AVEVA MARINE

Hull Model Concept User Guide

AVEVA Marine Hull Model Concept

Contents **Page**

Hull Model Concept

1 General Concepts

1.1 General

Shipbuilding is an industry with well established traditions. These traditions are also associated with rather special concepts and a somewhat special vocabulary. The shipbuilding vocabulary is partly universal but there are also local *dialects* that may cause confusion in contacts between people with different backgrounds. One such dialect is the one used within AVEVA Marine and its documentation.

The purpose of this document is to describe how different universal shipbuilding terms are used in the AVEVA Hull application when there is a suspicion that misunderstandings may occur. Some Hull specific terms are also introduced.

1.2 Product Information Model

The term ***Product Information Model*** refers to the complete information about all the physical parts that constitute the final product and all the different associations between them. If the term is used without further specification the Product Information Model comprises the parts and relations from all the different AVEVA Marine applications, e.g. the hull application, the different branches of outfitting, assembly planning, etc.

A narrower view on the model is the ***Hull Product Model*** that deals with the part of the product that is created and developed by the Hull application. This narrow concept excludes from the actual Hull Product Model the associated information created e.g. in assembly and weld planning.

The Hull Product Model is divided into two parts, the (Hull) Design Model and the (Hull) Parts.

The ***Design Model*** is the direct result of the design activities in the AVEVA Hull or Structural Design applications and the outcome will be information organised in the Hull Design Structure (*see Model Structures*). Typical for the Design Model is that it is implicit, i.e. that the parts are not expanded with full geometry, etc. and that the model is stored in nominal dimensions, i.e. modification for production requirements has not been applied in the Design Model.

In less strict use of the words the concept *Hull Model* normally should be interpreted as the Hull Design Model.

The ***Hull Parts*** are the result of automatic processes (**"automatic part generation"**) when parts are extracted ("released") from the implicit model by applying rules, e.g. for marking, bevelling, shrinkage compensation, etc. Parts may be plates and profiles. Plates are created from sheet material, profiles from bars.

1.3 Model Object Types

The Design Model is organised in the Design Structure as described in a separate document *Model Structures*, including the object types building up the Design Model. Some of this information is duplicated below when considered especially important for a general understanding of Hull.

The ***block*** in the Design Model is normally not equal to an assembly (a node in the assembly structure). It should primarily be considered a geographically constrained container of panels, used to establish the Design Structure (for more details, see the document referred to above).

The ***panel*** is the central modelling unit when creating the hull structure, especially in the internal structure. Panels may be planar (*"plane panels"*) in the internal structure or with arbitrary shape in sculptured surfaces (called *"curved panels"* even if they should happen to be planar).

In normal shipbuilding terminology a panel is a big flat structure consisting of several plates with mostly parallel stiffeners, typically in the flat of bottom, flat of side or in big platforms or bulkheads. Such panels may be assembled in *panel lines.*

The AVEVA Marine concept of a panel is much more general and a panel may range in size and complexity from a small bracket-like structure consisting of a plate only to a complete deck with hundreds of parts (plates, brackets, stiffeners, pillars, etc.) and attributes of different kinds.

Special types of plane panels are knuckled panels (with their sub-panels) and panels brackets, see.

The Reference Surface Objects (RSOs) represent the main inner structures, normally generated in Initial Design Surface/Compartments, but also to a limited extent possible to generate in Structural Design. The RSOs can be used for two purposes when creating the Design Model:

- To be given steel properties and be used to automatically generate large panels
- To be a stable location and boundary reference for manually generated panels

1.3.1 Plane Panel Concepts

A panel is restricted by a number of ***boundaries*** that may be curves, planes, other panels, profile sections, etc. Within the panel each boundary is responsible for a certain portion of the outer contour of the panel. These parts are called the ***limits*** of the panel. The connection points between limits are called the ***corners*** of the panel (not all knuckle points are corners and there need not be a knuckle at a corner!).

The ordinary ***plates*** of a plane panel are located in the ***mould plane*** of the panel. A panel is associated with a local co-ordinate system (uvw) and the uv-plane is located in the mould plane of the panel.

Brackets are normally smaller structures consisting of at most one plate, optionally with stiffening of different types and with notches (see below) at the corners. They are used to connect and to support other elements. Brackets may be *type standard brackets or panel brackets*. AVEVA Marine has an advanced facility for customer set-up of a bracket standard.

Clips is the AVEVA Marine denomination for the small plate pieces in profile cutouts used to reinforce the connection between the plate and the penetrating profile and/or to tighten the cutout. It is the common concept for what otherwise may be called lugs and collars.

Holes are closed openings in the interior of panels and profile webs.

Cutout is the AVEVA Marine denomination for openings for penetrating profiles along plate edges (or in webs of profiles). Alternative names sometimes used (however, not in AVEVA Marine!) are "slot" and "scallop". Cutouts may be associated with clips (see above).

Notch is the AVEVA Marine name of smaller, normally standardised openings in plate corners or along edges of plates and profiles, e.g. to give access for welding. They may sometimes be called "ratholes". AVEVA Marine has a large number of in-built notch types.

Profile is the common AVEVA Marine name of pieces that are used as stiffeners or pillars. Profiles are always modelled as though they should be manufactured from bar material but they may in the end be fabricated from plates. Profiles may be *stiffeners, flanges or pillars.*

- ***Stiffeners*** are normally welded with one edge against the plate or bracket of the panel. Stiffeners may be straight, curved or straight-and-knuckled.
- ***Flanges*** are welded against an edge of a panel or bracket or in a hole, normally symmetrically. They are often outside AVEVA Marine called *face plates* or *face flats*. Welded flanges may be straight or curved.

  Folded (or bent) ***flanges*** are not any real profiles but created by bending the plate along an edge. Folded flanges can be set both in panels and brackets.
- ***Pillars*** are free profiles, normally welded at the ends only and supporting decks and platforms in open areas.

When generating profiles a number of attributes are essential:

- The ***profile type*** specifies the shape of the profile and the parameters, relevant for control of its size.
- The ***endcut type*** specifies how the profile should be prepared in its ends.
- The ***connection code*** specifies details about how a profile end should be positioned relative to the element it is attached to.

Swedging is the AVEVA Marine term for the type of small corrugation often replacing welded stiffeners in superstructures, etc.

Excess is the AVEVA Marine term for material added to compensate for inaccuracies in manufacturing and assembly. Other common names for this are *overmaterial, stock material or green material.*

Compensation is excess of triangular shape (that may be combined with the normal constant-size excess).

(Material) quality is the AVEVA Marine name for what otherwise is often called the *grade*, i.e. the specification of certain properties of the used material, e.g. normal steel, high tensile steel, etc. In AVEVA Marine the material quality is always associated with a density selected by the customer.

For further details about Hull standards, see *Survey of Design Standards in AVEVA Hull in Chapter Standards*.

1.3.2 Curved Panel Concepts

Some of the concepts described for plane panels are relevant also for curved panel (panels in sculptured surfaces). However, some are special as described in this section.

The stiffeners in curved panels are called ***shell stiffeners***. As described in the document *Model Structures*, shell profiles are organised in structures in two levels. Objects on the upper reference level collect a number of parts along the whole ship, identified by a number, e.g. the longitudinal number. This type of reference object is called a ***shell profile,*** containing in principle only the references to a number of *shell stiffeners*.

A shell profile is either a *longitudinal frame* (or ***longitudinal*** for short) if oriented in the longitudinal direction of a ship or a *transversal frame* (or ***transversal***) if oriented mainly in the transverse direction.

1.4 Parts

Part is the AVEVA Marine name of each piece that finally has to be manufactured. It is a common denomination of both ***plate parts*** and ***profile parts***. However, it may happen these names are used in a less strict way. Part is often used as a synonym of plate part and a profile part may be called only *profile*.

There are two systems for naming of parts as described in detail in the document *About Naming*. The AVEVA Marine ***name*** is the internal name by which a part is stored in the data banks. The ***part name*** is the production oriented name marked on the part in the workshop. From a technical point of view this is an attribute that is formed when the part name is asked for.

The part name is normally built up by a number of constituents, e.g. names of blocks and assemblies, separated by delimiters. One position of the name distinguishes the name from that of other parts going into the same assembly. This is the individual number of the part (or of a number of identical parts). This individual portion of the part name is in AVEVA Marine called the ***position number*** (otherwise often called the piece number). The *position number* may be a string.

Example:

A part name may look like this: A2N-C3-4. If A2N and C3 are names of assemblies on different levels 4 would normally be the position number.

1.4.1 Special Profile Attributes

For profiles it makes sense so talk about the *length* in a number of different ways:

- The ***moulded length*** is the distance along the trace between the end points of the profile in the design model
- The ***used length*** is the minimum length of the raw bar required to manufacture the profile. It considers endcutting, bevel gaps, shrinkage compensation, etc.

Curved shell stiffeners are normally stored with one or several *inverse bending curves*. These are related to a certain technique for checking of the curvature of shell profiles.

1.4.2 Special Curved Plate Concepts

A shell plate must be *developed* (subject to an inverse forming operation) so that the plate part can be cut in a proper shape from a planar plate. This process is called ***(shell) plate development*** and should not be confused with ***shell expansion*** which is a process, creating a traditional expanded drawing view of the shell and shell related structures.

Shell plates are developed using a technique where the plate is cut into ***strips*** that are triangulated and rolled out independently in both directions from a neutral plane of the plate, the ***base plane.*** The intersection between the base plane and the plate is called the ***base line.***

Two ***roll axes*** (a first and a secondary) are normally calculated for a developed plate. They indicate where the rolls should be applied in a first step of forming the plates (by rolling).

For check of the shape of a shell plate it is normal to use ***bending templates***. These may either be *physical templates* (e.g. made of wooden board) or they may be *adjustable* (and thus reusable) templates of different types.

Shell plates may be assembled to a curved panel on a berth called a ***jig***. A jig may either consist of **jig pillars** (normally in a fixed pattern) or ***jig templates,*** i.e. of plates fitting to the shell and arranged in parallel rows.

1.5 Miscellaneous

This paragraph explains some general concepts related to or used in Hull.

Dotori is the marketing name of an advanced facility for automatic selection of fillet bevel types and calculation of continuously varying bevel angles.

Genauigkeit is the marketing name of an advanced facility for addition of reference marks in the shape of small triangles or short lines ("ticks").

Generic format is in the Hull concept the denomination of a couple of ASCII formats for neutral export/import of part information, both plate parts, nested plates, profile parts and nested profiles.

2 Model Structures

2.1 General

A ship or another similar product is big and complicated and consists of a very large number of parts. This means that the information must be organised in one way or other to be able to handle and manage. It is also reasonable to assume that different functions within a yard have different views on it even if, at the end, the product and all its pieces are the same.

Some possible views of a ship are exemplified below (quite independently of how the information is organised in AVEVA Marine).

- In the early project phase the main characteristics (dimensions, general arrangement, compartmentation, cargo carrying capacity, etc.) are in focus.
- The general arrangement and the compartments define normally the position of the main structures of the ship, like decks, platforms and bulkheads. In the basic design phase the functional properties of these main members are of interest, e.g. their position, thickness of plates, pitch between and dimensions of stiffeners. These properties are often evaluated without regard to the final breakdown into blocks, assemblies and parts. Thus the view on the hull structure in this phase mainly *functionally oriented*.
- At the end the workshops must manufacture all the parts of the ship down to the smallest details and also plan and decide how they should be assembled. The production view of the product is thus preferably *assembly oriented*.
- The purpose of the detail design phase is to transform the functional design into parts and assemblies as required for production in such a way that the design intent from the functional design is protected. Thus the detail design view on the product can be considered as a mixture of a functional view and an assembly view.

The purpose of this document is to describe how different views on the hull product model are supported by AVEVA Hull against the general background outlined above.

It is beneficial if the reader of this document is familiar with the definitions and the vocabulary as explained in *General Concepts*.

2.2 AVEVA Hull Views on the Hull Model

AVEVA Hull supports two different views covering all the parts of the hull structure of a vessel

- The first view is design oriented and it is used when the product model is built up in AVEVA Marine.
- The second view is the assembly view that describes the break down of the product into its assemblies.

In addition, there is a third more functionally oriented view on certain parts of the shell members that goes across the other two views. It is described at the end of this paragraph.

2.2.1 Design View vs Assembly View

The hull design view contains only hull objects and is *created* entirely within Hull and Structural Design (but may be *accessed* by any AVEVA Marine application).

The assembly model, on the other hand, is common to all AVEVA Marine applications, i.e. it contains elements from both the hull model and e.g. the outfitting model in the same structure. It is normally established in a separate AVEVA Marine application, Assembly Planning.

The assembly model is built up as a mapping onto the design model, i.e. the design model comes first and contains most of the physical attributes and the assembly model is mainly a way to collect parts into an assembly oriented structure. It contains also some information that is specific to the assembly process, e.g. orientation of an assembly when assembled.

Thus Hull offers two different views of *and two different access paths* to the product model. E.g. even if the model is built up via the design model it is often natural to extract information for production via the assembly structure.

All hull parts must be modelled and become part of the design model before they can be referred to from the assembly structure. (Outfitting steel is modelled in the Structure application). The detail design work means a successive breakdown and refinement of the design model in such a way that the design intent is protected. This adaptation for production should be done before the final population of the assembly tree can be done (or at least is quite meaningful).

Examples of the relation between the design model and the assembly model will be given after the description of the Hull design model.

2.2.2 Functional View on Shell Profiles

For shell profiles (both longitudinals and transversals) AVEVA Marine supports an additional and more functionally oriented view that is neither geographically oriented (as the hull design structure, see below) nor a production view (as the assembly structure).

Shell profiles, e.g. longitudinals at a certain distance from the centre line, are traditionally identified by a number that is common to all the different parts that are located at this distance along the whole ship. Towards the ends of the ship the trace curve has normally no longer the same fixed distance but the parts on it are still identified by the same number. Correspondingly, there is in AVEVA Marine a reference structure by which longitudinal parts belonging to a certain longitudinal may be referred to by the longitudinal number independent of the position in ship.

Thus, the shell stiffener parts may be part of three reference structures: The design structure, the assembly structure and a structure of type longitudinal or transversal (the latter structure being mandatory).

2.3 Design Structure of AVEVA Hull

The Hull design model is organised in a hierarchical structure with three or at most four levels as illustrated by the figure below. It may be considered as a compromise between the assembly view of the product and a functional view. Thus it is not it its concept a pure functionally oriented model but corresponds maybe more ideally to a view for the detail design. However, it is also so general that it lends itself to several different ways of organising the design work.

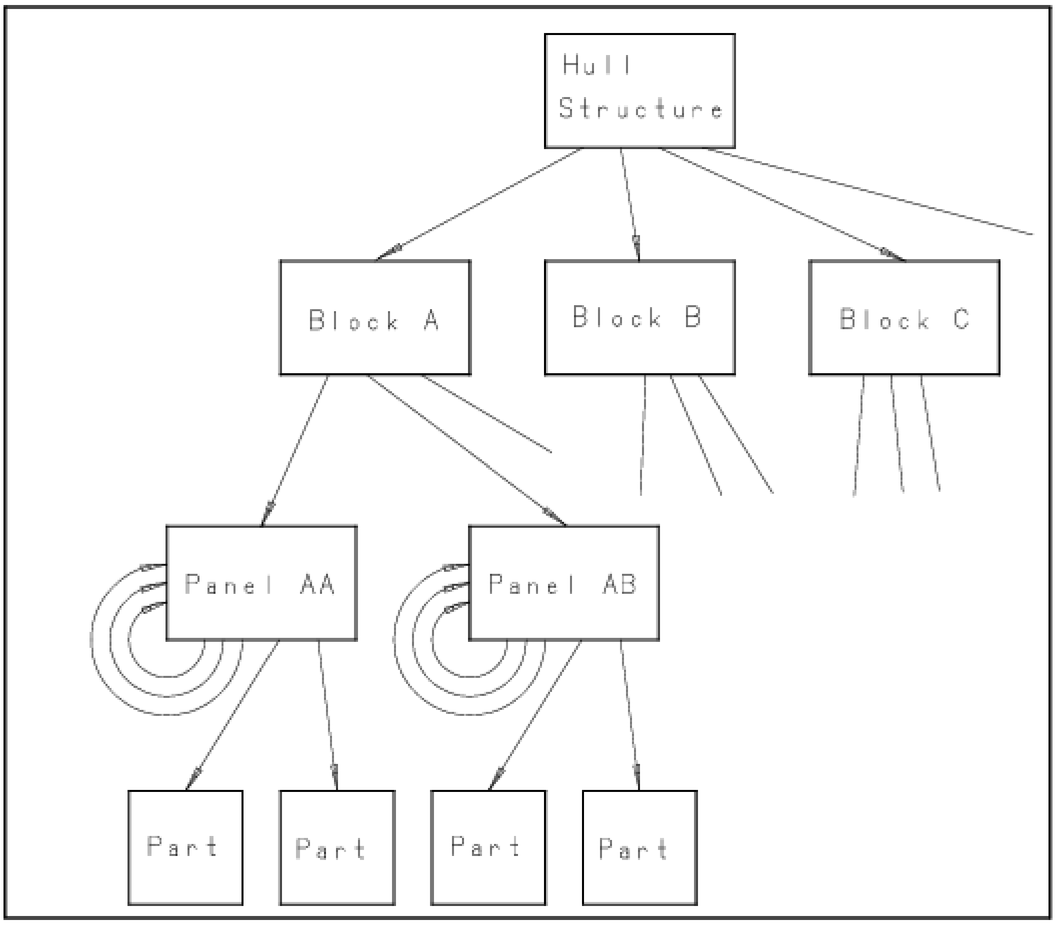


Figure 2:1. Hull model structure

At a first glance the structure in the figure above (and the mere vocabulary used) may suggest that the design model is very much like a simple assembly structure. The blocks then correspond to sections or high level assemblies, the panels correspond to sub-assemblies that are built up by parts on the lowest level. (The circular backward reference arrows at the panels indicate that the parts (in plane panels) normally are stored implicitly in the panel itself. More about that later!). The picture does not reveal the fact that brackets may be small "assemblies", i.e. contain stiffeners and/or flanges.

For a small yard with a simple organisation of the production it can actually be used both as a design model and an assembly model. However, in the general case the design structure and the assembly structure need not be the same. For one thing, the depth of the assembly tree structure is in principle unlimited compared to the limited depth of the design structure.

In principle, the partition into blocks should be considered as a way to divide the ship into regions of a suitable size for the *current design stage*. In an initial stage the whole ship may be considered as one block. In the detail design for production it is advantageous to let the block coincide with a main assembly of the assembly structure, but this in no way compulsory. Parts from one block may very well go into different assemblies. (Hull has certain tools to successively modify the break down of the model as the design work proceeds).

Other reasons for breaking down the model may e.g. be that the modelling of a block can be assigned to a single designer or a group of designers. Then most of the topological references in a block are local which makes it easy to work independently of the simultaneous design work in other blocks. However, it should be noted that all the time the whole model is available to all designers and there is no problem in making references across block limits.

Finally, a suitable design structure has also implications for the system performance, e.g. in case of frequent search operations in the model, especially when the model starts to get big.

The elements of the design structure will be described to some detail below.

2.3.1 Hull Structure Object

The Hull Structure object is the object that serves as the entry to the hull model via the design structure. It does not contain any relevant information except the references to all the blocks. It is created and updated via the hull utility module *inithull* at the same time as the Structure Reference object ("Structref"). The Hull Structure object is automatically updated each time a block object is created, modified or deleted.

The designer never really gets in direct contact with the Hull Structure object.

2.3.2 Block Object

Like the hull structure object the block objects do not carry any actual model information, except the location of its surrounding box in space. It is essentially a structure building object, referred to from the Hull Structure object and itself referring to the panels that belong to the block.

The blocks may be created as a feature of the *inithull* module or in a function of the Structural Design application. A block object is automatically updated each time a panel is created, modified or deleted.

The designer never really interacts with the block objects except when they are created. However, the block may be used as the "handle" by which information from the hull model is extracted in various situations.

When a block is created it is necessary to define its (rough) extension ("box") in space. The panels belonging to a block should preferably be located entirely within this box. The boxes of different blocks may overlap.

The same block may include panels both on portside and on starboard. If a block is restricted to a side section (e.g. a side tank) its extension should be restricted to its extension on portside. Panels valid for the starboard side (and even those *modelled and stored* on starboard) may nevertheless belong to this block. Thus a block can always contain panels within its explicitly defined block but also panels within the box when mirrored in the centre line plane.

A block over the Centre line should be defined with its true extension.

2.3.3 Plane Panel

The panels are divided into two types with entirely different principles: *plane panels* (in the internal structure of the ship) and curved panels in sculptured surfaces. Since they are so different they will be described in separate paragraphs. This paragraph concentrates on plane panels. For recommendations regarding panel names, see *About Naming*.

The panel is *The* model object of the internal structure, i.e. all modelling of the planar internal structure must be done via plane panels. The panel is also the level on which model information is stored in the data bank, e.g. a plate part or a stiffener can only be made part of the product model as included in a panel.

A panel may range in size and complexity from small bracket-like panels consisting of only a plate to big decks containing hundreds of parts. However, there are two conditions that must be fulfilled:

1. The ordinary plates of the panel must be located in one plane (for the special case of knuckled panels, see below).
2. It must be possible to create a closed outer contour of the panel. Thus it is normally inconvenient to create a complicated web ring structure as one panel.

The panel is a kind of "container" object. Panels consist of a number of parts, mostly implicitly stored, and a number of "features" that are required for the creation of the parts and for their adaptation to the production requirements. Thus, the hull structure data bank does not contain any explicit parts - they are realised in an automatic parts generation process when a certain region, e.g. a block or a panel, is ready for production. With this definition parts extracted for production are not considered to be part of the actual (and implicit) model.

The parts that a panel may consist of are *plates or profiles.*

Depending on how they are generated and used the part types can be further divided into sub-types.

The plates may be:

- Ordinary plates in the mould plane of the panel
- Bracket plates
- Clip (or collar) plates.

The profiles may be divided into the following types that are relevant mainly in the modelling phase (this classification is less relevant when it comes to production).

- Stiffeners (on the panel itself or on brackets)
- Welded flanges (often called face plates) (on the panel itself or on brackets)
- Pillars

In order for the parts to get all the information required to make them accurate and ready for production a panel may be generated with a number of *features* that contribute with information when the parts are extracted for production.

Examples of features in plane panels are:

- An outer contour of the panel as a whole,
- Seams that together with the outer contour will define the plate parts,
- Plate thickness and material quality,
- Bent flanges that will affect the geometry of plate parts,
- Bevel along the weld traces in edges of both plates and profiles,
- Profile characteristics and end cutting of profiles,
- Holes, notches and cutouts in both plates and profiles,
- Excess (overmaterial, green material),
- Swedging (small corrugation),
- Shrinkage compensation.

The picture below shows a panel with some parts of different types and with some types of features. The figure contains both a symbolic view of the panel (used in traditional working drawings) and a 3D isometric view of the same panel.

