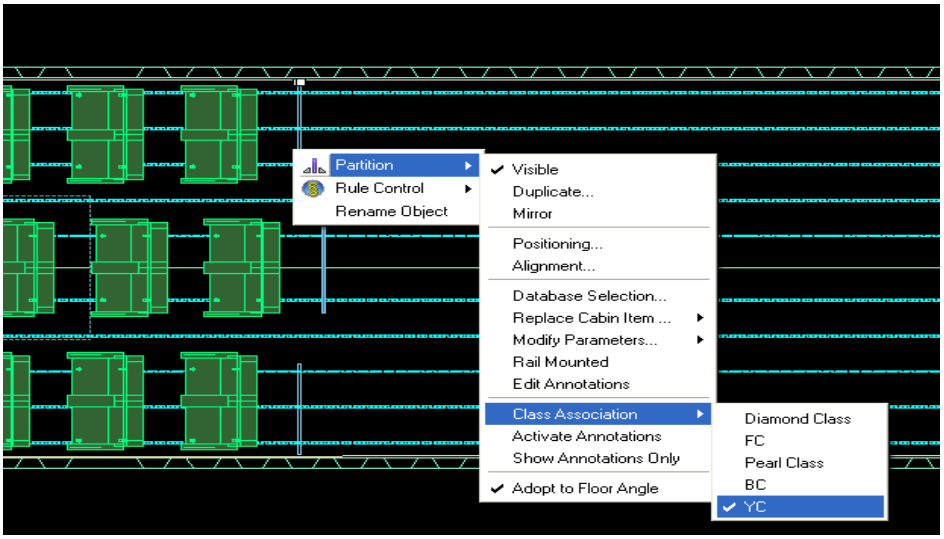


Fig. 4.37 Changing the class association

Making a step further, when installing the monuments in the Economy Class, and specifically, the galleys at the rear of the cabin, a rules violation can be seen:

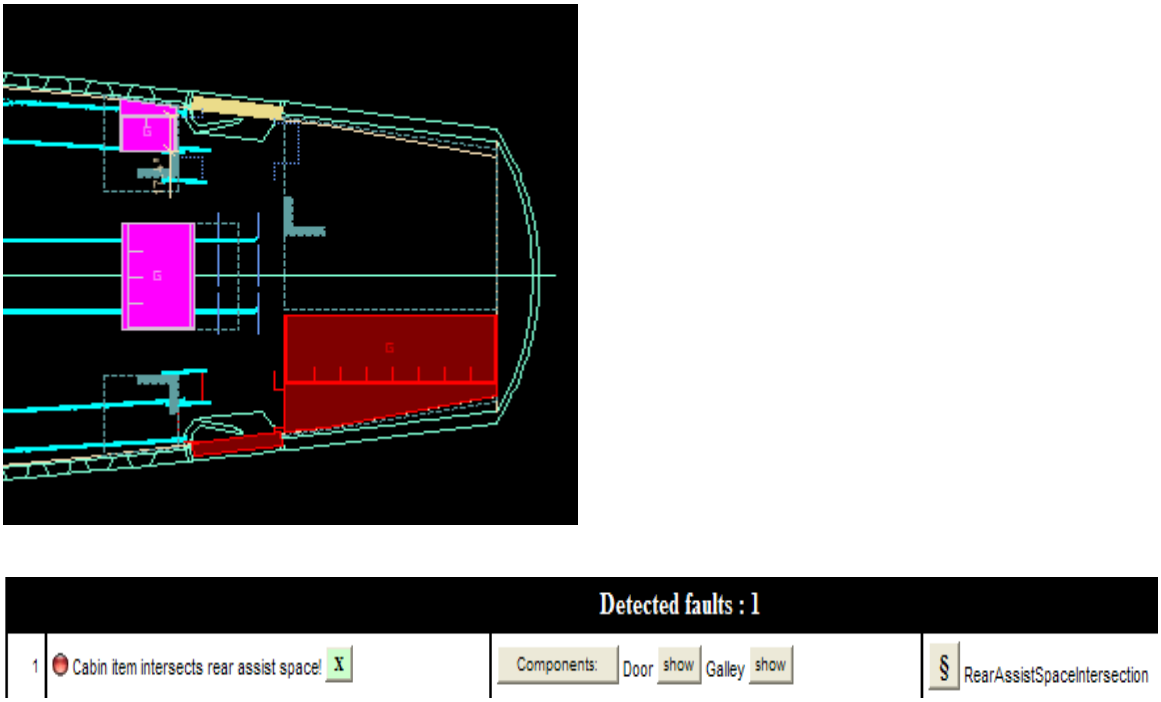


Fig. 4.38 A case of a Sidewall Extension necessity

This is because the sidewall extension clashes with the assist space next to the door and the solution is, like at the crew rest compartment in the first class, to define an offset for the sidewall extension by checking the "Extended to left sidewall" box and typing "8" (inches) in the "Front Offset" box.

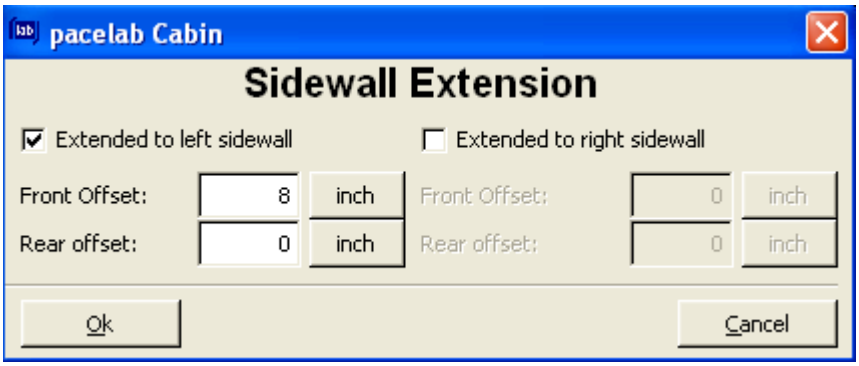


Fig. 4.39 The Sidewall Extension dialog

After installing the monuments, the seats should be installed in the Economy Class. Because in this Class it is very important to have as many seats as possible for maximum of efficiency, the seats will be installed for the first in their final position using the "Positioning" dialog. For this, a reference part of the seat will be used and this can be the front left or front right leg. Alternatively, the "Reference Position" radio button in the "Positioning" dialog can be used.

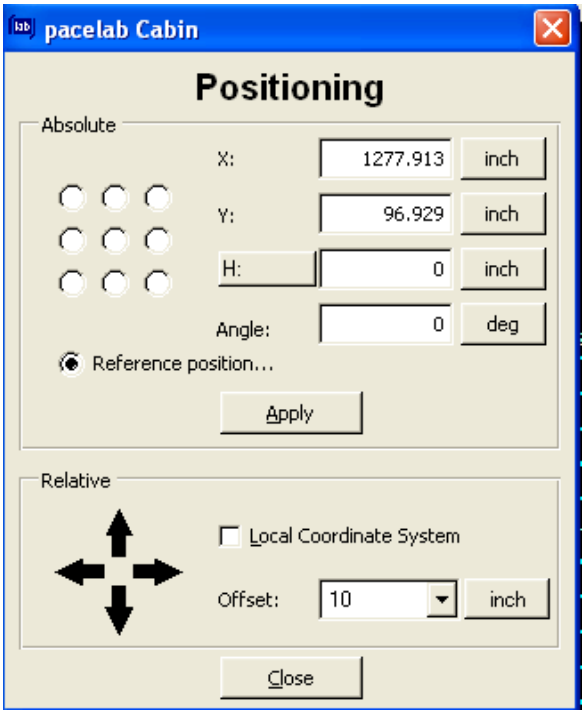


Fig. 4.40 Using the “Reference Position”

When installing the front seat on the centre/left row using the correct coordinates given by the Tutorial for Pacelab Cabin, a rules violation can be seen :

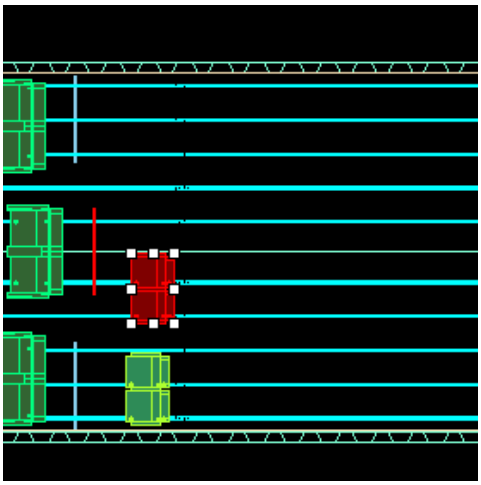


Fig. 4.41 Rules violation on the centre/left row

A way to lift up this rules violation is to replace the seat belts (with inflatable belts) as to reduce the head strike radius, but first, the seat block should be built. Otherwise, all the seats in the block would have the seat belts replaced.

In addition, before mirroring the seat blocks built on the left, the front seats should be modified some more. Moreover, before the front seats are behind a partition, they do not have a folding table in front of them and they need an in-armrest table. Double-clicking on the seats, a dialog will open and, in the “Armrest” menu, some parameters should be modified as following:

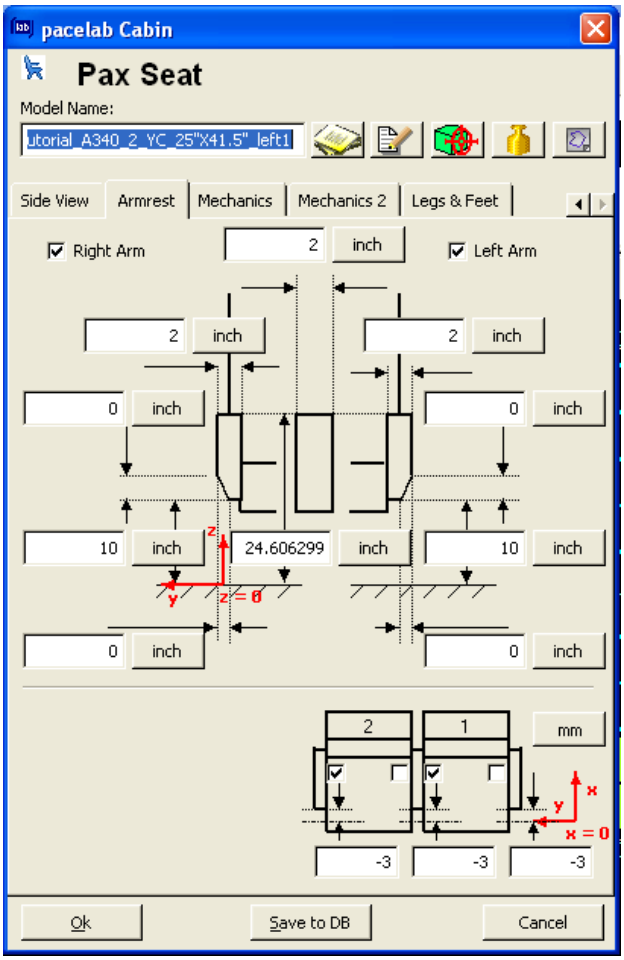


Fig. 4.42 Adding an in-armrest table

For adding an in-armrest table to the checked armrests, the two boxes next to the right armrests must be checked and for increasing the length of the armrests, a number (for example, -3 inches) must be typed into each of the lower part three fields. After doing this, the “Model Name” field should be modified because the parameters of the item have been modified. As can be seen, one can add a new type of seat to the Database by clicking on “Save to DB”. The only difference between the left front seat and the centre/left front seat is that there are different boxes checked. The above picture refers to the left front seat and the below pictures refers to the centre/left front seat :

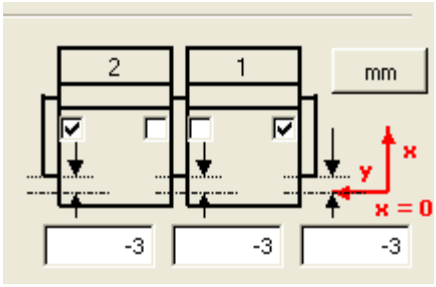


Fig. 4.43 Increasing the length of the armrests

The modifications can be seen in the layout:

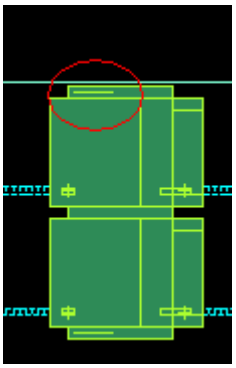


Fig. 4.44 The results after using the “Armrest” dialog

Making a step farther, the seats after the third doors must follow the non-constant section. For this, to install the left block seat, the “Local Coordinate System” and “Snap to rail” fields must be checked.

