# Dynamic Product Modeling with Inter-Features Associations: Comparing Customization and Automation

Bruno Lamarche[1] and Louis Rivest[2]

[1]Ecole de technologie superieure, Montreal, Canada, Bruno.Lamarche.1@ens.etsmtl.ca
[2]Ecole de technologie superieure, Montreal, Canada, Louis.Rivest@etsmtl.ca

**ABSTRACT**

The customization of CAD tools and the development of trade-oriented applications make it possible to eliminate tedious tasks of modeling that would otherwise monopolize the time of designers. This article reviews some concepts, such as entity typifying, associations, imprints and pseudo-imprints, used to associate features to form a dynamic product model. Concepts are tested in an aerospace case-study where CAD tools are used for modeling pockets in aircraft skin panels. Specific companies' know-how can be modeled by CAD tools through either Customization or Automation. Customization is achieved by implementing the concepts through User-Defined Features (UDF). A UDF is seen as a specific aggregation of constraints established between low-level geometrical objects, so as to transform an imprint into a pseudo-imprint. Once defined, this aggregation of constraints is manipulated as a single entity to accelerate design work. Automation goes beyond customization by performing UDF instantiation with limited human intervention. Automation decomposes the typified high-level technical objects, such as an assembly, into a series of low-level technical objects that feed the UDF. Time-gains measured reveal that customization is a better investment than automation.

**Keywords:** Features, associations, engineering knowledge, customization, automation, UDF

## 1. INTRODUCTION

A company's objectives during product development, generally, are to compress schedules, to lower cost, and to increase product quality. The processes that companies use to develop their products are pivotal in the attainment of these objectives. In order to enhance these processes, companies use a range of computer tools, including Computer Aided Design tools (CAD) to construct a digital mock-up of the product. Although CAD tools present many advantages, such as a reduction in the number of physical prototypes required for the validation of a concept or product, certain aspects related to their use can still be improved upon to maximize their efficiency.

In the aerospace industry, the product, a regional jet for example, is composed of several thousand parts. The average development cycle for an aircraft of this type extends over a period of several dozens of months and the number of change requests is significant. For example, in 2001 at Bombardier Aerospace, change requests nearly reached 14000 [8]. The implementation of associations within the product models can contribute to a decrease in the duration of a development cycle and also allow data consistency to be maintained within the models.

Researchers study different approaches to solve these issues. Deneux and Wang [3] proposed, in their model for functional "re-design", to establish links between the various knowledge concepts to be modeled. These concepts are either problems or solutions, and are represented at two levels: the conceptual level (represented by a symbol), or the parameter level (represented by alphanumeric values). The concepts are linked to one another at the conceptual as well as parameter level, which makes it possible to integrate knowledge into the model.

Zimmermann et al. [14;15] introduced a concept of links that he calls ULEO (Universal Linking of Engineering Objects). These links allow the transmission of information between different applications of the product development process. Engineering Objects, OR "Eos", are at the center of the model and correspond to all objects related to engineering (CAD assembly models, features, parts, tolerances, materials, etc.). EOs are divided into different classes

within a central taxonomy called UMEO (Unified Model of Engineering Objects) so that the various applications that access the models obtain the same information.

Hoffman and Joan-Arinyo [6;7] developed a neutral format of language for the representation of form features and constraints : "Erep – An Editable, high-level REPresentation for geometric design and analysis". They used this language to develop an application prototype to create UDFs (User-Defined Features) that they then applied to build a lubrification cylinder UDF inserted into a combustion engine connecting rod.

Bidarra et al. [1;2] proposed a structure to arrange the different classes of UDFs. First, they defined a UDF class and then described the form specifications and parameters of these features. They also dealt with the geometrical, topological and functional aspects that allow for the validation of specific user-created features.

Lederman et al. [10] collaborated with industrials on a project developing new methods to improve the precision, efficiency, and flexibility of predicting diverse data used in the preliminary design of an aircraft's structural elements. They used UDFs and VB scripts to create a "dynamic object", automatically adding frames and stringers in the aircraft's digital mock-up when the length of a section of the fuselage is modified. The model is linked and reacts to a finite elements analysis procedure, optimizing the number of structural elements in a section of fuselage in order to minimize the total weight of the aircraft.

Deneux's research deals with association concepts capturing specific design know-how. His approach is based on functions established at the beginning of the design and is directed towards "re-design". His model takes into account the inaccuracy of the know-how. Zimmermann's work puts emphasis on a structure of links standardizing the information between the different applications accessing the models. Bidarra looks at the management of UDFs modeled by object-oriented classes. Hoffmann's work focuses on the mechanisms underlying the creation of UDFs, which are important because they demonstrate the possibility of generating UDFs and of instantiating them within a model. These works are conceptual compared to Lederman's, whose work is more practical and closer to ours, which is focusing on the manipulation of UDFs and their potential to integrate precise engineering know-how as a solution to improve development time and data consistency.

