Links in CATIA

Julie Cyrenne, Dassault Systemes

Introduction

This article is the first of a series on the CATIA links. In this article, links are introduced and illustrated with the example of the design of a table. At each step of the design process, a link type is presented. The Five link types presented are the CCP link, the KWE link, the Instance link, the assembly constraint and the ViewLink.

Links Definition

A link is a dependency relation used to transmit geometric, parametric or positioning information between components. Links in CATIA are unidirectional and always point from the child to the parent. In other words, a child always knows what its parents are because this information is required to define its content. However, a parent doesn't know whether it has children, because children have no impact on the definition of the parent. This architecture allows for the re-use of parents in different context.

The update of a part only updates the content of the part itself (and not the elements copied from external files). To ensure that all modifications to the external files are reflected in the part, check the option 

in Tools/ Options/ Part Infrastructure/ General.

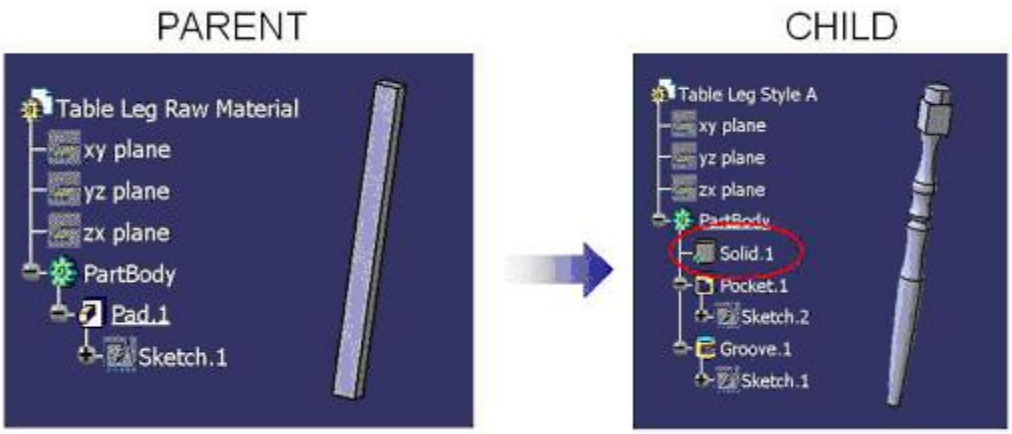
The Edit/ Links menu provides a list of all the external links of the part, if any, as well as the name and path of the pointed document and the status of the link.

Types of Links

Five types of links will be presented with the example of the design of a table: the CCP link, the KWE link, the Instance link, the constraint and the ViewLink.

- **The CCP link**

The first CATPart contains the geometrical definition of the raw material used to create the table leg. The second CATPart contains the geometrical definition of the finished table leg. Consistency must be ensured between the models: a modification of the raw material must be reflected in the finished leg. For that purpose, the raw material part geometry is copied and pasted in the finished part (with the link maintained), where the machining operations are performed. **Because the raw material and the finished leg will never be assembled together, this operation is performed outside the context of an assembly.**



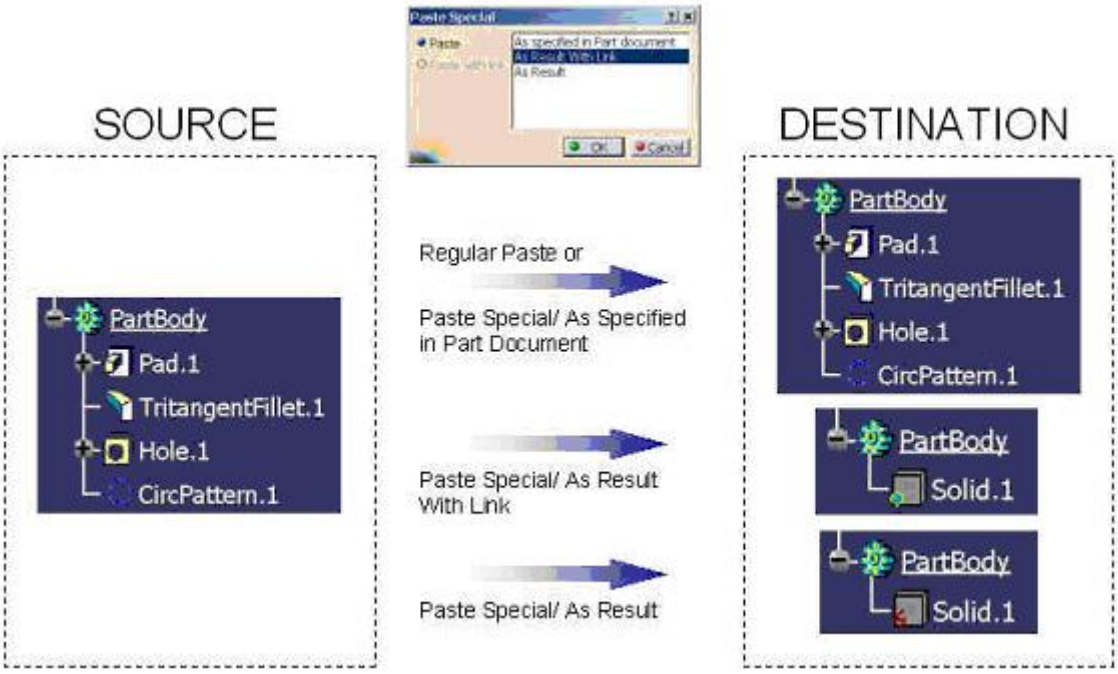
In this process, the raw material part is the parent: its definition is independent and is not affected by any modifications of the finished table leg. Therefore, no external links are created in the raw material part. On the other hand, the finished table leg is the child. Its geometrical definition is driven by the raw material part and will be affected by any modification of its geometry. This relation is represented by a CCP link in the finished table leg that points to the raw material.

The type of paste used is important in this operation. The figure below illustrates the three possible behaviors.