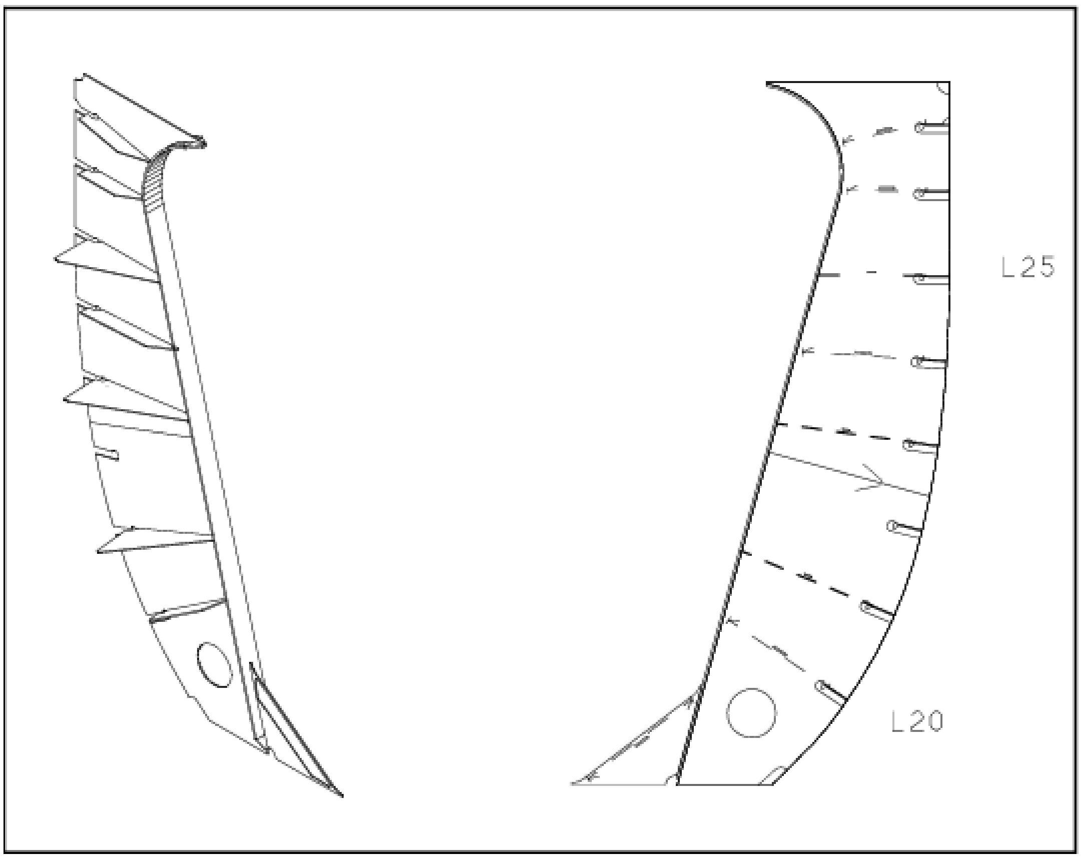


Figure 2:2. Example of a simple panel

The parts defined in a panel are normally not stored with their full geometry. As an example the plates are implicitly defined by reference to the outer contour of the panel and to seams that restrict them. Explicit information is the plate thickness and the material quality. When released as an explicit part different features will affect the geometry so that it becomes apt for production. E.g.:

- Bevel along seams and the outer contour will influence both the edge shape and the plate geometry, e.g. because of required bevel gaps.
- Holes, notches and cutouts will create open or closed "openings" in the plate geometry.
- If the plate is flanged the plate geometry will be expanded to include the folded flange.
- If the plate is swedged the plate will have to be expanded to compensate for the swedging.
- If excess is defined the plate will be made larger.
- If shrinkage compensation is asked for the plate will be slightly expanded to compensate for the shrinkage.
- Etc., etc.
- Finally the part will automatically get marking lines for all parts welded against it.

Thus the implicit model contains all the information necessary to make the plate become a "real" part for production.

- **Panels and Symmetry**

  AVEVA Marine allows a designer to benefit from the fact that the hull structure of a ship to a large extent is symmetrical. Below you will find some examples of the flexibility of Hull in this respect.

  - A panel may be generated in one half of the ship but be valid for both sides. Nevertheless individual parts for portside (PS) and starboard (SB) may be extracted from it.
  - A mainly symmetrical panel may have minor features or parts that are specific for PS or SB.
  - Any panel may be described on either PS or SB. Thus it possible to model and store on PS a panel valid for SB.

  When creating views of the model and extracting parts form the model the views and the parts will be correct independently of how panels have been modelled in these respects.

- **Panels, Special Cases**

  The panel concept is also used in a couple of special cases, namely as:

  - Panel brackets, and
  - Knuckled panels

  They are described below.

- **Panel Brackets**

Brackets are normally generated as standard brackets directly on the panel. However, sometimes the brackets are so special that they cannot be defined as standard brackets. Then they can be generated as small panels, using all the tools for available for panel generation.

However, such a panel bracket cannot be inserted directly into the hull structure (as belonging to a block). Rather it should be connected to one or several panels as a separately stored part (currently the only case of separately stored parts belonging to plane panels).

- **Knuckled Panels**

Plane panels are by definition planar, i.e. their plates must be located in one plane. Knuckled panels (including corrugated panels) are special in the sense that they may contain plates that are located in more than one plane.

AVEVA Marine handles this type of panel by letting all parts of a knuckled panel that are located in one plane be generated as individual "sub-panels". These sub-panels are then collected into a knuckled "main" panel and connected to each other for automatic extraction of parts across knuckles.

The knuckled main panel is inserted into the hull structure like all other panels. The sub-panels are (like panel brackets) not referenced directly from a block but only indirectly via the main panel.

2.3.4 Plane Panel Parts

As mentioned in the preceding paragraph the parts of a plane panel are normally implicitly stored in the panel and considered as part of the hull structure via the panel. (The circular reference arrows pointing back to the panel in the hull structure figure above indicate that the parts are (implicitly) stored in the panel "container".)

Panel brackets (see above) are the only separately stored parts in the hull structure in the model data bank).

(When released for production parts are hardly considered as "model objects". However, it is always possible to reach the released part from the model object (=the panel) and in the opposite direction it is possible to find back from the part to the implicit part in the panel. Actually, some attribute information is never really copied to the released parts but is always fetched from the model via the part. This means that such information may be changed and still there is no need to extract the parts again.

2.3.5 Curved Panels

Curved panels are part of the design structure, i.e. registered in blocks, in exactly the same way as plane panels.

However, there is an important difference between plane panels and curved panels. All the parts belonging to a plane panel are generated/created as an integrated part of the panel generation (with the exception of panel brackets that must be established separately and then are connected to the panel).
In contrast to this a curved panel is a pure "container object" and it serves only for collection of parts that have been generated "by themselves" before the panel can be established. No additional attributes can be added to parts via the panel. Thus the curved panel always has external references to its parts, quite in analogy with how a plane panel refers to its panel brackets.

The purpose of the curved panel is mainly to support the extraction of production output, calculation of support jigs, views in drawings, etc.

2.4 Mapping between Design and Assembly Structures

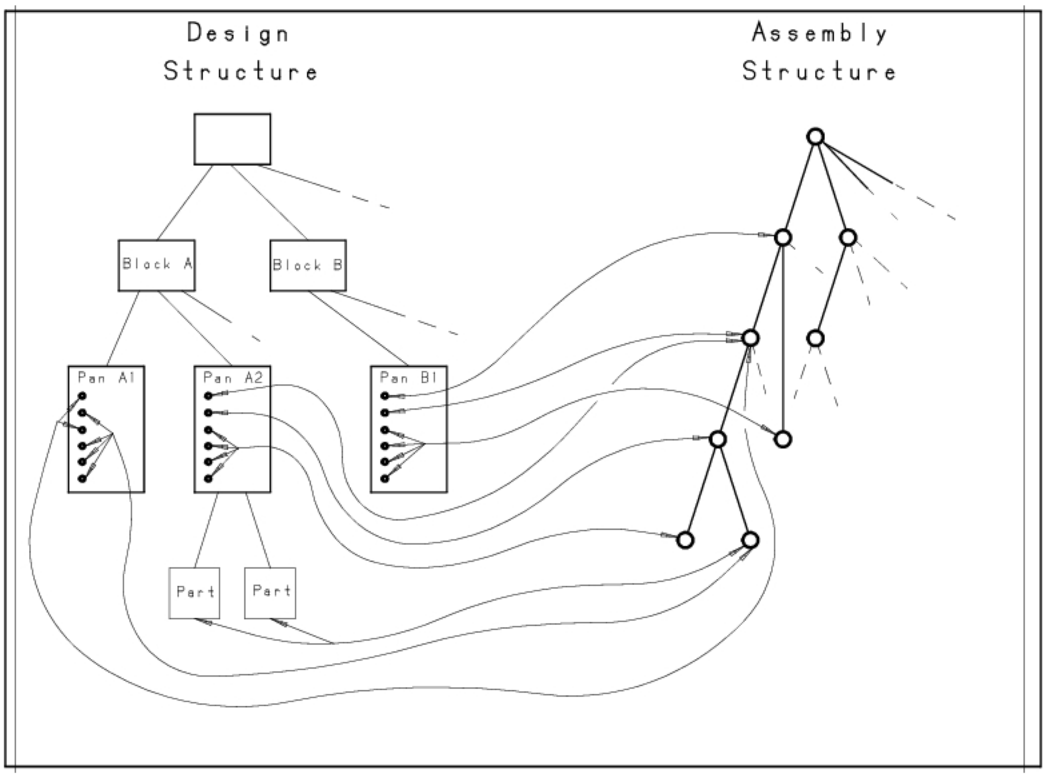
As already mentioned the *design structure* is built up in parallel with the creation of the hull structure (the panels) even if the block structure may be established with an otherwise empty model. This structure can be changed and further refined as the penetration of the design proceeds.

The complete *assembly structure* can in principle be created at the start of a project before any modelling work has been done. However, before the assembly structure can be populated by parts these must have been generated and inserted in the design structure.

The mapping between the assembly structure and the design structure is normally made via the Assembly Planning application. It creates bi-directional pointers (via names) between the nodes of the assembly tree and the parts of the design structure. It should be noted that these links are to the normally implicit parts in the design structure and not the extracted parts ready for production.

This mapping is arbitrary and is always on *part level*. Parts from a certain panel may be assembled in any assembly on any level and a certain assembly may contain parts from several different panels that may be located in different (design) blocks. As an example all the parts of a panel with the exception of a few brackets may be welded together in a low-level sub-assembly. The brackets may be welded several stages later in the assembly process, e.g. when two main sections are welded together.

The figure in the next page illustrates a possible mapping between a design structure and the assembly structure that is based on the same model. Each dot in the panel boxes of the design structure represents a part and the lines with errors represent the links between node in the assembly tree and the parts.


Design Structure
Assembly Structure
Block A
Block B
Pan A1
Pan A2
Pan B1
Part
Part

3 Object Types

3.1 General

AVEVA Marine is a system for handling of design and production information related to ships and similar products. This information may be of many different types, e.g. model information describing the hull structure (plates and profiles), pipes, cables, equipment, etc. But it may also contain information derived from model data, e.g. drawings, parts lists, etc.

Most of this information is stored as objects in data banks in formats that in the bottom is the same for the storing of a plate part, a pipe or a drawing. The purpose of this document is to give a brief introduction to certain types of objects that are specific to the hull application and also to describe how they are related to each other.

3.2 Hull Objects, General

The hull information in the data banks is stored in two distinct object types, both implementations one level above the most basic object type. (Other applications use the same object types but may also have specialised object types of their own.)

1. The first object type is the so called CAT (Contour And Table) object. In principle all hull objects except drawings are stored in this type of object. Therefore, the CAT object may be regarded as the object for hull information even if it is used for general purposes in other applications.

   The internal structure and contents of such an object depends on the type of object that it describes. The type of object is identified by an object code (object code 1 ="data type") that is registered centrally by AVEVA Marine. The model objects of Hull are all stored in CAT objects and so is much of additional information regarding standards, etc.
2. The second type of object used by Hull is the picture object that stores pictures (e.g. model views) and drawings. This object type is used - with small variations - in all AVEVA Marine applications (and is above all the object type of Drafting). However, the model objects of Hull maps into the picture object in a way that is specific to hull information. Nevertheless, information from Hull and other applications may without problems be combined into drawings, i.e. into one picture object.

   The emphasis of this document is on the Hull model data objects but a short concluding section will describe how the structure of the model objects is reflected (mapped) into the picture (drawing) objects.

3.3 Objects and Data Banks

The AVEVA Hull Information is organised in a number different logical data banks that normally also are physically different. Some of these may be common to other AVEVA Marine applications.

The table below contains a complete list of all the data banks that may be used and updated by the Hull application (all data banks are in principle available for reading by all applications, e.g. to create composed drawings). The table contains the name of the data bank, the environment variable with the name of the data bank and finally a short description of the main contents of the respective data bank.

Name	AVEVA Marine env. var.	Contents
Form databank	SB_CGDB	Form information (surface, hull curves, seams/butts).
Structure databank	SB_OGDB	Hull model information.
Plate databank	SB_PLDB	Plate parts, ready for production, extracted from the hull model.
Profile databank	SB_PROFDB	Profile parts, ready for production, extracted form the model.
Nesting data bank	SB_NPL	Nested plates and burning sketches.
Nesting standard db	SB_NSTD	Standard plates, hooks, burning sketch forms, etc. for plate nesting.
Profile nesting db	SBH_NEST_PROFDB	Nested profiles.
Drawing databank	SB_PDB	Drawings, sketches of nested profiles.
Picture databank	SBD_PICT	Separately stored views, panel pictures.
Receipt drawing db	SBH_RECEIPT	Receipt drawings from runs via Prod. Program Interface.
Assembly databank	SB_ASSDB	Assembly data bank
Production database	Oracle database	Extracted part information for user written production programs
Standard databank	SBD_STD	Drawing forms, etc. (common to all AVEVA Marine applications)
Topology databank	SB_REFDB	Topology objects
TID databank	SB_TID	Surfaces (RSOs) and Compartments

Table 3: 1.

Paragraphs below will described the most significant object types in the different data banks listed in the table below.

3.4 Objects in the Form Data Bank

The form data bank contains information related to the sculptured surfaces of the ship, i.e. the ship surface itself and pure curve information derived form the surfaces. Additionally, there are some tables that keep record of the names of objects in this data bank and of the objects stored there.

Each object type is associated with a registered "data type" (object code 1) with values according to the table below

Data Type	Description
1	Ship surface
2	Hull reference object
3	Co-ordinate Tables, general
4	Co-ordinate Limit Table, general
5	Frame Curves
1005	Inclined planar curve, most similar to a frame curve
6	Waterline Curves
1006	Inclined planar curve, most similar to a waterline curve
7	Buttocks curves
1007	Inclined planar curve, most similar to a buttock curve
8	Seams and Butts
12	Arbitrary Curve, stored from a drawing
13	Reference curves for marking of reference lines
55	Arbitrary plane

Table 3: 2.

3.4.1 Ship Surface

Normally, sculptured surfaces are stored and handled (created, modified, intersected) outside the actual Hull application, e.g. in AVEVA Initial Design. Any surface related information is fetched from this "outside" source via calls to the surface processing module.

Anyhow, an object with the appropriate name must be available in the data bank for each of the surfaces to be used in a certain project. These objects are created at the same time as the names are noted in the Hull Reference object (see next paragraph). This is done via a feature of the hull utility *inithull.*

3.4.2 Hull Reference

The hull reference object is a small table containing information about names and name rules of objects in the form data bank. The AVEVA Marine modules access the name of this object via the environment variable SB_HREF.

The hull reference object is created or modified by a feature of hull utility *inithull.*

The object contains the following information:

- Some characteristic dimensions of the ship. (length between perpendiculars, half-breadth, etc.)
- The name of the main hull surface

- The name of deck surface.

  If any of these surfaces should not be available the corresponding names should be empty.
- The group names of frame, waterlines and buttocks.

  The names main curves of these types are composed by a "group name" concatenated with a curve number (e.g. a frame number). These group names are defined in this object.
- Co-ordinate tables for frame, waterline and buttock curves.

  These tables contain the x-, z- and y-co-ordinate, respectively, of the plane in which the curve with a given number is located.
- Group names of seams/butts.
- The names of limit tables along the x and z-axes for seams.

  One of these tables contain the minimum and maximum co-ordinates along the x-axis of all seams, the other the same information for the z-axis.

It is possible to define additional sculptured surfaces for which the names have to be defined as well in the hull reference object. Curves created in these additional surfaces are named according to the same rules as curves in the main surfaces. To separate them from the main surface curves the group names of these additional surfaces have an additional "suffix" by which the group name extended. These suffixes are specified together with the additional surfaces for which they are valid.

It is good rule (but not compulsory) to let all names start with the ship letters.

3.5 Objects in the Structure Data Bank

The structure data bank contains model information about the steel structure of the ship, i.e. all information that has steel related to it. The model information in the structure information is stored according to nominal dimensions and the adjustments for production are made when parts are extracted for production. Examples of such adjustments are shrinkage compensation, excess, shell plate development, development of knuckled pieces, changes for (varying) bevel angles and bevel gaps.

Additionally the structure data bank contains some table information and objects that describe miscellaneous types of hull standards, set up by the customer.

The table below lists the most important types of objects and the "data types" (object code 1) that are specific to them

Data Type	**Description**
50	Structure reference
9	Longitudinal frames
10	Transversal frame
11	Shell stiffener
51	Hull Structure Object
52	Block object
58	Endcut table

Table 3: 3.

Data Type	Description
59	Shrinkage compensation object
60	Swedging object
61	Cutout standard object
70	Hull standard objects (for part naming, bevel, flanging, etc.
100 - 499, 700 - 999	Plane panels
500 - 599	Curved panels

Table 3: 3.

3.5.1 Structure Reference Object

The structure reference object is a small table containing information about names and name rules of objects in the structure data bank. The AVEVA Marine application modules access the name of this object via the environment variable SB_SREF.

The structure reference object is created by a feature of hull utility *inithull.*

The object contains the following information:

- The "ship letters"
  It is suggested that the names of all hull objects of a certain project should start in the same one or two letters. By using ship letters it is easy to differentiate between production names and the internal names of AVEVA Marine. It makes it also possible to have several projects in the same physical data bank with the "same" names. (This refers to the names used internally in the data banks only - they will normally disappear when the internal names are mapped onto the production oriented part names).
  If not used, the ship letters should be assigned an empty string
  When used, the ship letters must be explicitly added to the objects when they are created and also e.g. to the group names of frames, etc.
- The name of the Hull Structure object, i.e. the object via which all whole structure design model may be accessed (i.e. the “root object” of the design structure).
- The group names of longitudinal and transversal frames. The names of these types of objects are created by a "group name" plus a number (in the same way as e.g. frame curves).
- The names of the extension table for longitudinal and transversal frames along the x-axis (min-max co-ordinate values). The names of limit tables for the extension along the Y- and Z-axes are formed by adding 'Y' and 'Z', respectively, to this name.
- The project name of the current project. This name may be used as a part the production oriented part names (and may thus be considered as an "external" correspondence to the ship letters that are for internal use).

3.5.2 Structure of the Structure Data bank

The structure data bank contains the design structure of the ship under design. The model information is stored in a number of different object types. There is further a certain structure between the different objects in the data bank. This is described to some detail in a separate *Design Structure of AVEVA Hull*.

4 Topology

4.1 General

In Product Information Models there may be dependencies between the objects. These dependencies are nearly always saved as some kind of references to objects or part of objects. This is certainly true for the Product Information Model and specifically for the Hull Model. Such references constitute the topology of the Product Information Model.

In AVEVA Hull each design model object holds references to the objects that are part of its definition, e.g. the surfaces, curves or panels defining a plane panel boundary. Thus each object can be regenerated using its references. This might create e.g. an updated outer contour if the referred objects have changed its position or shape.

Besides the references defining an object the Hull Model also contains references to the depending objects. For e.g. a deck this would be references to all webs attached to it. This is essentially the same information as the defining references but seen from the other end. The depending references are stored in special topology objects.

In order to benefit from the topology it is essential that it is well defined. This means e.g. that a bulkhead that ends at a deck should have a limit definition that is a reference to that deck rather than just a coordinate. Then the bulkhead outer contour will be dependent on both the position of the deck and even the thickness of the deck plating. Also circular references should be avoided. If e.g. two deck panels are connected they should not refer each other directly in the boundary definition. Either both can refer a predefined plane or one can be defined by a coordinate and the other refers the first one.

Many references are on component level, e.g. a bracket referring a stiffener on the same panel or on another panel. If e.g. the boundary definition of a bulkhead refers a deck and then a stiffener on that deck refer a stiffener on the bulkhead this may seem like a circular reference as the bulkhead refers the deck and vice versa. In reality it is not as the topology has to be viewed on component level. Thus the boundary of the bulkhead is dependant on the deck position and plating while the deck stiffener is dependant on the bulkhead stiffener. So it is quite possible to sort out e.g. in which order to regenerate the components of the panels.

4.2 Topology Objects

A special kind of object is used to store topology references. The objects are automatically created and updated by the AVEVA Marine applications. One object is created for each design model object.

For Hull Models created with versions of Tribon prior to M1 version 3 it is necessary to run a utility program (sj903) in order to create the topology objects. The system can work without topology objects, but certain functions will not work properly. It is also possible to recreate the topology objects for any project at any time using this utility program.

The objects contain references down to component level, or rather on component group level. Each object contains both the references defining the object and the references to the dependant object component groups.

The updating of topology objects is done when the object, currently only plane panel, is stored.

4.3 Using the Topology

The topology objects are used in a number of functions. The most visible is the Topology function in which defining and dependant objects are highlighted.

Another less visible but as important use is in the **Planar/Panel/Recreate** function. This function will recreate all the activated panels. Naturally these panels may have internal dependencies among them, and to get the correct result they must be regenerated in the correct order. The order is then derived using the topology objects. This may result in that some panels are only partly generated at first as some components may depend on components on other panels that have not yet been regenerated.

Also in the functions **Planar/Panel/Copy** and **Planar/Panel/Move** the topology objects are used to sort the panels and components in the correct order.

When using functions such as **Planar/Model/Split** STI it may be that the topological order among the component groups within the panel becomes disturbed. This results in that e.g. the panel cannot be created from the scheme, as the statements are not in the correct order. However when storing the panel the component groups are sorted in the topologically correct order at the same time as the topology objects are updated.

5 About Naming

5.1 General

As described in a separate document AVEVA Marine in general and Hull in particular support two different views of the product model:

- The design view

and

- the production (assembly) view.

The design view is kept internally in AVEVA Marine and is the responsibility of the design departments. The production view is the view of the product mainly considering how it should be manufactured and assembled.

The two views are based on the same product model and consist of the same parts. However, quite different tools are used to access a part via the two views and the name of a part is not the same. Thus, a part has one design name and another name for production. (Traditional part programming systems do normally not distinguish between design and production names. One reason is that parts programming often takes place at such a late stage that the production planning has already been done).

In AVEVA Marine the design names are assigned as soon as a part is created. They are here called the AVEVA Marine names, are normally set by the designer and are used internally in AVEVA Marine and in the drawing offices. The AVEVA Marine names are fixed once they have been set.

The production names are set at later stages when the break down into assemblies has been made. They are called part-names and are typically painted onto parts in production, are the names used in production lists, etc. The part-names are dynamic in the sense that they are formed when asked for. If the rules for the structure of part-names are changed or if any of the constituents of the name is modified then a new part-name will be delivered the next time it is asked for.

AVEVA Marine has functions that support the mapping from the AVEVA Marine names to the part-names. This support comprises a function for automatic setting of position (piece) numbers and a module to control how the part-names for different types of parts should be built up from available attributes, for details. If in a simple production the design and production views are identical the part-names may be based on the design structure.

However, the purpose of the current document is restricted to give some information about rules for *AVEVA Marine names* of model objects in the hull structure and of some related support objects.

5.2 General about AVEVA Marine Names

5.2.1 Ship Letters

The concept of *ship letter(s)* is relevant for names of *model* objects in the hull data banks (thus, they are normally not used for drawings and other non-model types of objects). The purpose of the ship letters is:

1. The name of an object should immediately indicate to which project an object with a given name belongs. This makes it possible to store model objects from different projects (e.g. sister ships) in the same data-banks even if they otherwise have the same structure of the naming (in most cases separate data-banks are used).
2. Further, the existence of the ship letters indicates that this is a AVEVA Marine name and not a part-name.

The ship letters (sometimes one but normally two) are registered in the Structure Reference object and should be added in the beginning of names even if the names are given directly by the user. I.e. the ship letters are never added automatically to a name by AVEVA Marine.

However, it should be pointed out that the use of ship letters in names is optional and thus a matter of *convenience.* The Hull application will do perfectly well without them.

Note: In all examples of names in this document the ship letters are supposed to have been defined as AA. The ship letters of the current project should replace AA when the name rules below are applied for a customer project.