This article discusses how concepts such as entity typification and persistent associations established between typified entities, to derive pseudo-imprints from imprints, can be implemented to customize and automate domain-specific modeling tasks. Indeed, associating intra-part and inter-part features lead to a dynamic product model that captures some of the design intent within the models. The resulting product model allows automatic change propagation. An aerospace case study is presented. Customization of a generic CAD system is first used to capture the task-specific knowledge that controls the insertion of lightening pockets into the skin panel of an aircraft fuselage. We then apply Automation to Customized UDFs to further facilitate pockets instantiation.

Section 2 of this article is devoted to the industry case study. Section 3 introduces the fundamental concepts. Section 4 presents both the Customization and the Automation prototype. Section 5 compares the results obtained with Customization and with Automation.

## 2. INDUSTRY CASE STUDY

The industry case, drawn from the aeronautical field, concerns the lightening of skin panels of different fuselages of an aircraft using chemically machined pockets. Among the many parts of an aircraft, we find stringers and frames. These structural elements particularly interest us, because they are used as a reference in the geometric definition of the pockets located in the skin panel's assembly (SPA). The number of SPA varies from one type of aircraft to another, but one counts them by ten for a small one.

One of the objectives guiding the design of an aircraft is to minimize its weight. In fact, the heavier the aircraft, the less possible will it be to fit the passengers and/or goods on the same flight [9]. One way to reduce the weight is to vary the skin panel thickness of the aircraft's SPAs. For example, near the structural elements, the thickness must be greater, because the mechanical constraints on the SPA are greater. In contrast, the areas of the SPA that are not constrained as much require less thickness, such as midway between the structural elements (figure 1).

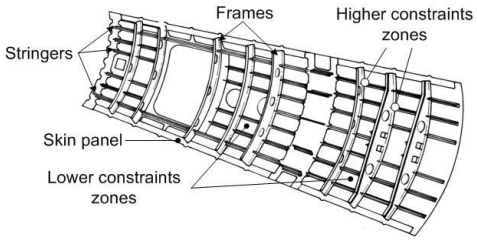


Fig. 1: Example of a skin panel assembly (SPA) of the central fuselage [12].

Thinnings in the skin panel (called lightening pockets) are applied to lower constraints zones. Chemical machining is used to "carve out" the metal to the desired thickness and tolerances. To do this, one completely covers the skin panel with a neoprene mask in which cutouts are made using cutout templates, so that the acid eliminates the metal in the desired spots. The geometrical shape and the outline of the pockets are obtained by applying design rules to the boundaries delimited by the frames and stringers that surround the pocket. Obtaining this shape is a long and tedious task that could be automated.

### 2.1 Anatomy of an SPA

SPAs are comprised of, amongst other things, a skin panel covering the exterior of the aircraft, and structural elements of two types: stringers and frames. Stringers are longitudinal parts of the fuselage of an aircraft. They are radially distributed in the fuselage and are positioned using a system of radial plans.

The anatomy of a stringer is rather simple and its form, longitudinally, will be defined according to its place in the aircraft. For example, in the central fuselage, they will be straight, while in the forward and after fuselage they will be curved to fit the main internal surface known as "Inside Mold Lines" or "IML". For the construction of the digital mock-up, the IML is obtained by offsetting the main external surface, "Outside Mold Lines" or "OML", by the thickness of the skin panel towards the inside of the fuselage.

The frames are transverse structural elements of the fuselage. As with the stringers, the frames are part of the aircraft's skeleton that help rigidify the aircraft's structure, and impose the physical form of the fuselage. During product development, the frames are positioned in the digital mock-up of the aircraft using a system of planes called "Fuselage Stations" or "F.S." and the shape of a frame corresponds to the shape of the IML. An opening, called a cutout, is made in the frame at the intersection of a stringer. Transition geometry is required on the frame and joggles are thus defined at the cutout. These two operations (cutout and joggle definitions) are dictated by the rules laid down by the manufacturer. Moreover, these openings occasionally require that one rigidify the frame using embossing, which are called "beads". "Lightening holes" can also be used in order to lighten the frame and make way for the hydraulic system, cables, etc.

The skin panel or "skin" is generally made of aluminum alloy sheeting, the thickness of which can vary from about 0.04" to 0.13" (approximate values) and covers the skeleton of the aircraft. Because of the shape and thickness of the skin panel, conventional manufacturing of lightening pockets using machining tools is difficult and very costly. This is the reason chemical manufacturing is used to "engrave" the pockets into the skin in the desired places.