1/ If features are pasted using the regular paste or the paste special ‘as specified in the part document’, all the features are copied and no link is maintained to the source file.

2/ If features are pasted using the paste special ‘as result with link’, the geometry is copied (and not the features) and the link is maintained to the source file. This is the only type of copy that will create the CCP link.

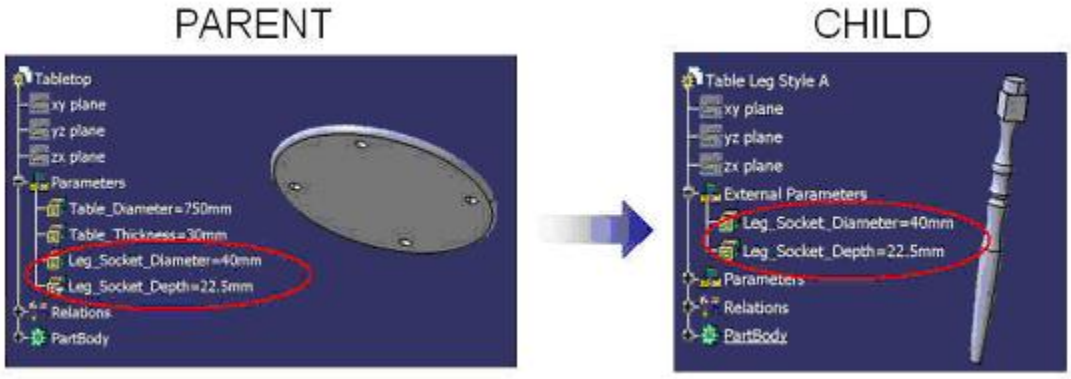
3/ If features are pasted using the paste special ‘as result’, the geometry is copied (and not the features) and no link is maintained to the source file.



The CCP link is created in a part where geometry is pasted using the 'Paste special as result with link' functionality if the copy/paste operation is performed from part to part **and outside the context of a product**.

- **The KWE link**

The leg socket is the common interface between the table leg and the tabletop. The dimensions of the socket are stored in parameters in the tabletop part. For the table leg to match the tabletop, those dimensions must match. For that purpose, the dimension can be copied (with the link maintained) from the tabletop to the table leg.

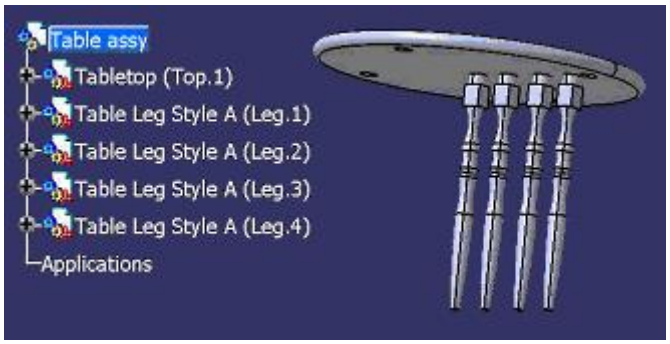


This link is in all points identical to the CCP link, except that parameters are copied instead of geometry.

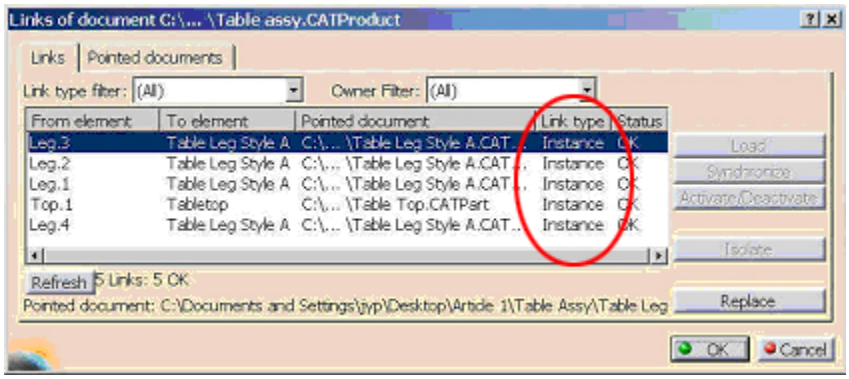
The KWE link is created in a part where parameters are pasted using the 'Paste special as result with link' functionality.

- **The Instance link**

To create a table assembly, the tabletop and table leg components must be inserted in the table assembly CATProduct.



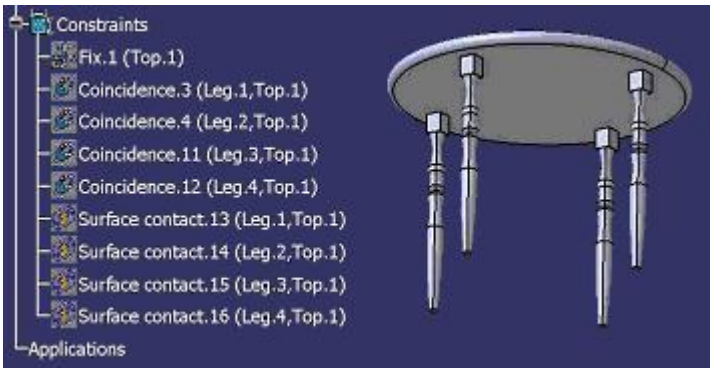
In this process, the table assembly is the parent and the tabletop and table legs are the children. In the table assembly, an instance link is created for each component. This link carries the name and path of the files.



The Instance link is created in a CATProduct where CATParts or CATProducts are inserted.

- **The Constraints**

Assembly constraints are used to position the parts of the assembly. Each leg will require an axis-to-axis coincidence and a surface-to-surface contact constraints to position it.



The **assembly constraints allow to position parts and products <u>in the context</u> of an assembly**. This type of link cannot be visualized in the Edit/ Links menu. The links are stored in the CATProduct and apply to the instance of the components. In other words, an assembly constraint applies to a specific representation of a part in a specific assembly. For example, if the Leg.1 instance is constrained, all the other table leg instances are still free.