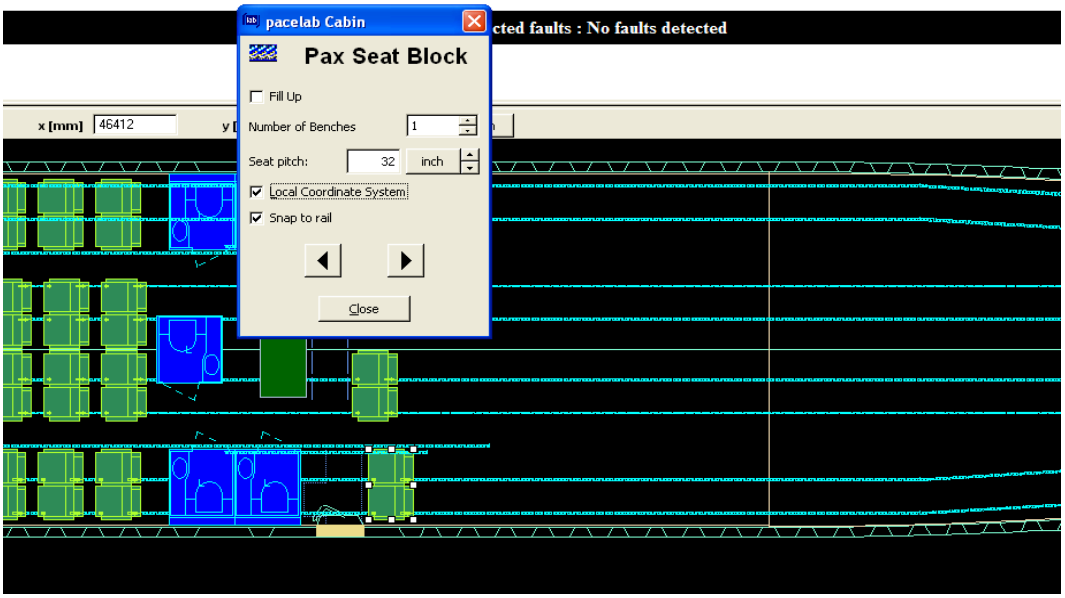


Fig. 4.45 Checking the “Local Coordinate System” and “Snap to tail” boxes

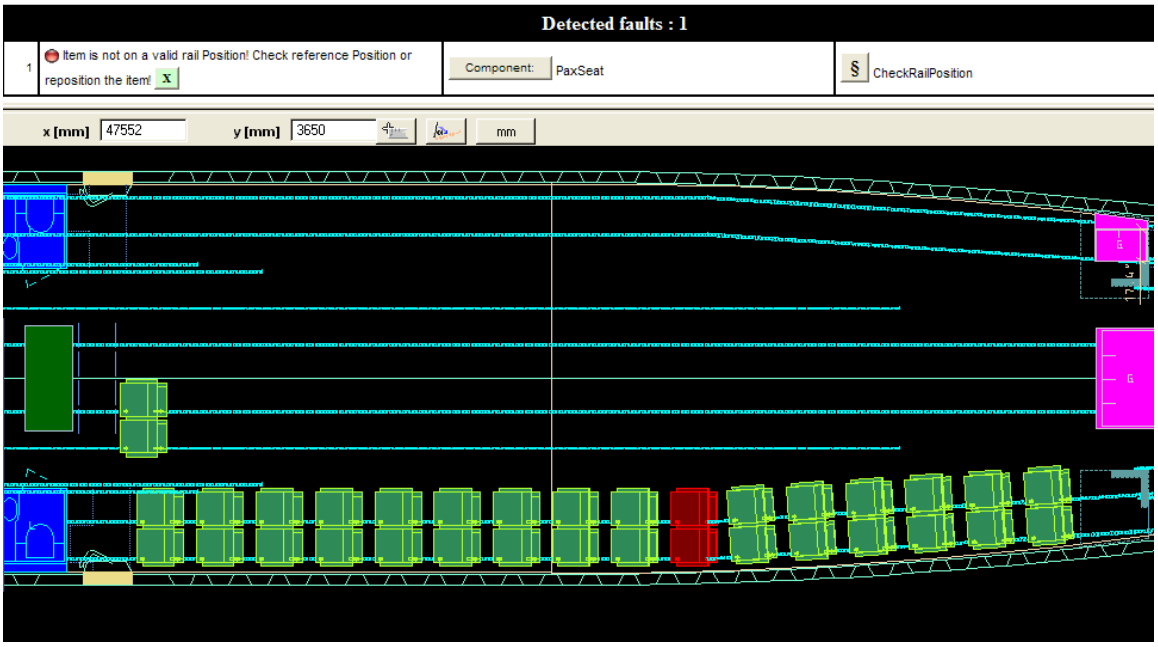
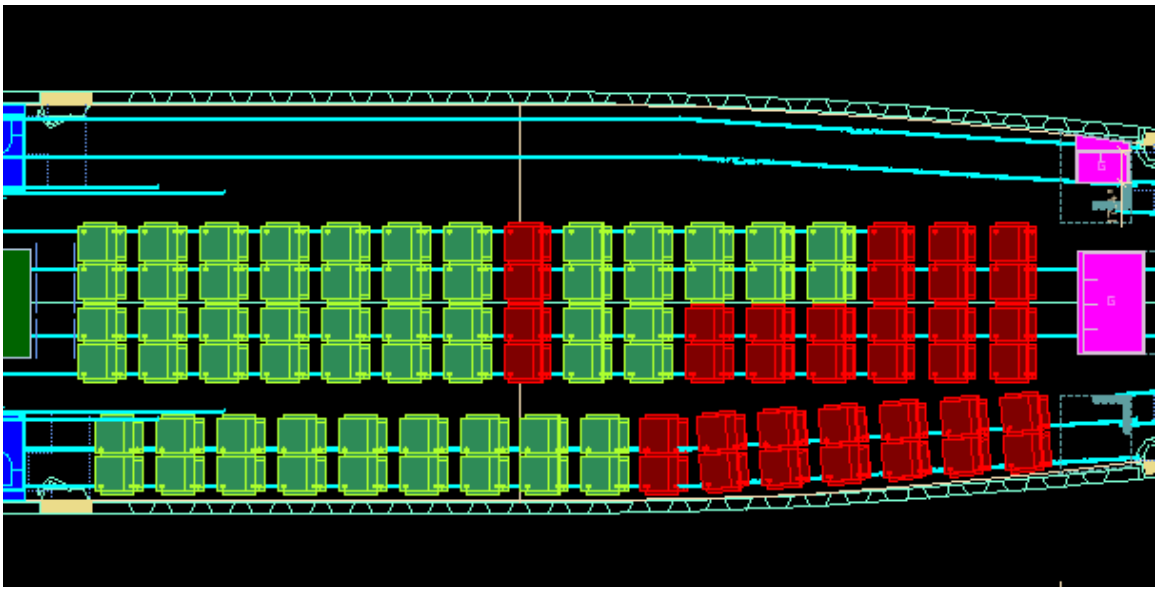


Fig. 4.46 A Seat Block which follows a non-constant section

The seats follow the seat rails as they move from the constant to the non-constant section of the cabin. Checking the "Snap to rail" and "Local Coordinate System" caused the seat to be rotated as it is installed (the legs are fitted to the position of the rails, forcing the seat to rotate) and checking the "Local Coordinate System" determined the position of the next seat. If global (aircraft) coordinates are used, all seats would be installed in a horizontal line (constant Y) and using the local coordinate system installs the next seat directly behind and parallel to the previous one.

As there can be seen, there is a rules violation, which will be solved after the installation of the other seats. But what can be done is to make a small correction for increasing the passenger comfort and this correction means to increase the distance between the rotated seats. By selecting all the oriented seats, opening the "Positioning" dialog and by checking the "Local Coordinate System", the seats are moved along their body axes. After building the centre/left row and making mirror, the Economy Class will look like this:



Detected faults : 37			
1	Longitudinal aisle blocked by cabin items! X i	Components: PaxSeat show PaxSeat show	§ CheckLongitudinalAisle
2	Longitudinal aisle blocked by cabin items! X i	Components: PaxSeat show PaxSeat show	§ CheckLongitudinalAisle
3	Longitudinal aisle blocked by cabin items! X i	Components: PaxSeat show PaxSeat show	§ CheckLongitudinalAisle
4	Longitudinal aisle blocked by cabin items! X i	Components: PaxSeat show PaxSeat show	§ CheckLongitudinalAisle
5	Longitudinal aisle blocked by cabin items! X i	Components: PaxSeat show PaxSeat show	§ CheckLongitudinalAisle
6	Longitudinal aisle blocked by cabin items! X i	Components: PaxSeat show PaxSeat show	§ CheckLongitudinalAisle

Fig. 4.47 Rules violations on the Economy Class

At this stage, there are quite a few red items in the cabin, each of them represents one or more rule violation and all of the rule violations are listed in the Browser pane. A good way to manage with the rule violation is to reduce the list to manageable proportions.

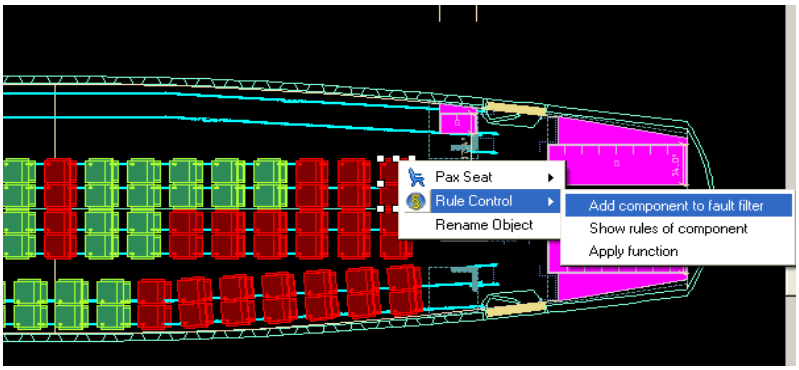


Fig. 4.48 A way to manage the rules violation list

In the end, the solution for the resulting rules violation is to change the seatbelt type (there is a violation in the head strike radius), to change the number of benches in the

centre row and, for the Rear Seat Studs, the solution is to change the orientation of the legs (represented by the number in the red circle) like in the picture:

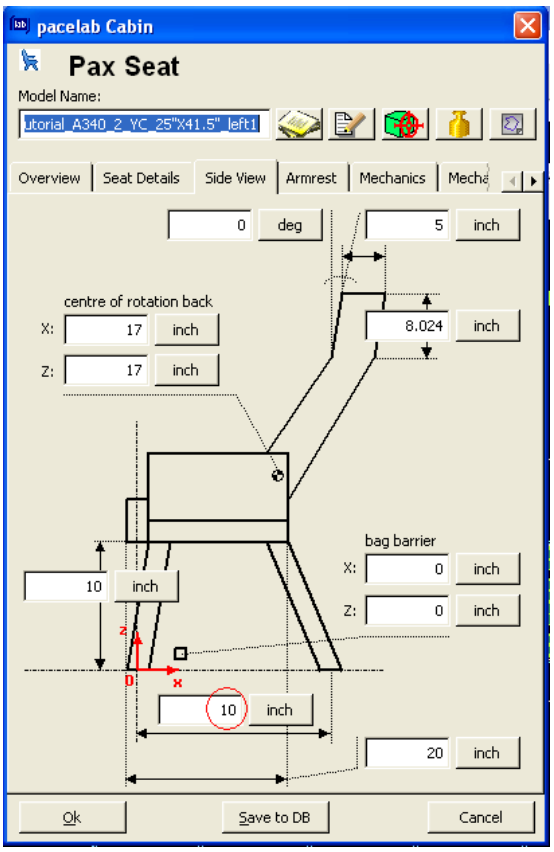


Fig. 4.49 Lifting the rules violation for the Rear Seat Studs

4.2.5 Results for Layout

After lifting up all the rules violation and building the right block in the Economy Class, the results can be seen by going at the Pacelab tree and selecting “Work Results”:

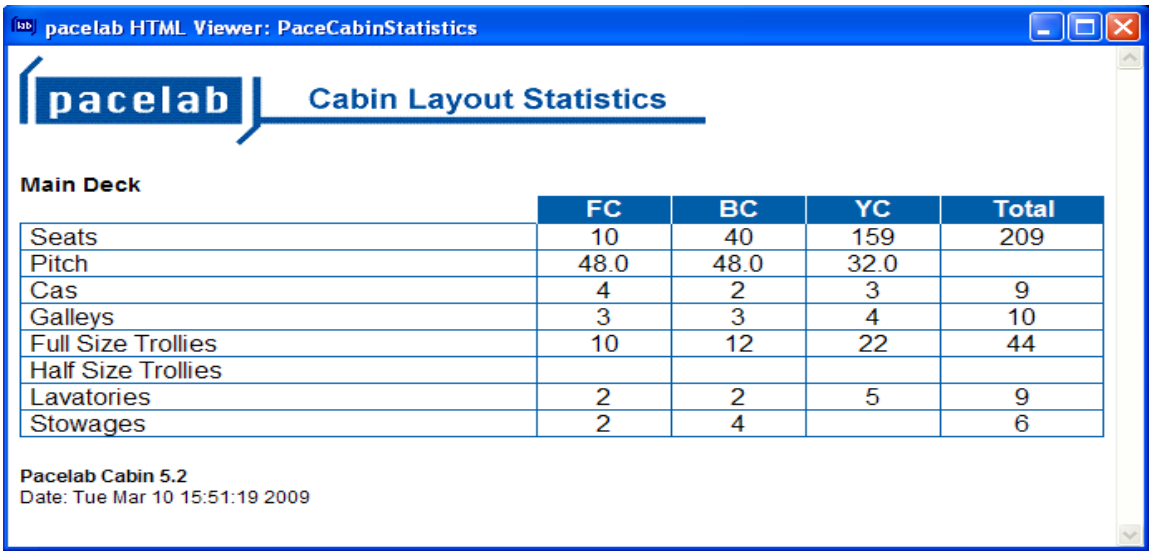


Fig. 4.50 Work results

At the end, the Cabin Layout should look like in the following picture in 2D:

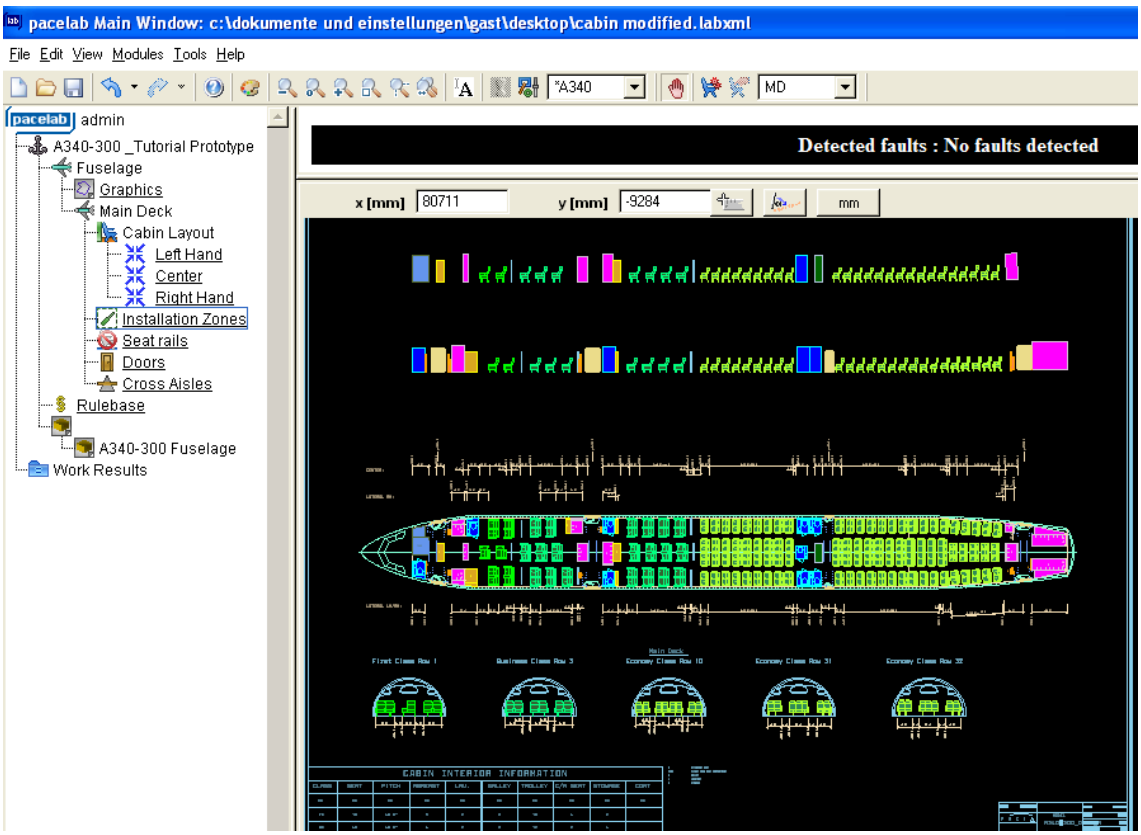


Fig. 4.51 The Cabin Layout

Also, the cabin can be seen in 3D using 3D Viewer:

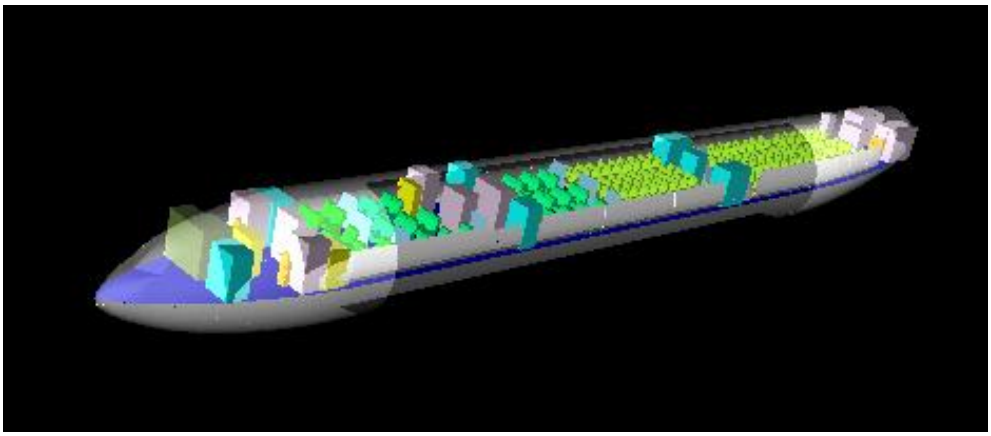


Fig. 4.52 3D Viewer

However, Pacelab Cabin provides not only sophisticated 2D and 3D impressions of the future cabin, but also a very good Rules Engine which uses some rules from the Certification Specifications.