### 2.2 Lightening pockets

Michaud [11] classifies skin panels into four distinct families (table 1).

| Families | Production Volume | Curvatures | | Number of pockets | | Pocket Depth | | Pocket shape | | Islands in pockets |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Simple | Double | Multiple | Few | Uniform | Multiple | Simple | Complex | |
| First | 45% | √ | √ | √ | | √ | | √ | | |
| Second | 30% | √ | √ | | √ | √ | | √ | √ | |
| Third | 20% | √ | √ | √ | | | √ | √ | √ | |
| Fourth | 5% | √ | √ | | √ | | √ | √ | √ | √ |

Tab. 1: Classification of Skin panels by Michaud [11].

The first family represents 45% of our industrial partner's chemically manufactured parts and gathers parts counting multiple lightening pockets of uniform depth, such as simple curvature skin panels (found in the central fuselage) and double curvature skin panels (found in the after and forward fuselage). The second family represents 30% of the volume of manufactured parts and is composed of parts with few pockets, but more complex contours. The third family represents 20% of the total volume, and groups parts having lightening pockets of multiple depths. Lastly, the fourth family contains parts with island-shaped pockets and accounts for only 5% of the total volume. Our case study focuses on the first family since it represents the largest group.

During the time we worked with our industrial partner in 2003, the design work was done using the CAD tool CATIA V4 in order to construct a tridimensional digital mock-up of an aircraft. Designing the lightening pockets is a rather manual process. In fact, the designer himself carries out all the low level operations in order to obtain a digital model representing the skin panel with its lightening pockets. For a regional jet, we estimated the total number of pockets in both simple and double curvature skin panels covering the central, after, and forward fuselage, as well as the wings and horizontal and vertical tail units, to be more than 2500.

The principle of pocket modeling consists of creating an extrusion, the contour of which corresponds to the pocket's geometry, which is then extracted from the volume corresponding to the skin panel. We broke down the process into fourteen general steps : (1) the designer loads the SPA into the computer's memory, (2) he identifies the structural elements surrounding the pocket to be created, (3) he extracts the limit lines from the stringers and frames lying on the IML surface, (4) he transfers the limit lines to the skin panel model, (5) he projects the limit lines onto the IML surface, (6) he shifts the limit lines, (7) he proceeds with a "relimitation" of the limit lines in order to obtain the rectangular geometry of the pocket's contour, (8) he defines the fillets of the four corners of the pocket's rectangle, (9) he uses the IML surfaces to shift what will delimit the bottom of the pocket, (10) using the previously defined contour, he creates an extrusion going beyond the IML on one side and the OML on the other, (11) he trims the extrusion created previously using the surface defining the pocket's bottom, (12) he creates the fillets on the boundary of the extrusion on the OML side, (13) he assembles the extrusion representing the pocket to the other pockets in the model, (14) he subtracts the extrusions from the skin panel in order to obtain the lightening pockets.

These steps allow for a better understanding of the work carried out by the designer using CATIA V4 in order to obtain the skin panel with its lightening pockets. In the case of a regional jet, more than 35,000 low-level, repetitive and tedious operations are involved in creating the 2500 pockets. This trade-oriented work could be delegated to a computer by means of a Customization tool, such as a UDF, or by a trade-oriented application which could bring partial or complete Automation of the process, thus freeing the designer.

The fourteen-step process is subject to rules governing the design of the pockets. They are rather simple and can vary from one manufacturer to another. We were able to document them during our collaboration with our industrial partner, but before detailing them, we will use the nomenclature described in figure 2 in order to situate the different geometrical elements that make up the pockets.

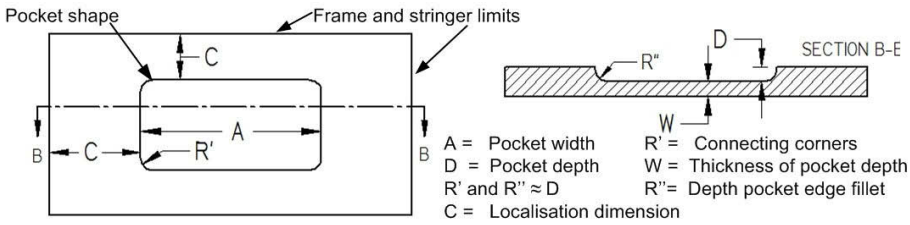


Fig. 2: Nomenclature of lightening pocket design parameters.

In figure 2, the lines outlining the rectangle that surrounds the pocket represent the limits formed by the stringers and frames. Label C, "Localization dimension", is the distance required between a structural element and the beginning of the pocket. "Pocket depth", labeled D, represents the quantity of metal removed during the chemical machining of the skin panel. Label W, "Thickness of pocket depth", corresponds to the metal remaining after machining. "Connecting corners" and "Depth pocket edge fillet" are similar to pocket depth.