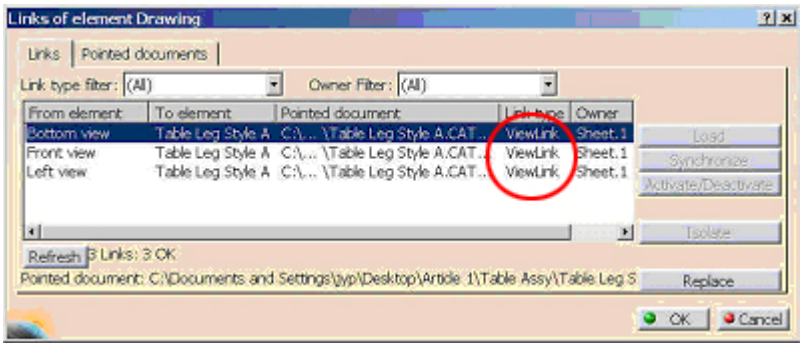
If the table leg style A was replaced by a table leg style B in the table assembly, all the constraints would be broken. The use of publications prevents this behavior. If the same publication name (Leg_Axis) is applied to both table legs' axis, the assembly constraint will be reconnected automatically when the table leg is replaced with the second one. Publication is a topic that will be detailed in a future article.

The constraint is created between two part/ product instances. The use of publications will allow to reconnect the constraints automatically when parts are replaced.

- **The ViewLink**

The table leg CATPart contains the 3D geometrical definition and a CATDrawing contains the associated 2D representation. The views are generated from the CATPart using Generative Drafting functionalities. Whenever the part is modified, the drawing views indicate an update is required to reflect the changes.

This process does not create any external link in the CATPart (the parent). In the CATDrawing (the child), a ViewLink points to the CATPart (the parent). The ViewLink link is created for each generative view of a drawing. The information carried by the link is the name and path of the source model used for the projection.



The ViewLink is created between in the CATDrawing for each generated view. The link point to the object used to create the view (CATPart or CATProduct)

Conclusion

The design of the table has illustrated five types of links: CCP, KWE, instance, assembly constraint and ViewLink. There are many other types of links, such as MML, Material, Shape, etc. The next article will present the ‘design in context’, which involves context and import links.

Links in CATIA - Part 2: Design in context

Julie Cyrenne, Dassault Systemes

Introduction

In the previous article, we saw how to create parts and assemble them in a product. However, interfacing parts are often designed using elements from one another to ensure they match perfectly. This is called design in context and will be explained in this article. For the purpose of illustration, we will use a simplified representation of an aircraft floor beam.

The Import Link

A CATProduct contains two CATParts, a parent and a child part. As discussed in the previous article, this means the CATProduct has two instance links pointing to the two parts.

The parent part does not require any external information to complete its geometrical definition. The Edit/ Links panel of this part would reveal no external links.

The child part utilizes a feature (wireframe or surface element) from the parent part to complete its geometrical definition. In the event where the child part is opened outside the context of the product, its geometry must be resolved. For that purpose, when an external feature from a parent is selected to define a child part, that external feature is copied with a link in the child part. The linked wireframe or surface geometry is stored in a geometrical set named 'External References'. The Edit/ Links panel would show two links:

1. An import link pointing to the parent file that provides the geometry.
2. A context link pointing to the product. This will be the object of the next article.

The same links would be created if a geometrical feature was copied from a parent part and pasted 'as result with link' in a child part in the context of a product.

What's the difference between the CCP link and the Import link?

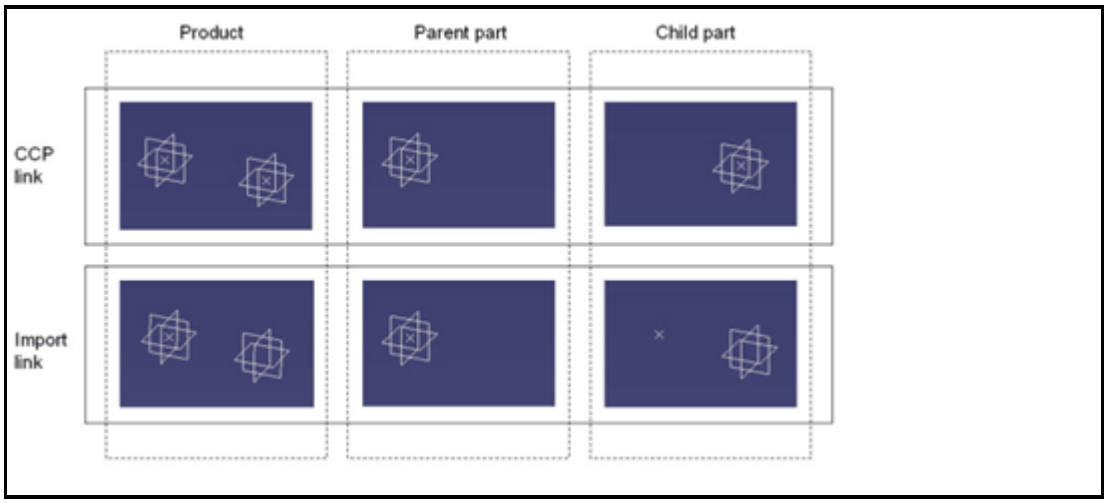
The previous article presented CCP links as the link resulting from the copy and paste special 'as result with link' of a geometrical feature between two parts *outside the context of a product*. Because it is outside the context of the product, the link is carried by the part reference and the positioning is not taken in account. Note that when a CCP link is created, it remains a CCP link when the two parts are inserted in a product.

In the image below, a point is created at coordinates (0,0,0) in the parent part. The point is copied and pasted 'as result with link' in the child part, opened in a second CATIA window. In the child part, the resulting linked point is also located at (0,0,0). The two CATParts are then inserted with their origins not coincident

in a CATProduct, and therefore, the points are not coincident.