4.3 The Rules Engine

4.3.1 Rules Definition

When designing the cabin layout, certification specifications are implemented in the design. Such rules are for example the positioning of the monuments, the distribution of emergency exits, or the required emergency equipment (**EASA 2009**).

Also, Rules describe knowledge in the form of When/Then statements. They combine a condition with an action which is to take place if the condition is met. Also, the conditions part states a premise which is typically the violation of a certain constraint, e.g. a certification regulation or a company design standard. The actions part says that this violation is to be reported in a certain error message format (**Tutorial 2009**).

Moreover, rules consist of rule properties and the rule code. Rule properties are general information about the rule and its behaviour, e.g. full text, update information, status (active/inactive), etc. The rule code consists of two parts: a Conditions part and an Actions part (**Tutorial 2009**).

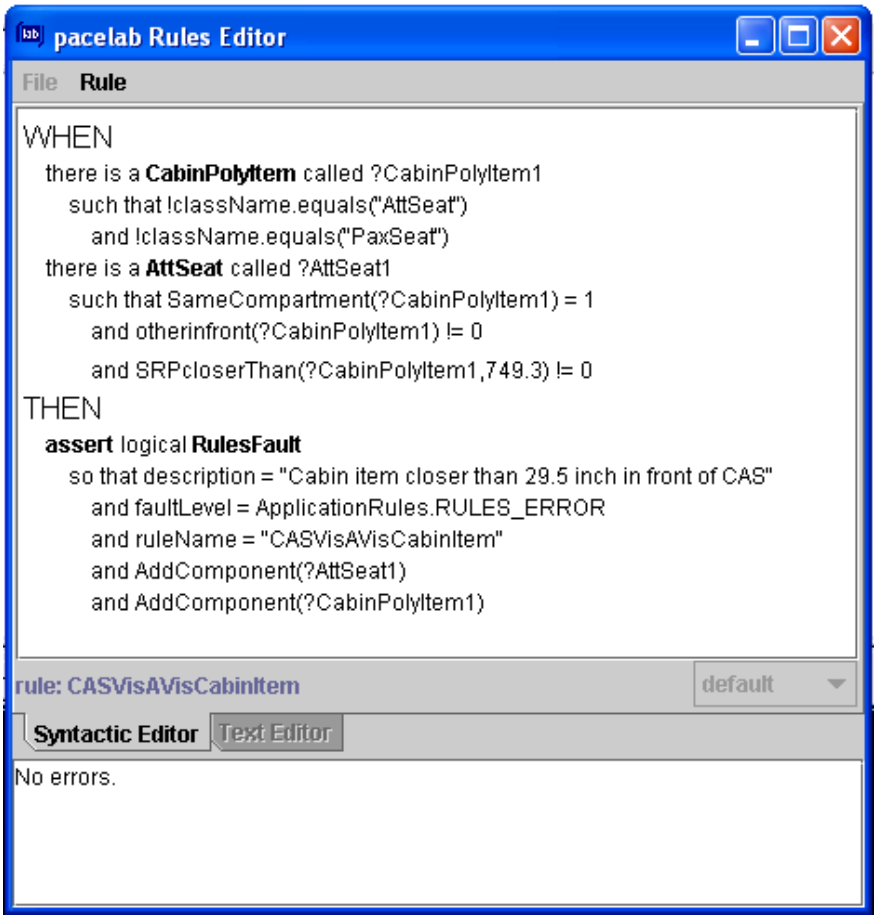


Fig. 4.53 The rule code

According to **Tutorial 2009**, the Conditions part contains:

- Layout objects (classes) affected by the rule;
- Slots which provide elementary information on the objects concerned to define them more accurately in the context of this rule, e.g. dimensions, location, etc.;
- Methods which describe the relation of the object to another object, e.g. “intersects” (“otherobject”).

In addition, when the rule is implemented and active, the Rule Engine will go through all objects contained in the Conditions part of any of the rules and check them for violations of constraints according to the methods specified in the rule and when violations are found, the Actions part comes into play (**Tutorial 2009**).

In addition, the Actions part states that an error message is logically asserted due to the violation and describes what information about the error is to be reported to the user:

- Error message to be shown to the user;
- type of error (fault or warning);

- name of the rule;
- Layout objects (classes) affected by the rule.

However, for a better insight in the Rules Engine, some essential techniques for working on the Knowledge DB are required.

4.3.2 Essential Techniques for Working on the Knowledge Database

The Knowledge Database contains all rules ever created in the organization. However, rules cannot be loaded individually but only as part of a fixed data structure. The following hierarchy applies within the Knowledge Database: (Predefined) Rulebase/Current Rulebase – Rule collection – Rule folder – Rule.

Predefined Rulebase is a complete and ready-made set of rule collections. There can be added that the Predefined Rules can be considered as Default Rules. Predefined Rulebases can only be loaded as a whole. For a user without a Knowledge Organizer license, the Predefined Rulebase is always identical with one of the Current Rulebases which is defined by comprising all rules, folders and collections currently loaded.. A person with a Knowledge Organizer license may load and create collections independently of any Predefined Rulebases, even though the two may coincide here.

Making a step further in the hierarchy, the Rule Collection contains a set of rule folders. Collections represent groups of rules that are linked together due to their content. Rule collections are the smallest unit that can be downloaded from the Knowledge Database and the Rule Folder contains a set of rules and the Rule itself is the smallest unit, consisting of rule code and rule properties (**Tutorial 2009**).

This hierarchy is also reflected in the Pacelab tree where the rule base currently loaded is shown.

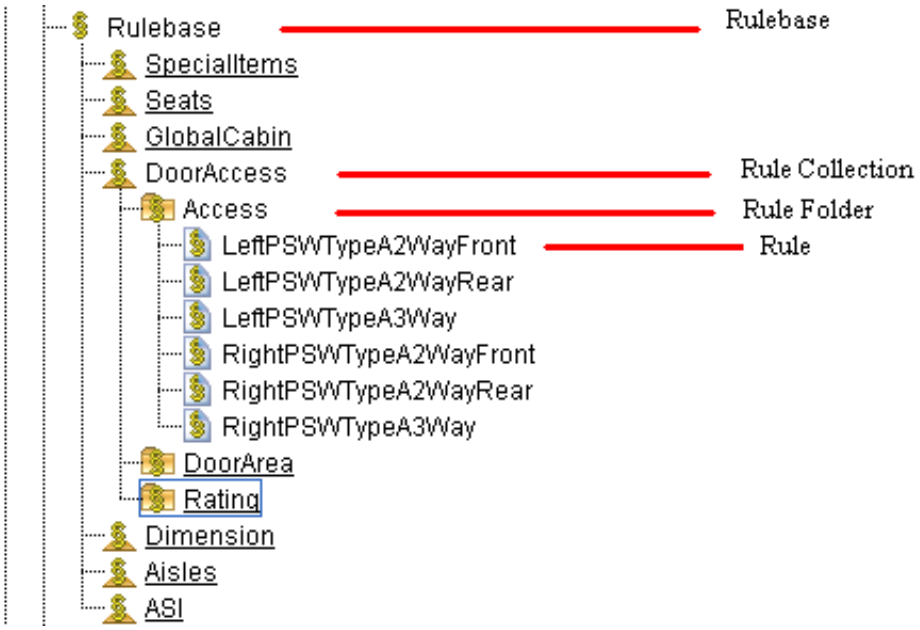


Fig. 4.54 The hierarchy

For operating on the Knowledge Database, there is always need to:

- connect to the Knowledge Database in the right mode;
- load a rule base or collections you want to work on into the current rule base;
- lock the collections you want to work on for editing.

If the user wants to search the complete Knowledge Database, he has to load all available rule collections into the current Rulebase first or – if he wants to search a specific Predefined Rulebase only, he has to load this one (in the following examples, only the Predefined Rulebase will be presented). After, he must right-click the “Rulebase” folder and select “Rule Tree/Search rules” from the context menu and the “Search Rules” dialog will open:

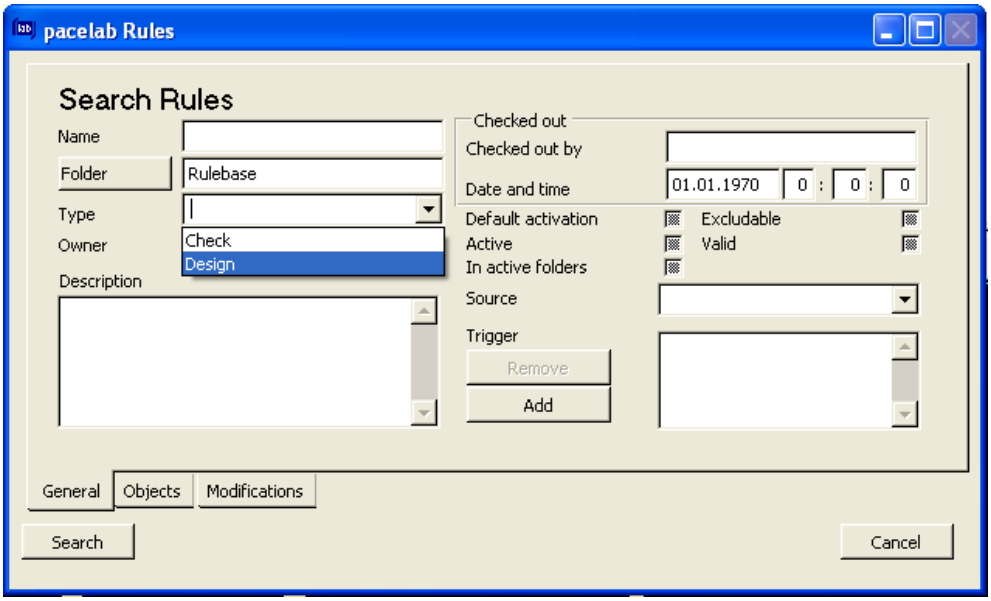


Fig. 4.55 The “General” tab

If, for example, there will be chosen “Design” at the Owner, after clicking “Search”, all matching rules will be shown in the upper right corner of the browser pane. The Fig. 4.56 shows that there are 89 rules for the “Design”.

Number of matching rules: 89	
1	§ ASI-Check20mm
2	§ ASI-Check20mmRecline
3	§ ASI-CheckHeadStrikeRadius
4	§ ASI-CrossAislesIntersection afterDeformation
5	§ ASI-CrossAislesIntersection maxBreakover
6	§ ASI-CrossAislesIntersection maxRecline
7	§ ASI-DoorAreaIntersection afterDeformation
8	§ ASI-DoorAreaIntersection maxBreakover
9	§ ASI-DoorAreaIntersection maxRecline
10	§ ASI-PAXSeat
11	§ ASI-Setback
12	§ Baby Bassinet

Fig. 4.56 Matching rules

There can be noticed that the "Search Rules" dialog consists of three tabs to organize search criteria in subcategories. They correspond to the tabs of the "Rule Info" and the "Edit Rule Attributes" dialogs:

- "General" tab;
- "Objects" tab;
- "Modifications" tab.

However, some of the attributes are user-editable while some are generated by the program and cannot be edited. For example, "Trigger" option limits the search to rules in task folders with a specific trigger and the user can edit the list of triggers by using the "Remove" and "Add" button on the left of the list of triggers or the "Excludable" option must be checked when searching for rules that can be deactivated (**Tutorial 2009**).

In the "Objects" tab, the user can specify a pattern of the rule code by selecting objects contained in the Conditions part (premise) and in the Actions part of the rule. The respective object class does not have to be contained explicitly in the rule code. It is sufficient if it is subsumed under an umbrella class (**Tutorial 2009**).

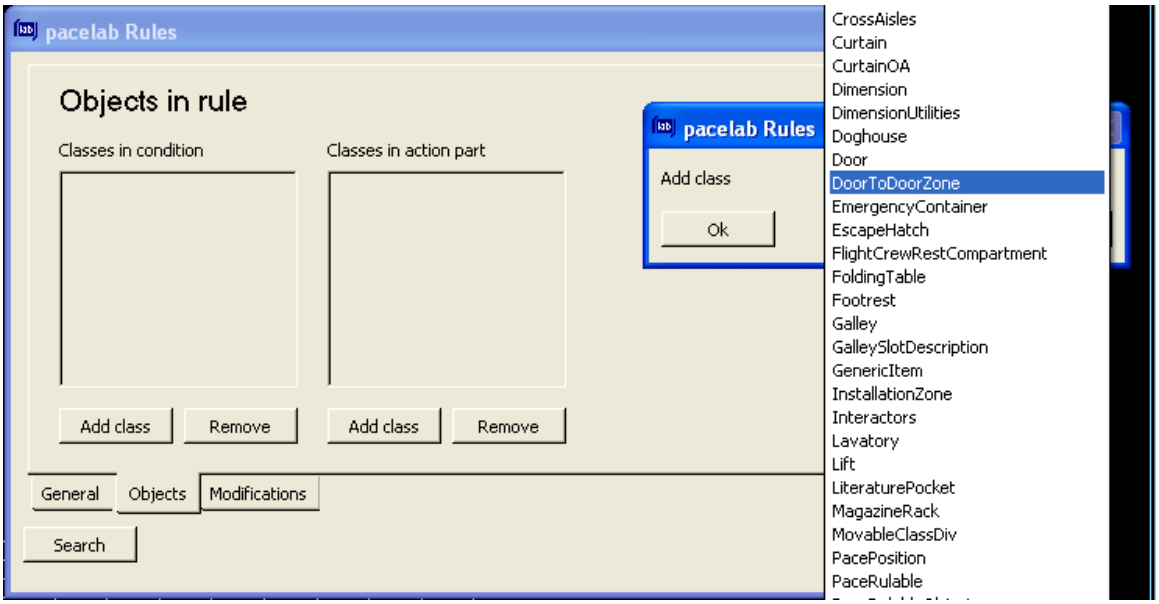


Fig. 4.57 Adding classes

Also,

the "Modification" tab enables the user to search for rules according to the critera:

- Last Modification: a substring of the ID of the author of the last modification;
- Last Modification past: a date and time after which a rule has been modified (**Tutorial 2009**)

Regarding the connection to the Knowledge Database, there are three connection modes: “Not connected” (not available and/or not used which means that the user cannot load new rule collections from the DB and none of the editing made in the local copy can be stored in the Knowledge Database), “Read only” and “Read/Write” (Data can be read and written from/to DB). Loading requires at least a “Read only” connection to the Knowledge Database. If the database is write-protected, the user is not able to select “Read/Write”.