There are a few simple rules the designer must apply to create a pocket that should be emphasized:

1. The localization dimension defines the outline of the pocket in reference to the parts that surround it; this dimension is about 0.100" ;
2. The radius of the connecting corners is about 0.5" ;
3. The pocket depth is a function of a part of its thickness and corresponds to the metal removed during the chemical machining; we will consider that pocket depth to be about 1/3 of the skin thickness.

## 3. TERMINOLOGY

Before we define the fundamental concepts, we will go over the terminology that is associated with those concepts and which is used throughout this article. One of the concepts is the "feature". We use the definition provided by Deneux [3] and that is inspired by the one proposed by FEMEX , "Feature Modeling Experts Working Group" : a feature is an information unit which describes an aggregation of properties of a product model relevant to a specific view of the product. In this paper, we adopt the design point of view.

We also adopt some definitions proposed by Tremblay et al. [13] in their work on change propagation in a PLM context, as well as the essence of some definitions from Giguère et al. [4;5].

| | |
|---|---|
| Task : | A task is defined as a part of a job to be done [13]. |
| Technical Object: | The term technical object encompasses, in a non-exhaustive way, geometric elements, features, parts, assemblies, documents, articles, functions, tasks and containers [13]. |
| Know-How: | The know-how is a set of knowledge related to an association that links objects. Know-how corresponding to a specialized task will be formalized in a link that associates two technical objects [13]. |
| Association: | Essentially, an association indicates the existence of a dependency between two technical objects [13]. Links and constraints are different classes of associations. |
| Link: | A link defines a formal dependency between different technical objects. It represents know-how relative to a task which uses a procedural language or a similar artificial intelligence. A link can be established inside a CAD tool (using knowledge management modules or UDFs as in CATIA V5) or outside the tool by using APIs [13]. |
| Constraint: | A constraint represents a specific formal and indecomposable dependency between technical objects. They are terminal level associations [13]. Example: an offset between two planes. |
| Persistent Association: | A persistent association's main feature, with respect to time, is its permanence. Only destroying or modifying the association can end the dependency between the different technical objects which it associates. A persistent association eases change propagation when there is a modification. For example, a dimensioning constraint permanently established between two lines maintains the geometrical constraint when one of the two lines is modified [13]. |
| Derivation link: | A specialized link which expresses a unidirectional dependency between a target feature and a reference feature. It plays an essential role in change propagation [4;5]. |
| Imprint: | Information specific to a reference feature determined by extraction from a group of reference parts [4;5]. |
| Pseudo-Imprint: | Information specific to a target feature. It is automatically determined by the application of a design knowledge (encapsulated in the derivation link) to an imprint extracted from the design context [4;5]. |
| Customization: | Means of using CAD tools to encapsulate task-specific knowledge in order to re-use it. For instance, UDF as defined with Catia V5 and NX are customization tools. |
| Automation: | Means of defining and using procedural programs that encapsulate trade-oriented knowledge to reduce the amount of interactions required from a user to achieve a modeling task. For instance, automation is generally achieved using APIs from the CAD system. |

## 4. FUNDAMENTAL CONCEPTS

One of the fundamental concepts used in this work is the typification of entities. Object typification allows associating methods attributing specific behavior to a technical object or a group of technical objects in a given context. In our case, the structural parts, such as the stringers, frames, and the pocket "feature" are typified in order to give them meaning in the design context, and in order to carry out operations using them in a design task.

Two other fundamental concepts of our work are the imprint and the pseudo-imprint. A feature needs to interact with its context according to the requirements of the task to be carried out, in order to retrieve information from the context and to use it. Thus, the imprint of a feature is information pertaining to reference features determined by an extraction process from a set of reference parts. Conversely, the pseudo-imprint is information pertaining to a target feature. It is automatically determined by the application of design know-how (associated to a derivation link) to the imprint extracted from the design context. In a CAD context, the imprint and pseudo-imprint will usually be geometrical information. Giguère used these concepts for the first time in 2001 during the development of an application of insertion and automatic updating of the joggles and cutouts in a frame where it intersects with a stringer. His results clearly show that the time-gain potential offered by these concepts is more than 90% [4;5].

The application process of these two concepts is broken down into three distinct steps. The first step consists of extracting the imprint from a reference feature or group of features. In the second step, engineering know-how is applied to the imprint to derive the pseudo-imprint. Finally, the pseudo-imprint is applied to a target feature. The reference feature, imprint, derivation link, pseudo-imprint and the target feature form a group that Giguère calls a "meta-feature".