5.2.2 Different Principles of Names

There are different principles for the names of model objects in AVEVA Marine as specified to some detail below. Independently of how the names are formed their length must not exceed 24 characters (it should be noted that the length of part-names does not have the same limitation). The different types of names are:

1. Explicit names, given by the designer.
2. Names, automatically generated by AVEVA Marine, based e.g. on user given names.
3. Names, built up by a "group name" and a user defined number ("composed names").

- **Explicit Names**

Explicit names are (in principle) selected and set by the designer. They can be chosen arbitrarily but it is almost always convenient to follow certain rules that may be set up by each customer. If ship letters are defined the names should start in these letters.

Example:

Panel names may be selected arbitrarily but it is *convenient* to let the panel name start with the name of the block it belongs to, because then it is not necessary to explicitly specify the block.

- **Derived Names**

Typical examples of derived names are the names of all the parts that are extracted from a panel in the automatic part generation of AVEVA Marine. E.g. a plate part extracted from the panel AA123-4 will get name AA123-4-1S or AA123-4-1P unless otherwise specified (see below) and the designer has no possibility to affect that.

- **Composed Names**

  It is quite frequent that certain object types should be identified (and in most cases referred to) by numbers (set by the designer) rather than by the full names by which they are stored in the data banks. Examples of such object types are the traditional main hull curves (frames, waterlines, buttocks), longitudinal and transversal frames, seams and butts, etc. These objects are always either curves in the sculptured surfaces of the ship or based on such curves.

  The numbers of these objects must always be given by the designer but the name by which they are stored in the data-bank is formed by AVEVA Marine concatenating a prefix (a "group name") with the user-given number. The group names are individual to different object types. The used numbers should always be smaller than 10000 (with individual deviations for certain object types).

  Example:

  Suppose that the group name for seams and butts is AAS and that the designer has given a seam the number 599. Then the name of the seam object in the data-bank will be AAS599.

- **Composed Names of Objects in Multiple Surfaces**

  As already mentioned objects with composed names are always related to curves in sculptured surfaces. In order to allow the same numbers to be used for objects in different surfaces (in case the project has several different sculptured surfaces) it is necessary to specify a surface specific extension of the group names. This *surface suffix* consists normally of one letter and is defined in the Hull Reference object together with the names of the used surfaces.

  The main hull need not have any surface suffix.

  Example:

  Suppose that there is a seam with number 123 in an additional surface with surface suffix C and that the group name for seams is AAS. Then the name of that seam will be AASC123.

5.3 Specific Name Rules

This paragraph contains rules that should be used or are valid for different types of Hull objects. The reader is supposed to be familiar with the different types of model objects that may occur in the data banks.

The valid characters in Hull Object names are [A-Z], [0-9], . (dot), - (hyphen) and _ (underscore).

5.3.1 Hull Structure Object

The Hull Structure object is the root object in the design structure. Its name can be chosen arbitrarily and it is defined when creating the Structure Reference object in the hull utility module *inithull.*

However, it is suggested that its name should be AAHULLSTRUCT.

5.3.2 Block Objects

The block objects are on the level closest below the Hull Structure object. They can be named arbitrarily and they are created as a feature of the hull utility *inithull* or alternatively in the Structural Design application.

However, it should be kept in mind that the block name normally is the first part of the names of panels belonging to that block. Therefore, the block names should preferably be kept short, typically like AA123.

5.3.3 Panels

The names of panels can be chosen arbitrarily. However, for normal panels it is convenient to let the panel name start by the name of the block that the panels belongs to, followed by a dash (-), e.g. a panel belonging to block AA124 should preferably have a name looking like AA123-4.

When parts are extracted from the panel their names will be formed by an extension of the panel name. This should, therefore, not be too long, preferably never longer than 16 characters.

There are some special types of panels with somewhat modified recommendations regarding the names.

Panel brackets are not registered in any block and their names may be chosen quite arbitrarily. However, since most panel brackets belong to one "owner" panel it might be convenient to let that owner name be part of the panel bracket name. E.g. panel brackets belonging to panel AA123-4 may be called AA123-4B01, AA123-4B02, etc.

Similarly, sub-panels of knuckled panels can be named arbitrarily but may preferably be given names that relate them to the main panel they are parts of. Supposing the same panel as above they may e.g. be called AA123-4K01, AA123-4K02, etc.

Remark:

For curved panels there are currently some additional rules (that are temporary). The symmetry status of the curved panel has to be indicated a suffix at the end of the name.

- The names of symmetrical panels (valid portside and starboard side) should end in a digit,
- Portside specific names should end in a P,
- Starboard specific names should end in an S,
- Names of panels (in principle symmetrical) over the centre line should end in SP.

5.3.4 Parts from Plane Panels

When a panel is "split" the part components of the panel will be stored as parts of their own in their respective data-banks. The names of these extracted parts are always generated quite automatically and they are built up as specified below.

The following types of panel components may result in parts of their own:

- Plates
- Brackets
- Clips (collars)
- Stiffeners
- Flanges

- Pillars

Each of the parts has a unique sequence number within their type of part within the panel. E.g. plates are numbered 1, 2, 3, likewise stiffeners are numbered 1, 2, 3,… (S1, S2, S3, ..), etc. ("holes" in the number series are allowed). These sequence numbers are used when forming the AVEVA Marine names of the parts.

When extracting parts from symmetrical panels (i.e. when the panel is stored in one copy that is a valid for both the portside (PS) and starboard (SB) side) there is an option regarding how to extract parts. By default individual occurrences are stored for PS and SB. Optionally one part may be stored, valid for both PS and SB in the same way as the panel. Slightly different naming rules are required in these cases.

The rules for names in different cases are described by examples in the table below. The panel name is AA123-4 and the parts are all supposed to have sequence number 9.

The second column contains the names when the parts should be stored like the panel regarding symmetry, i.e. with no individual copies of parts for PS and SB. Columns 3 and 4 show the names of the PS and SB part, respectively, when individual copies are stored PS and SB. These rules will apply also if the panel itself is valid e.g. only PS. E.g. the parts from a panel valid only PS will be named as in column 3. The name as in columns 3 and 4 is the normal situation.

Column 4 contains (within parenthesis) a further addition to the names of plate parts in (or symmetrically over) the centre line. E.g. if AA123-4 is a centre line girder the plate name will be AA123-4-9SP.

Part type	**AVEVA Marine name (SB+PS)**	**AVEVA Marine name PS**	**AVEVA Marine name SB /(in CL)**
Plate	AA123-4-9	AA123-4-9P	AA123-4-9S(P)
Bracket	AA123-4-9B	AA123-4-9BP	AA123-4-9BS(P)
Clip	AA123-4/C9	AA123-4/C9P	AA123-4/C9S
Stiffener	AA123-4/S9	AA123-4/S9P	AA123-4/S9S
Flange	AA123-4/F9	AA123-4/F9P	AA123-4/F9S
Pillar	AA123-4/P9	AA123-4/P9P	AA123-4/P9S
Stiff. on bracket	AA123-4-9B/S1	AA123-4-9B/S1P	AA123-4-9B/S1S
Flange on bracket	AA123-4-9B/F1	AA123-4-9B/F1P	AA123-4-9B/F1S
Doubling plate	AA123-4-9D	AA123-4-9DP	AA123-4-9DS

Table 5: 1.

Built profiles may optionally be split into two different parts, one from the web of the profile, the other from the flange. The names the resulting parts are formed by adding a W (for the web) and F (for the flange) to the end of the profile name as in the table above.

E.g. if profile AA123-4/S9P should be split into a web part and a flange part their names will be AA123-4/S9PW and AA123-4/S9PF, respectively.

In addition to the "normal" plate piece parts as described above it is also possible to extract so called "assembly parts" consisting of several piece parts. Such parts may be extracted from both plane panels and (at least almost) planar shell panels and they are used to generate NC information for panel lines via the PLCM module. The names of assembly parts are formed by concatenating the panel name with the name of the lowest assembly of the involved piece part plates separated by a dash (-).Suppose that there is a seam with number 123 in an additional surface with surface suffix C and that the group name for seams is AAS. Then the name of that seam will be AASC123.

Example:

Suppose that the assembly name of the piece part plates is ABC in the panel with name AA123-4. The name of the assembly part they constitute will be AA123-4-ABC.

5.3.5 Parts in Curved Panels

The parts in a curved panel are shell plates and shell stiffeners. They both exist as objects of their own in the structure data bank as. Shell plate and shell stiffeners will be transferred to the plate/profile data banks when running “Curved Part Generation“ (sf831d).

In Tribon versions earlier than M2SP2 shell stiffener and shell plate were renamed (in the structure data bank) when they were added to a curved panel. (They got a new name based on the panel name). This handling is now changed so that the plates and stiffeners will keep their original name even after they have been added to a curved panel. This change of naming also affect the naming of the plates and stiffeners in the plate/profile data bank, as will be explained in the following sections.

Note: *In Tribon M3 and later versions the new naming rules are the default.* This can be overruled by setting SBH_CPAN_RENAME_PARTS to YES. Please note that the old naming rules will only be available during a transition period and may be obsolete in future versions of AVEVA Marine.

Important: The flag “**SBH_CPAN_RENAME_PARTS**” must be used with caution! Once you have started to generate curved panels with the new/old naming rules, you should stay in this mode. It is NOT recommended to jump back and forth between new and old naming rules (at least not within the same project).

- **Naming of Shell Stiffeners in the Structure Data Bank**

The name of a shell stiffener in the *structure data bank* should in principle be completely free. However, currently a stiffener must apply to the following rules:

`<FreeName>-S<sequence_number>[<symmetry>]`

- `<FreeName>` is any string valid in a object name. For example: it can be the name of the curved panel that the stiffener belongs to.
- `<sequence_number>` is a free running number that may be selected arbitrarily to give the shell stiffener a unique name in the structure data bank
- `<symmetry>` is empty (missing) if the stiffener is valid both on port-side (PS) and on starboard side (SB). If PS specific P should be added, if SB specific S should be added.

Example:

A shell stiffener that belongs to the curved panel AA567-8P and that is PS specific may get the name AA567-8-S25P. It may also get a name *not* based on the panel name, e.g. STIFFENER-S25P

- **Naming of Shell Stiffeners in the Plate/Profile Data Bank**

  When transferred to the plate and/or profile data bank the stiffener will get a name based on the curved panel name:

  `<panel_name>/S<sequence_number>[P|S|SP][W|F]`

  - `<panel_name>` is name of the curved panel to which the stiffener belongs
  - /S is always added
  - `<sequence_number>` is a neutral sequence number assigned to the stiffener when it is added to the curved panel. Assembly Planning uses the same number when referencing a stiffener in a curved panel.
  - P or S will be added if individual parts port-side and starboard should be stored
  - W and F, respectively, are added if built profiles are split into different parts for web and flange.

  The portside version of the symmetrical shell stiffener AAL100-S1 that belongs to the panel AA567-8 may get the part name AA567-8/S5PP (if the sequence number 5 was assigned to the stiffener when added to the curved panel).

  If the same stiffener is split into web and flange parts their names will be AA567-8/S5PPW and AA567-8/S5PPF, respectively (in analogy with the rules for plane panel profiles).

- **Naming of Shell Plates in the Structure Data Bank**

  The naming shell plates in the structure data bank is free. The plate name can be any string that applies to the general rules of a AVEVA Marine name.

- **Naming of Shell Plates in the Plate Data Bank**

  When transferred to plate data bank, the plates will receive a name based on the panel to which they belong:

  `<panel_name>-<sequence_number>[S|P]`

  The sequence number is a neutral sequence number assigned to the plate when added to the curved panel. Assembly Planning uses the same number when referencing a plate in a curved panel.

5.3.6 Profiles in the Shell

Shell profiles may be divided into longitudinal frames (or longitudinals for short) and transversal frames (or transversals). As has been described in ref. HG 2 longitudinals and transversals consist of objects on two levels, the *shell profile* and the *shell stiffener*.

The shell profile object collects all shell stiffeners belonging to a certain longitudinal/ transversal and that are referred to by a number and it does not itself contain any physical profiles. The actual profile information (one profile per object) is stored in the shell stiffener object.

- **Shell Profiles**

  The names of shell profiles are composed names and the structure of the name is

  `<group_name>[<surface_suffix>]<number>`

Example:

Suppose a case with:

• Longitudinal group name:	AAL
• Empty surface suffix	
• Longitudinal number:	250

Then the longitudinal name will become AAL250. This name is formed automatically and will never be used directly by the designer - he will always refer to the longitudinal by its number.

Longitudinals and transversals have separate group names which are stored in the Structure Reference object and defined as a feature of the hull utility inithull (ref. HB14). Recommended names are AAL and AAT, respectively.

Associated with the longitudinals and transversals are also *limit tables* (one for each co-ordinate axis) with the extensions of the shell profile as a whole. The name of the x-table is defined at the same time as the group names. The names of the y- and z-limit tables are formed by adding Y and Z, respectively, to the given name of the x-table.

Recommended names are AALLIM and AATLIM, respectively.

The group names and the limit table names as described above should be extended with the surface suffix. Then the shell profiles from different surfaces will be registered in separate limit tables and may have the same numbers as those in other surfaces (not yet implemented - currently the surface suffix will be disregarded).

- **Shell Profile Numbers**

There are currently some rules for the numbering of shell profiles that should be adhered to (see the figure below). They are to some extent temporary. Violation of these rules will mainly affect the automatic generation of shell profile sections in symbolic hull views.

- The numbers should always be integer numbers.
- **Longitudinal** numbers should be multiplied by 10 compared to their numbers in drawings, etc. This is in order for the numbers of "inserted" longitudinals towards the ends of a ship to be integers as well.
- The numbers should be in the range 5 -999.
- When output in drawings the number will be divided by 10.
- The number of longitudinals in the deck should be increased by 1000 compared to the shell longitudinals at the same distance from the Centre Line. The deck longitudinals are referred to by the increased number but before being output in views the added 1000 will be subtracted.
- Longitudinals that are specific for starboard should have their numbers increased by 2000 compared to the longitudinal in the same position on portside. This is valid for profiles both in the shell and in the deck. Thus starboard specific longitudinals in the deck should have numbers, larger than 3000.
- There are similar rules for the numbering of **transversals**.
- The numbers of transversals are normally equal to the frame numbers but *must not be negative*. However, they may be multiplied by 10 in case there the ship has half frames.

- The transversal numbers should be in the range 1 -4999.
- Starboard specific transversals should have their numbers increased by 5000 (which is subtracted when the number is output in drawings/views).
- There are no specific rules for the numbers of transversals in deck. Their numbers may e.g. be increased by 1 compared to the shell transverse.

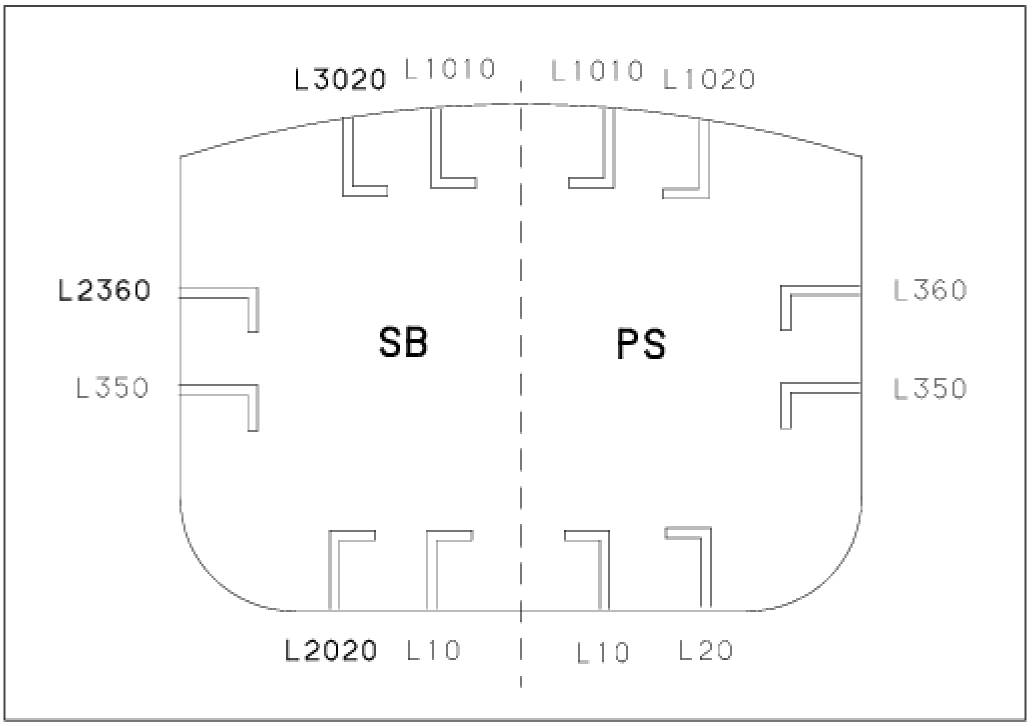


Figure 5:1. Rule for numbering of longitudinals.

Note: If free naming of seams and shell profiles is activated, a shell profile may be given an arbitrary name.

5.3.7 Hull Curves

It is customary in traditional shipbuilding to use sets of principal curves like frames, waterlines and buttocks. They play a less important role in today's AVEVA Marine when it is possible to refer directly to the surface and no predefined curves of these types actually are required. Anyhow, in most cases sets of such curves are created in the form data bank. These curves are always planar.

There is an additional type of hull curve that is essential for modelling of the shell plating, namely the seam/butt. Seams may be planar but may also have an arbitrary shape.

Common to the naming of all these curves is that they have composed names, i.e. that they are identified and referred to by a number set by the designer. When the curves are stored in the data-bank their names are formed by concatenating a group name with the user given number.

The curve numbers should follow certain rules:

- They are normally positive (however, frame curves may have negative numbers).
- They must be smaller than 10000.

- Special for frames is that they should be in the interval [-899,2766] (at least if the numbers of the frame curves are used as number of frame positions as well).

The group names are individual to each type of curve and are registered in the Hull Reference object and set-up by a feature of the hull utility *inithull.* The following group names are recommended:

- AAX for frames
- AAY for buttocks
- AAZ for waterlines
- AAS for seams/butts

The group names will be extended by the surface suffix of the surface from which they are cut.

The hull curves (frames, waterlines, buttocks) are associated with co-ordinate tables that specify the x-, y- and z-co-ordinates, respectively, of each of the numbered curves. The names of these tables are defined and registered simultaneously with the group names and are suggested to be AAXTAB, AAYTAB and AAZTAB, respectively. These names should be extended by the surface suffix so that both the curves and the tables may be kept apart for curves from different surfaces (not yet implemented).

The seams/butts are instead associated with two co-ordinate limit tables that keep record of the extension along the x- and z-axes for each seam/butt. The names of these tables are recommended to be AASLIMX and AASLIMZ, respectively.

Note: If free naming of seams and shell profiles is activated, a seam may be given an arbitrary name.

5.3.8 RSOs

Normally RSOs are created in the Initial Design Surface application but it is also possible to create them from panels in the Structural Design application. In Structural Design the RSO names always start with the prefix "_RSO_". If not given by the user when the RSO is created, the system will add the prefix automatically. The maximum length of the RSO name is 25 characters including the prefix.

5.3.9 Other Objects

The data-banks contain a large variety of objects and object types in addition to those mentioned above. When it comes to names of these objects the user may either give them arbitrary names or they may get fixed names by AVEVA Marine that the user is never really confronted with.

Examples of the former category are drawings and nestings. However, it is recommended that a customer sets up rules of his own for how these objects should be named. But this is nothing that is required by AVEVA Marine.

Examples of objects with fixed names are miscellaneous set-up tables for design standards. AVEVA Marine assign fixed names to these objects and they have been selected with a structure to minimise the risk for name collisions.

Finally, there is one very important (and general) aspect on names of objects in the data-banks. *Names of objects that may be processed simultaneously by a* AVEVA Marine *program must have unique names*. Otherwise AVEVA Marine will in some cases issue an error message but in other cases the result may be errors that are difficult to penetrate and there is also a risk to get corrupted objects.

E.g. it is quite possible to give a drawing the name of a block because these objects are stored in different data-banks. If, however, a view is generated in a drawing via the block object with the same name then problems will occur. When the block is read into the work area of the program to be sequenced for panels to be included in the view (drawing) there is already an object (the drawing) with the same name and this causes problems.

5.3.10 Setup for free naming of seams and shell profiles

If a project is setup to enable free naming of shell profiles and shell seams, an NSEQ database has to be setup. This is done in the following way:

1. Create a new database of type Name sequence. This database will be update only, and if Global is to be used, one database must be created for each site that at any time will create shell seams or shell profiles (i.e. longitudinals or transversals).
2. Add the database to the appropriate MDB's

If global is not used at all, nothing more is required. Otherwise, name sequences must be defined manually via the .Net PML interface like follows:

1. Create a new name sequence for shell seams for each surface in the project. The name of each sequence must be _CSEAM_ followed by the name of the surface and a final _. For example, for a surface named HULL the sequence name should be _CSEAM_HULL_.
2. Define a interval of valid numbers for the sequence by setting start number and maximum limit. It is essential that the interval do not overlap the interval at any other site. Sequence numbers may be in the range from 0 to 21477483647.
3. Repeat steps 1 and 2 for longitudinals and transversals. The procedure is the same, but the names of those sequences should have a prefix of _CLONG_ and _CTRANS_, respectively, instead of _CSEAM_.

It is possible to create new surfaces directly from within Hull Design (in the Structural Design and Curved Hull applications). Whenever that is done, corresponding name sequences must be defined as described in the steps above before any seams or profiles are modelled in the surface.

For more details on the .Net PML interface to name sequences, see the *.NET Customisation Reference Manual* and the *.NET Customisation User Guide*.

5.4 Automatic AVEVA Marine Names

Some functions can generate a large number of model objects at the same time. In these cases it may be quite time-consuming to enter the AVEVA Marine names of these objects manually. In Planar Modelling there is an option to generate the names of plane panels automatically.

The solution is built on the assumption that the panel names follow a convention. This convention must be possible to express programmatically using the data given in a normal panel statement (see *Hull Planar Modelling, Design Language of AVEVA Hull Modelling*). For example the name could be composed of the block name, the location, the symmetry code and a running number to make it unique if the other parts of the name coincide for two panels. The name could then be e.g. **AA123-FR26SBP_3** for the third symmetrical panel on frame #26 in the block AA123.

If a Vitesse Trigger named **_TBhook_AutoPanelName.py** exists in the AVEVA Marine Vitesse library directory it will be activated when a function in Planar Modelling needs a panel name to create a new panel. An example of such a trigger is delivered and installed together with Hull. It is recommended that the trigger be modified to implement the name convention at the site, after installing AVEVA Marine. Alternatively to remove the trigger if automatic naming is not wanted.

The trigger replaces the need to explicitly give the panel name when creating new panels, but in most cases it is possible to override the trigger and give the panel name explicitly. See the documentation of the modelling functions in the chapter *Planar Modelling, Interactive AVEVA Planar Hull Modelling Functions*.

For a description of AVEVA Marine Vitesse, see *Developer's Toolkit, Vitesse*.

6 Standards

6.1 General

There are standards of many different types and on many different levels related to AVEVA Hull and its use. Examples of general types of standards are:

- Rules for drawing layout (should transverse structures be shown "portside/ looking aft" or "starboard/ looking forward", which line types and symbols should be used in symbolic drawings, etc.).
- Rules for naming of structural members and parts both in design and production.
- Rules for use of symmetry.
- Etc.

These and similar matters are described in different parts of the AVEVA Marine documentation.

The focus of this document is on another aspect of standards, namely on the design standards used in the hull design and production. The design standard typically affects the physical properties of parts and members compared to the more routine oriented types of standards, exemplified above.

The different types of design standards involved in the use of Hull are all described to a required detail level in separate documents, devoted to the feature in question. The purpose of this document is to give an overview of the different types of design standards available in Hull and of the principles according to which they have been implemented and are available for customer adaptation and set-up.

6.2 Principles for Implementation of Design Standards

The design standards in AVEVA Marine have been implemented according to three different main principles.