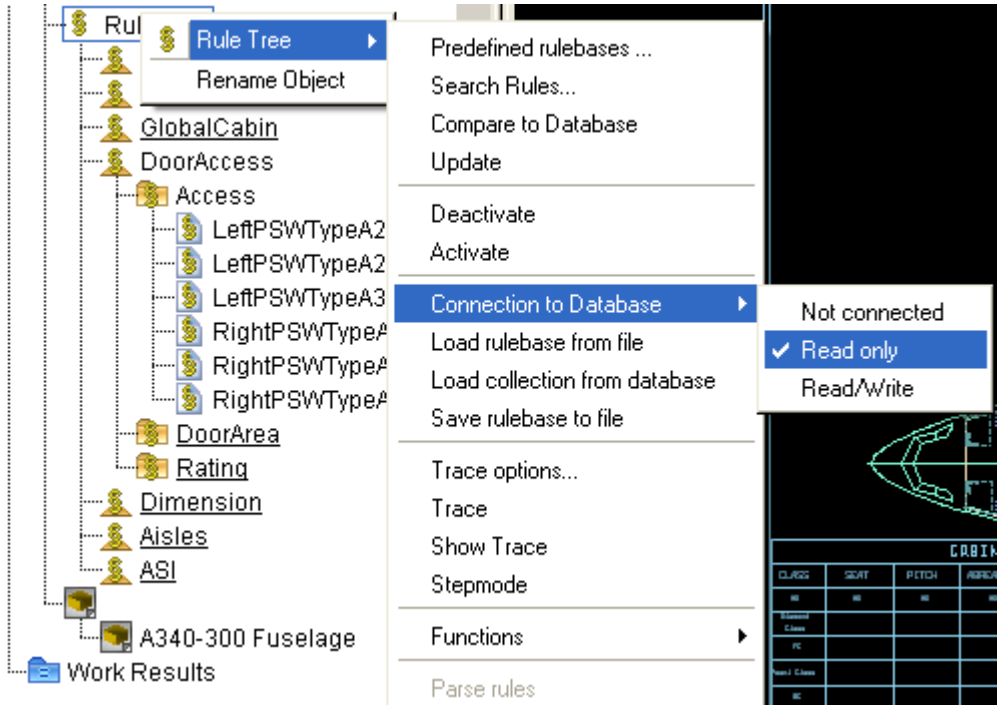


Fig. 4.58 Connecting to Database

There should be added that the Default (the Predefined Rulebases) are loaded automatically when the program is started and when a project is loaded, the Predefined Rulebase last saved with the project is loaded. Also, the current rule base is a local copy of rule collections loaded from the DB and the user can load rule collections from the Knowledge Database into the current rule base at any time and, also, he may remove collections from the current rule base (and this is not the same as deleting them from the Knowledge Database) (**Tutorial 2009**).

Loading a collection from database is a very easy process. It can be done by right-click the “Rulebase” folder and choose from there “Rule Tree/Load collection from database”. In the following example, the “Dimension” Collection will be loaded and it will be added

in the Rules Tree. At any time, it can be removed from the Rules Tree (by right-click on it), but this does not mean that it will be deleted from the Database.

What is interesting to notice is that the rule base can be saved (in a similar process as loading) without the rest of the project and this enables the user to exchange rule bases between users or edit layout using several rule bases (**Tutorial 2009**).

Another important step is locking Rule Collections before editing and saving to the Knowledge Database.

> *The locking mechanism ensures that only one user at a time can edit a collection and the rules it contains and only when the changes have been saved to the database, the data is unlocked and other users are able to edit it* (**Tutorial 2009**).

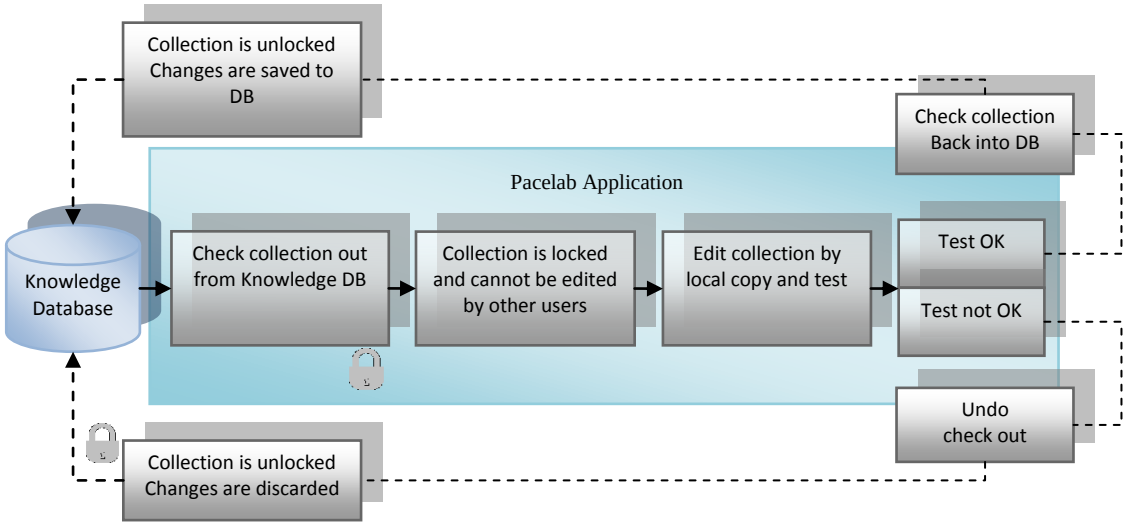


Fig. 4.59 Working with Collections

For locking a collection or some folders and rules contained in the collection, the user must ensure "Read/Write" access to the database, right-click the collection in the rule tree and select "Rule Tree/Check out collection from database" from the context menu. This is the why in which the system checks if the collection and the objects in it are unlocked in the database and, if it is so, it locks the object in the database and shows the locked collection icon (open box) as an indicator that the user is not allowed to edit the collection (**Tutorial 2009**).

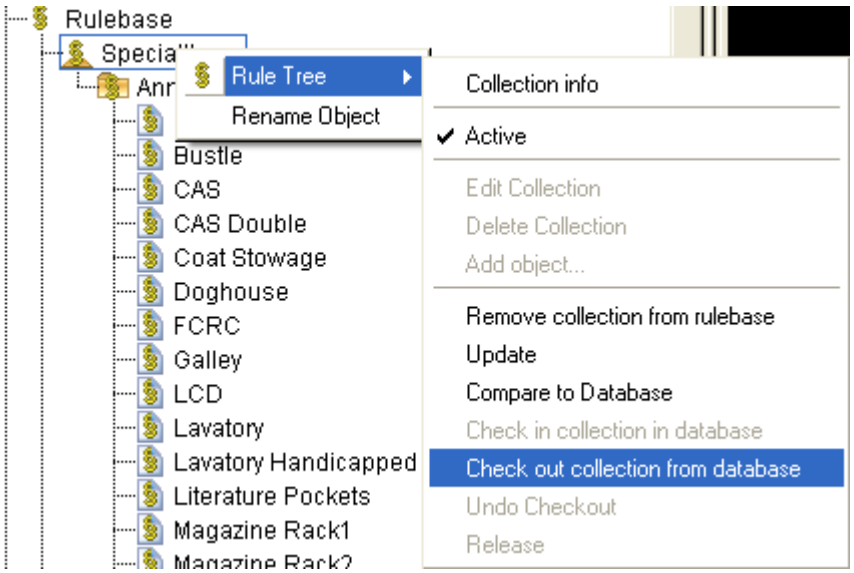


Fig. 4.60 Locking the Collection

These steps enumerated above – connecting to the Database and locking a collection – are necessary especially when the user wants to edit or create a rule.

4.3.3 Editing existing Rules

The Fig. 4.61 shows the workflow for editing existing rules:

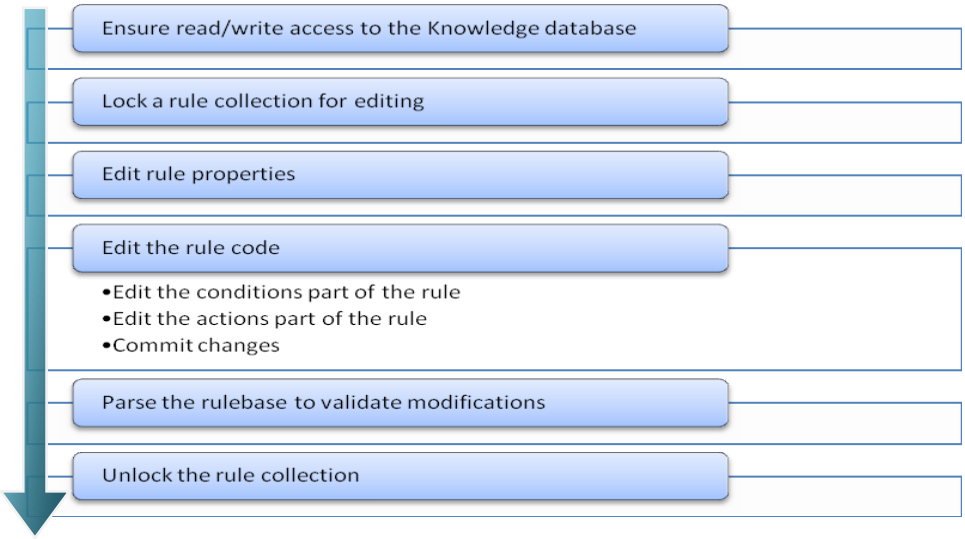


Fig. 4.61 The workflow for editing the existing rules

If the user wants to obtain details on classes, slots and methods implemented in application, he must access the Function Editor and select “Class Documentation” from the Function menu as in the Fig. 4.62:

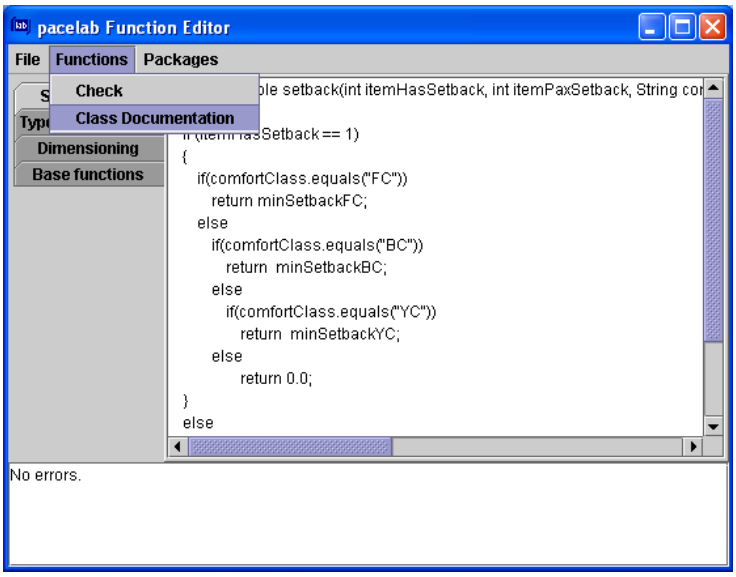


Fig. 4.62 The Class Documentation

For editing the rule properties, the user have to assure that he is on the read/write mode and that the rule collection he chooses is locked and after he may right-click the rule he wants to edit from a collection and select “Rule Tree/Edit rule” from the context menu. In the next example, the “SpecialItems” Collection will be chosen and from there “Seats/Pax/Check 20mm” Rule.

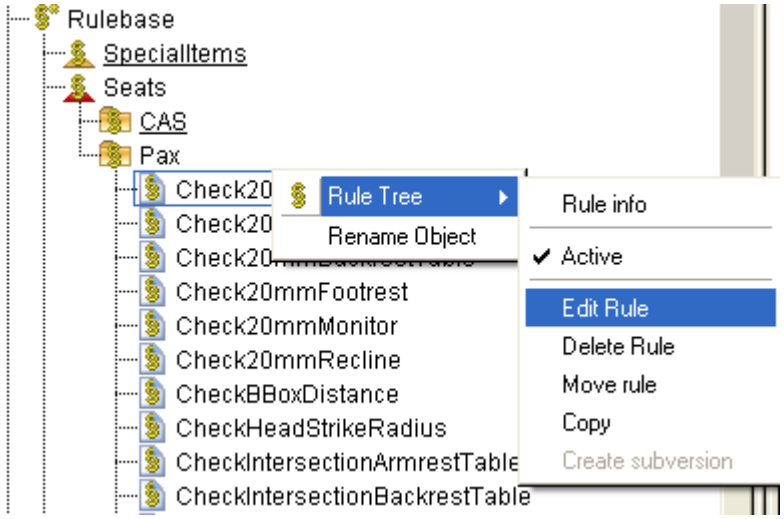


Fig. 4.63 Edit Rule

Choosing “Edit Rule”, the “Edit Rule Attributes” Dialog will open. As there can be seen, the description of the rule is: “Check if a cabin item is closer than 20 mm to pax seat (AN 64)” or, in other words, there must be minimum distance 20mm to surroundings. There should be noticed that not all the fields can be edited, some of them are only general information about the rule.

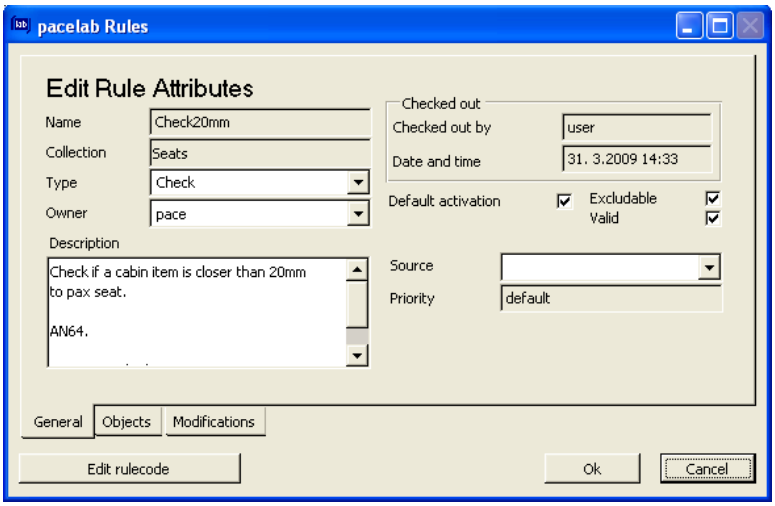


Fig. 4.64 “Edit Rule Attributes” Dialog

In the “General” tab, there are fields like Default Activation, Excludable, Valid. According to **Tutorial 2009**, the Default Activation expresses the state of activation after loading the rule from the Knowledge Database. Also, if the “Excludable” box is checked, the rule can be deactivated and if the “Valid” box is checked, the rule is valid (is not outdated or otherwise). However, invalid rules cannot be activated.