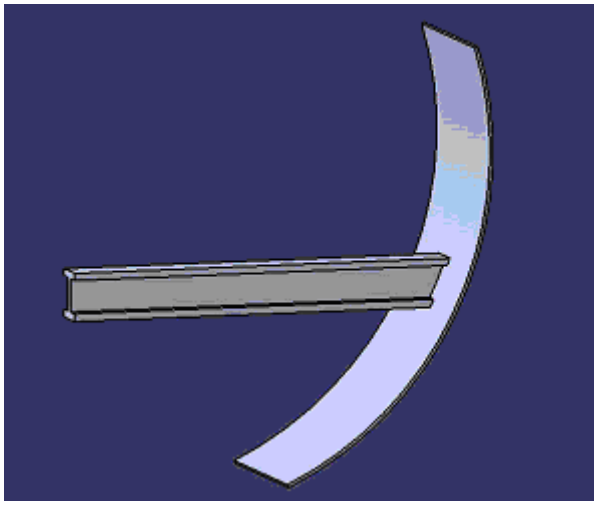
This article presents the **import link** as the link resulting from the copy and paste special 'as result with link' of a geometrical feature between two parts *in the context of a product*. **Because it is inside the context of a product, the link is carried by the part instance and the positioning is taken in account**.

In the image below, a CATProduct is created and two CATParts are inserted with different origins. A point is created at coordinates (0,0,0) in the parent part. The point is copied and pasted 'as result with link' in the child part. The child part's resulting linked point is coincident with the parent part's point. When the child part is opened in a new CATIA window, we can see that the point is located at (x,0,0).



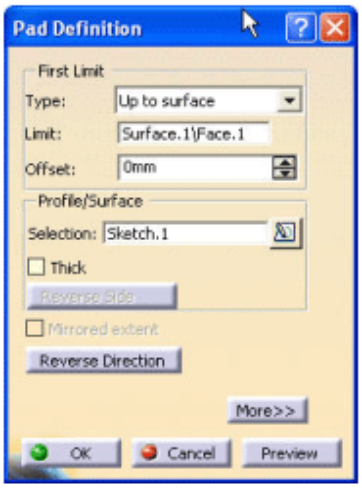
The Beam example

In our example, the CATProduct contains two CATParts. The parent part, Fuselage.CATPart, contains a portion of the fuselage skin. The child part, Beam.CATPart, contains an H-beam that is flat on one extremity and is limited by the inside surface of the fuselage on the other extremity.

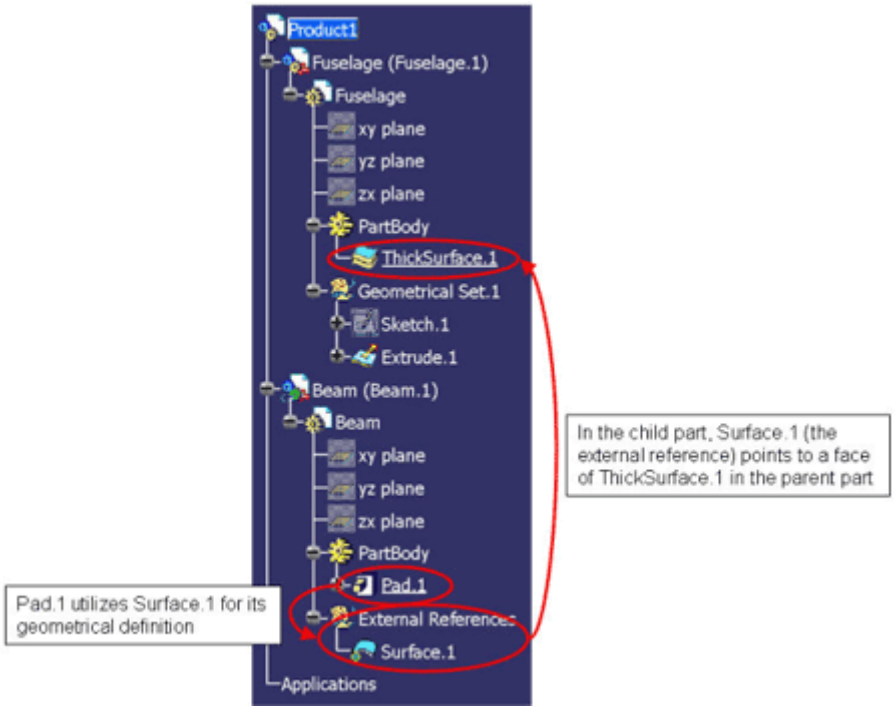


Fuselage.CATPart does not require any external reference to define its geometrical definition. Verification to the Edit/ Links menu confirms it has no external links.

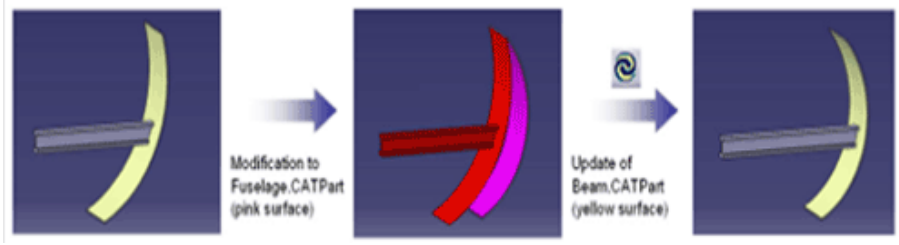
The beam's geometry is a Pad with an 'Up to surface' limit type. The selected limit is the inside wall of the fuselage skin.