1. The standard is completely "glued" into the code, i.e. any required changes must be done by AVEVA and new programs must delivered when a standard of this type has been extended. Standard of this type is closed.
2. AVEVA Marine offers a set of tools (a "toolbox") adapted for the needs in certain areas. By combining these tools a customer may create his own standard in a certain area. Such a standard is open to the extent the toolbox is complete enough and it can be created and set up by the customer.
3. The standard is completely open, i.e. a customer may set up his own standards without being restricted by the tools of AVEVA Marine.

In making a choice between these different ways of implementing the standards one has to consider a number of aspects:

- How often need the standard be changed? - If the frequency of new or changed standard is very low you may accept a standard that is less flexible.
- On the other hand: A standard that often need be changed, e.g. between each project, need be open so that changes can be made rapidly and without dependency on resources outside the company.
- One should observe that there is often a trade off between flexibility and ease of use. An in-built standard is immediately available whereas a completely open standard may have to be set up or adjusted before it can be used.
- An in-built standard makes it easy to co-operate with other users of the application (and with your system supplier!). The in-built standard is common to all users and exchange of data and services can be done without problems. An open standard may cause troubles in such situations, e.g. because two co-operating parties may have used different names for the same thing or - which is worse - the same names for different things. In any case, it is not enough to supply model data, relevant parts of the standard set-up need be transferred as well.

A compromise combining some of the advantages of all alternatives is that the system supplier - AVEVA - delivers information required for the set-up of an open standard. As long as customers stay with this standard it will to a large extent have the same benefit as the in-built standard but with an option for easy adaptation when needed.

6.3 Survey of Design Standards in AVEVA Hull

The different types of design standards in Hull are described in separate paragraphs below on a survey level. Details about each type of standard can be found in separate documents referred to. The standards are also roughly categorised according to the alternatives listed above.

6.3.1 Holes

Holes are closed openings in the interior of panels and in webs of profiles. AVEVA Marine has a number of in-built types of standard holes for the most commonly used ones (round, manholes, etc.). Arbitrary sizes of these holes may can be created by selecting appropriate parameters.

Any closed curve can be used as a hole when the in-built standard is insufficient.

Quite new types of standard holes must be added by AVEVA.

6.3.2 Notches

Notches are normally small openings in edges and corners of panels and along traces of profiles. AVEVA Marine has a very large number of in-built notch types. Arbitrary sizes of these notches can be generated if the user selects appropriate parameters.

Any open curve intersecting the outer contour of a panel (or edge of a profile web) can be used as a notch when the in-built standard is insufficient.

Quite new types of standard notches must be added by AVEVA.

6.3.3 Profile Cutouts

Cutouts are in AVEVA Marine openings around penetrating profiles in plates and webs of profiles. The customer standard of cutouts can be set up in two different ways.

1. AVEVA Marine contains "patterns" for a number of more or less universally used cutout types. These can easily be instantiated by the customer (by defining clearances, gaps, etc.) to create a customer specific standard.
2. Cutouts that do not fit into any of these patterns may be described in geometry macros.

A customer can set up his cutout standard by a combination of these two methods. Thus, the cutout standard is completely open.

6.3.4 Clips

Clips (lugs, collars) are small plate pieces, associated with cutouts. The clips standard must be established in geometry macros in much the same way as cutouts in alternative 2 above. Thus, the clip standard is quite open.

Clips in AVEVA Marine cannot have any holes an can currently not be associated with cutouts in profiles.

6.3.5 Profile Types

The profile standard is an example of standard that currently is closed in AVEVA Hull supports only a restricted number of standardised and commonly used profile types, identified by reserved type numbers and with predefined size parameters that must be given in a certain order. Arbitrary profile sizes can be used for most profile types. New profile types must be implemented by AVEVA.

6.3.6 Profile Endcuts

AVEVA Marine has an in-built standard for end treatment of profiles. This standard contains a large variety of endcut types for different types of profiles. The standard is partly open since new instances may be created by the customer of the endcut types supported by AVEVA Marine, e.g. with differences in notch sizes, fixed angles in flanges, etc.

A set-up file for definition of instances of all supported endcut types is delivered together with Hull.

Quite new endcut types must be implemented by AVEVA.

6.3.7 Profile End Connection Types

The endcut standard describes how a profile should be shaped at its ends. There is an additional standard related to the ends of profiles, namely details about how they should be connected to other members they are attached to. This standard is called the connection codes of AVEVA Marine. AVEVA Marine is delivered with a comprehensive set of connection codes. However, a customer may create his own instances of supported basic pattern of connection. From a practical point of view, this standard can be considered to be completely open.

6.3.8 Stiffener Connections

The automatic generation of panels in Structural Design is made on RSOs with properties. One og these properties is the stiffener connection defined as being either TIGHT, SNIPE or OVERLAP. The underlying connection between end connection types and profile endcuts into these stiffener connection types for different connections is stored in a standard object named __SBED_STIFF_ENDS__ .

An example set-up file is delivered together with the Template Project in AVEVA Marine.

6.3.9 Folded Flanges

The concept "Flange" is in AVEVA Marine used for both welded flanges ("face plates") and folded (or bent) flanges. Welded flanges are treated like other profiles when it comes to standards. Folded flanges may vary both regarding their general characteristics (e.g. bending radius) and the shape of the flange ends in a way that requires adaptation for the customer.

In AVEVA Marine there is a toolbox that may be used to control /set up standards for both the general characteristics of a flange and the end shape. For the end shapes the customer may select from a restricted number of types that may be instantiated in any number with special settings of angles, radii, position of knuckle points, etc.

To the extent the basic types supported by AVEVA Marine meet the customer needs this standard is open.

6.3.10 Brackets

Brackets are rather complex when considered as standardised elements. Different factors affecting a bracket are:

- Its outer contour that - in turn - consists of bracket toes of different types, connected by a free side.
- The bracket may have notches at the corners.
- The bracket may be stiffened, either by a welded or folded flange or by stiffeners.
- The bracket may have standardised connections to the surrounding hull structure.
- The bracket may be associated with rules for how its size should be derived from the surrounding structure.

All these aspects are considered in the AVEVA Marine facility for set-up of a customer bracket standard. The toolbox for brackets consists of standardised toes and standardised connections with rules for how certain parameters of the bracket should be calculated automatically. In addition the standards for notches, profiles and flanges as described above are used also in the customer set-up of brackets.

Standard brackets can currently not have holes and cutouts for penetrating stiffeners.

Even if standards can be set up for the majority of a customer's brackets there will always be some that are so special that they cannot be set up as a standard. Such brackets may be generated as so called panel brackets.

The bracket facility of AVEVA Marine allows a customer to define a bracket standard of his own. In doing that some parameters of the brackets are given specified values, e.g. toe heights, toe lengths, etc. whereas other parameters are supposed to be evaluated or given by the user when the bracket is used.

A set of bracket parameters for a certain bracket type is said to form an instance of the bracket type.

6.3.11 Bevels

AVEVA Marine has an advanced toolbox for the definition of a customer standard for bevelling of edges in both plates and profiles.

This toolbox contains separate basic bevel types for butt welding, for fillet welding with fixed bevel angles and for fillet welding with varying bevel angles. The latter type of bevels may be automatically selected as a function of both the plate thickness and the connection angle. The customer set-up of the bevel standard means instantiation of the basic bevel types.

Associated with the bevels themselves are also options for control of the layout of bevel notes in drawings and burning sketches.

6.3.12 Swedging

Swedging is the AVEVA Marine name of the type of small corrugation used e.g. in superstructures of ships. AVEVA Marine has a toolbox that supports a customer in defining his own standards with respect to swedging, regarding:

- Type,
- Indication in drawings,
- Material to be added to nominal plate sizes for the swedging.

6.3.13 Knuckling of Plates

Knuckling in relation with folded flanges is considered together with flanges (see above). For arbitrary knuckling in plates AVEVA Marine has a facility that allows a customer to define different bending radii and - for each bending radius - the stretching/compression as a function of the bending angle.

6.3.14 Shrinkage Compensation

AVEVA Marine has an advanced facility to compensate parts (both plates and profiles) for shrinkage. The compensation can be controlled by manual input but is normally supposed to be evaluated and applied automatically from tables that may be loaded with shrinkage values according to the experiences of the individual yard.

7 Co-ordinate Systems and Relative Positions

7.1 Co-ordinate System

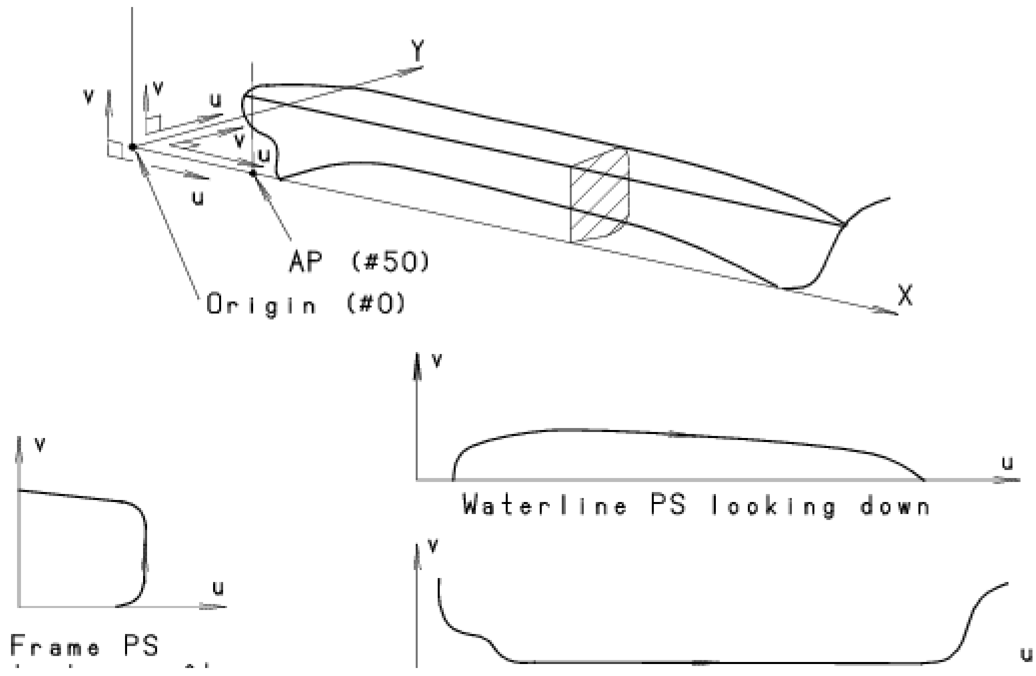
All the parts or objects of a ship or any other product described in AVEVA Marine are positioned in a co-ordinate system. A co-ordinate system may be local and positioned in other co-ordinate systems in hierarchies with any level of depth. However, in the end all local co-ordinate systems must be positioned relative to one global co-ordinate system that is located in the product in a well defined way.

In the case of the AVEVA Hull application the product is normally a ship or a ship like vessel. In that case there are rather strict rules for how the global co-ordinate system is located (see the figure below).

- The origin should be placed in the aft end of the ship, preferably in the bottom and in the symmetry plane (the Centre Line plane (=CL)). It is a good practice shown in the figure (but not necessary!) to locate the origin aft of the ship so that all x-co-ordinates are positive. Another alternative is to place the origin at the aft perpendicular.
- The x-axis should be directed in the forward direction.
- The y-axis should be in the direction against portside.
- The z-axis should be in the upward direction (co-ordinates may be negative)

Planar parts of the hull model (e.g. plane panels, brackets, etc.) are described in the uv-plane of local co-ordinate systems (uvw). These co-ordinate systems may be located quite arbitrarily in the ship co-ordinate system (xyz) of the ship. However, when the uv-plane coincides with a principal plane of the xyz-system the local co-ordinate system is as a standard positioned as shown in the figure below. In these cases the u- and v-co-ordinates will be identical to one of the x-, y- or z-co-ordinates which may be convenient. However, this is a matter of convenience and is in no way compulsory.

The co-ordinates are (in the Hull data banks) always expressed in millimetres.



7.2 Relative Positions

A ship is a big product and the co-ordinates may get so big numerical values (especially in the longitudinal direction of the ship) that it becomes impractical to use them directly. This problem could be solved by use of properly located local co-ordinate systems. However, there is a traditional way of specifying the longitudinal position in ship-building that eliminates this problem. Specified positions along the x-axis are assigned a frame number and the longitudinal position is most often specified as a distance for or aft of a specified frame position. In Hull it is possible to specify the longitudinal position accordingly and a similar facility has also been implemented for distances from the Centre Line and above the Base Line.

7.2.1 Frame Positions

In accordance with traditional shipbuilding practice a ship has a varying number of frames, each with an associated position along the x-axis. It is then possible to specify the x-position by e.g. X=FR45 +100 or X=FR45 -150 which means positions 100 forward of and 150 aft of frame position 45, respectively.

The following rules apply to the frame numbering in AVEVA Marine.

- The frames must always be numeric, i.e. they must not contain any letters. On the other hand they may be negative.
- The number of the frames should be in the range [-899,2766]
- The total number of frames is currently restricted to 500, unless the frames are consecutively numbered. In the latter case the frames may have numbers in the range [-99,500], i.e. 600 in total.
- The relation between frame number and frame position may be quite arbitrary, e.g. they may be increasing with increasing x-co-ordinates, decreasing with increasing x-co-ordinate or set without any specific order with relation to the frame position.
- The distance between frames may vary arbitrarily.

- The frame numbers and the corresponding positions are stored in a table, set up by the customer.

Remarks:

- It is conventional in shipbuilding to locate frame number 0 at the aft perpendicular and to let the frames in the aft peak be identified by letters: A, B, C, etc. The rules above do not allow this denomination. Recommendation: replace the letters by negative numbers (A--> -1, B--> -2, etc.).
- In some regions of the world it is customary to have numbered frames only at web frames and to identify intermediate frames by adding letters to the main frame number, e.g. 56, 56A, 56B …., 57, 57A, 57B, … . Recommendation: Change this example to 56, 561, 562,…., 57, 571, 572, … (or to 560, 561, 562, …., 570, 571, 572, …).

7.2.2 Distances from BL and CL

Frame positions are in most cases defined at those locations along the ship where there are transversal hull members, either frames or webs, etc.

In a similar way there are in most ships characteristic distances from the Centre Line (CL) and above the Base Line (BL) where hull members are located. E.g. longitudinals in the bottom and in the side in the midship section are located at positions which normally also define the position of stiffeners in decks, platforms, bulkheads, etc., and the position of girders. By referring to these positions one may define locations along the y- and z-axes as simple as e.g. Y=LP10 +100 and Z=LP35 -100. (LP10 +100 means 100 mm in portside direction from Longitudinal Position number 10 in the bottom, LP35 -100 means 100 mm below Longitudinal Position 35 in the side).

From a practical point of view it is recommended to let the longitudinal positions and their numbers coincide with the numbers and positions of actual longitudinals in the midship section. However, it should be noted that the longitudinal positions form a grid that need not have any direct relation with the physical longitudinal frames. E.g. if some longitudinals are replaced by girders there are "holes" in the numbering of longitudinals. However, the longitudinal positions should include all the positions, also those where there are no longitudinal frames. The figure below shows schematically a typical midship frame with suggested longitudinal positions.

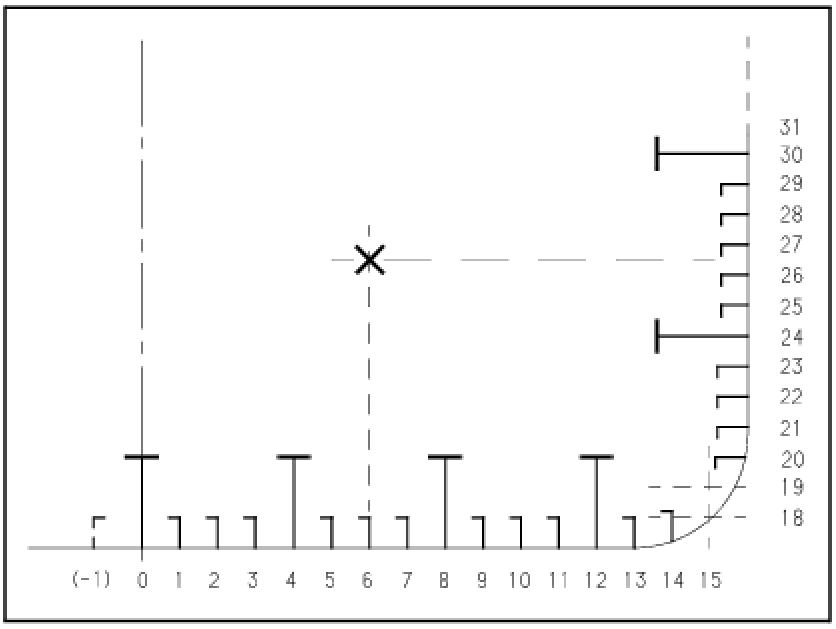


Figure 7:1. Midship section with longitudinal positions.

The point as the cross in the figure above may be located by Y=LP6, Z=LP26.5

The following rules should be considered:

- The positions and the numbers should be related to those of actual longitudinal frames, if possible.
- The longitudinal numbers should be in the interval [0,999]
- The numbers for horizontal positions (along the y-axis) and vertical positions (along the z-axis) should not be the same.
- It is quite possible to define a longitudinal position in the CL plane, i.e. where y=0. This position may have number 0.
- The relation between increasing/decreasing numbers and increasing/decreasing distances is arbitrary similar to what is stated for frames. This should be decided by the rules for longitudinal numbering, used by the yard.
- The longitudinal numbers and their positions are stored in a table set up by the customer (thus, there is no direct connection to the generated physical longitudinal frames).
- Longitudinal positions in the bottom are normally only defined on portside. Reference to the corresponding positions on the starboard side is done by negating the longitudinal number, e.g. Y=LP-20+100.

7.3 Set-up of Co-ordinate Tables

The tables for frame and longitudinal positions are defined via an ordinary text file in a special format, described below. This text file is read by a function of the hull utility *inithull* and if the input is correct a resulting table object __SBH_GENTAB__ will be stored in the structure data bank (associated with the environment variable SB_OGDB). The name of the file may be arbitrary.

The input file is organised in "record types" with identical layout as described below. The format is free but it is recommended to have one record per line. The line width is limited to 80 characters. The number of records is unrestricted.

Record layout:

```
<type>  <F_No>  <step_No>  <L_No>  <F_coord>  <step_coord>
```

The interpretation of the terms are:

`<type>`	The type of coordinates: = 20: Frame coordinates = 30: Horizontal longitudinal positions (along y-axis) = 40: Vertical longitudinal positions (along z-axis).
`<F_No>`	Number of the frame, etc. which the First co-ordinate (`<F_coord>`) is associated with.
`<step_No>`	Steps by which the numbers should be incremented.
`<L_No>`	Number of the frame, etc. that should be the last number assigned a coordinate by this record
`<F_coord>`	The coordinate value assigned to `<F_No>`.
`<step_coord>`	Steps in coordinates.

This should be interpreted in the following way:

Frame `<F_No>` will get co-ordinate `<F_coord>`,
frame `<F_No>+<step_No>` will get co-ordinate `<F_coord>+<step_coord>`,
etc.

The order between the records is arbitrary but it is recommended to give record types of the same type in one sequence.

Example:

The example shows a simple case with constant partitions between both frame and longitudinal positions. Frames with partition 745 start in frame number -10 at the x-co-ordinate -7450 and the last frame number is 295. Longitudinal positions are as in the figure above with partition 750 and 675, respectively.

```
20   -10   1   295   -7450.0   745.0
30     0   1    15       0.0   750.0
40    18   1    31     600.0   675.0
```

In addition to the updated table object its contents will also be listed. If *inithull* has been run via the Job Launcher (JL) the list file will be available according to the standard for runs via JL. If *inithull* has been run directly the resulting file will be stored in the print directory of the current project with the file extension. *lst* added to the input file name.

8 Run Mode Control

8.1 General

It is possible to customise and control the operation of the AVEVA Hull application in different ways. This control can take place on different levels.

1. The lowest level of control is on module or program level.
2. The second level of customer control is connected to specific "facility" of AVEVA Hull that may be common to several modules or programs.

   Examples of such features are the customisation of part names, bevel standard, standards for bent flanges, etc.
3. The highest level of control concerns the behaviour of AVEVA Hull on an overall level.

A brief overview of the principles for these levels of control is given in this document. On the highest level the document also specifies some parameters and their use.

8.2 Control on Module Level

The control on module or program level normally takes place via default parameters (sometimes called "ip's") which are gathered in a file (default or ip file), specific to the module. The default parameters are described in the documentation of the module in question.

Normally these Hull default files/ip files must be placed on the directory associated with SB_SHIP. This is explicitly valid for:

- ppanparts
- bendtempl
- cpanparts
- profnest
- matplate
- jigpillar
- platejigs
- mark3axis
- wcog
- Nesting
- PLCM

Therefore, when the name of a default file is given e.g. as the value of an environment variables then the full path name shall not be given, only the file name. The rest of the path name will be fetched by the system from SB_SHIP.

The reason for this is that the name of the used default file is often stored in objects. As an example, in Nesting the default file name is stored in the nested plate object. If, for some reason, there is a need to modify the directory structure and thereby give a new name in the environment variable SB_SHIP then the stored file name will still be valid since it is not the full path name.

(For other types of files, such as data files, control files, etc. the full name must be given.)

8.3 Control on Facility Level

The concept *facility* is here used for special features that may be used in one or several different modules. The control on facility level may take place in different ways:

1. Via a file in ASCII format that specifies a special customer's rules. The file is read into the application program at run time and may normally be checked for errors by the *inithull* module.

   Examples are:

   - Set-up of connection codes,
   - Definition of default cutouts,
   - The association between a user number of a cutout and the macro that creates it.
2. Via objects stored in the hull structure data bank. These objects in turn may be created and checked by *inithull* and they are in most cases described in a language based on the general Interpreter Language (TIL) syntax. The resulting objects have normally fixed and predefined names but they should sometimes be controlled by environment variables.

The detailed conditions can be found together with the documentation of each separate facility.

8.4 Control on Application Level

The control of the application as a whole may take place in several different areas.

One of those is of course the environment, i.e. data banks, directories, etc. This is all documented and described together with the tools for project management.

Another matter of general importance for a project is to select names for the current project, to set up rules for frame numbering and spacing, ditto for the longitudinal spacing and numbering, etc.

These matters are normally handled by the module for initialisation of a hull project, called *inithull.*

Finally there are some means of controlling the overall behaviour of Hull application, mainly related to how it identifies, handles and stores parts that are symmetrical with respect to the centre line or specific for portside and starboard, respectively. This control takes place via a number of environment variables and the main purpose of this paragraph is to give a detailed description of their use and effect. The environment variables are normally supposed to be defined in the project file of the current project. The use of these variables is not always independent of each other as the description below will illustrate.

8.4.1 Symmetry Status of Panels

In the Hull application a panel may be symmetric, portside (PS) specific, starboard (SB) specific or be located in/over the centre line.

In a symmetric panel the definition of each part will result in two physical copies, one used on PS, the other on SB. Furthermore, and independent of the symmetry status of the panel, it may be stored in the panel data bank either on PS or SB.

The symmetry status of the panel must always be explicitly specified when the panel is modelled and a try to model a panel without it will be prevented.

The symmetry status is specified via special codes that in the Design Language are corresponding to the following keywords.

SBPS	Symmetric panel, valid in one copy for PS, another for SB
P	A PS specific panel.
S	A SB specific panel.
SP	A panel in/over the centre line (CL), e.g. a centre line girder or decks and platforms with parts on both sides of the CL.

P- and S- marked panels may well extend somewhat into the opposite half of the vessel but must not have complete parts stored on the "opposite" side. If so, they should be treated as SP panels instead.

(For curved panels there is a temporary exception to what is stated above. For them the symmetry must be controlled by suffixes of the panel names (P, S and SP corresponding to the equal codes above). Names of symmetrical panels should end in a digit. Moreover, all panels (except those over CL) should be described on PS.)