Editing the rule code (command which can be accessed from the “General” tab – “Edit rulecode”) is the actual editing of the rule and it consists of the “Conditions” part and the “Actions” part. In the following picture, the rule code for the example is shown:

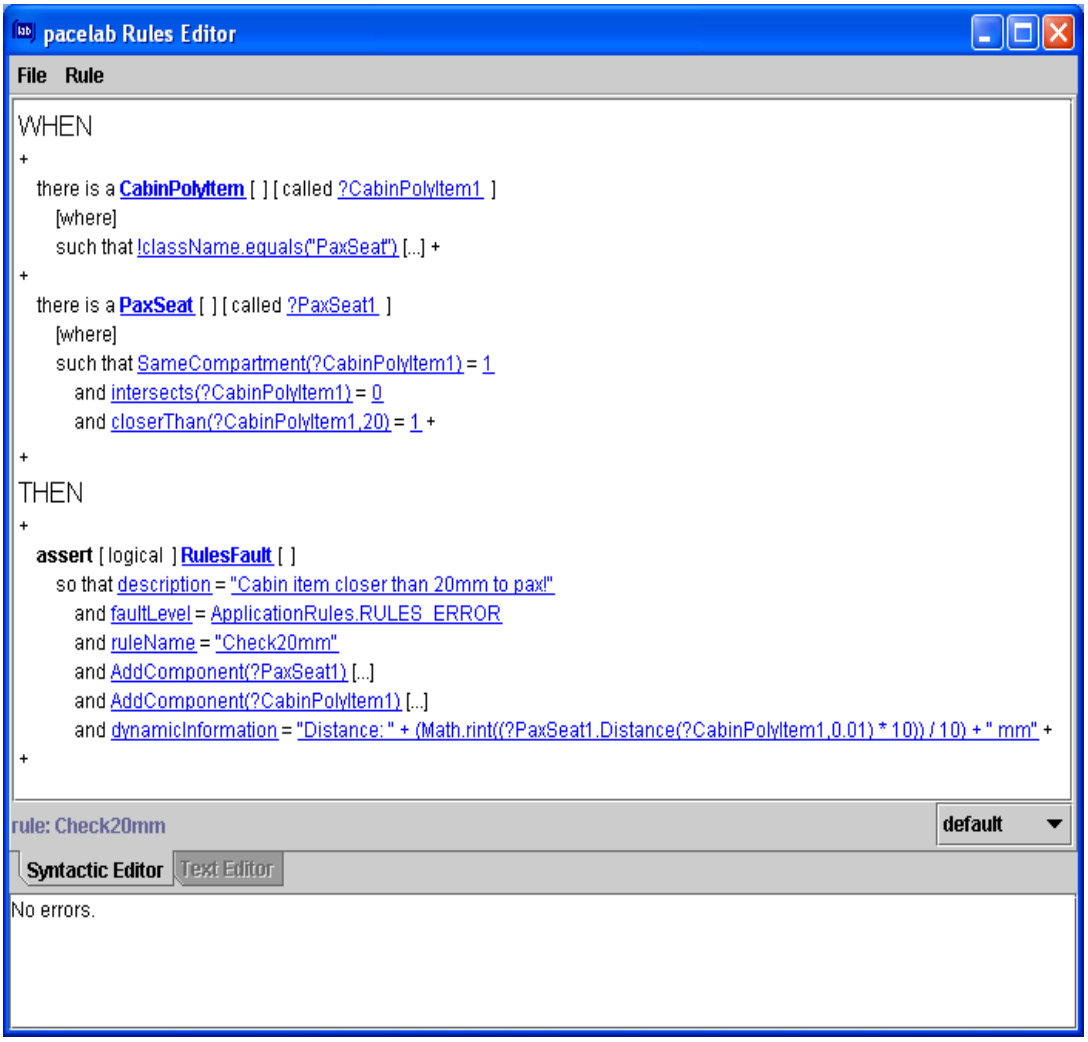


Fig. 4.65 Pacelab Rules Editor

There should be noticed that by clicking on the blue lines, the user can change the items in blue with others from a list which appears. Also, other actions can be added by clicking in front of the rows as is shown in the picture:

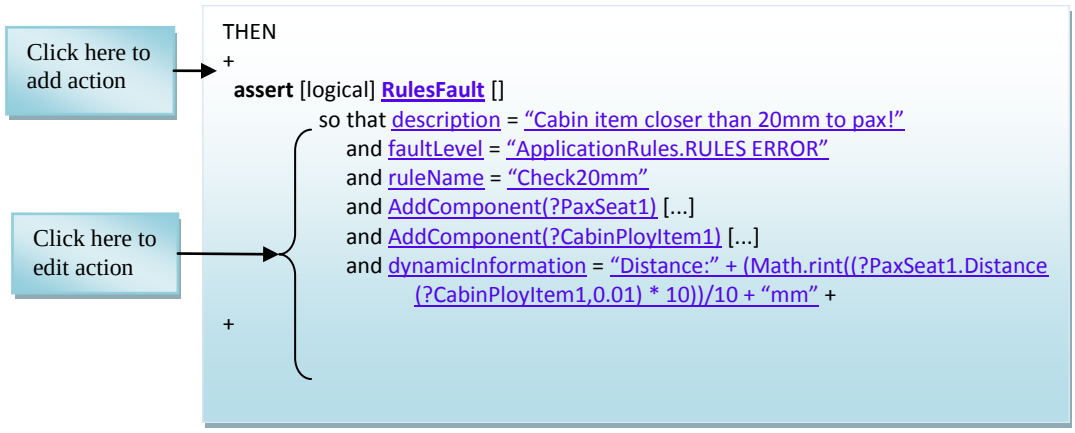


Fig. 4.66 Adding and editing actions

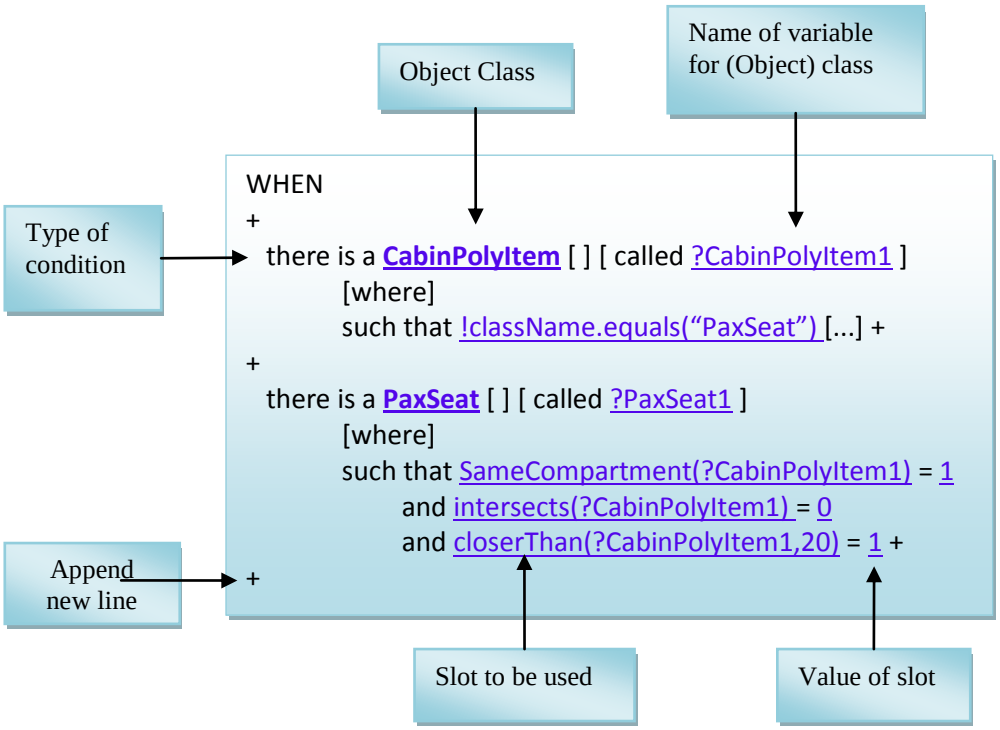


Fig. 4.67 Editing an existing rule

Here are described the items in the Fig. 4.67 (according to **Tutorial 2009**):

- the type of condition is an introductory statement to be selected from menu (the most widely is: "there is");
- the object class specify the objects to be involved in the rule check;

- the name of variable for object class expresses the name of the variable which is used to bind an object matching a condition to this variable in order to be able to use this object either in another condition or in the action part of the rule;
- "Append new line", like the name says, is used to add a new line;
- the "Slot (or method) to be used" represents the slots which are required to describe the object class more precisely (there must be noticed that methods describe the relationship between objects);
- the "value of the slot (or method)" is the field in which the user can select or type a name, a substring or a number required by the syntax of the slot.

Table 4.1 The code signs interpretations

Name of predicate	Explanation
<none>	No test. Used if left value is Boolean
=	Tests if left value is equal to right value
!=	Tests if values are not equal
>	Tests if left value is greater than right value
>=	Tests if left value is greater or equal to right value
<	Tests if left value is less than right value
<=	Tests if left value is less or equal to right value
is a	Tests if left object is of given class
equals	Tests if left object is the same as the right object

To give an example of how the cabin layout reacts at a possible change of the rule code, the maximum distance will be changed from 20 mm to 300 mm. In this way, after choosing "Parse Rule" from the "Rulebase" context, there will be rules violations on the cabin layout and many element will be turned to red. Afterwards, the Cabin Layout should look like this:

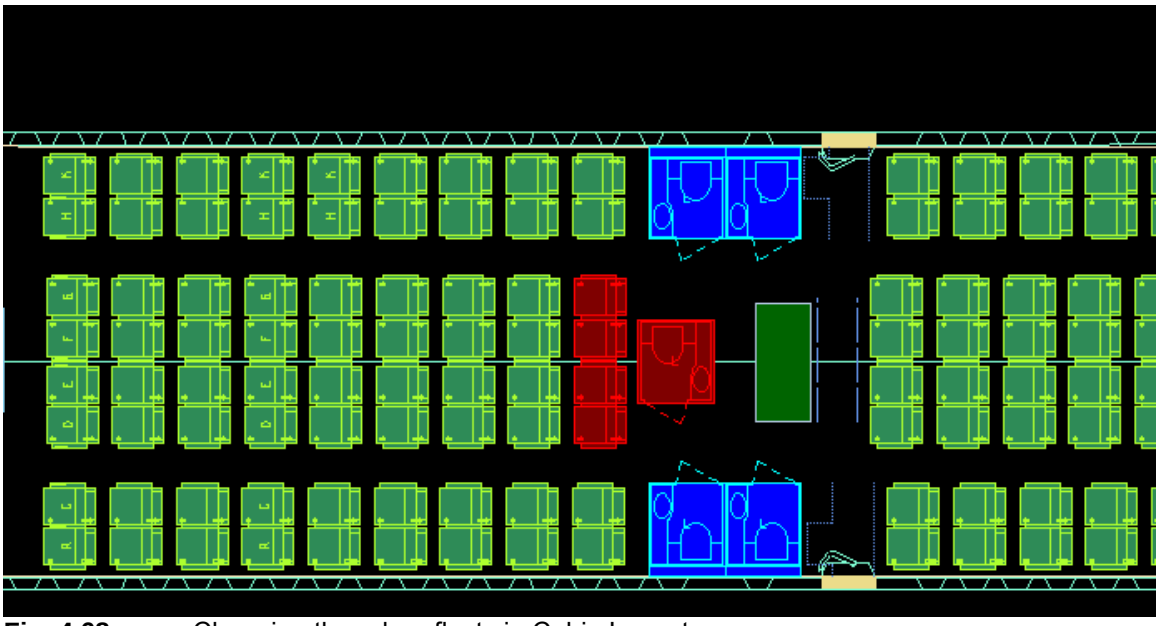


Fig. 4.68 Changing the rule reflects in Cabin Layout

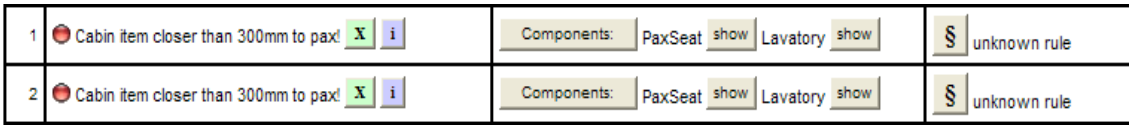


Fig. 4.69 Rules violations due to the changes in the rule code

In the next example, the rule which checks if the cross aisle of type A (2-way access, in the front) properly overlap with a left passage way of a type A door will be modified.

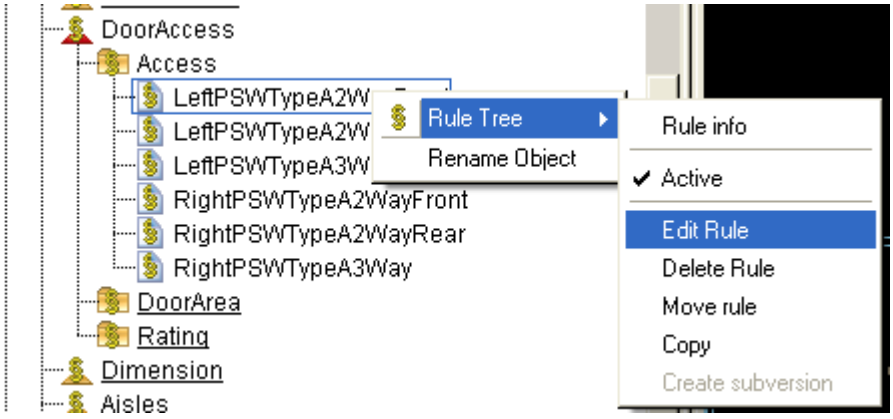


Fig. 4.70 Rules for door access

According to **FAA 2009**, FAR 25, article 25.814 (a) chapter (Emergency exit access), "there must be a passageway leading from the nearest main aisle to each Type A, Type B, Type C, Type I or Type II emergency exit and between individual passenger areas". The same source (Federal Aviation Administration, chapter 25.814 (i)) says that "the access

must be provided by an unobstructed passageway that is at least 10 inches in width for interior arrangements".

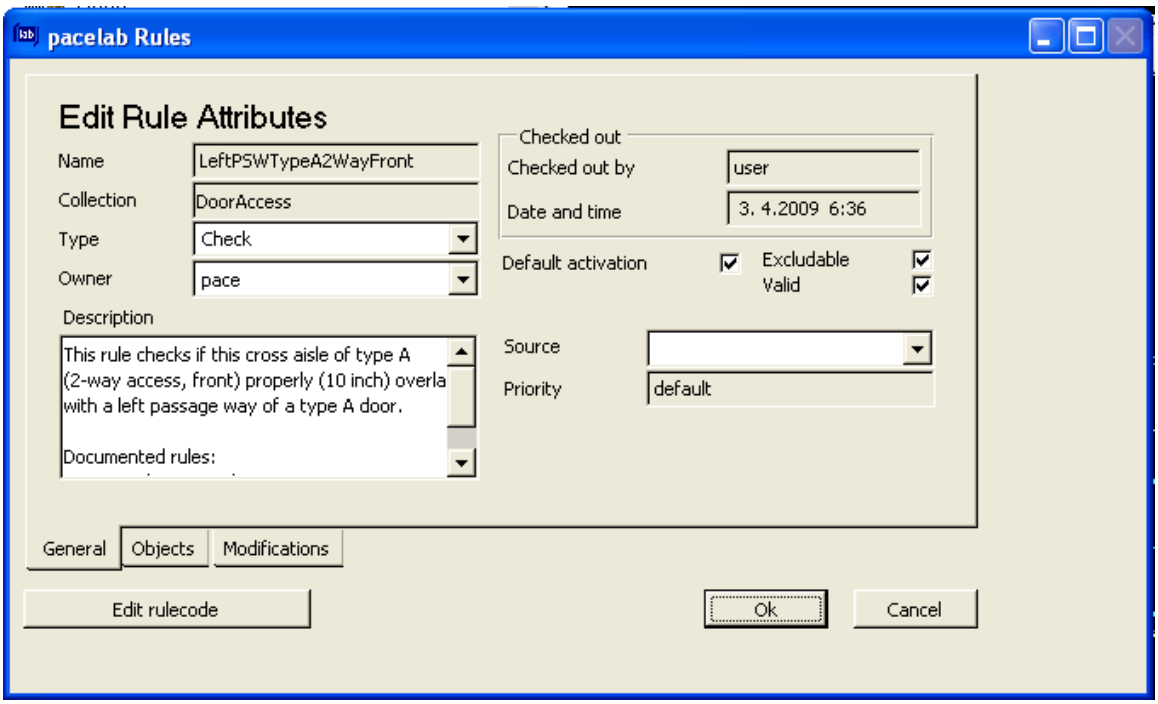


Fig. 4.71 Cross aisle Type A door 2-way access

As in the first example, after locking the rule, the user can open the "Edit Rule Attributes" where he can see the description of the rule and choose from there "Edit Rule Code".