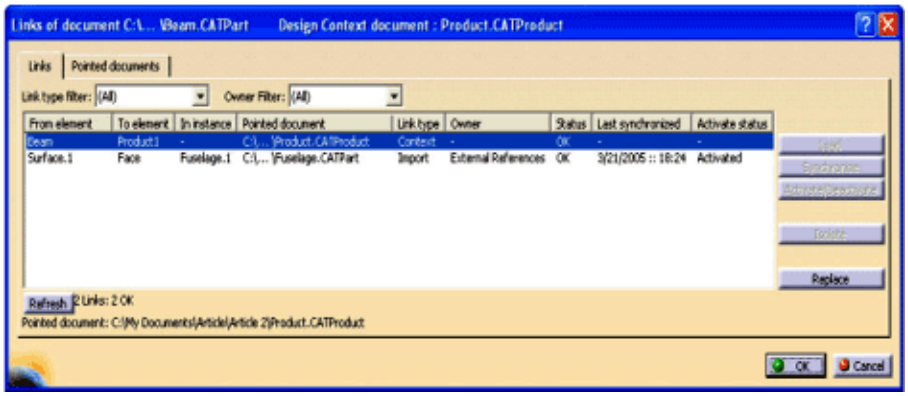
The inside surface of the fuselage wall is copied with a link in the 'External References' geometrical set of Beam.CATPart.



Because the surface is linked, any modification to the fuselage will be reflected in the beam geometry. In the image below, the fuselage geometry is modified. In the middle image, the beam and its external reference are red as they are not updated to the new fuselage definition (pink). After the update, the beam's external reference and the beam definition match the fuselage.

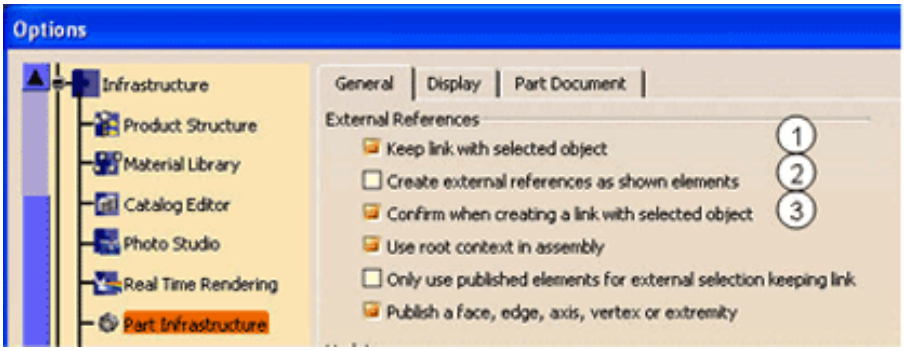


The Edit/ Links panel of Beam.CATPart shows a context link pointing to the product and an import link pointing to Fuselage.CATPart.



Setting the options for design in context

The menu Tools/ Options/ Infrastructure/ Part Infrastructure/ General tab contains the options pertaining to contextual design.



1. *Keep link with the selected object*

   If this option is not selected and a feature from a parent part is selected for contextual design, the feature is copied 'as result' in the child part. The feature will show the red lightning bolt characteristic of datum geometry. 

   A modification to the parent geometry will not be reflected in the child part.

   If the option is selected, the feature is copied 'as result with link' in the child part. The feature will show the green P or the green point, for published and non-published features, respectively (publication will be the subject of a subsequent article). 

   This way, the imported feature will be synchronized with its parent. **This is the <u>recommended configuration</u> for a true associative contextual design.**

2. *Create external references as shown elements*

   If this option is selected, any geometry copied in a child part in context will be created in show mode. This option has no impact on the part definition, so it's more question of personal preference and clutter in the workspace.

3. *Confirm when creating a link with selected object*

   If this option is selected, a dialog box will warn the designer every time an external link is created. This is useful to avoid unwanted links (e.g. mistakenly selecting the xy plane from a part whose origin is coincident with the active part would create a link with no added value). <u>The recommended configuration is to enable this option.</u>

Conclusion

The import link, described in this article is appropriate for design in context. The CCP link is similar, but because it does not take in account the positioning, it should not be used for design in context. The next article will present the context link, which is created along with the import link.

Tips and Techniques

Links in CATIA, Part 3: Context Link

Julie Cyrenne, Dassault Systemes

Introduction

In the previous article, we saw how to do design in context, where the child part (also known as contextual part) utilizes elements from the parent part to complete its geometrical definition. As a result, the parent part has no external link while the child part has 2 external link types: import and context. This article will explain when and how the context link is created. All the concepts will be illustrated using the example of a spline.

The Context Link

In CATIA V5, a context link is created when an import link is created in a CATPart. However, there may only be one context link per part, regardless of the number of import links. The context link points from a CATPart (the child) to a CATProduct (the parent).

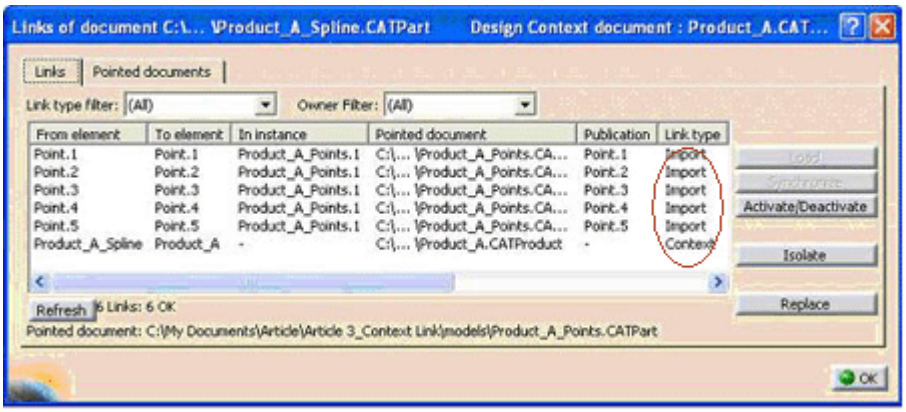


Figure 1: Edit / Links panel showing 5 import links but only one context link

The context is the product in which a part designed in context is associatively designed. In other words, if this product is opened, all the parent parts of the contextual part (part with import links) will be present and all the links will be resolved. You can also think of the context as the entity which needs to be opened for synchronizing links and updating a contextual part.