8.4.2 Storing of Parts from Plane Panels

As mentioned in the previous paragraph panels may be stored mirrored in the centre line plane compared to where they are physically located. E.g. a panel valid SB may be stored PS.

Then there are in principle two ways of handling the parts that are extracted from the panel. Either they may be stored in the same location as the panel or they may be stored where they are physically located.

AVEVA Marine supposes that the plate parts are stored where they are *physically located independently* of if the panel has been generated on the opposite side of the centre line.

8.4.3 Individual Storing of Parts from Symmetrical Panel

Each part in a symmetrical panel contains the "pattern" of two parts, one on portside (PS) and one on the starboard side (SB). When AVEVA Marine extracts parts from the panel individual instances of the parts on PS and SB are normally created automatically. These instances are stored where they physically belong independently of how they have been modelled.

However, there is an option to store only one instance of parts of symmetrical panels. By assigning the environment variable SBH_NO_IND_SYMPARTS to any value the individual storing of parts from symmetrical panels is inhibited. Then the parts will, from a storing point of view, be handled in parallel with the panel itself.

When symmetrical parts are stored individually the parts PS and SB are identified by adding a suffix of P and S, respectively, at the end of the name. This goes for both plate parts and profile parts. For plate parts over/in the centre line plane SP will be added at the end of the name.

Example:

The symmetrical panel AA123-4 will result in parts with the following names:

AA123-4-1P (plates)

AA123-4-1S

AA123-4-2P

AA123-4-2S

AA123-4-3P

.

AA123-4-1BP (bracket plates)

AA123-4-1BS

.

AA123-4/S1P (stiffener parts)

AA123-4/S1S

.

(The generated parts PS/SB need not be exact mirror images of each other. Also "symmetrical" panels may have minor differences PS/SB which will be transferred to one of the parts only. This may also be valid for some of the marking.)

8.4.4 Plate Parts from Panel Brackets

Type standard brackets are always generated individually even if many of them should happen to be identical. However, in a project there are normally a relatively large number of brackets that cannot be set-up as type standard brackets. In AVEVA Marine they may have to be generated as small panels, so called panel brackets as described e.g. in the *Design Language of Hull Modelling*. Panel brackets are normally individual, i.e. occurring in one instance only, but they may also be used in several places, even in different assemblies. However, when the plate part is extracted from the panel bracket there is only one part. On the other hand there is a need to have individual occurrences of this bracket part in all situations where it is used, possibly with different position numbers and assembly names and always with different locations.

AVEVA Marine uses in this situation a technique of creating individual almost empty parts for all instances of the panel bracket. These instances contain only some administrative information and a pointer to the plate extracted from the panel bracket itself. Thus these plate parts do not contain any plate geometry themselves and when accessed this information is inherited from the panel bracket plate part pointed to.

8.4.5 Compensation for Thickness of Shell Plates

Traditionally ships have had only one shaped hull and this hull has had an inner surface coinciding with the moulded surface. In such projects there is no (or at least little) need to consider the thickness of the shell plates when modelling parts against the shell.

However, the situation is different for vessels with double skins where both surfaces are non-planar or in vessels with a "planar" outer hull. In these cases it is essential that the plate thickness can be considered, preferably automatically. Such a feature should take into regard both the actual thickness and its "distortion" because of the (varying) angle along the section curve between the part and the surface.

Hull has a feature that considers both of these aspects. It has the following characteristics:

- Hull curves and traces of shell profiles will be intersected and stored in the model as intersections with the moulded surface.
- When a hull curve is used as a boundary of a panel it will be transformed to consider the shell plate thickness (and the intersection angle). This will also be done for the "implicit" curves, created when reference is made directly to the surface.
- Likewise, when a shell profile section is referred to, it will be moved to the correct side of the shell plate against which it is welded. This means that cutouts for it and stiffeners or brackets attached to it will get a correct size. (The movement from the moulded surface will occur in the plane of the trace curve, i.e. not necessarily in the direction of the web of the profile).

This feature is activated by setting the environment variable SBH_ENABLE_SHPLACOMP to any value. This should be done only in projects where the feature is relevant since it will have some implication on the system performance.

Remark:In simple cases and with simple geometry it is possible to consider the shell plate thickness by "manually" compensating for the plate thickness when modelling parts against the shell. However, as soon as references to shell profiles are involved the automatic feature must be activated.

The figures below show the result when a panel has been generated against the shell in a ship with the plate thickness inwards. In the left figure a try has been made to compensate for the plate thickness by a parallel movement of the "moulded" curve. In the right figure the new feature has been activated and the designer has not had to bother about the plate thickness.

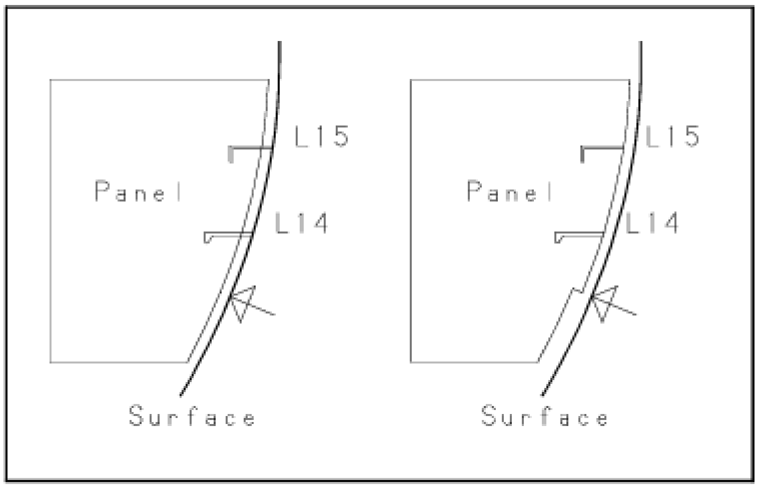


Figure 8:1. Panel against shell without and with automatic plate thickness compensation.

8.4.6 Reference Point of Profile Section

- **General**

  Profiles (stiffeners, longitudinals, transversals) are in most cases welded against a plate surface, normally perpendicular to the surface but often also with an inclination against the plate.

  Other components (connected stiffeners, brackets, clips and cutouts) are depending on the calculated position and the height of the profile, and therefore it is important that the position of the reference point of the profile is well defined. This point is normally placed on the lower edge of the web on the mould line side for different types of profiles.

  The purpose of this paragraph is to describe the options that are available in AVEVA Marine and to control how the reference point should be evaluated.

- **Optional Positions**

  AVEVA Marine offers three different options for placing of the reference point of a profile.

  1. It is supposed always to be located in the plate surface, see figure below, i.e. the fact that an inclination of the profile may "lift" the reference point from the plate surface is neglected. This is the default option.

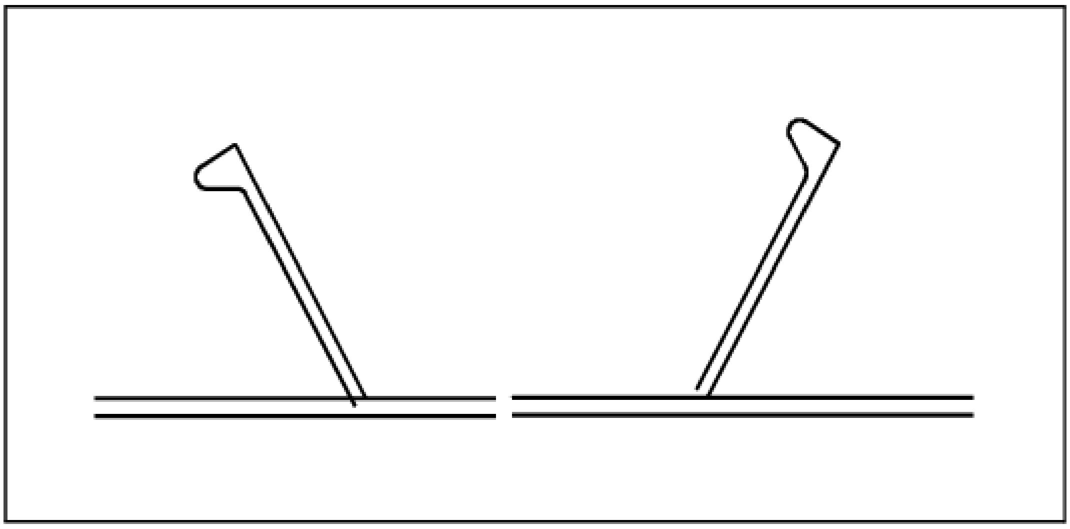


Figure 8:2. The default option for placing the reference point of a profile.

  2. The second option is that the reference point is located as though the edge of the web welded against the plate is rectangular, see figure below. Compared to option 1 there is only a difference when the angle between the profile web and the plate surface is larger than 90 degrees. In this case the profile section will be "lifted up" having the base of the non mould line side on the same level as the surface.

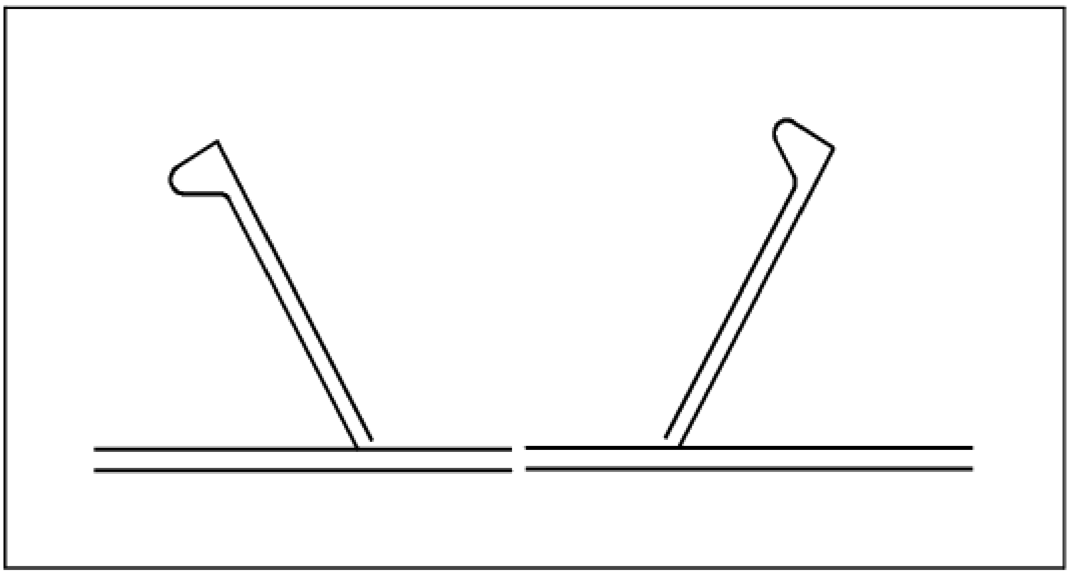


Figure 8:3. The reference point located as though the edge of the web welded against the plate is rectangular.

3. The third alternative is a compromise between the other two alternatives, namely that the midpoint of the web is located in the plate surface. This is the case both when the angle between the web and the surface is smaller or larger than 90 degrees.

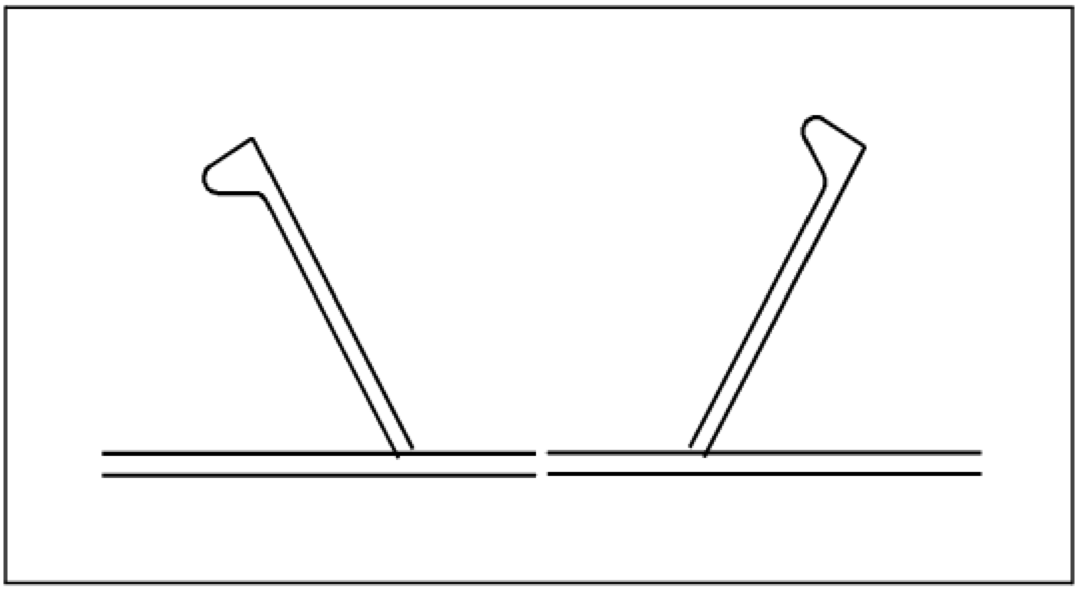


Figure 8:4. The midpoint of the web located in the plate surface.

- **Customer Selection**

  The option for profile sections is selected in AVEVA Marine by the environment variable SBH_PROFPT_ADJUST.

Option 1	is the default situation, i.e. will be in effect if the variable is unassigned.
Option 2	is selected by assignment of the value "TRUE".
Option 3	is selected by assignment of the value "MID".

9 Functional Descriptions

9.1 General about Functional Descriptions

Parts in the AVEVA Hull Product Information Model are identified by names and by different identification codes, used internally in AVEVA Marine. The names may carry some information about the part. At least when it comes to part names (the production oriented names) the name normally contains some information about which assembly the part belongs to.

To support more explicit description of parts AVEVA Hull also allows descriptive name *tags* to be assigned to Reference Surface Objects, to Panels or to parts of a panel. These tags are called functional descriptions and they are strings that *follow* the parts throughout the development of the project. In this document the term *Functional Description* is normally abbreviated to FD.

In addition to serving as pure name tags (which might be valuable in request functions like Info/Model and in parts lists), the FD:s may also be used to control certain automatic evaluations. An example of such a use is that the functional properties may control the automatic calculation of weld sizes or the automatic generation of steel (hull panels) from the Reference Surface Objects.

Below is given a general description of the functional descriptions in AVEVA Marine and the pre-defined descriptions built into AVEVA Marine.

9.2 User-defined Functional Descriptions

Functional descriptions are strings (up to 75 character long). They may be fixed strings but can also contain predefined parameters that may be replaced by actual values fetched from the part that the tag is fixed to. FD:s can be assigned to RSOs. panels and to physical parts in panels, i.e. to plates, stiffeners, welded flanges, pillars, brackets, doubling plates and clips.

Each single FD consists of two parts: The FD string and an functional code associated with that string. The code is an integer (in the range 0 -32767) that must be unique for a certain FD, i.e. there is a one-to-one relation between the string and the code. FD:s are stored in model objects via the code which thus is the "handle" by which the FD string is accessed.

An FD with code 0 is the default FD. This default FD is applied everywhere where no explicitly defined FD exists. An undefined FD (i.e. a code without any corresponding string) may be possible to use in some cases. Such an FD may be defined at a later stage, but if it is asked for before being defined an error message may be issued (depending on the function where it is asked for).

9.3 Pre-defined Functional Descriptions

AVEVA Marine is reserving the functional codes in the interval 9000 to 10999 and they will be automatically defined according to the description below when the functional description object is created

The pre-defined functional descriptions are divided into groups, represented by smaller intervals within the full interval specified above .

Any information if Watertight or Non-tight structures are given by the use of Data Types. Further information on this can be found in *Hull / Planar Modelling / Design Language / Panel Statement*.

9.3.1 Groups

The following groups of structures are used:

- Decks
- Longitudinal bulkheads
- Transversal bulkheads
- Girders and Stringers
- General web frame
- Shell
- Miscellaneous

- **Decks**

Decks are associated with Functional Codes in the interval 9000 to 9099.

Functional Code	Functional Description
9000	General Deck
9001	Weather Deck
9002	Upper Deck
9003	Main Deck
9004	Strength Deck
9005	Accommodation Deck
9006	Navigation Deck
9007	Platform Deck
9008	Deck in Superstructure
9009	Inner Deck
9010	Inner Bottom
9011	Double Bottom
9012	Tank Top

Table 1:

Functional Code	Functional Description
9013	Sloping Inner Deck
9014	Sloping Inner Bottom
9015	Forcastle Deck
9016	Poop Deck
9017	Cargo Deck

Table 1:

- **Longitudinal Bulkheads**

  Longitudinal Bulkheads are associated with Functional Codes in the interval 9100 to 9199.

Functional Code	Functional Description
9100	General Longitudinal Bulkhead
9101	Centerline Bulkhead
9102	Inner side Bulkhead
9103	Topside Tank Sloping
9104	Hopper tank Sloping
9105	Wash Bulkhead
9106	Corrugated Bulkhead

Table 2:

- **Transversal Bulkheads**

  Transversal Bulkheads are associated with Functional Codes in the interval 9200 to 9299.

Functional Code	Functional Description
9200	General Transversal Bulkhead
9201	Aft Peak Bulkhead
9202	Collision Bulkhead
9203	Wash Bulkhead
9204	Corrugated Bulkhead

Table 3:

- **Girders and Stringers**

  Girders and Stringers are associated with Functional Codes in the interval 9300 to 9399.

Functional Code	Functional Description
9300	General Girder
9301	General Stringer
9302	Centerline Girder
9303	Deck Girder
9304	Topside Tank Vertical Strake

Table 4:

- **Web-frame Members**

  Web-frame Members are associated with Functional Codes in the interval 9400 to 9499.

Functional Code	Functional Description
9400	General Web-Frame Member
9401	Floor
9402	Hopper Transverse Web
9403	Topside Tank Transverse Web
9404	Vertical Web
9405	Deck Transverse

Table 5:

- **Shell members**

  Shell Members are associated with Functional Codes in the interval 9500 to 9599.

Functional Code	Functional Description
9500	General shell plate
9501	Keel
9502	Bottom Shell
9503	Bilge
9504	Side
9505	Sheerstrake
9506	Gunwale
9507	Bilge Keel

Table 6:

- **Miscellaneous**

  The miscellaneous structures are associated with Functional Codes in the interval 9600 to 9699.

Functional Code	Functional Description
9600	General miscellaneous
9601	Hatch Coaming
9602	Hatch Cover
9603	Stool

Table 7:

10 XML Interfaces

There is a growing need to build interfaces between shipbuilding software systems so that product data defined in one system can be used in other systems. To support this, data formats have been defined that can be accepted by the industry and implemented with reasonable efforts.

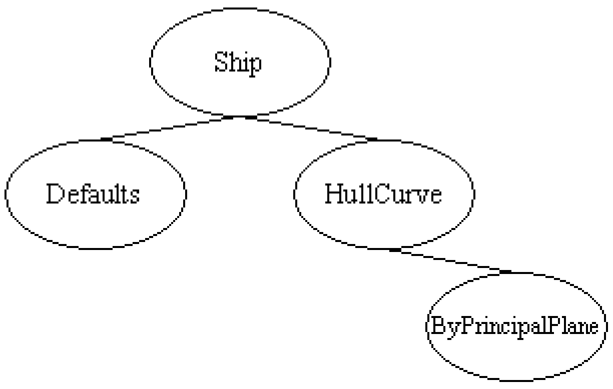
10.1 Introduction to XML

XML stands for eXstensible Mark-up Language. It originates from the more advanced data format SGML. XML is often mentioned together with web technology and HTML but it is important to remember that XML is basically a way to *describe and structure data*. It can be used in a great variety of applications, not only in web applications.

XML is a self describing data format where data is marked with “tags”. Let us look at a simple example:

```
<?xml version="1.0" encoding="UTF-8"?>
<Ship>
   <Defaults Surface="SPHULL" XMin="FR40" YMin="0" XMax="FR80"/>
   <HullCurve ObjId="SPX951">
      <ByPrincipalPlane X="FR30"/>
   </HullCurve>
</Ship>
```

“<Ship>” is a tag as well as “<Defaults>”. An XML document is organized as a tree structure with one single root element:



In this example “Ship” is the root element. Is has two child elements: “Defaults” and “HullCurve”. “HullCurve” has also a child element: “ByPrincipalPlane”.

10.1.1 Tags, elements and attributes

The basic building blocks of a XML file are *elements* and *attributes*. Let us look at the sample file again:

```
<Ship>
   <Defaults Surface="SPHULL" XMin="FR40" YMin="0" XMax="FR80"/>
   <HullCurve ObjId="SPX951">
      <ByPrincipalPlane X="FR30"/>
   </HullCurve>
</Ship>
```

In this file “Ship”, “Defaults”, “HullCurve” and “ByPrincipalPlane” are elements. The data within the element tags are called attributes. The “Defaults” element for instance, has four attributes: “Surface”, “XMin”, “XMax” and “YMin”. “Surface” is the *attribute name* and “SPHULL” is the *attribute value*. Please note that an attribute value must be enclosed within double quotes.

It is the attributes that actually carry the data. Elements organize the attributes into logical groups. If you compare an XML document to a file system, the elements are the directories and the attributes are the files.

An element in the XML file is represented by one or two tags. In this example there are two:

```
<Ship>
</Ship>
```

<Ship> is called a “start-tag” and “</Ship>” is the “end-tag”. In XML, every start tag must have a corresponding end tag.

Important: XML is case sensitive: <Ship>, <SHIP> and <ship> will be regarded as three different tags.

There is an alternative way to write an element, with only one tag:

```
<Ship/>
```

This is called an empty element tag, because when an element written like this it may not have any child elements.

10.1.2 Well-formed XML

When an XML document applies to all the rules that is set up for a valid XML document, it is said to be a *well-formed* XML document. We have already mentioned some of those rules:

- All start tags must have a matching end tag
- XML is case sensitive
- Attribute values must be enclosed within double quotes

There are many more rules. The complete definition of XML can be found on the web site of the World Wide Web Consortium: www.w3c.org

In this document we will only mention the basics.

Free Layout

The layout of an XML document is free. An XML parser does not care about new line and extra blanks. The following two documents are equivalent:

Document 1

```
<Ship>
   <Defaults Surface="SPHULL" XMin="FR40" YMin="0" XMax="FR80"/>
   <HullCurve ObjId="SPX951">
      <ByPrincipalPlane X="FR30"/>
   </HullCurve>
</Ship>
```

Document 2

```
<Ship><Defaults Surface="SPHULL" XMin="FR40" YMin="0"   XMax="FR80"/
><HullCurve ObjId="SPX951"><ByPrincipalPlane X="FR30"/></HullCurve></Ship>
```

Nested tags are not allowed

Another special feature of XML is that tags may not be nested, the following document is not well-formed XML:

```
<Ship>
<HullCurve>
</Ship>
</HullCurve>
```

Free order of attributes

The attributes of an element can always be given in an arbitrary order. The next two elements are equivalent:

```
<Box XMin="FR10" XMax="FR100" YMin="0" YMax="5000"/>

<Box YMax="5000" YMin="0" XMax="FR100" XMin="FR10"/>
```

10.1.3 Vocabularies

In XML the tags are not predefined, an application must define its own tags. A set of tags used by a specific application is often referred to as an “vocabulary”. When describing a vocabulary you typically state:

- What elements and attributes are accepted by the application
- In what order the elements must be given
- If elements/attributes are required or optional
- Data types of attributes.
- Minimum and/or maximum value attributes
- Default values of attributes

The vocabulary of an application can be described in a separate document. There are several formats for describing an XML vocabulary of which the most common ones are:

- DTD. Stands for “Document Type Definition”. This is one of the first formats that were created for defining vocabularies.
- XML Schema. This format is developed by the world wide web consortium, W3C.
- XML Data Reduced. This is a subset of XML Schema.