.

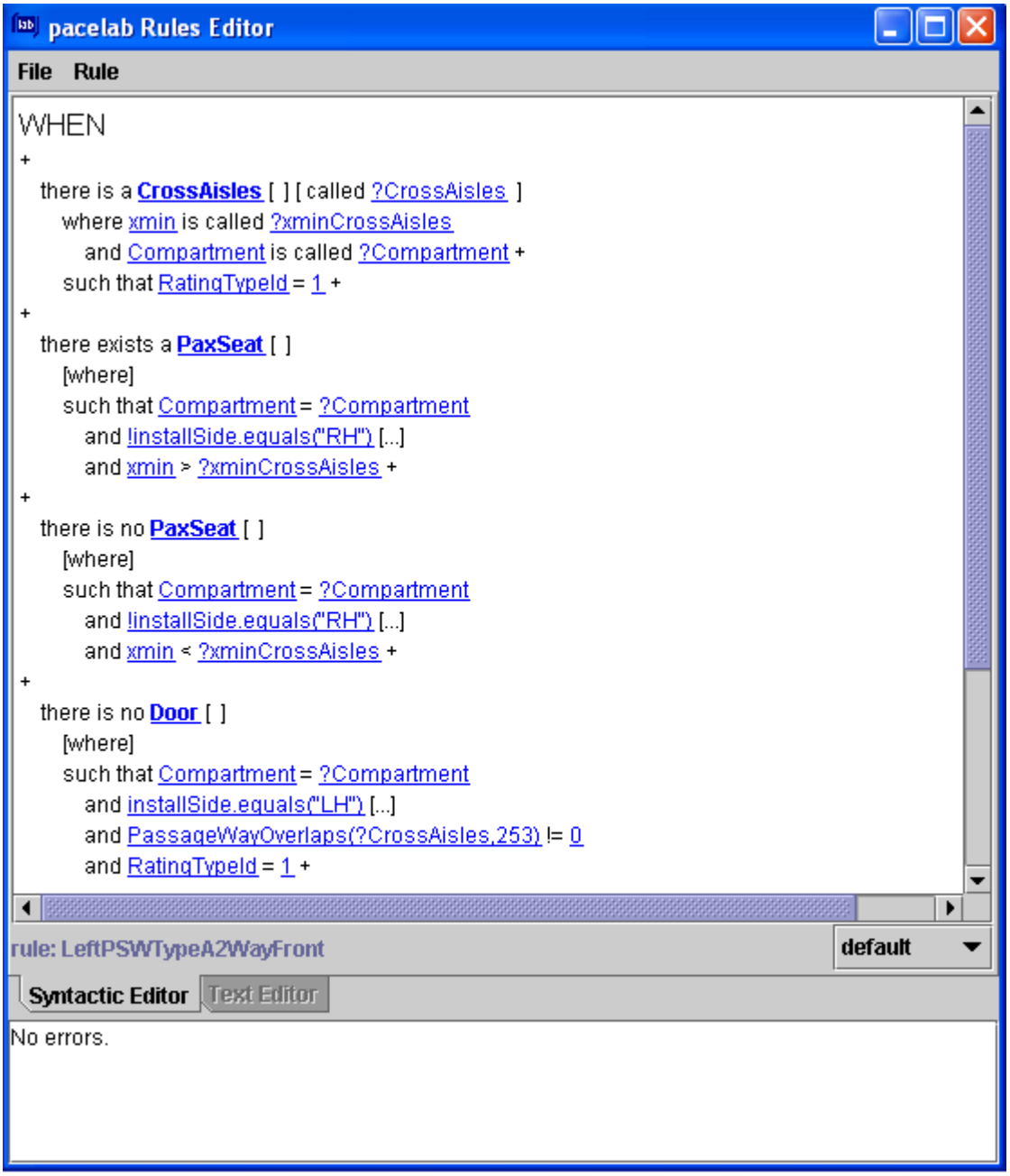


Fig. 4.72 The “When” part of the Rule Code

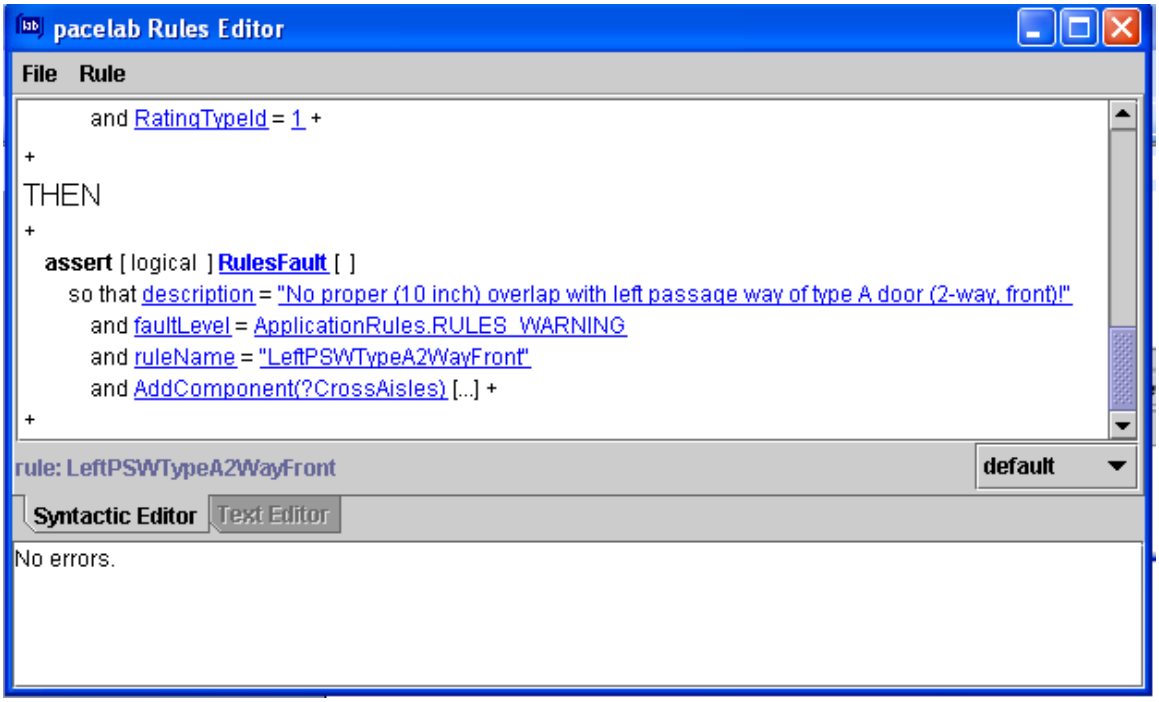


Fig. 4.73 The “Then” part of the Rule Code

The user can change the distance of 253 mm with 500 mm, or, in other words, to create a more exigent rule which says that the width must be almost double than before:

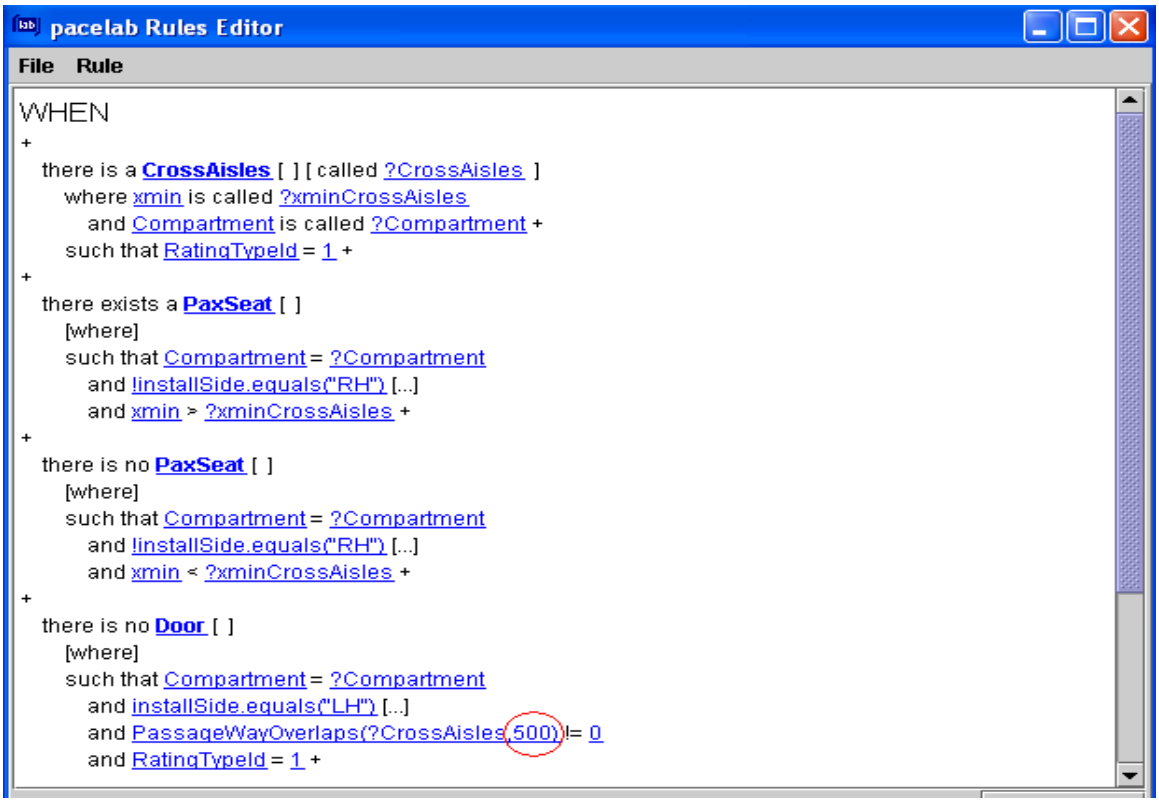


Fig. 4.74 Changing the rule

The rule violation is represented in the Layout by the two vertical red lines:

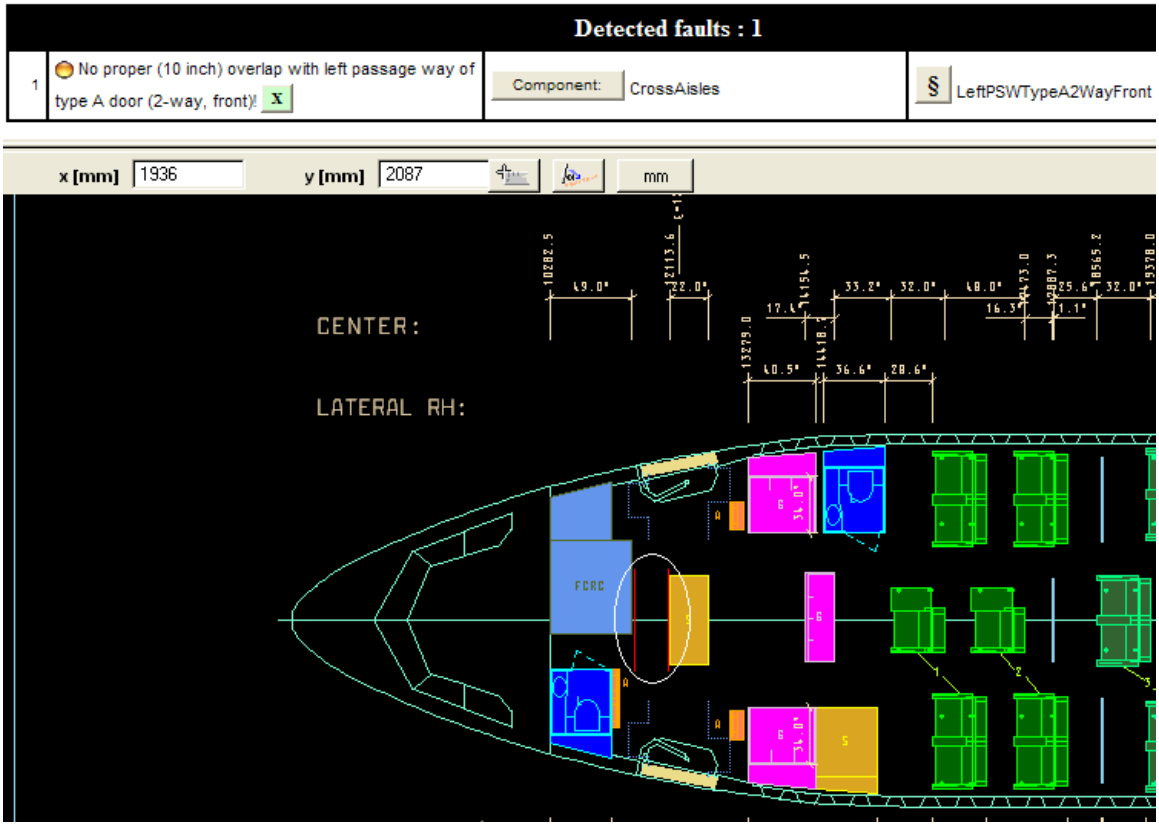


Fig. 4.75 The results on the Layout after modifying the rule

The last example shows that Pacelab Cabin incorporates rules from the Federal Aviation Administration. FAA issues and enforces regulations and standards related to the manufacture, operation, certification, and maintenance of aircraft.

4.3.4 Creating New Rules

Usually, the knowledge database needs to be updated so that new requirements can be implemented. Therefore, the user needs to add new rules in the rule tree. In this subchapter, the way of creating rules from scratch will be presented.

The first step in accomplishing this is to add a rule from the Object Palette in a folder in Rules Tree, like in Fig. 4.76.

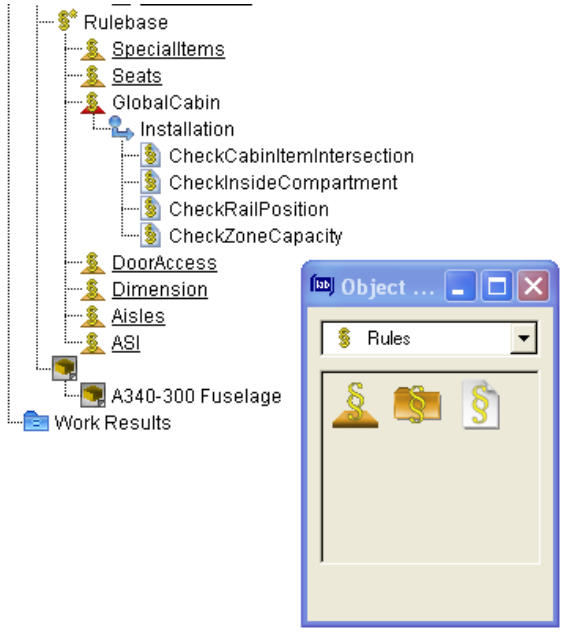


Fig. 4.76 Adding a rule to the Tree

It must be noticed that a rule can be added only to a folder and not to a Rules Collection. In this example, the folder in which the rule will be created is the "Global Cabin". The steps of creating the rules are similar to those for editing a rule. Considering this, the next step is to lock the "Global Cabin" folder and afterwards, the user can start working on the rule.