Giguère's application of imprint and pseudo-imprint concepts used a reference feature, the imprint of which was defined from a single part, the stringer. He applied joggle and cutout design rules to the imprint, thus creating two pseudo-imprints. Each pseudo-imprint is linked to a target feature and contains either the joggle or cutout geometry and applies to only one part, the frame. The derivation link, linking the reference feature to the target feature is unidirectional. The information is unidirectional from the reference to the target. Moreover, this derivation link is persistent, in that during a change made on a stringer (either to its shape or position), the propagation is made automatically and updates the pseudo-imprint applied to the target feature. Figure 3 a), illustrates the derivation link in Giguère's application case.

In the case of lightening pockets, the imprint is jointly determined by four parts and four reference features: the two frames and the two stringers bordering the pocket to be created and their limits. The derivation link is formalized by the pocket design rules that are applied to the imprint to obtain the pseudo-imprint. Then the pseudo-imprint is applied to a single target feature, that is, the pocket to be created in the skin panel. Figure 3 b), shows the three design application steps in the creation of a lightening pocket.

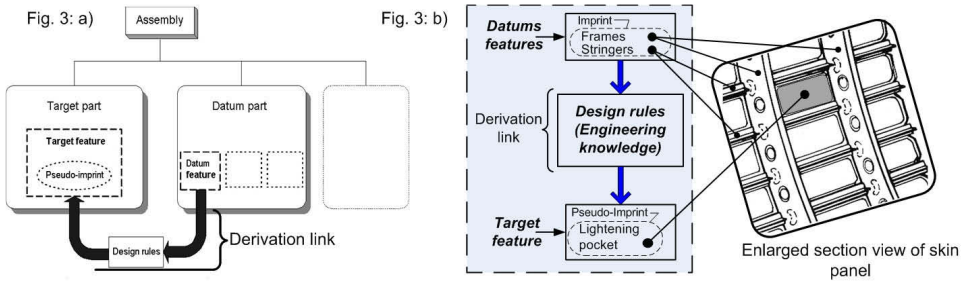


Fig. 3: a) Derivation link in Giguère's joggle and cutout application case [4;5],
Fig. 3: b) Derivation link between frames and stringers and a lightening pocket as a target feature.

Lastly, we have the concept of the User Defined Feature, or UDF. Hoffmann and Joan-Arinyo [7] distinguish two stages in the UDF mechanism. The first is its definition and corresponds to the creation process of the feature prototype. The second stage corresponds to the use of the UDF by instantiating it in an adequate context.

The use of UDF allows, amongst other things, the customization of a modeling task by encapsulating some task-specific knowledge (UDF prototype creation step). This specific knowledge is next re-used by instantiating the UDF. Another advantage related to the use of UDF is to reduce the number of geometrical elements that encumber the specification tree. Figure 4 shows the general structure of a UDF prototype.

A UDF can be seen as a black box, into which inputs are entered (in figure 4, $I_1$, $I_2$ and $I_3$). These inputs are fed with low-level technical objects obtained from the design context (can be a parameter value, such as a distance, or

geometrical elements, such as a plane or a surface, etc.). These entities can be found within the current CAD model itself, within models of an assembly, or within another external model. These entries are then managed by the UDF, the process of which consists of "replaying" some modeling steps using the supplied entries (such as offsetting an input plane, etc.).

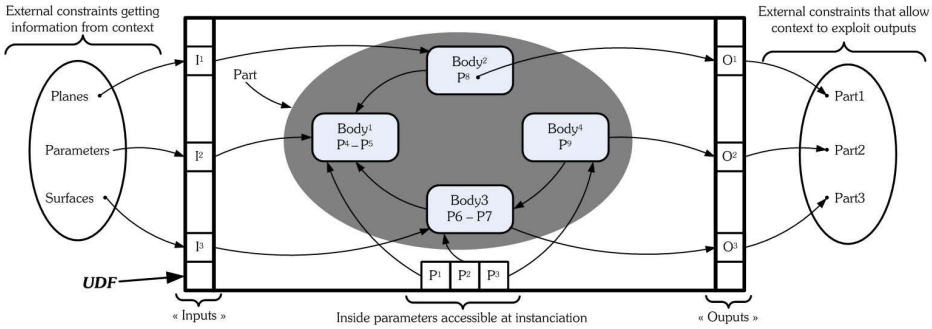


Fig. 4: Structure of a UDF prototype (User-Defined Feature).

The outputs ($O_1$, $O_2$ and $O_3$ in figure 4) of the UDF are defined at the time the UDF prototype is created, and correspond to the UDF's geometrical result, for example, a prism or a cutout. It is also possible to add outputs to a UDF, such as a plane or surface, so that they can be utilized in the instantiation context.

Parameters can be "published" by the UDF prototype so as to control the behavior of the UDF instance. These parameters feed the UDF's internal constraints and are independent from the geometrical context (as is the case of $P_1$, $P_2$ and $P_3$ in figure 4). A UDF, some of its outputs or its published parameters, can also become entries for another UDF, creating a UDF chain of sorts. One of the strategies of our work is based on this principal.