Choosing a context

In Tools/ Options/ Infrastructure/ Part Infrastructure/ General, there is one option associated with the context link: ‘Use root context in assembly’.

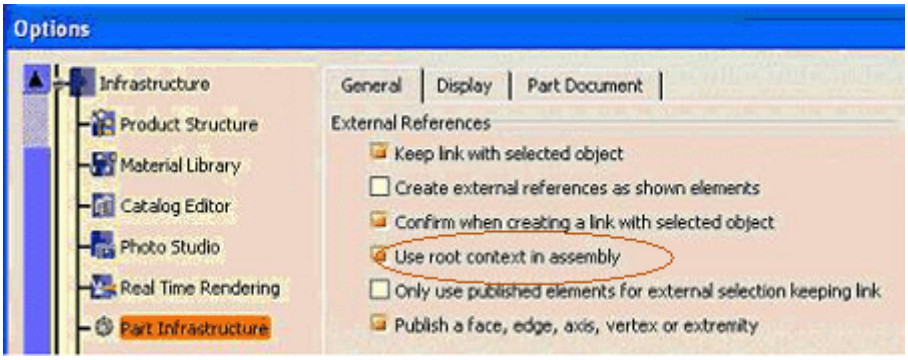


Figure 2: 'Use root context in assembly' option

When the option 'Use root context in assembly' is not selected, the minimum context is used. In other words, when creating a part's first import link, the first product that is a common parent of the two parts involved in the link becomes the context of the contextual part. The two left cases in Figure 3 below illustrate the result of using the minimum context.

When the option 'Use root context in assembly' is selected, the root product opened in session becomes the context of the child part. The right case in Figure 3 below illustrates the result of using the root context.

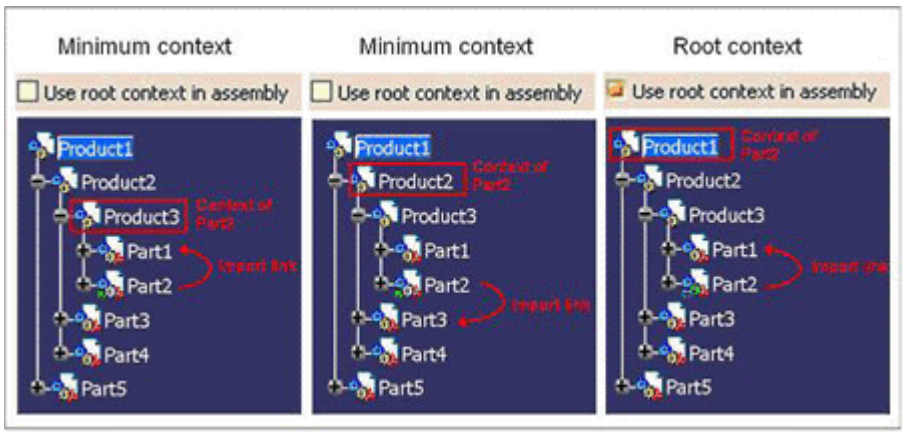


Figure 3: Impact of the 'Use root context in assembly' option

Once the context link is created, it does not change if the contextual part is opened in another configuration (unless explicitly changed by the designer as described in the 'Managing the context' section)

Considerations when choosing a context

1) It is impossible to create links with parts outside the context

In the example below, the Profile part contains a sketch of an I-beam profile. The Beam part is a pad using the Profile's sketch, hence creating an import link in the Beam part. The context link of the Beam points on Product2, as the 'use root context in assembly' was not selected. Later, if the designer wants to split the Beam with the Surface, he will get an error message (Figure 5 and Figure 6). Since the Surface part is outside Product2 (the context), it is impossible to create an import link from the Beam to the Surface.

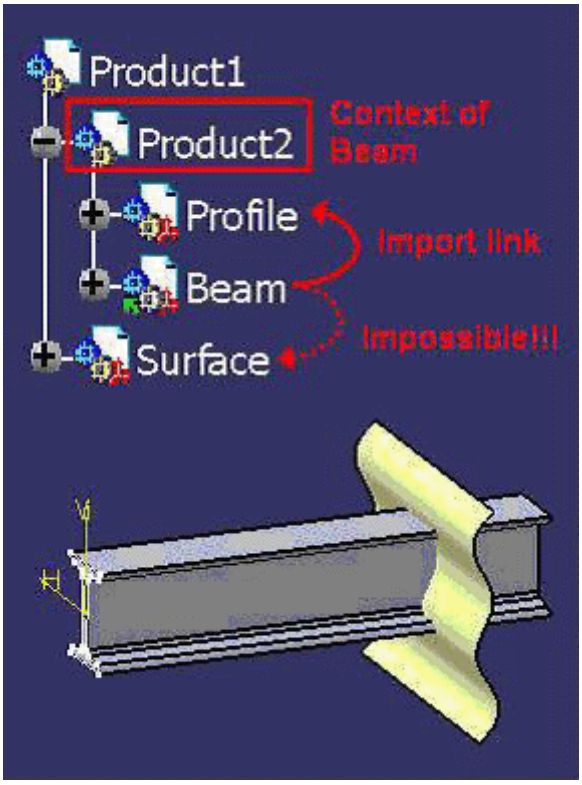


Figure 4: Example for 'impossible to create links outside the context'

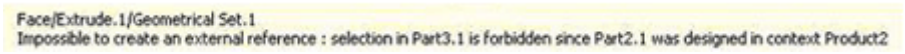


Figure 5: Pop-up message when trying to select an object outside the context for contextual design

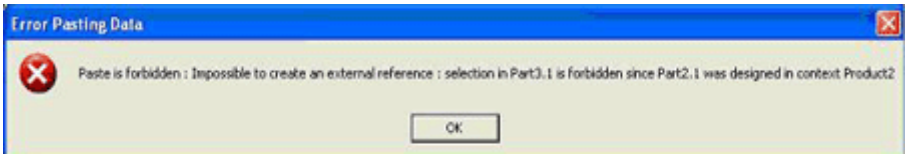


Figure 6: Error message when trying to paste with link an element from outside the context

2) It is impossible to update links if the context is not loaded in session

In the example below, the Cylinder is a pad defined up to the Surface part. The import link points from the Cylinder to the Surface. Since the option ‘use root context in assembly’ was active when the import link was created, the context link points from the Cylinder to the Product1. If a designer wants to work in Product2, any modification to the Surface will not be represented in the Cylinder because the context product is not in session (even though all the parts involved in the import link are loaded in session).