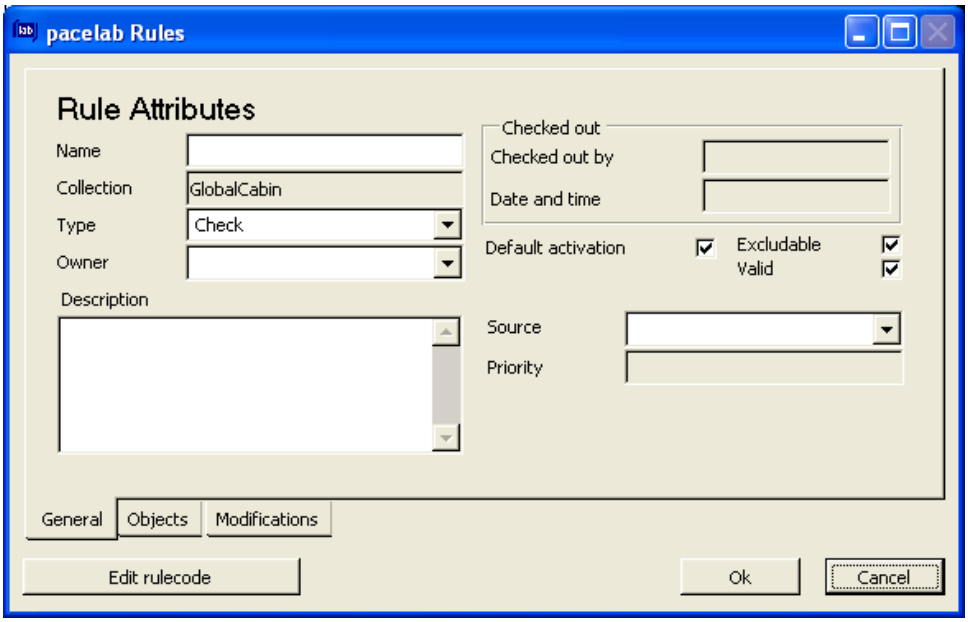


Fig. 4.77 Rules Attributes window

After the “Rule Attributes” Dialog opened, the user can edit the characteristics of the rule:

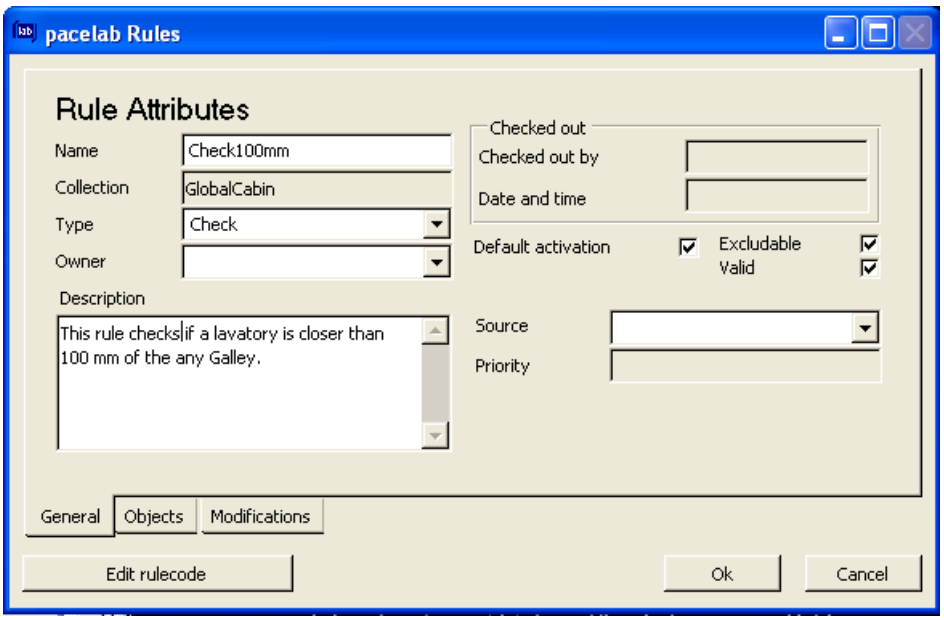


Fig. 4.78 Editing the Rule Attributes

In this example, the description of the rule which the user wants to create is: “the rule checks if lavatory is closer than 100mm of the any Galley” and the name of the future rule is “Check100mm”.

After the rule is defined in this way, it can be seen in the Rules Tree:

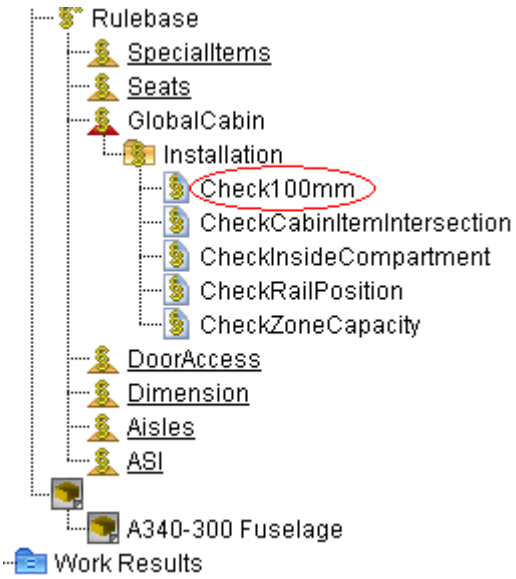


Fig. 4.79 The new rule appeared in the Rule Tree

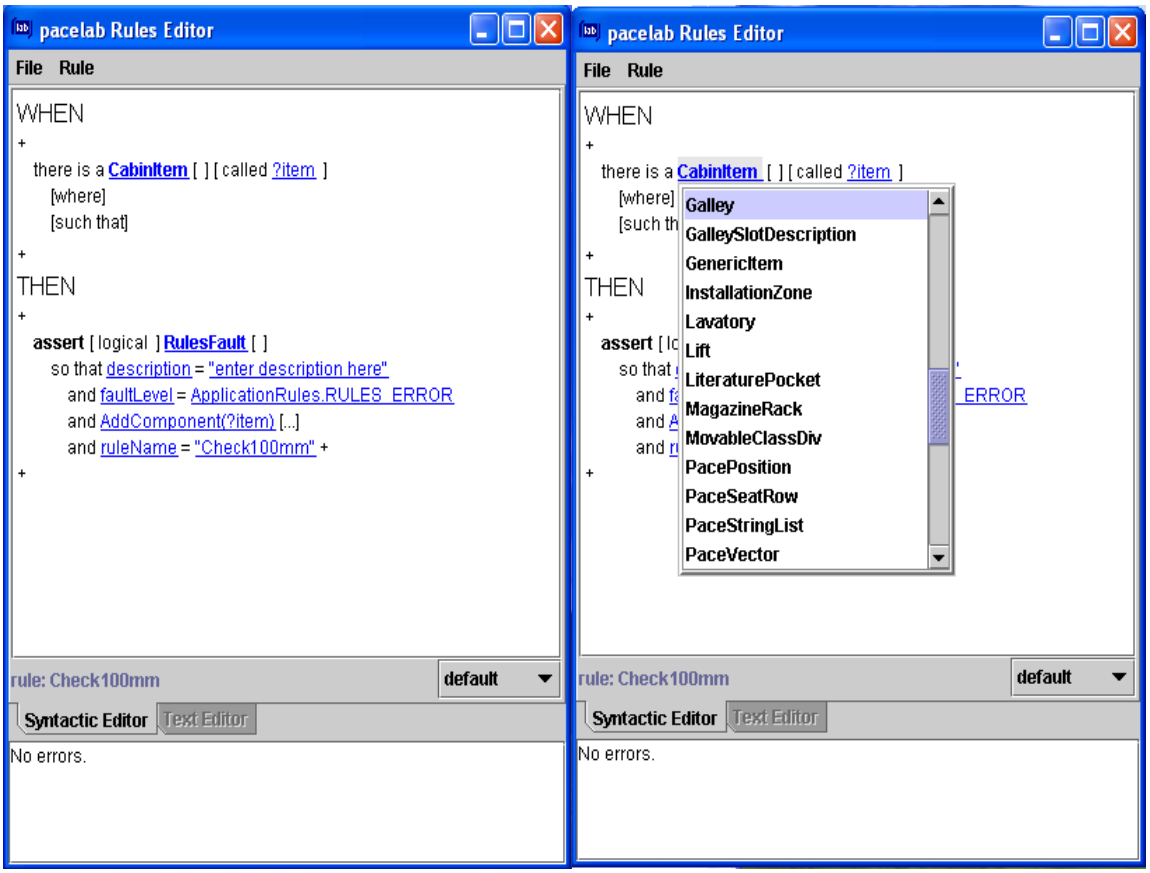


Fig. 4.80 Rule Code Editor

In the Fig. 4.80, the cabin item is chosen from a class of objects. The approach is the object oriented programming, also used by the Java programming language, which was adopted for creating the rule base in Pacelab Cabin.

After creating the code, the syntax can be verified by choosing from the “Rule” Menu, “Check” command. A simple error (in the red circle) can be seen in the next picture. When the user is sure that the syntax is correct, he can perform: “Commit the rule”.

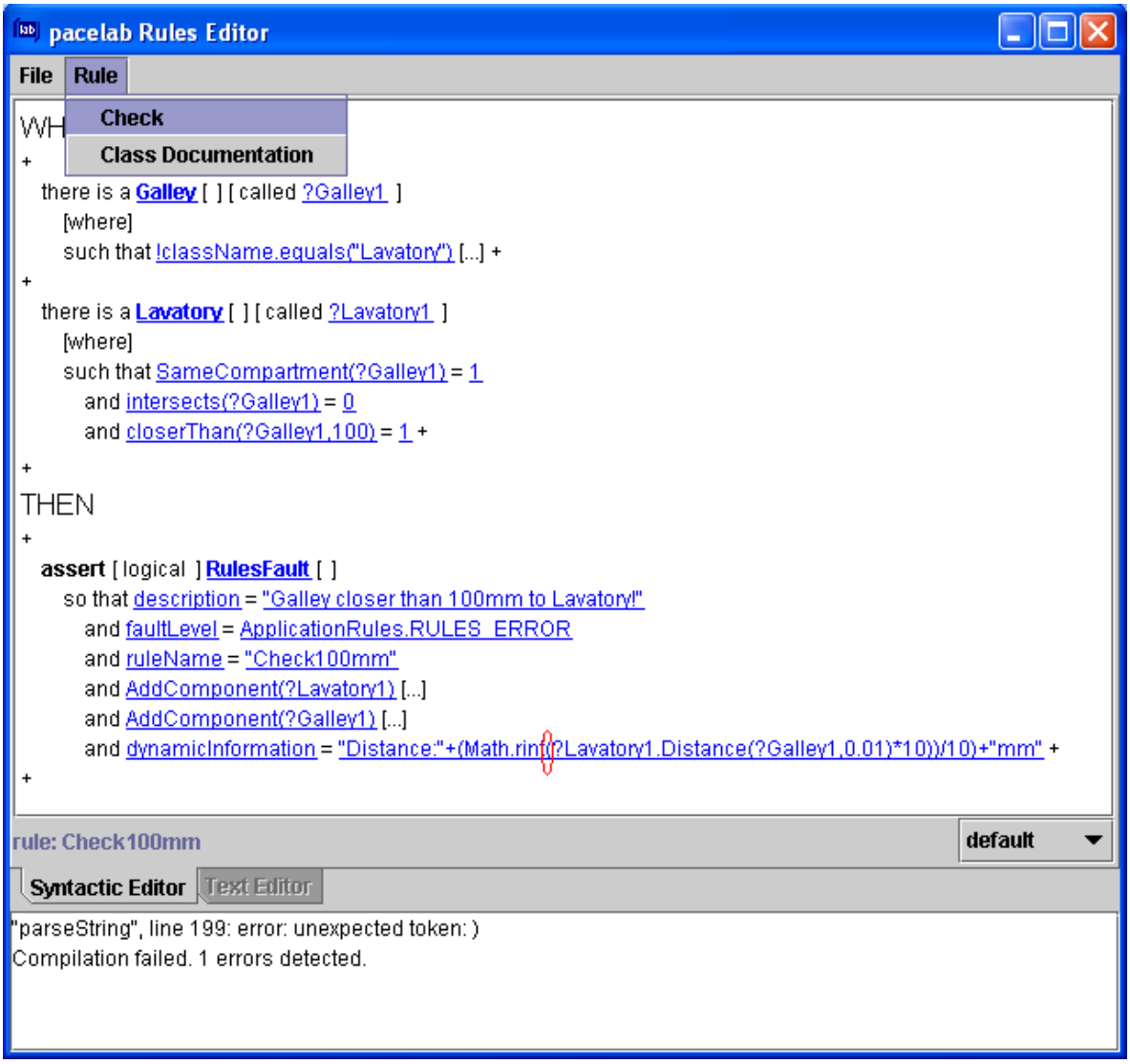


Fig. 4.81 The Check operation

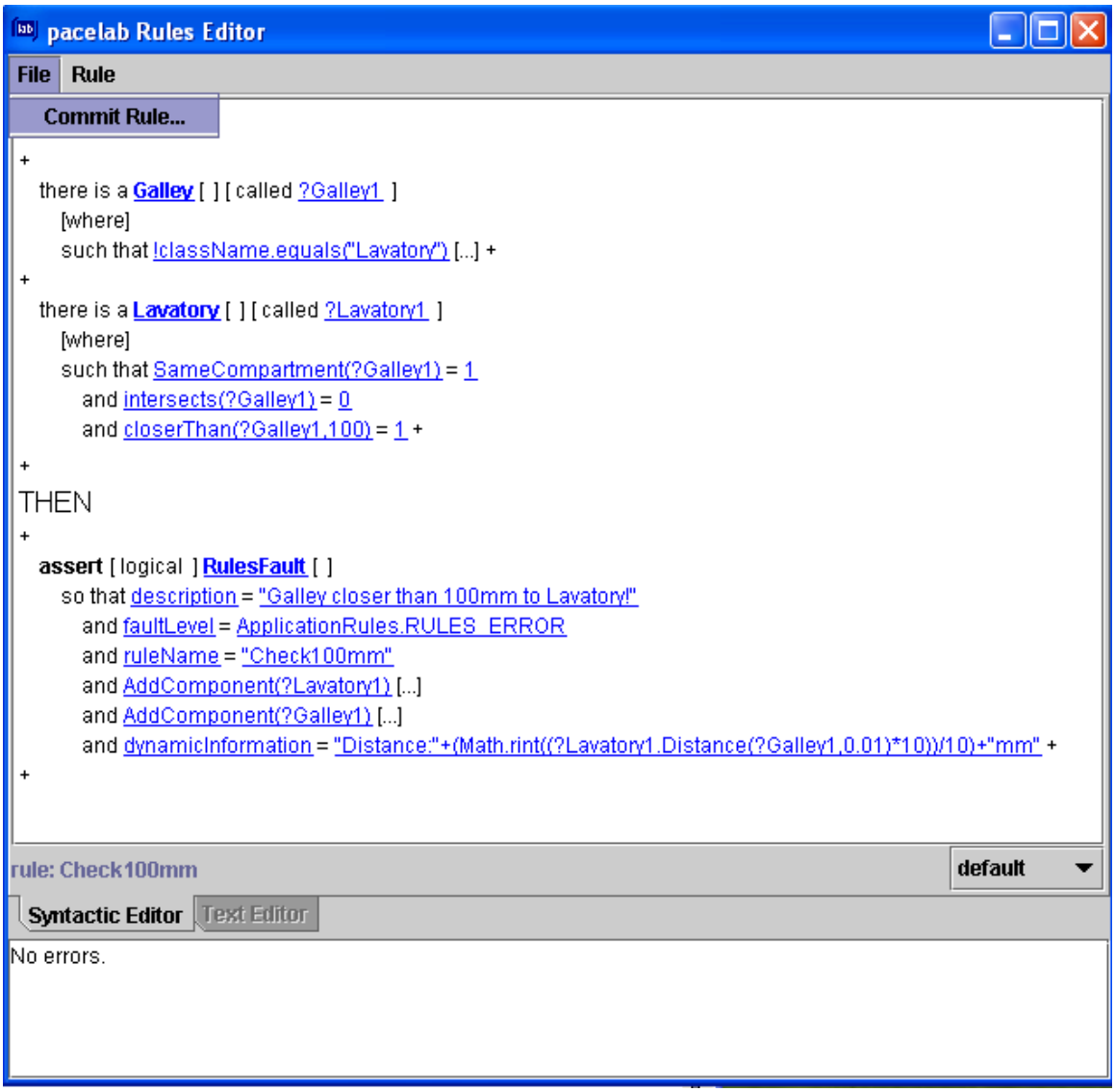


Fig. 4.82 Committing the rule

In the end, the effects of creating a new rule can be seen on the layout:

Fig. 4.83 The Results of applying the Rule to the Layout

An example of how the rule functions in the program is shown in Fig. 4.83. A lavatory and a galley are coloured in red in the First Class, due to the fact that the rule has been violated: the distance between a lavatory and a galley must be more than 100mm, as the rules states, and this distance has not been respected.