If an XML document should be validated against a special vocabulary it must have some kind of reference to this vocabulary. You will find this reference in the root element of the XML document, in this example an XML schema:

```
<Ship xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

```
xsi:noNamespaceSchemaLocation="CurvedHull.xsd">
...
</Ship>
```

Here the root element “Ship” has two special attributes. The first one, called “xmlns:xsi” is actually a reference to the definition of XML Schema language itself. The second attribute, “xsi:noNamespaceSchemaLocation“, points out the schema file defining the vocabulary that this document applies to.

10.1.4 Create and Edit XML Files

Since XML is a text file it can be created and edited in any text editor such as “Notepad” or “Wordpad”. However, there are many XML tools available on the market that provides “intelligent” editing of an XML file. The tools may for instance:

- Check that the file is a well-formed XML document.
- Present the document in a “tree view”, that can be expanded/collapsed.
- Automatically add closing tags when editing
- Validate the document against its vocabulary, if any.
- Support “context-sensitive editing”. When you edit a document that applies to a special vocabulary and you want to insert something new, the tool will let you choose among all element/attributes that are valid in the current context. This will very much facilitate the editing if the vocabulary is complex.

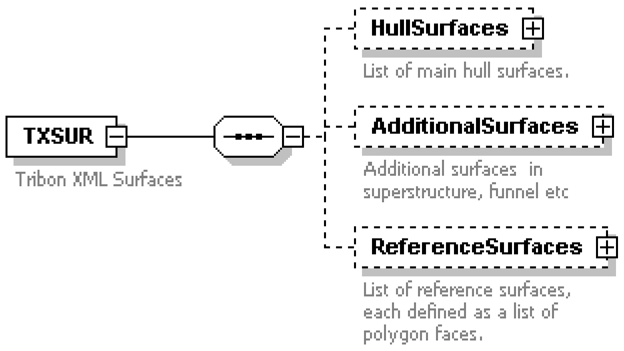
There are many XML tools available on the market as freeware, shareware or commercial. Some of them offers all the functionality described above while others only support parts of the functionality.

10.2 TXSUR

TXSUR is used to describe surfaces in the system. Currently there are three major types, HullSurface, AdditionalSurface (funnel etc) and ReferenceSurface (RSO’s).

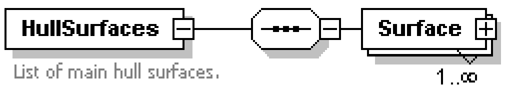
10.2.1 Basic Syntax of TXSUR

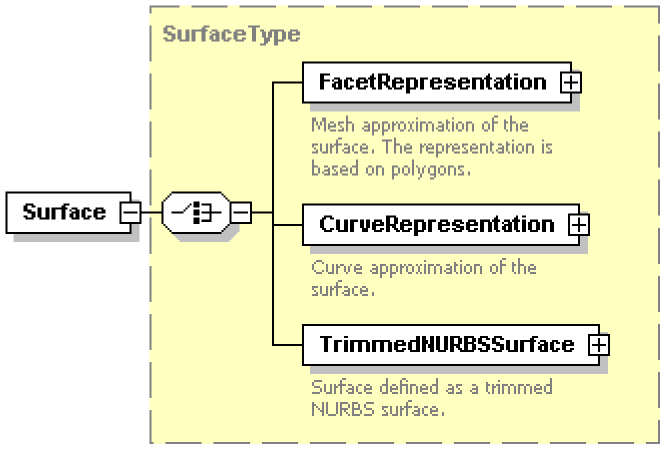
The XML format has one root element, TXSUR, with a number of different child elements containing different surfaces used in the model.

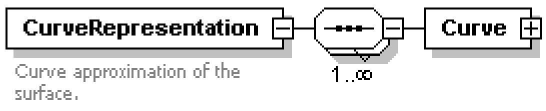

TXSUR
Tribon XML Surfaces
HullSurfaces
List of main hull surfaces.
AdditionalSurfaces
Additional surfaces in superstructure, funnel etc
ReferenceSurfaces
List of reference surfaces, each defined as a list of polygon faces.

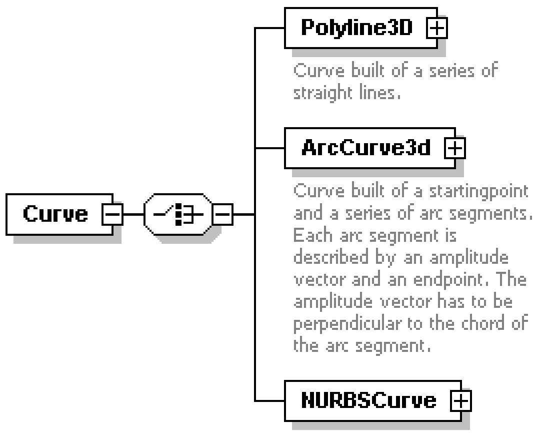
- **HullSurface Element**

  Here the hull surface is defined. Currently, (M3SP4), only **CurveRepresentation** is available. The other representations are yet to be implemented.




SurfaceType
Surface
FacetRepresentation
Mesh approximation of the surface. The representation is based on polygons.
CurveRepresentation
Curve approximation of the surface.
TrimmedNURBSSurface
Surface defined as a trimmed NURBS surface.



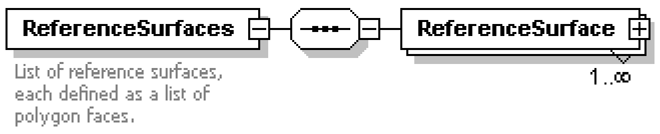

Curve
Polyline3D
Curve built of a series of straight lines.
ArcCurve3d
Curve built of a startingpoint and a series of arc segments. Each arc segment is described by an amplitude vector and an endpoint. The amplitude vector has to be perpendicular to the chord of the arc segment.
NURBSCurve

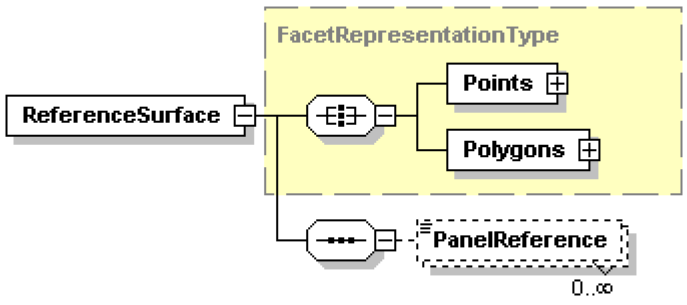
- **AdditionalSurfaces Element**

  Here additional surfaces are defined, such as funnels and others. Currently, (M3SP4), only **CurveRepresentation** is available. The other representations are yet to be implemented. The format is the same as for **HullSurface**.

- **ReferenceSurfaces Element**

  Here all the Reference Surfaces (RSO's) are defined.


ReferenceSurfaces
List of reference surfaces, each defined as a list of polygon faces.
ReferenceSurface
1..∞



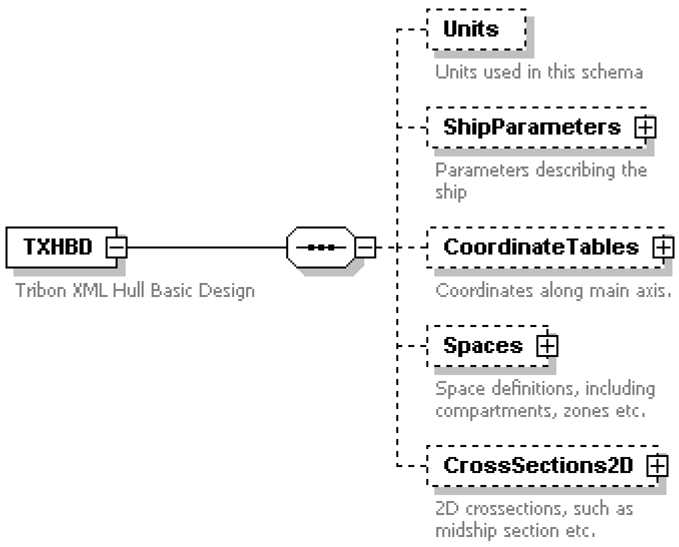
10.3 TXHBD

TXHBD is used to describe the model in an Early Design oriented way. The format includes Units (the units used in the schema), Ship Parameters (LOA, LWL etc), CoordinateTables (frame table and longitudinal tables), Spaces (compartments etc) and CrossSections2D (a collection of transverse sections).

See also Schema_TXHBD.chm

10.3.1 Basic Syntax of TXHBD

The XML format has one root element, TXHBD, with a number of different child elements containing different aspects of the model.


Units
Units used in this schema
ShipParameters
Parameters describing the ship
TXHBD
Tribon XML Hull Basic Design
CoordinateTables
Coordinates along main axis.
Spaces
Space definitions, including compartments, zones etc.
CrossSections2D
2D crossections, such as midship section etc.

- The **Units** element describes the units used in this schema.
- The **ShipParameters** element contain general data about the ship, such as speed and length between perpendiculars.
- **CoordinateTables** contain frame and longitudinal coordinate tables.
- The **Spaces** element describes the ships division into zones and compartments.
- The **CrossSections2D** contain a collection of transverse sections, such as midship section.

- **Units Element**

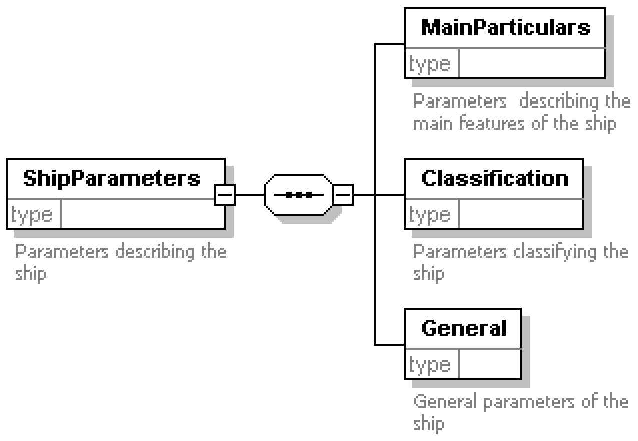
  Here the units for length, weight, area and so on are defined.

Example:

```
<Units Length="mm" Area="mm2" Volume="mm3"
     Weight="kilogrammes" Angle="degrees" Velocity="knots"/>
```

- **ShipParameters Element**

  Here overall parameters for the ship are stored.

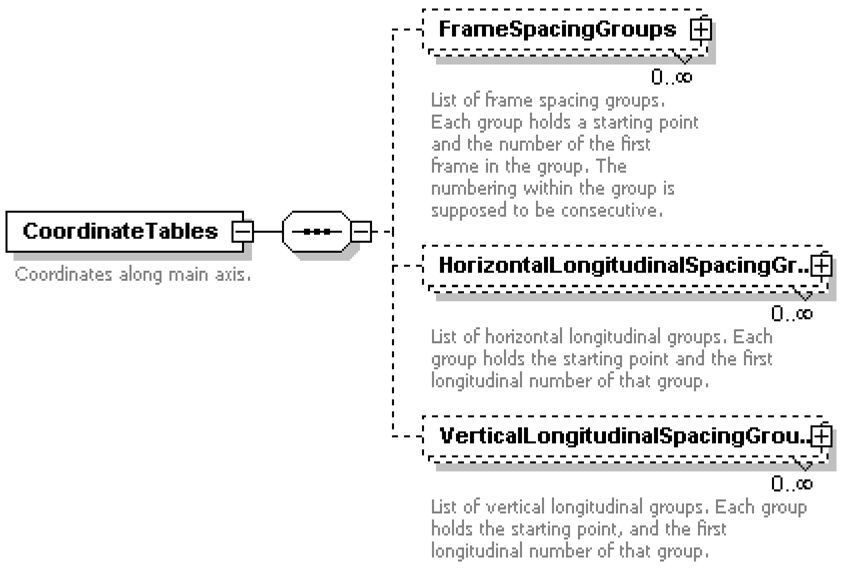

MainParticulars
type
Parameters describing the main features of the ship
ShipParameters
type
Parameters describing the ship
Classification
type
Parameters classifying the ship
General
type
General parameters of the ship

Example:

```
<ShipParameters>
    <MainParticulars
         LBP="2.1400000000E005" LOA="2.1900000000E005"
         LWL="2.1600000000E005"
         RuleLength="2.1000000000E005"
         Draft="2.5600000000E004" Depth="6.0000000000E003"
         BOA="2.1000000000E005" BWL="2.0800000000E005"
         Cb="6.9000000000E-001" Cx="9.7000000000E-001"
         Cp="7.0700000000E-001" Cw="7.7000000000E-001"
         LCF="2.0000000000E004" LCB="1.9000000000E004"
         BilgeRadius="2.5000000000E003"
         RiseOfFloor="0.0000000000E000"
         RakeOfKeel="0.0000000000E000"
         SternOverhang="5.5000000000E003"
         StemOverhang="4.0000000000E003"
         ShellThickness="1.9000000000E001"
         HalfSiding="3.0000000000E004"
         MinimumZ="0.0000000000E000"/>
    <Classification
         Yard="The yard" YardNo="214" ClassSociety="Class"
         IMONumber="1234" ShipName="THESHIP"
         ShipType="Container" CallLetters="CallMe"
         Flag="FLAG" PortOfRegistry="Yokohama"
         GrossTonnage="18900" NetTonnage="8500"
         MouldedDisplacement="8.4000000000E003"
         LightShip="1.0000000000E004" ServiceArea="Tropics"
         Owner="The Carrier Co" YearOfBuild="1976"/>
    <General
         MaxSpeed="1.9500000000E001"
         ShipBuildingProjectName="A1024"/>
</ShipParameters>
```

- **CoordinateTables**

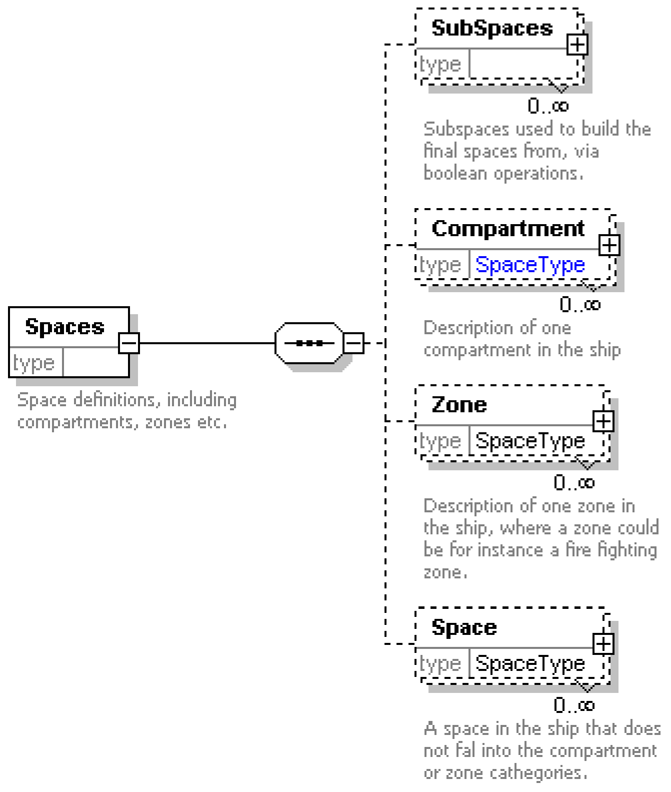
  Here the different tables for defining spaces along the main axis are stored.

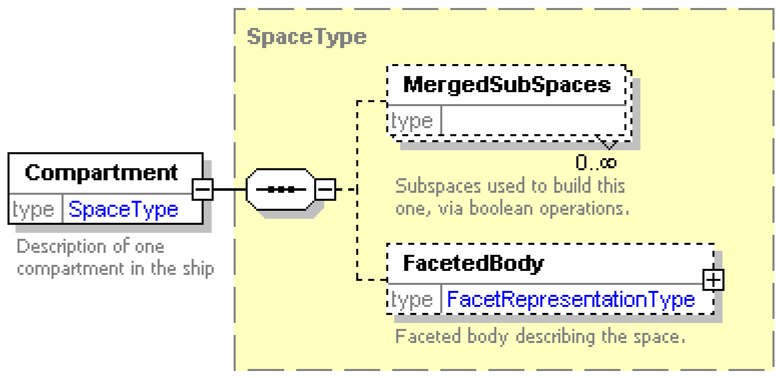

CoordinateTables
Coordinates along main axis.
FrameSpacingGroups
0..∞
List of frame spacing groups. Each group holds a starting point and the number of the first frame in the group. The numbering within the group is supposed to be consecutive.
HorizontalLongitudinalSpacingGr..
0..∞
List of horizontal longitudinal groups. Each group holds the starting point and the first longitudinal number of that group.
VerticalLongitudinalSpacingGrou..
0..∞
List of vertical longitudinal groups. Each group holds the starting point, and the first longitudinal number of that group.

- **Spaces Element**

  Here the division of the ship into compartments, zones and spaces is described.

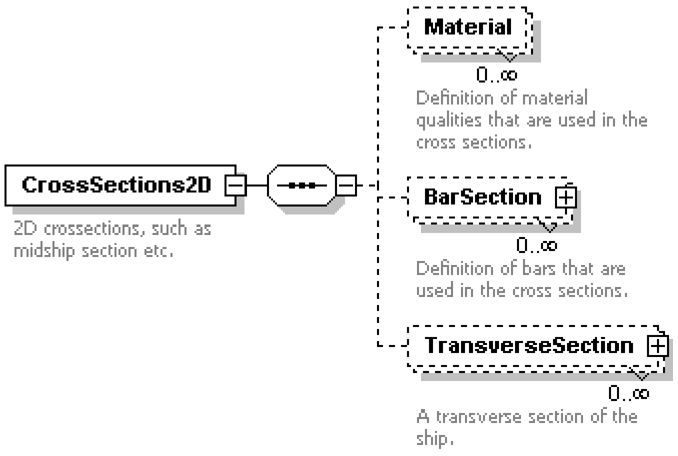
  The only portion of this element that is implemented so far is the compartment part with its attributes, and an accompanying facet body to define its volume.


SubSpaces
type
0..∞
Subspaces used to build the final spaces from, via boolean operations.
Compartment
type SpaceType
0..∞
Description of one compartment in the ship
Spaces
type
Space definitions, including compartments, zones etc.
Zone
type SpaceType
0..∞
Description of one zone in the ship, where a zone could be for instance a fire fighting zone.
Space
type SpaceType
0..∞
A space in the ship that does not fal into the compartment or zone cathegories.


SpaceType
MergedSubSpaces
type
0..∞
Subspaces used to build this one, via boolean operations.
Compartment
type SpaceType
Description of one compartment in the ship
FacetedBody
type FacetRepresentationType
Faceted body describing the space.

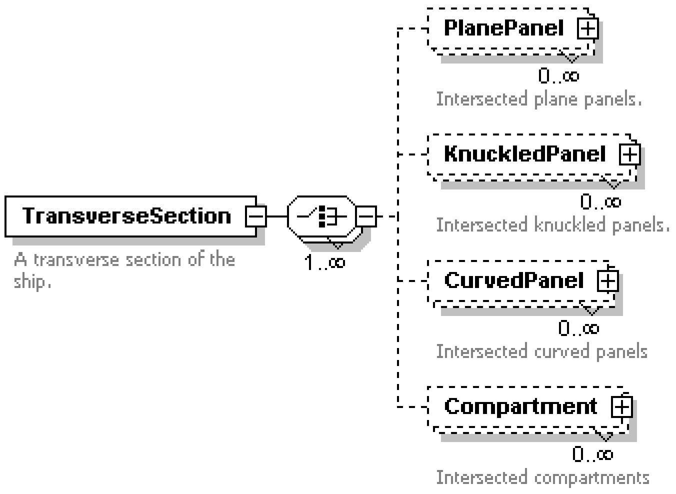
- **CrossSections2D Element**

  CrossSections2D is a collection of **TransverseSection** of the ship, together with the **Material** and **BarSection** used in the transverse sections.


Material
0..∞
Definition of material qualities that are used in the cross sections.
CrossSections2D
2D crossections, such as midship section etc.
BarSection
0..∞
Definition of bars that are used in the cross sections.
TransverseSection
0..∞
A transverse section of the ship.

- ## TransverseSection Element

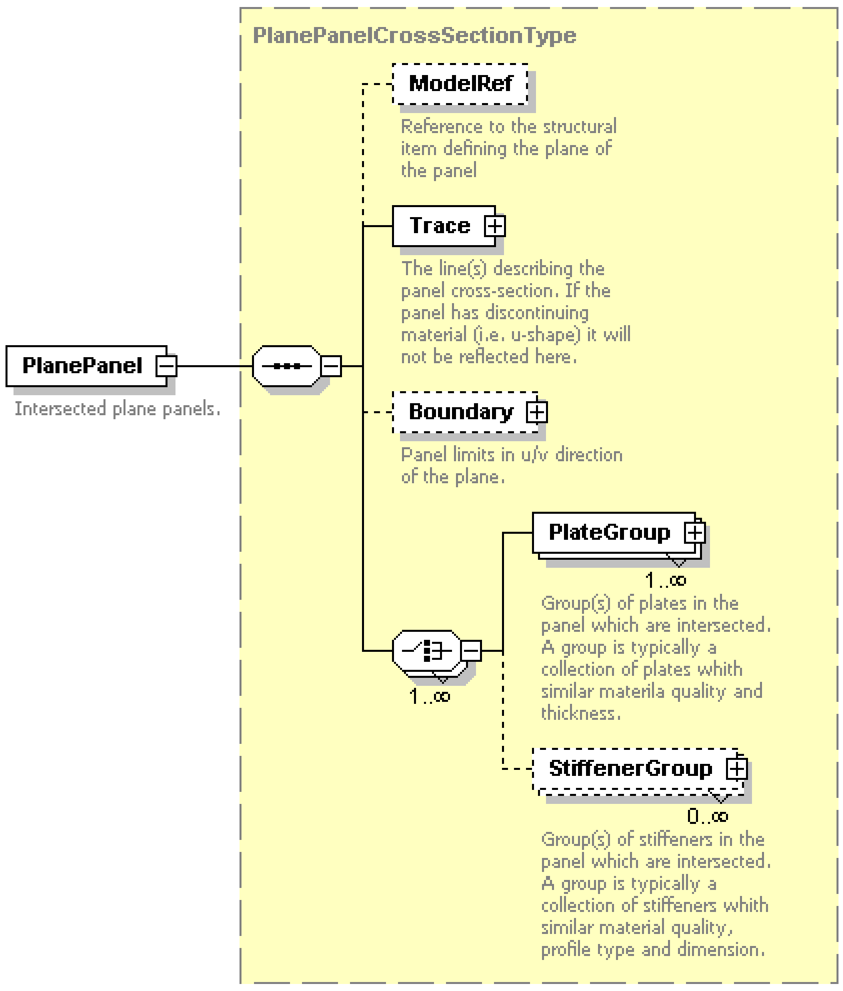
  Here a transverse section at a specific x coordinate of the ship is described. All the intersected panels and compartments are listed. The **TransverseSection** element has attributes for **Name** and **X** (x coordinate). **Name** is the x coordinate in the same format as input by user, i.e. FR88-125.


PlanePanel
0..∞
Intersected plane panels.
KnuckledPanel
0..∞
Intersected knuckled panels.
TransverseSection
A transverse section of the ship.
1..∞
CurvedPanel
0..∞
Intersected curved panels
Compartment
0..∞
Intersected compartments

- ## PlanePanel Element

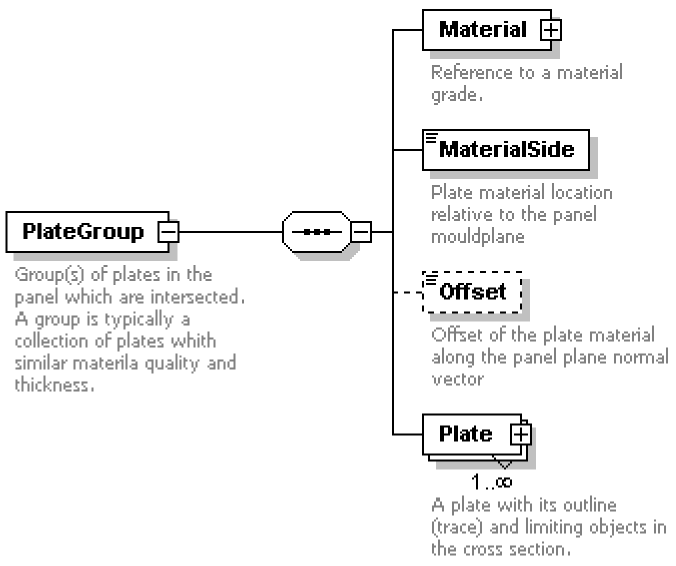
  PlanePanel describes an intersected plane panel. It has **Instance** attribute defining if it is reflected or not, **FunctionalProperty** stating the function and **Tightness** telling whether the panel is water tight or not.

  The **ModelRef** element holds a reference to a Reference Surface Object if exists. The **Trace** element describes the outline of the panel in the intersection plane. The **Boundary** element lists the referred objects, which limits the panel in the intersected plane, such as other panel, hull etc.


PlanePanelCrossSectionType
ModelRef
Reference to the structural item defining the plane of the panel
Trace
The line(s) describing the panel cross-section. If the panel has discontinuing material (i.e. u-shape) it will not be reflected here.
PlanePanel
Intersected plane panels.
Boundary
Panel limits in u/v direction of the plane.
PlateGroup
1..∞
Group(s) of plates in the panel which are intersected. A group is typically a collection of plates whith similar materila quality and thickness.
1..∞
StiffenerGroup
0..∞
Group(s) of stiffeners in the panel which are intersected. A group is typically a collection of stiffeners whith similar material quality, profile type and dimension.

- **PlateGroup**

The **PlateGroup** element holds the common data for a **PlateGroup**. **Material** refers to a material quality in **CrossSections2D**. **MaterialSide** enumerates the material side (aft, forward etc). **Offset** gives the offset value from the mould line.


Material
Reference to a material grade.
MaterialSide
Plate material location relative to the panel mouldplane
PlateGroup
Group(s) of plates in the panel which are intersected. A group is typically a collection of plates whith similar materila quality and thickness.
Offset
Offset of the plate material along the panel plane normal vector
Plate
1..∞
A plate with its outline (trace) and limiting objects in the cross section.

- **Plate**

  The **Plate** has a **Trace** element describing the outline and a **Limit** element(s) which holds the referred object and the intersection point for the limit in the plane.

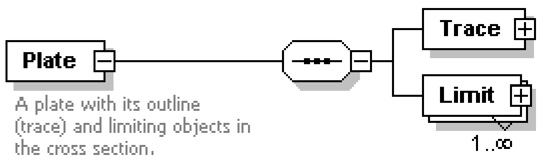
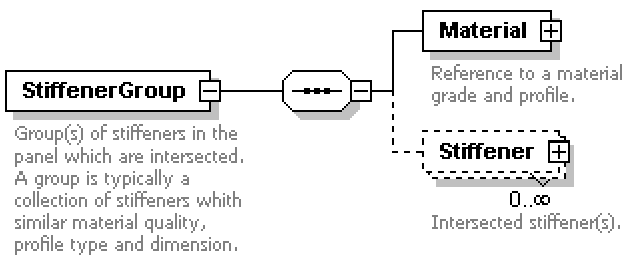


Plate
A plate with its outline (trace) and limiting objects in the cross section.
Trace
Limit
1..∞

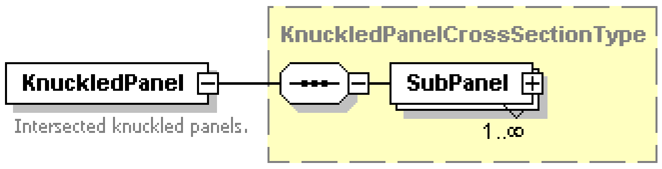
- **StiffenerGroup**

  The **StiffenerGroup** element holds the common data for a **StiffenerGroup**. The **Material** attribute refers to a material quality and profile in **CrossSections2D**.


Material
Reference to a material grade and profile.
StiffenerGroup
Group(s) of stiffeners in the panel which are intersected. A group is typically a collection of stiffeners whith similar material quality, profile type and dimension.
Stiffener
0..∞
Intersected stiffener(s).

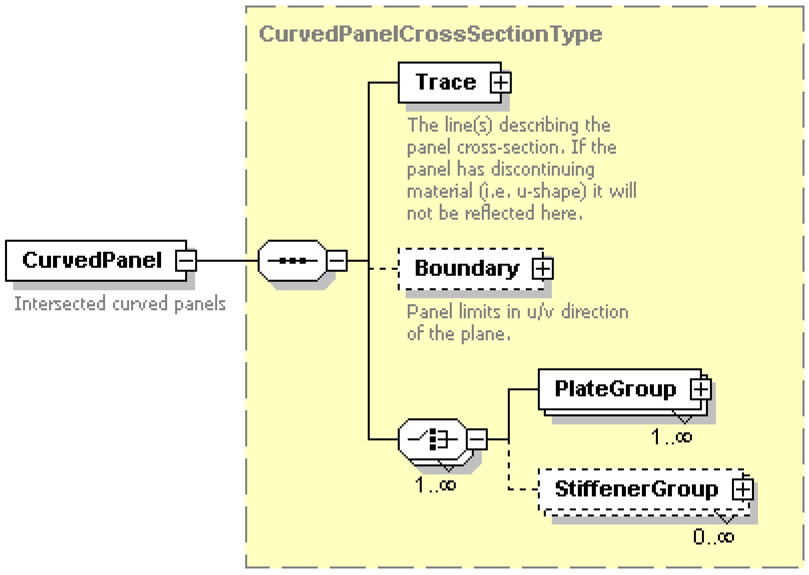
- **KnuckledPanel**

  The **KnuckledPanel** is a collection of **SubPanel**, which is actually PlanePanels. For further information, see PlanePanel section.



- **CurvedPanel**

  The **CurvedPanel** elements describes the shell. It is very similar to a **PlanePanel**. The only difference is that a curved panel has no **ModelRef** since it always refer to the hull surface. For further information, please see **PlanePanel**.


CurvedPanelCrossSectionType
Trace
The line(s) describing the panel cross-section. If the panel has discontinuing material (i.e. u-shape) it will not be reflected here.
CurvedPanel
Intersected curved panels
Boundary
Panel limits in u/v direction of the plane.
PlateGroup
1..∞
1..∞
StiffenerGroup
0..∞

- **Compartment**

  The **Compartment** element describes an intersected compartment (i.e. tank) with its **Boundary** panels. The boundary panels are listed as **Limit** elements with **ModelRef** element child's.

10.4 TXHSTL

TXHSTL describes ship´s steel structures and has the flexibility to transfer steel structure product data on basic, overall level or on a very detailed level.

One of the first applications of TXHSTL is to transfer steel structural data from Structural Design to PrimeShip from ClassNK and Patran/Nastran from MSC.

TXHSTL exists in two versions, one for modelling (input) purposes, called TXHSTL-M, where M stands for Modelling and another one for reporting (output) purposes, called TXHSTL-R. Their main structures are similar, but there are differences in the way the details regarding the plate and profile parts are described. The M version is focused on topology and parameters while the R version focuses on geometry.

10.4.1 TXHSTL-R

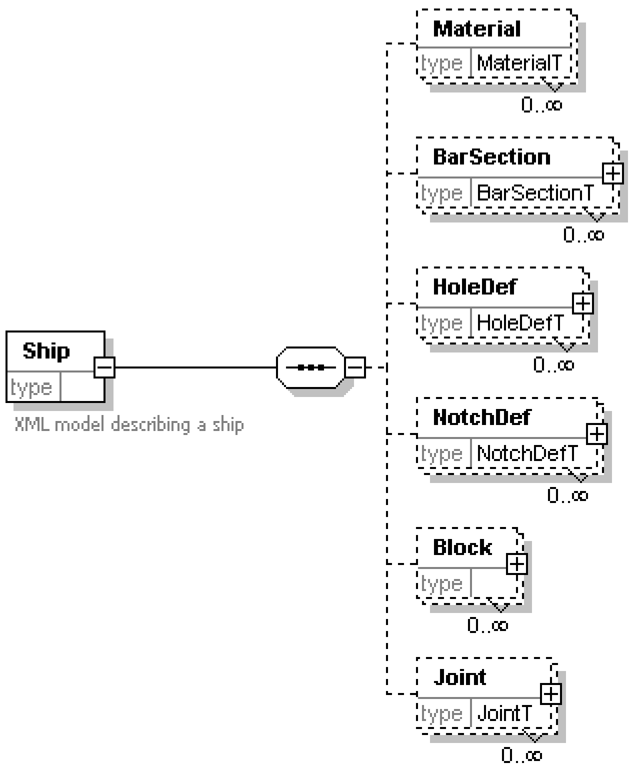
This section describes the reporting format for hull steel structures. The vocabulary is described in detail in the XML schema called TXHSTL-R.xsd. Below is a link to a presentation of this schema in HTML format.

TXHSTL-R.chm

The sections *Navigating*, *Element Relations* and *Attributes* will shortly explain how to use this appendix.

- **Basic Syntax of TXHSTL-R**

The input file has one root element, <Ship>. The root element has a number of child elements of different type.



- **Definition of Standards**

The first four of the elements below the Ship element describes some kind of standard that is referred to from the steel structure parts under the Block elements. The Joint elements are used together with a special option described below.

- **Material element**

  The material element contains the properties of a given material grade. It contains a Grade attribute also used as the unique identifier and density, Young's modulus, the Poisson ratio, the yield stress, the ultimate stress and optionally the thermal expansion coefficient. See the example below.