This example shows how, in a context of concurrent engineering, the root context should be used with caution!

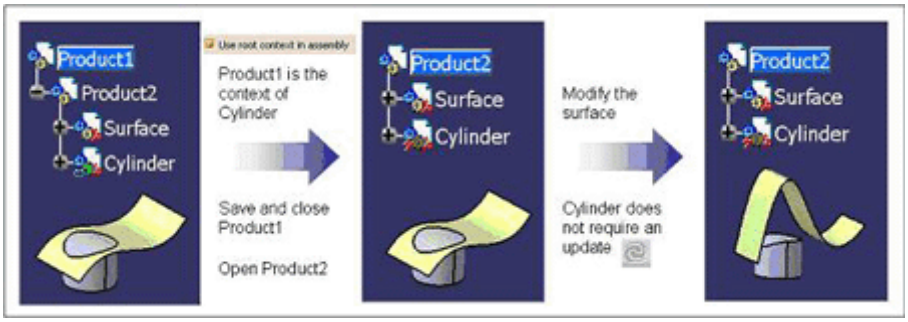


Figure 7: Modification and update of a contextual part outside its context

3) In specific use cases, it may be mandatory to work with a specific context

I don't want to enter into details as this is outside the scope of this article, but an example is when building a 'structure exposed' assembly in ENOVIA, the root context must be used,

Managing the context

It is possible to change the context of a part to a higher level or a lower level product.

Open Right-click on the part whose context should be changed, go to 'Components' and 'Define Contextual Links'. In the panel that appears, simply click on OK. The new context of the part will be the active product in session (regardless of the option 'use root context in assembly')

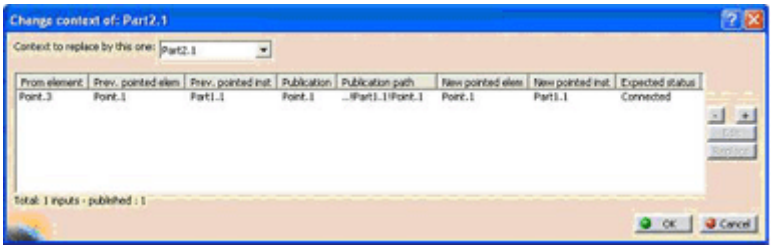


Figure 8: 'Define contextual links' panel

Icons

There are four CATPart icons to inform on the part's context. You may understand their purpose from the images in the examples. The next article will explain the meaning of each icon,

	Non-contextual part
	Definition contextual part
	Instance of the definition contextual part
	Instance of a contextual part

The spline example
Two designers are working in their separate assemblies, Product_A and Product_B.

In Product_A, the first CATPart, Points_A, contains 5 points that are published. The second CATPart, Spline_A, has import links to the 5 points as it uses them for the definition of the spline. Whether the option ‘use root context in assembly is enabled or not is irrelevant: because Product_A only has one level, the context link points to Product_A in either case.

Product_B is architected the same way.

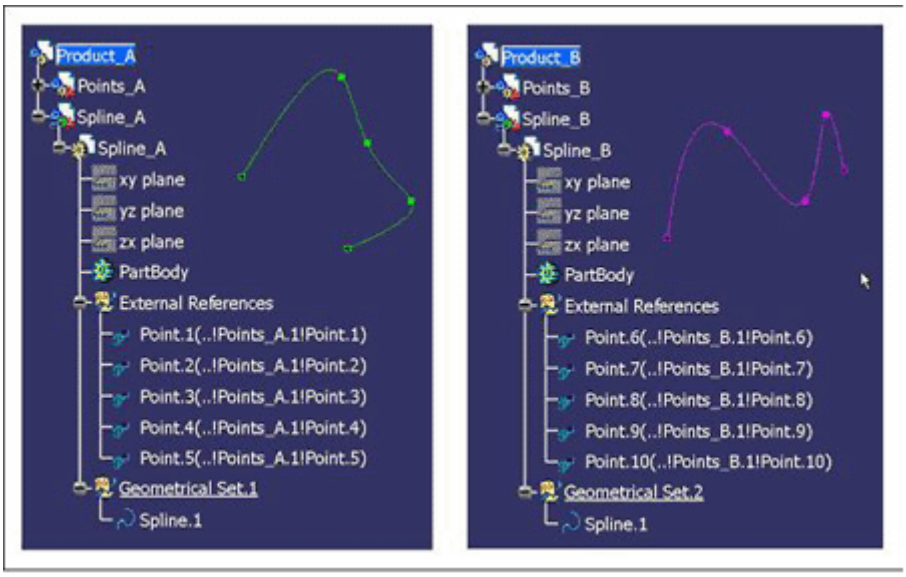


Figure 9: The spline example: designer's initial state

Once both designer’s work is done, the two products are inserted in a higher level product, Integration_A-B. The two splines must be connected by using the last point of Point_A as a point for Spline_B. The figure below shows how the software will not allow this operation.