Due to this freedom to create rules, the user can have a better control on his work and can adapt the layout to new requirements.

4.4 Application of KBE in Pacelab Cabin

The Cabin Layout has to take many design limitations into account: certification rules, aircraft boundary conditions (from structures and aircraft systems) and customer requests. For this reason, it is almost incomprehensible for a human mind to optimize the cabin layout without compromising these design limitations (**Scholz 2009**).

In order to respond at these challenges, KBE allows manipulating the geometry and annexed knowledge and supports the investigation of multiple what-if on the design. Pacelab Cabin is a program which uses the same approach. The knowledge is gathered in the *knowledge databases* represented by the sum of rules, coming either from the certification specifications or from airliner requirements. As shown in the previous paragraphs, the Rule base is represented in the form of a tree. Advanced users can access

the knowledge – the rules, can edit it, and update it according to the needs. Therefore, the creation of the cabin layout is flexible enough to match a large range of requirements.

There are two type of rules in Pacelab Cabin: rules provided by the knowledge base incorporated in the program and other rules (internal rules) which are used to draw the conclusions (if the first rules are respected or not). Furthermore, there is a method of using the rules in the knowledge base to derive a conclusion - the so-called inference engine - which takes the rules provided by the knowledge base and uses internal rules of inference to draw a conclusion.

In Pacelab Cabin Program, the user can easily insert, modify and delete the rules from the Database and this is because of the interrelation of elements beyond the rules in the knowledge base. The rules combined in objects allow the effective manipulation of engineering knowledge.

As **La Rocca 2009a** states, rule-based design, object-oriented modelling and parametric CAD represent the cornerstones of KBE technology. Indeed Pacelab Cabin combines these attributes. The object oriented programming is used for creating new rules, while 2D and 3D visualizations illustrate the cabin layout.

By using the advantages of KBE, Pacelab Cabin, allows the time and cost reduction for engineering applications, especially in the early phases of the negotiation with the customers. Therefore, the engineers can focus more on the conceptual design activities.

However, the knowledge database, represented by the rules in Pacelab Cabin is not complete. For creating certified cabin layouts all the rules in the certification specifications must be considered and transformed into valid constraints. It can be concluded that there is the need to continuously update the rule base, according to both Aviation Safety Agencies and airlines requirements.

5 Summary

In this work, the theory of Artificial Intelligence (AI) and Knowledge Based Engineering (KBE), its possibilities to an application in the field of cabin conversion and refurbishing are investigated. Within large aircraft companies, like Boeing, Lockheed Martin and Airbus, KBE is already a mainstream technology since years.

To remain competitive in the aerospace industry, it is no longer enough to be able to design smart products, but one also needs to design them in a smart way. The company PACE has produced the Pacelab Cabin software – a tool for creating Cabin Layouts. The program uses a Rules Engine based on KBE approach (which has roots in Artificial Intelligence). It returns 2D and 3D representations of the Cabin Layout, and provides the possibility to define rapid modifications with little effort. As a consequence Pacelab Cabin can be used especially as a marketing instrument during negotiations with the customers. This can be done with minimum preparation, allowing the increasing communication between the company using Pacelab and its customers.

However the knowledge base of the program is subject to continuously upgrades. A further direction in which this paper can be continued is to investigate the available knowledge and to represent this knowledge by creating new rules to be implemented into the program. Another investigation direction is the adaptation of this program to the cabin refurbishing tasks, like generating delta part lists along with the layout.

List of References

AP 2289 AIRBUS PROCEDURE: AP2289 – *Develop new Cabin & Cargo (DnCC).* – The Processes

Barr 1981 BARR, Avron; FEIGENBAUM, Edward A: *The Handbook of Artificial Intelligence (Vol. 1)*. Los Altos : William Kaufmann, Inc., 1991.

Brachman 1991 BRACHMAN, J. Ronald; LEVESQUE, J. Hector; REITER, Raymon: *Introduction to the Special Volume on Knowledge Representation*. New York : Prentice Hall International, 1991.

Breuker 1987 BREUKER, J.A.; WIELINGA, B.J.: *Knowledge Acquisition for Expert Systems*. New York : Plenum Press, 1987.

Chapman 2007 CHAPMAN, C.; PRESTON, S.; PINFOLD, M.; SMITH, G.: *Utilising enterprise knowledge with knowledge-based engineering*. Geneva : Inderscience Publishers, 2007.

Cooper 1999 COOPER, S.; FAN, I-S; LI, G.: *A best practice guide – Achieving competitive advantage through Knowledge-Based Engineering*. Department of Trade and Industry, UK.

Cunis 1991 CUNIS, R.; GUENTER, A; STRECKER, H.: *The PLAKON-Book*. Menlo Park : American Association for Artificial Intelligence, USA.

Dhar 1985 DHAR, V.: *An Approach to Dependence Directed Backtracking Using Domain Specific Knowledge*. New York : NYU Working Paper, USA.

EASA 2009 EUROPEAN AVIATION SSAFETY AGENCY. URL: http://easa.eu.int/ws_prod/g/doc/Agency_Mesures/Certification_Spec/decision_ED_2003_01_RM.pdf (2009-06-18).

FAA 2009 FEDERAL AVIATION ADMINISTRATION: *§25.813 Emergency exit access*. URL: http://edocket.access.gpo.gov/cfr_2009/janqtr/pdf/14cfr25.813.pdf (2009-06-30)

Felfernig 2000 FELFERNIG, Alexander; GERHARD, Friedrich; JANNACH, Dietmar; STUMPTNER, Markus; ZANKER, Markus: *UML as knowledge acquisition frontend for Semantic Web configuration knowledge bases*. University of South Australia, Advanced Computing Research Centre.

Fischer 2002 FISCHER, Michael D.: *Indigenous knowledge and Expert Knowledge in Development*. Harwood : Silatoe and Bicker, 2002.

Giesecke 2005 GIESECKE, Raphael: *Airbus Competence Training (ACT) – Cabin: Develop new Cabin & Cargo*. Airbus Hamburg, 2005.

Graham 1997 GRAHAM, Deryn; BARRETT, Anthony: *Knowledge-Based Image Processing System*. London : Springer, 1997.

Green 1986 GREEN, C.; BARSTOW, D.: *On program synthesis knowledge, Readings in artificial intelligence and software engineering*. San Francisco : Kaufmann Publishers Inc., 1986.

Guenter 1990 GUENTER, A.; CUNIS, R.; SYSKA, I.: *Separating Control from Structural Knowledge in Construction Expert Systems*. Charleston : USA, 1990.

Hickman 1990 HICKMAN, Frank R.; PORTER, David; LAND, Lise; MULHALL, Tim; KILLIN, Jonathan L.: *Analysis for Knowledge-Based Systems: A Practical Guide to the KADS Methodology*. Prentice Hall Professional Technical Reference, 1990.

Kopisch 1991 KOPISCH, Manfred: *Configuration of a Passenger Cabin of an Aircraft with an Expert System*. Univ. Hamburg, 1991.

Kopisch 1992 KOPISCH, Manfred; GUENTER, Andreas: *Configuration of a Passenger Aircraft Cabin based on Conceptual Hierarchy,*

Constraints and flexible Control. London : Springer-Verlag, 1992.

La Rocca 2007a LA ROCCA, G; VAN TOOREN, M.J.L: *Enabling distributed multidisciplinary design of complex products: a Knowledge Based Engineering approach*. URL: http://search.tudelft.nl/en/?q=La%20Rocca (2009-06-30)

La Rocca 2009a LA ROCCA, G; VAN TOOREN, M.J.L: *A Modular Reconfigurable Software Modelling Tool to Support Distributed Multidisciplinary Design and Optimisation of Complex Products*. URL: http://search.tudelft.nl/en/?q=La%20Rocca (2009-06-30)

La Rocca 2009b LA ROCCA, G; VAN TOOREN, M.J.L: *Development of Knowledge Based Engineering Techniques to Support Aircraft Design and Multidisciplinary Analysis and Optimization*. URL: http://search.tudelft.nl/en/?q=La%20Rocca (2009-06-20)

La Rocca 2009c LA ROCCA, G; VAN TOOREN, M.J.L: *Knowledge Based Engineering to support aircraft multidisciplinary design and optimisation*. URL: http://search.tudelft.nl/en/?q=La%20Rocca (2009-06-20)

Lexicon 2004 LEXICON DEFINITION.
URL: http://www.lexicon-definition.de (2004-09-20).

Luger 2004 LUGER, George; STUBBLEFIELD, William: *Artificial Intelligence: Structures and Strategies for Complex Problem Solving (5th ed.)*.The Benjamin/Cummings Publishing Company, 2004.

McCarthy 1955 MCCARTHY, John: *Artificial Intelligence, Logic and Formalizing Common Sense*. Stanford: Computer Science Department, 1990.

McDermott 1980 MCDERMOTT, J.: *A Rule-Based Configurer of Computer Systems*. Pittsburgh: Carnegie-Mellon University, 1980.

Webster 2009 MERRIAM-WEBSTER ONLINE DICTIONARY.
URL: http://www.merriam-webster.com/dictionary (2009-06-18).

Minsky 1972 MINSKY, Marvin; PAPERT, Seymour: *Artificial Intelligence*. University of Oregon Press, 1972.

Nawijn 2009 NAWIJN, M.; VAN TOOREN, M.J.L: *Automated Finite Element Analysis in a Knowledge Based Engineering Environment*. URL: . http://zoeken.tudelft.nl/nl/?q=NAWIJN (2009-06-30)

Niţă 2009 NIȚĂ, Mihaela; SCHOLZ, Dieter: *The Process Chain to a Certified Cabin Design and Conversion*. Hamburg : Aero – Aircraft Design and Systems Group, 2009.

PACE 2009a PACE, Aerospace Engineering and Information Technology Gmbh: *Airbus Upgrade Services Center uses PACE-software to streamline refurbishment of passenger cabins*. URL: http://www.pace.de/en/index.php?PHPSESSID=21a4d3f94cd65924d4fcf6380ac4ae72&lang=en (2009-04-10).

PACE 2009b PACE, Aerospace Engineering and Information Technology Gmbh: *Case study SCHEMGEN*. URL: http://www.pace.de/en/index.php?PHPSESSID=21a4d3f94cd65924d4fcf6380ac4ae72&lang=en (2009-04-10).

Poole 1998 POOLE, David; MACKWORTH, Alan; GOEBEL, Randy: *Computational Intelligence, a Logical Approach*. New York : Oxford University Press, 1998.

Probst 1999 PROBST, G.; RAUB, S.; ROMHARDT, K.: *Managing Knowledge*. London : Wiley, 1999.

Russell 1994 RUSSELL, J. Stuart; NORVIG, Peter: *Artificial Intelligence. A Modern Approach*. New York : Prentice Hall International, 1994.

Scholz 2009 SCHOLZ, Dieter: *Aircraft Cabin and Cabin System Refurbishing – Optimization of Technial Processes (CARISMA)*. Hamburg University of Applied Sciences, Department of Automotive and Aeronautical Engineering, 2009.

Schut 2008 SCHUT, E.J.; VAN TOOREN, M.J.L: *Development and Implementation of Knowledge-Based Design Process Primitives*. URL: http://zoeken.tudelft.nl/nl/?q=SHUT (2009-06-30).

Seeckt 2004 SEECKT, Kolja: *Cabin Configuration using PaceLab Cabin*. Hamburg : University of Applied Sciences, 2004.

Simon 1956 SIMON, H.A.; NEWELL, Allen: *The Logic Theory Machine*. Pittsburgh : Carnegie Mellon University, 1956.

Stokes 2001 STOKES, M.: *Managing Engineering Knowledge: MOKA Methodology for Knowledge Based Engineering Application*. New York : American Society of Mechanical Engineers, 2001.

Trauşan 1998 TRAUSAN, Matu Stefan: *Knowledge-Based, Automatic Generation of Educational Web Pages*. Romanian Academy and National Research Council for University Research, 1998.

Tutorial 2009 PACE, Aerospace Engineering and Information Technology Gmbh: *Pace Documentation*. URL: http://www.pace.de/en/index.php?PHPSESSID=2239b2f0e61b8e4238f31e9cf891a48e&lang=en (2009-30-06).

van der Laan 2004 VAN DER LAAN, A.H.; VAN TOOREN, M.J.: *Parametric Modeling of Movables for Structural Analysis*. URL: http://zoeken.tudelft.nl/nl/?q=VAN+DER+LAAN&start=0&lang=nl (2009-04-15).

van Tooren 2009a VAN TOOREN, M.J.L.; NAWIJN, M.; BERENDS, J.P.T.J.; SCHUT, E.J.: *Aircraft Design Support using Knowledge Engineering and Optimisation Techniques*. URL: http://zoeken.tudelft.nl/nl/?q=VAN%20TOOREN (2009-05-20).

van Tooren 2009b VAN TOOREN, M.J.L.; LA ROCCA, G.; KRAKERS, L; BEUKERS, A: *Design and Technology in Aerospace. Parametric Modelling of Complex Structure Systems including Active Components*.URL: http://zoeken.tudelft.nl/nl/?q=VAN%20TOOREN (2009-05-20).

Vermeulen 2005 VERMEULEN, B.; VAN TOOREN, M.J.L; PEETERS, L.J.B.: *Knowledge Based Design Method for Fibre Metal Laminate Fuselage Panels*. Long Beach: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, 2005.

Wikipedia 2009a WIKIPEDIA, The Free Encyclopedia. URL: http://www.answers.com/topic/artificial (2009-04-15).

Wikipedia 2009b WIKIPEDIA, The Free Encyclopedia. URL: http://en.wikipedia.org/wiki/Artificial_Intelligence (2009-02-01).

Wikipedia 2009c WIKIPEDIA, The Free Encyclopedia. URL: http://en.wikipedia.org/wiki/Knowledge_based_engineering (2008-12-06).

Williams 2009 WILLIAMS, Tom: *The Management of Programmes at Airbus*. URL: http://www.fzt.haw-hamburg.de/pers/Scholz/dglr/hh/text_2009_06_04_Management_of_Programmes_at_Airbus.pdf (2009-06-30).