The UDF concept materializes in modern CAD tools and in certain platforms allowing the creation of features defined by the user to respond to specific modeling needs. That is the case of CATIA V5, NX, and SolidEdge as well, which has a similar tool called "System Library Document".

Considered globally, we would say that Customization, by means of UDF, aggregates low-level associations (constraints) that would otherwise be established manually between low-level geometrical entities, so as to let the user interactively instantiate these higher level associations (links). This Customization tool (UDF) is next used as a building block to define the Automation program that decomposes the parts into features and into geometric entities that form the imprint needed by the UDF.

The Automation, achieved by a VB program, basically automates the work of instantiating the UDF. Automation can be partial or complete. Partial automation refers to the case where the user has to interact with the application in order to select the parts (such as a stringer) that contribute to define the reference feature imprint, while the program takes care of decomposing the parts to identify the low-level geometrical entities (such as a plane) needed to feed the UDF instances. Complete automation refers to the case where the user interaction is limited to designating the assembly on which the program will take action; in this case, the program decomposes the assembly into parts, and decomposes parts into low-level geometrical entities needed to feed the UDF instances.

The UDF is therefore a customization and automation building block. It allows us to create a pocket feature that encapsulates some task-specific design know-how that can be re-used. The designer or a program can then instantiate the pocket feature where appropriate.

## 5. THE APPLICATION PROTOTYPE

The application prototype development's main objectives are to validate the concepts described above and to help measure the time-gains. The first step in this customization/automation project is to identify the different parts, features, geometrical elements and parameters involved in order to structure their associations. In our specific case study, inserting lightening pockets in a SPA requires structuring the hierarchy of associations between pieces of information

defining the stringers, frames, skin panel, and lightening pockets. For example, defining a frame requires considering its shape type (C or T profile, etc.), the OML surface controlling the shape of the frame in contact with the skin panel, the F.S. which positions the frame in the mock-up, and finally, the skin panel thickness for positioning the frame on the IML. Studying the hierarchy of associations between the parts, features, parameters, and geometrical elements of this case study led us to define the interdependence graph shown in figure 5.

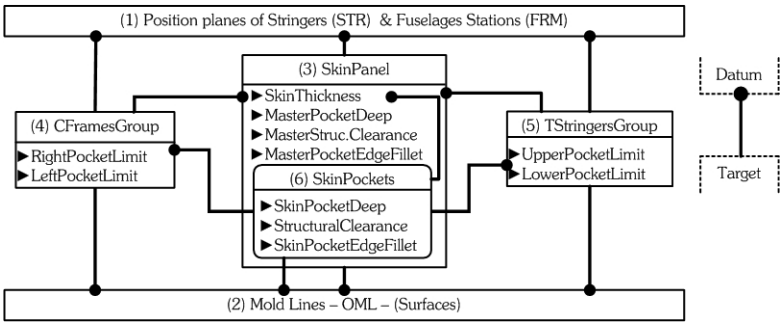


Fig. 5: Interdependence graph of the parts, features, geometric elements and parameters.

This graph defines the order according to which the parts should be included in the assembly, as well as the necessary accesses to external information for each part, geometric element and parameter. The blocks called "Position planes of Stringers (STR) & Fuselage Stations (FRM)" and "Mold Lines-OML-(Surfaces)" are the models that must appear at the top of the assembly hierarchy, because they do not depend on any upstream part. The model containing the "Skin panel" can follow, because its definition only depends on models already present in the assembly. Next, the stringers and frames are positioned and associated to previous models as needed. Finally, each pocket, "SkinPockets" in the graph, is instantiated in the skin panel model and associated to 4 out of the 5 parts already existing in the assembly. The interdependence graph is thus a map describing the set of associations that are the very engine of the assembly model's dynamism. Besides, constructing this graph helps avoid cyclical references. Considered globally, this graph is useful both to achieve customization and automation.

### 5.1 Customization Prototype with UDFs

For the development of the customization prototype, we used CATIA V5, foremost because it is widely used in aeronautics and is used by our industrial partner. The next development step was to define the methodologies that would be used to interactively design the stringers, frames, and lightening pockets in order to create the lightening pocket UDF prototype. The design method that was developed under CATIA V5 for a lightening pocket needs six repetitive steps, instead of the 13 repetitive steps under CATIA V4. The defined methodology is valid both with single and double curvature OML surfaces, as required to support all the needs of the first skin-panels family defined by Michaud [11]. The UDF was next created based on this interactive modeling methodology. We finally built a typical SPA containing 9 stringers and 11 frames, which require a total of 80 lightening pocket UDF instances. It should be noted that customization was facilitated by the availability of persistent associations in CATIA V5, as well as because of the existence of the UDF construction tool available with this platform.