```
<Material Grade="A27" Density="7790" YoungsModulus="205000" PoissonRatio="0.3"
YieldStress="270" UltimateStress="490" ThermalExpansionCoefficient="0.011"/>
```

- **BarSection element**

  The BarSection element defines a bar crossection. Each bar type has its own element containing the characteristic dimensions as attributes. Optionally a BarSection can also have a section properties element containing area, neutral axes and moments of inertia. See the example below:

```
<BarSection BarSectionId="IBar240*240*10*17">
    <IBar Height="240" Width="240" WebThickness="10"
FlangeThickness="17"/>
    <SectionProperties Area="10600" NeutralAxisU="103"
NeutralAxisV="60.8" InertiaU="11259" InertiaV="3923"/>
</BarSection>
<BarSection BarSectionId="LBar180*90*10*10">
    <LBar Height="180" Width="90" WebThickness="10"
FlangeThickness="10"/>
    <SectionProperties Area="2620" NeutralAxisU="117.2"
NeutralAxisV="18.5" InertiaU="880" InertiaV="151"/>
</BarSection>
```

- **HoleDef element**

  The HoleDef element defines a hole shape. Each standard hole type has its own element containing the characteristic dimensions as attributes. For a more detailed explanation see the *Hull Standards*. The example below is a definition of a manhole.

```
<HoleDef HoleDefId="HO800*600">
    <Manhole Height="800" Width="600"/>
</HoleDef>
```

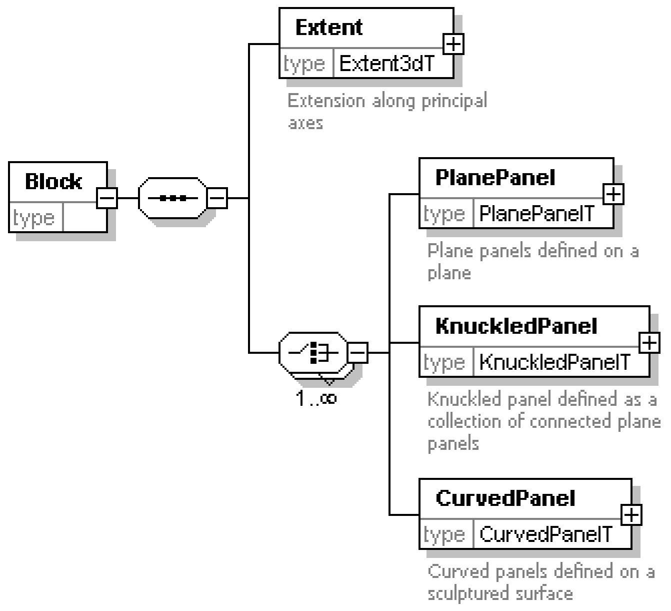
- **NotchDef element**

  The NotchDef element defines a notch shape. Each standard notch type has its own element containing the characteristic dimensions as attributes. For a more detailed explanation see the *Hull Standards*. The example below defines a VUF notch.

```
<NotchDef NotchDefId="VUF250*150*50">
    <VUF Width="250" Height="150" Radius="50"/>
</NotchDef>
```

• Block element

The Block elements contain the steel structure. The Ship element can have multiple Block elements that in turn can have multiple panel elements.


Extent
type Extent3dT
Extension along principal axes
Block
type
PlanePanel
type PlanePanelT
Plane panels defined on a plane
KnuckledPanel
type KnuckledPanelT
Knuckled panel defined as a collection of connected plane panels
CurvedPanel
type CurvedPanelT
Curved panels defined on a sculptured surface
1..∞

The block also has an ObjId attribute and a circumscribed box (Extent). A number of basic types are used throughout the steel structure described below.

- **Basic Types**

- ObjId, GroupId and CompId

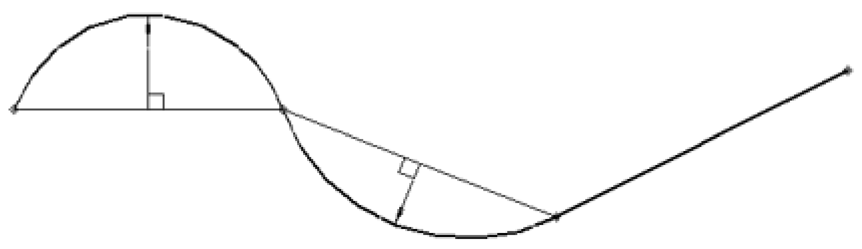
These attributes are used in the model whenever a model object such as a PlanePanel, a group of components such as a StiffenerGroup or a single component such as a Stiffener must be uniquely identified.

- ModelRef

A ModelRef element refers another object or a part within an object. ObjType is the type of object while ObjId is the model object name unique within the database. CompType is the component type while CompId is the component identity unique within the model object.

- Geometry

All geometry in TXHSTL-R is expressed as curve contours made up of circular arc segments. An endpoint and an amplitude vector define a segment. The start point of the segment is the endpoint of the previous segment. The amplitude vector is defined as the vector from the mid-point of the chord between the two points going perpendicular to the top of the arc. A zero-length amplitude vector gives a line segment. A contour has a start point and a number of segments. For a closed contour like a hole, the curve start point coincides with the curve endpoint.

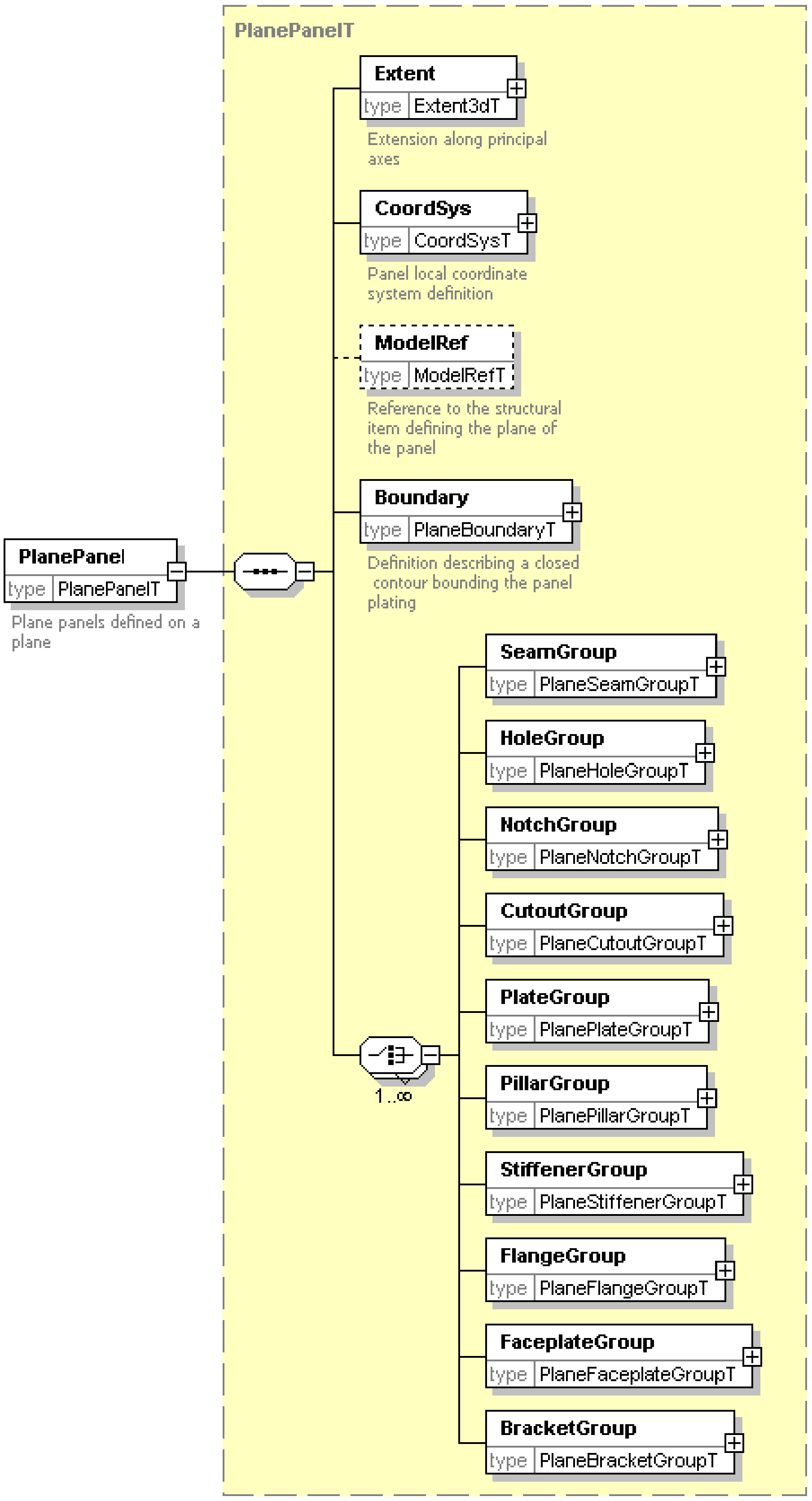


The circular arcs can be both two-dimensional and three-dimensional. For a two-dimensional curve there is belonging transformation information.

- **VectorContour3dT**

  This vector contour is used together with a Contour3dT to describe the orientation of a local coordinate system along the contour.

- **PlanePanelT**


PlanePanelT
Extent
type Extent3dT
Extension along principal axes
CoordSys
type CoordSysT
Panel local coordinate system definition
ModelRef
type ModelRefT
Reference to the structural item defining the plane of the panel
Boundary
type PlaneBoundaryT
Definition describing a closed contour bounding the panel plating
PlanePanel
type PlanePanelT
Plane panels defined on a plane
SeamGroup
type PlaneSeamGroupT
HoleGroup
type PlaneHoleGroupT
NotchGroup
type PlaneNotchGroupT
CutoutGroup
type PlaneCutoutGroupT
PlateGroup
type PlanePlateGroupT
PillarGroup
type PlanePillarGroupT
StiffenerGroup
type PlaneStiffenerGroupT
FlangeGroup
type PlaneFlangeGroupT
FaceplateGroup
type PlaneFaceplateGroupT
BracketGroup
type PlaneBracketGroupT
1..∞

This type is used to define the PlanePanel element. All geometry data, except shapes, inside the plane panel such as contours are given relative to the coordinate system described by the CoordSys element. This is true for both 2D and 3D contours. The 2D contours lie in the mould plane of the panel e.g. the outer contour. The 3D contours also have the third coordinate and can describe components outside the mould plane of the panel e.g. pillar trace lines.

- **CoordSysT**

  A local coordinate system is defined by an origin, the third and the first axis (w- and u-axis).

- **PlaneBoundaryT**

  The SimpleContour is a closed contour bounding the panel. It is described as a 2D contour and supposed to be located in the u-v plane of the local coordinate system. It represents the panel boundary excluding cutouts and notches. The DetailedContour includes cutouts and notches.

  Each Limit has a startpoint that can be used to extract the part of the outer contour generated by this limit. Then the startpoint of the next limit is used as the endpoint for the contour part of the current limit.

- **Plane<x>GroupT**

  All plane panel components are organized in groups. The group holds data common to all components in the group such as the bar crossection in a stiffener group. All groups have a GroupId and all components have a CompId. A group corresponds to a Plane Panel Input Scheme statement.

- **PlaneHoleT**

  The contour is defined in the u-v plane of the local coordinate system.

- **PlaneSeamT**

  The trace is defined in the u-v plane of the local coordinate system.

- **PlanePlateT**

  The SimpleContour and the DetailedContour elements represent the plate boundary in the panel mould plane. The position along the panel normal (w-axis) is given by the thickness and offset values of the superior plate group.

- **PlaneStiffenerT**

  The Trace element represents the description line of the stiffener in the mould plane. The Offset element gives distance along the panel normal that the trace should be moved when taking plate thickness into consideration. The Inclination element represents the inclination along the trace. For each nodepoint in the trace contour there exists a corresponding direction vector in Inclination. To create a solid, take the Crossection from the BarShape element and move it along the trace. The origin of the Crossection contour is mapped to the trace and the v-axis should coincide with the vector given by Inclination.

- **PlaneFlangeT**

  See *PlaneStiffenerT*.

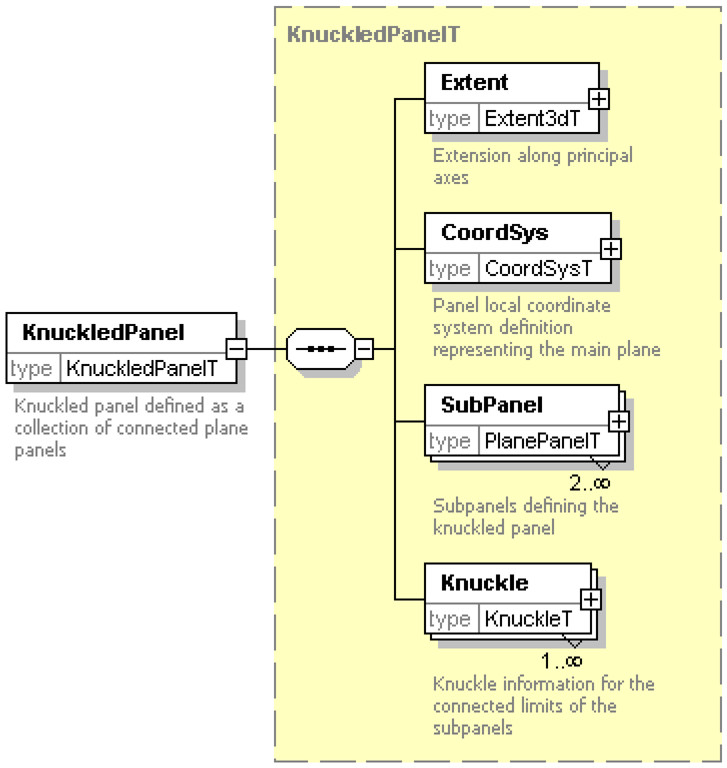
- **PlanePillarT**

  See *PlaneStiffenerT*.

- **PlaneBracketT**

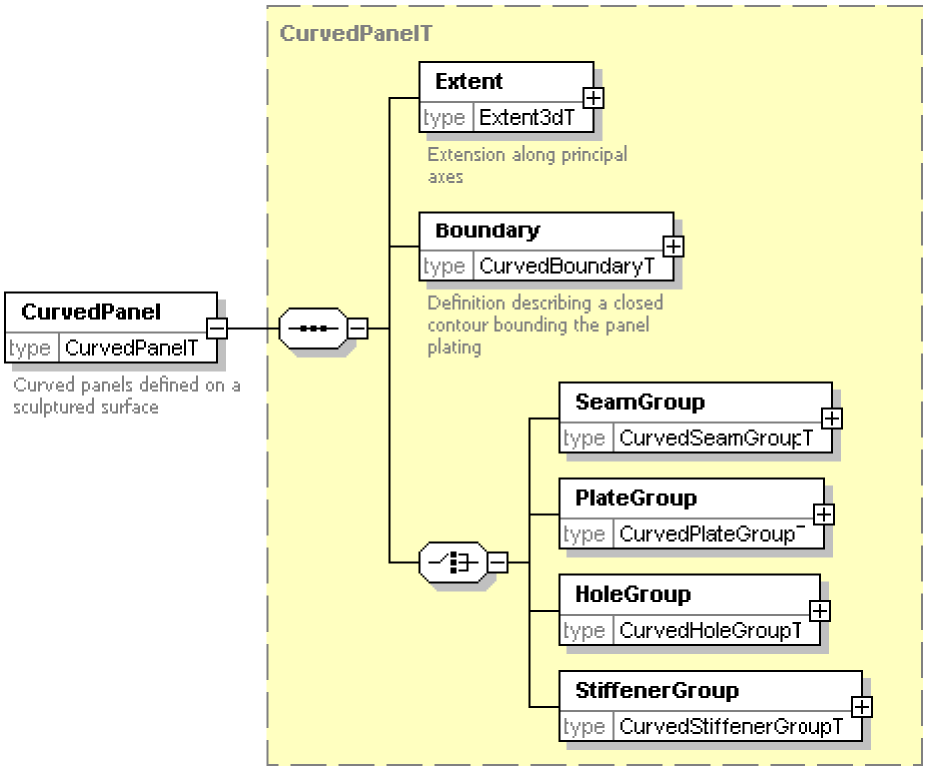
  The bracket has a local coordinate system of its own. The contours contained in it are described in this coordinate system.

• **KnuckledPanelT**


KnuckledPanelT
KnuckledPanel
type KnuckledPanelT
Knuckled panel defined as a collection of connected plane panels
Extent
type Extent3dT
Extension along principal axes
CoordSys
type CoordSysT
Panel local coordinate system definition representing the main plane
SubPanel
type PlanePanelT
2..∞
Subpanels defining the knuckled panel
Knuckle
type KnuckleT
1..∞
Knuckle information for the connected limits of the subpanels

The knuckled panel contains an Extent and a CoordSys element just as a PlanePanelT. It also contains a number of SubPanel elements together with some knuckle information. The SubPanel elements are of type PlanePanelT making a knuckled panel a collection of plane panels.

- **CurvedPanelT**



The curved panel contains in many respects a subset of the plane panel elements. However all geometry data inside the curved panel is described in the global coordinate system.

- **FacetSurfaceT**

A plate in a Curved Panel may have a facet surface as a point matrix. This matrix of points is used to create an approximate triangular facet description of a curved plate surface. Each strip row contains the same number of points and to create a strip of triangular facets the points from two adjacent strip rows are used.

- **Idealization**

The output using TXHSTL-R is very useful when a larger part of the model should be exported e.g. to an FEM system. When the model should be used for FEM meshing, the geometry needed is however not the normal one. Instead an idealized form of the geometry is used. In order to support this activity an idealizer option has been developed.

The Finite Element Idealiser option makes it possible to export the geometry in a form that is adapted to the way a FEM mesh is built up. It is still within the same XML format containing e.g. material data and all other attributes. But the geometry is exported as if the plates had no thickness and profiles had no height, width or thickness.

The following entities are affected:

- Panel and plate bounding contours. Normally these contours are adapted to the plate thickness of surrounding panels but with the Idealiser they are extended to meet the mould planes of the surrounding panels.

- Stiffener trace lines. Stiffeners that are connected to the flange of other intersecting stiffener are stretched to meet the mould line of these stiffeners. All gaps are removed and if the stiffener end is within a certain distance from another stiffener trace or end, the ends are connected.
- Faceplates and pillars will also have possible gaps removed at the ends.

Brackets will be moved if they connect to e.g. stiffeners as the stiffeners act as if they had no height.

10.4.2 TXHSTL-M

This section describes the modelling format for hull steel structures. The vocabulary is described in detail in the XML schema called TXHSTL-M.xsd. Below is a link to a presentation of this schema in HTML format.

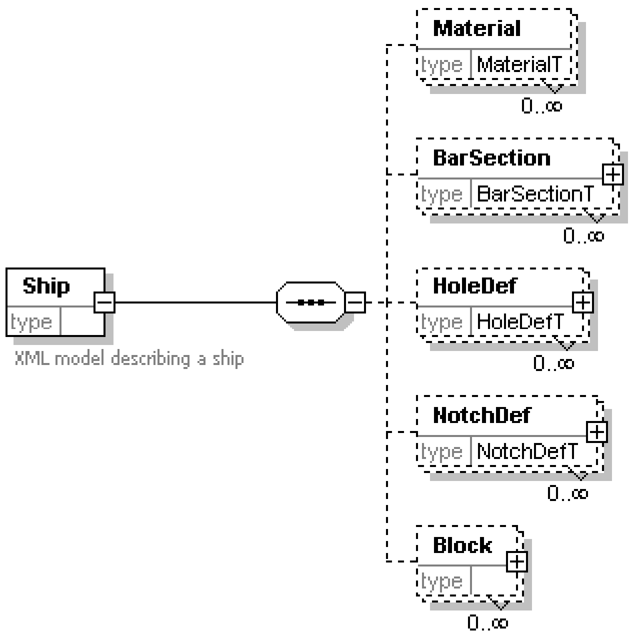
TXHSTL-M.chm

The sections *Navigating*, *Element Relations* and *Attributes* will shortly explain how to use this appendix.

The modelling format TXHSTL-M is very similar to the reporting format TXHSTL-R regarding the overall structure. The difference is that as –R focuses on geometry and material the –M format is built up mostly by topological references and material. However it is in most cases also possible to use bare geometry to create the model even of that is not the ideal way.

- **Basic Syntax of TXHSTL-M**

The input file has one root element, <Ship>. The root element has a number of child elements of different type.



The first four of the elements below the Ship element describes some kind of standard that is referred to from the steel structure parts under the Block elements.

- **Material element**

  The material element contains the properties of a given material grade. It contains a Grade attribute also used as the unique identifier together with density, Young's modulus, the Poisson ratio, the yield stress, the ultimate stress and optionally the thermal expansion coefficient. See the example below.