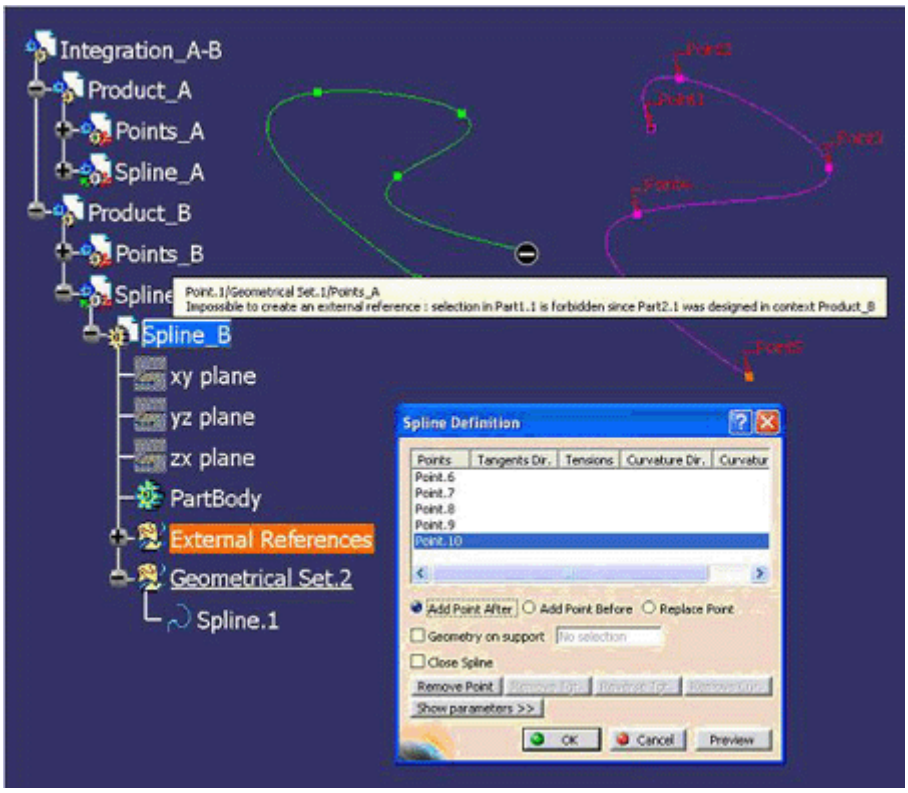


Figure 10: The spline example: Spline_B's context does not allow to import with link elements from Points_A

In order to enable this operation, the context of Spline_B should be changed from Product_B to Integration_A-B using 'Define Contextual Links'. Once the context is changed, Figure 12 shows how the operation will be allowed. The context of Spline_A does not need to be changed: the part does not have any external links outside Product_A to create.

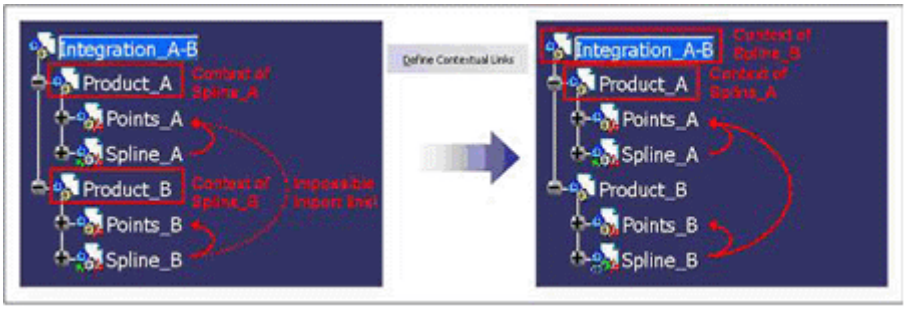


Figure 11: The spline example: changing the context of Spline_B

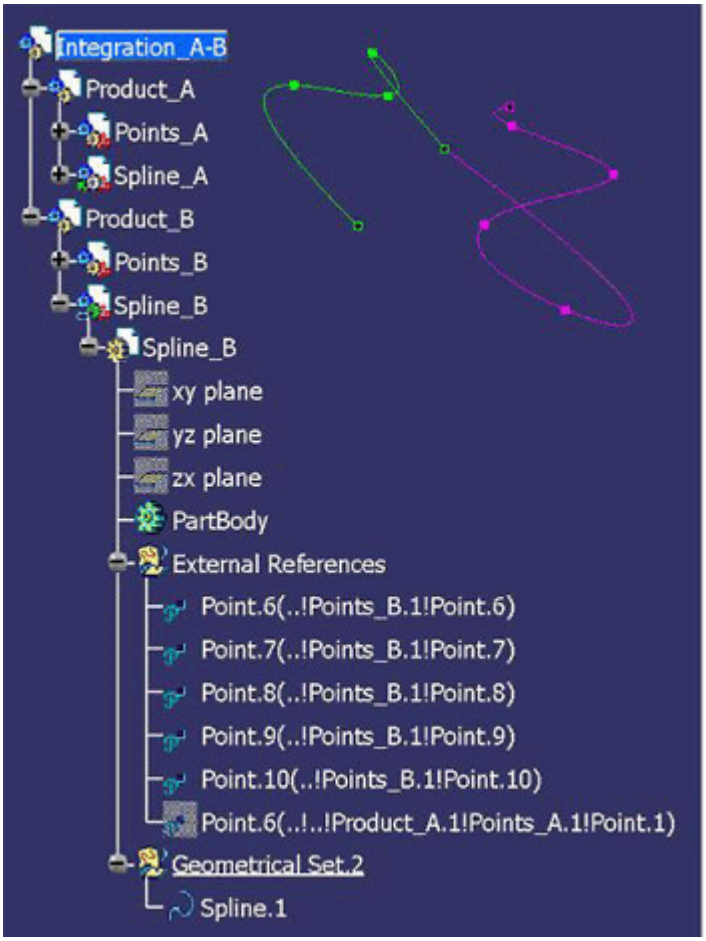


Figure 12: The spline example: Spline_B successfully imported with link an element form Points_A

Conclusion This article explained the context link and the means to manage it. There is no good or bad way: it depends on concurrent engineering, data architectures, etc.

One useful source of information to interpret the context is the CATPart icon. This will be explained in the next article.

Julie Cyrenne
Dassault Systemes