In this prototype, customization is achieved by encapsulating task-specific knowledge within UDF that aggregate low-level associations (*constraints* as per Tremblay et al. [13]) between parameters and low-level geometrical objects, such as the planes that express the frames and stringers limits that define the imprint of the pocket design context. Actually, UDF allow the designer to manipulate multiple constraints as a single Derivation link (as per Giguère et al. [4;5]) and facilitate task-specific knowledge re-use. Overall, this work shows that the concepts help organize the customization work using UDFs. Moreover, a UDF is used next to develop the automation prototype.

### 5.2 Automation Prototype with a VB Program

From a functional point of view, there were two goals for the automation prototype. First, it had to assist the designer, step by step, in the insertion of a pocket into a skin panel, and second, it had to automatically insert (without user interaction) all the lightening pockets that could fit into a skin panel at each quadrilateral bordered by two stringers and two frames. The development of an automation program could have been carried out using various tools, such as

Scripts, Visual Basic and C++ in order to query and manipulate CATIA V5 objects. Our prototype basically exploits Visual Basic APIs (Application Programming Interface) available from CATIA V5.

The third step consisted of programming a partial automation application that could automatically instantiate the pocket feature, defined as an UDF, into the SPA at a specific quadrilateral indicated by the user who selects two stringers and two frames. The application then searches the assembly for an OML surface and a thickness parameter. The program instantiates the pocket UDF by feeding all its inputs with the required low-level geometrical elements found in the parts selected by the user. Indeed, the partial automation prototype relieves the user from decomposing the parts (e.g. stringer) into features (e.g. Stringer Lower Limit) and features into low-level geometrical elements (e.g. a plane).

The final step consisted of programming a complete automation application that could automatically instantiate all the lightening pockets that could fit into a skin panel. In this case, the user chooses to insert all the pockets. The application searches for pockets that could be already present in the SPA and asks the user if he wants to keep or delete them. The application then searches for the frames and the stringers (typified objects) in order to compute the total number of pockets to be inserted. Finally, the application validates this number with the user and starts inserting the pockets into the SPA without any further user interaction. The complete-automation prototype relieves the user from selecting all the parts that define the design context of all pockets, and thus automatically carries out the work of decomposing the assembly (the SPA) into parts (e.g. stringer).

## 6. RESULTS OBTAINED WITH THE CUSTOMIZATION AND THE AUTOMATION PROTOTYPES

After customizing and automating the insertion of lightening pockets, we performed some performance tests. First, we measured the time required to insert the 80 pockets into the SPA with the help of the UDF tool (customization). Then we measured the time spent by the completely automatic application to carry out the same task. The results can be compared with those observed in industry.

Interviews conducted in the industrial environment reveal that about half a day of work is spent manipulating low level geometrical elements using CATIA V4, to interactively define all the 60 pockets needed in a sample skin panel belonging to the first family defined by Michaud [11]. Based on this estimate, the time required for a designer to model and insert a pocket into an SPA similar to ours is about 4 minutes per pocket, or 320 minutes for 80 pockets. This time applies for simple, rectangular-shape pockets. It should be noted that the time needed for modeling, under Catia V4, a complex lightening pocket driven by several structure elements could reach up to 45 minutes. This type of complexity is mainly found in the second and third skin panel families defined by Michaud [11].

By using the customization prototype, we needed 45 minutes to insert 80 pockets into the SPA. Using the complete-automation prototype we needed 4.7 minutes to carry out the same modeling task (table 2). By comparing the results, we conclude that customizing CATIA V5 by aggregating persistent constraints into UDFs to perform a specific task yields a time-gain of 86% (i.e., 275 min, or 4h and 35 min.).

| | Catia V4 | Catia V5 | |
|---|---|---|---|
| | Manual work (min.) | Customization (min.) | Complete Automation(min.) |
| Insertion of 80 pockets | 320 | 45 | 4.7 |
| Time-gains from Customization | 86% | - | - |
| Time-gains from Complete Automation | 98.5% | 90% | - |

Tab. 2: Results obtained with Customization and Complete Automation.

The additional gain achieved with the complete-automation prototype is 90% over the time measured with Customization. However, this savings represents only 40 minutes. The significance of these numbers is better seen when transposed to an aircraft development program. As a rough estimate, assuming that most pockets in the first skin panel family can be handled by the automation prototype, that this family accounts for 45% of an airplane skin panels (same as the 'production volume') and that a typical regional jet has 2500 pockets, we could say that complete automation could save, in this case (45% x 2500 pockets x 4 min./pocket x 98.5% reduction), 4433 minutes (73h and 53 min.) on an aircraft program. This potential saving appears rather limited when compared to the 20 days of work or so required to develop the complete-automation program. On the other hand, the potential saving brought by

Customization would be 3870 minutes (86%, thus 64h and 30 min.), which is significant when compared with the few hours needed to prepare the UDF used here as a customization tool.

## 7. CONCLUSION

We made one further step in the validation of concepts of entity typification, associations, imprints and pseudo-imprints by applying them in the industrial context of modeling the lightening pockets located in the skin panels of aircraft fuselages. These concepts were materialized through UDF definition. Customization was compared to partial and to complete automation. Customization was performed by defining UDF prototypes that actually encapsulate task-specific knowledge that is re-used every time the UDF is instantiated. Conceptually, a UDF is a specific aggregation of constraints (low-level associations) established between low-level geometrical objects, so as to transform an imprint into a pseudo-imprint that belong to a target feature. Once defined, this aggregation (a link) is manipulated as a single entity by the designer. This simplifies and accelerates his work, but the various low-level technical objects that feed this link still have to be selected manually from the context at instantiation. Automation goes further than customization by performing UDF instantiation work with limited human intervention. Conceptually, such automation decomposes the typified high-level technical objects, such as an assembly, into a series of meaningful low-level technical objects that feed the UDF. Moreover, changes brought to the assembly can be automatically propagated to impacted features of the dynamic product model using inter-features associations. The time-gains achieved here by the customization of CAD tools, over the traditional method, is 86%, which is significant when compared with the magnitude of the implementation effort it requires. The additional improvement brought by complete automation is however outweighed, in our case, by the effort required to develop the application.

## 8. REFERENCES


[1] Bidarra, R.; Bronsvoort, W. F.: Validity maintenance of semantic feature models, Proceedings of the Symposium on Solid Modeling and Applications, 1999, 85-96.

[2] Bidarra, R.; Idri, A.; Noort, A.; Bronsvoort, W.: Declarative user-defined feature classes, Proceedings of the ASME Design Engineering Technical Conferences, Atlanta, USA, 1998.

[3] Deneux, D.; Wang, X. H.: A knowledge model for functional re-design, Engineering Applications of Artificial Intelligence, 13(1), 2000, 85-98.

[4] Giguère, F.; Rivest, L.; Desrochers, A.: Improving Design Productivity and Product Data Consistency; From Design Intent to Solid Models through Technological Links, Proceedings of the International IFIP Conference FEATS, 2001.

[5] Giguère, F.; Rivest, L.; Desrochers, A.; Maranzana, R.: Les caractéristiques contextuelles : une solution pour accroître la productivité en CAO. Revue Internationale de CFAO et d'Informatique Graphique., 17/1-2, 2001.

[6] Hoffmann, C. M.; Joan-Arinyo, R.: Erep -- an editable, high-level reprensentation for geometric design and analysis (P. Wilson, M. Wozny, M. Pratt ed.), North Holland: Geometric and Product Modeling, 1993.

[7] Hoffmann, C. M.; Joan-Arinyo, R.: On user-defined features, Computer-Aided Design, 30(5), 1998, 321-332.

[8] Holding, J.: Séminaire sur le Partage des données électroniques sur les produits: Solutions client-fournisseurInstitut des matériaux industriels du CNRC, Ministère de l'Industrie du Québec, 2002.

[9] Jenkinson, L. R.; Simpkin, P.; Rhodes, D.: Civil jet aircraft design, Reston, Va.: American Institute of Aeronautics and Astronautics. 1999.

[10] Ledermann, C.; Ermanni, P.; Hanske, C.; Kelm, R.; Wenzel, J.: Associative parametric CAE methods in the aircraft pre-design, Aerospace Science and Technology, 9(7), 2005, 641-651.

[11] Michaud, M.: Méthodologie de modélisation unifiée pièce-outillage en CAO aéronautique : application aux tôles et gabarits de découpe, Master Thesis, École de technologie supérieure, Montréal, 2004.

[12] Soors, P.: Dessin d'outillage en aéronautique. Outremont: Modulo, 1984.

[13] Tremblay, T. G.; Rivest, L.; Msaaf, O.; Maranzana, R.: The role of associations in CAD and PLM for handling change propagation during product development, 13th ISPE International Conference on Concurrent Engineering: Research and Applications, Antibes, France, 2006.

[14] Zimmermann, J. U.; Haasis, S.; Van Houten, F. J. A. M.: Applying universal linking of Engineering Objects in the automotive industry - Practical aspects, benefits, and prototypes.28th Design Automation Conference, Sep 29-Oct 2 2002, Montreal, Que., Canada, 2002

[15] Zimmermann, J. U.; Haasis, S.; Van Houten, F. J. A. M.: ULEO - Universal linking of engineering objects, CIRP Annals - Manufacturing Technology, 51(1), 2002, 99-102.