```
<Material Grade="A27" Density="7790" YoungsModulus="205000" PoissonRatio="0.3"
YieldStress="270" UltimateStress="490" ThermalExpansionCoefficient="0.011"/>
```

  By defining a number of these elements as input, a material standard can be built up inside the Product Information Model to be included in the following XML reports according to TXHSTL-R. The Grade attribute is the key to access the other values. Each time a <Material> element is given the material is either updated or created depending on whether it existed or not.

- **BarSection element**

  The BarSection element defines a bar crossection. Each bar type has its own element containing the characteristic dimensions as attributes. Optionally a BarSection can also have a section properties element containing area, neutral axes and moments of inertia. See the example below:

```
<BarSection BarSectionId="IBar240*240*10*17">
    <IBar Height="240" Width="240" WebThickness="10"
FlangeThickness="17"/>
    <SectionProperties Area="10600" NeutralAxisU="103"
NeutralAxisV="60.8" InertiaU="11259" InertiaV="3923"/>
</BarSection>
<BarSection BarSectionId="LBar180*90*10*10">
    <LBar Height="180" Width="90" WebThickness="10"
FlangeThickness="10"/>
    <SectionProperties Area="2620" NeutralAxisU="117.2"
NeutralAxisV="18.5" InertiaU="880" InertiaV="151"/>
</BarSection>
```

  By defining a number of BarSection elements with a SectionProperties element as input a bar section standard can be defined inside the Product Information Model. When making XML reports according to TXHSTL-R the stiffeners and other profiles are matched against this standard and provided there is a match the sectional properties are included.

- **HoleDef element**

  The HoleDef element defines a hole shape. Each standard hole type has its own element containing the characteristic dimensions as attributes. For a more detailed explanation see the *Hull Standards*. The example below is a definition of a manhole.

```
<HoleDef HoleDefId="HO800*600">
    <Manhole Height="800" Width="600"/>
</HoleDef>
```

  The holes can be defined either via a HoleDef element and a reference to it or directly via the HoleType attribute and the Parameters element in the HoleShape element under the HoleGroup element. When defining via a HoleDef element the HoleDefId of this element must match the HoleDefId of the HoleShape element.

- **NotchDef element**

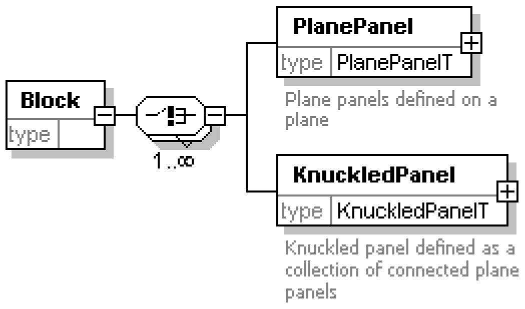
  The NotchDef element defines a notch shape. Each standard notch type has its own element containing the characteristic dimensions as attributes. For a more detailed explanation see the *Hull Standards*. The example below defines a VUF notch.

```
<NotchDef NotchDefId="VUF250*150*50">
    <VUF Width="250" Height="150" Radius=”50”/>
</NotchDef>
```

  Notches can be defined in a similar way to holes.

- **Block element**

  The Block elements contain the steel structure. The Ship element can have multiple Block elements that in turn can have multiple panel elements. Also see the documentation for *TXHSTL-R* for a further description of common elements and types.


PlanePanel
type PlanePanelT
Plane panels defined on a plane
Block
type
1..∞
KnuckledPanel
type KnuckledPanelT
Knuckled panel defined as a collection of connected plane panels

The block name is given by its ObjId attribute. If the block already exists it is updated. The same goes for the panels, groups and components. So to update e.g. the dimensions of one of the holes in the panel ES123-730 the input below can be used.

```
<Ship>
     <Block ObjId="ES123">
          <PlanePanel ObjId="ES123-730">
               <HoleGroup GroupId="5">
                   <HoleShape HoleType="HO">
                        <Parameters>800 400</Parameters>
                   </HoleShape>
          </PlanePanel>
     </Block>
</Ship>
```

10.5 TXHCM

This section will explain how to define curved hull modelling objects in an XML input file.

Important: When generating curved hull objects from an input file, there are some important restrictions. Please see *Important Restrictions*.

10.5.1 Schema File for the Input Language

The input language for curved hull is in XML format. Naturally, this language has a special XML “vocabulary”, i.e. there is a set of XML elements and attributes that are valid for this input language. The vocabulary is described in an XML Schema file (to be precise: there are two). The schema files, named “CurvedHull.xsd” and “HullBasic.xsd”, can be found in the directory SB_SYSTEM\xml.

The xsd-files are rather difficult to read if you view them in their raw text format. If you have an XML tool (XML Spy for instance) it will present the syntax rules in a more readable format, as a graphical tree for instance. If you do not have such a tool you may read an appendix to this document. It is a graphical presentation of the xsd-files in HTML format. Click in the following link to see the appendix in help browser:

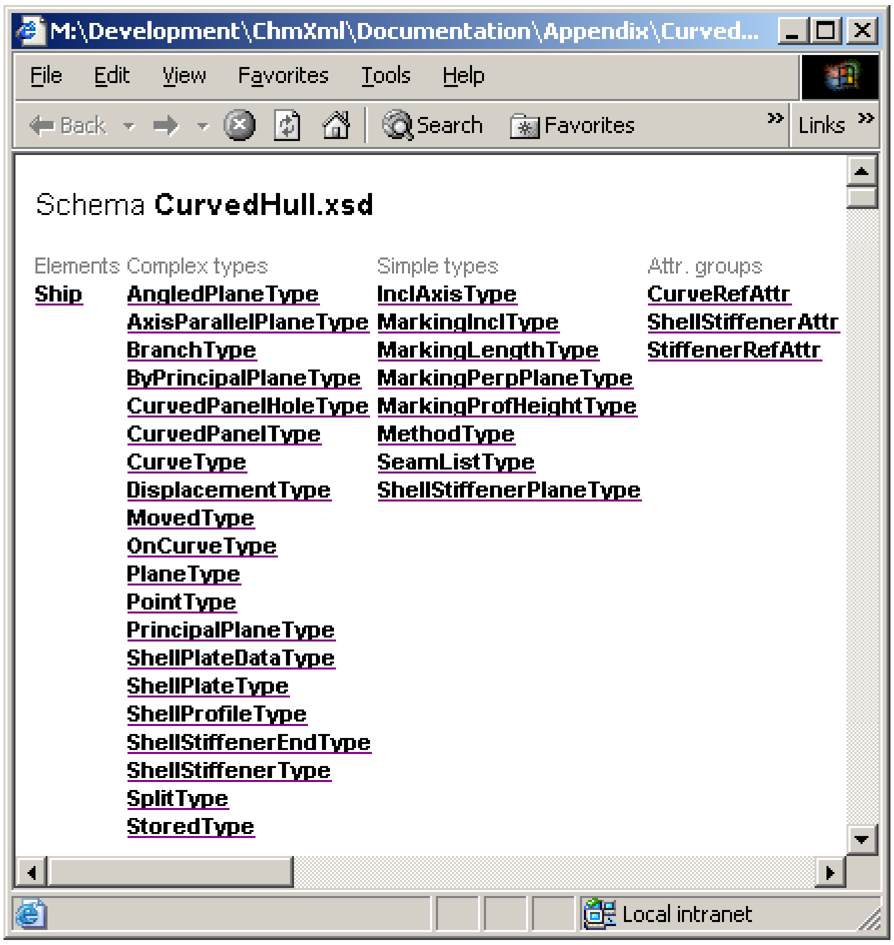
Schema_Curved_Modelling.chm

The next section will shortly explain how to use this appendix.

- ### Navigating

The Schema file appendix shows the tree structure of the input file in a graphical format. You can navigate in the tree by clicking on the elements.

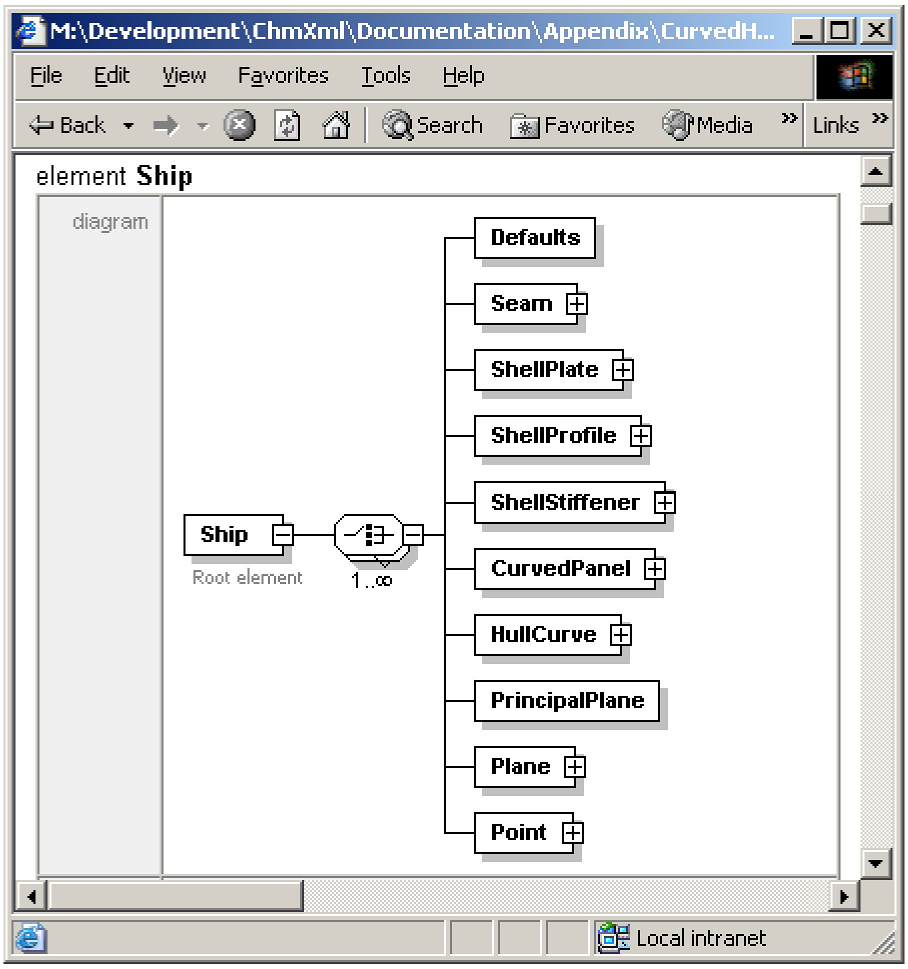
The start of the appendix is list of all element and attribute types:



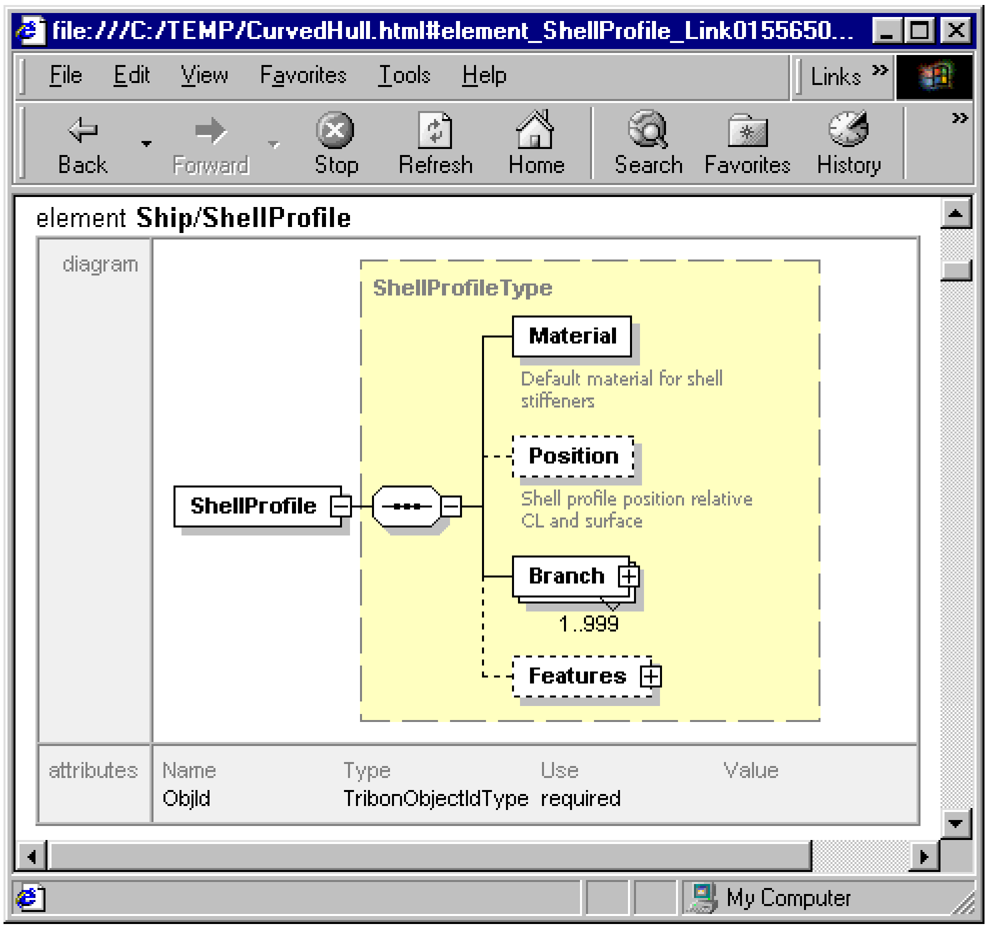
Schema **CurvedHull.xsd**

Elements	Complex types	Simple types	Attr. groups
Ship	AngledPlaneType	InclAxisType	CurveRefAttr
	AxisParallelPlaneType	MarkingInclType	ShellStiffenerAttr
	BranchType	MarkingLengthType	StiffenerRefAttr
	ByPrincipalPlaneType	MarkingPerpPlaneType	
	CurvedPanelHoleType	MarkingProfHeightType	
	CurvedPanelType	MethodType	
	CurveType	SeamListType	
	DisplacementType	ShellStiffenerPlaneType	
	MovedType		
	OnCurveType		
	PlaneType		
	PointType		
	PrincipalPlaneType		
	ShellPlateDataType		
	ShellPlateType		
	ShellProfileType		
	ShellStiffenerEndType		
	ShellStiffenerType		
	SplitType		
	StoredType		

Click on “Ship” in the upper left corner. Ship is the root element of a curved hull input file and thus the root of the tree structure:



Some of the child elements have a ‘+’ sign to the right. This indicates that this element also have child elements. You may now click on the “Shell Profile” element for instance and get to description of this element:

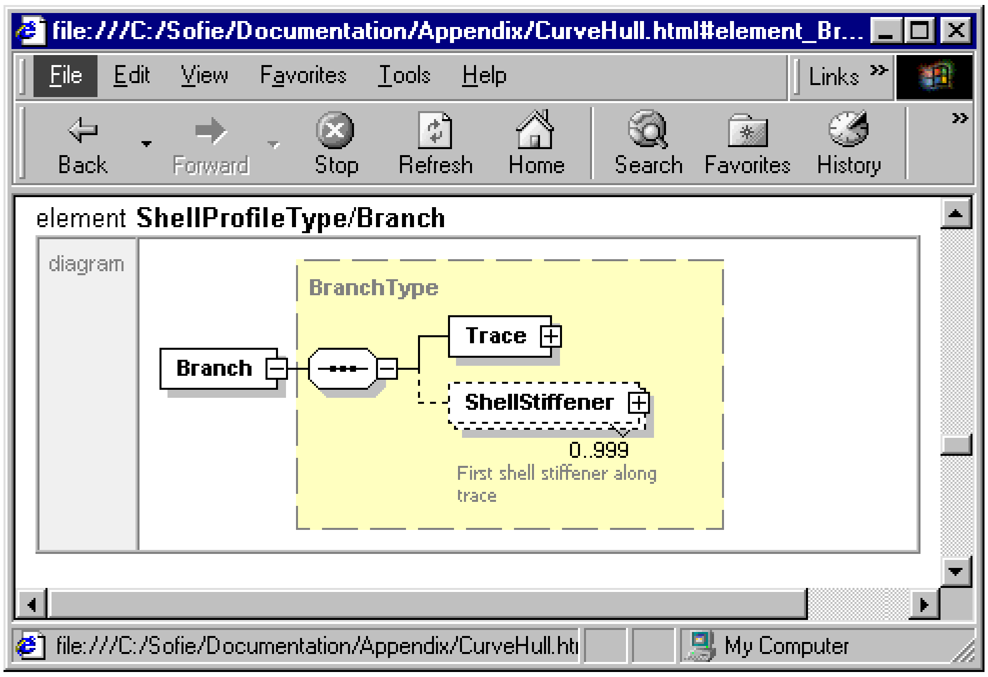

file:///C:/TEMP/CurvedHull.html#element_ShellProfile_Link0155650...
File Edit View Favorites Tools Help
Links
Back Forward Stop Refresh Home Search Favorites History
My Computer

element **Ship/ShellProfile**

diagram
ShellProfileType
ShellProfile
Material
Default material for shell stiffeners
Position
Shell profile position relative CL and surface
Branch
1..999
Features

attributes	Name	Type	Use	Value
	ObjId	TribonObjectIdType	required	

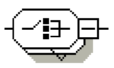
Clicking on “Branch” will take you to the Branch element definition:


file:///C:/Sofie/Documentation/Appendix/CurveHull.html#element_Br...
File Edit View Favorites Tools Help Links
Back Forward Stop Refresh Home Search Favorites History
element ShellProfileType/Branch
diagram
BranchType
Branch
Trace
ShellStiffener
0..999
First shell stiffener along trace
file:///C:/Sofie/Documentation/Appendix/CurveHull.ht
My Computer

- **Element Relations**

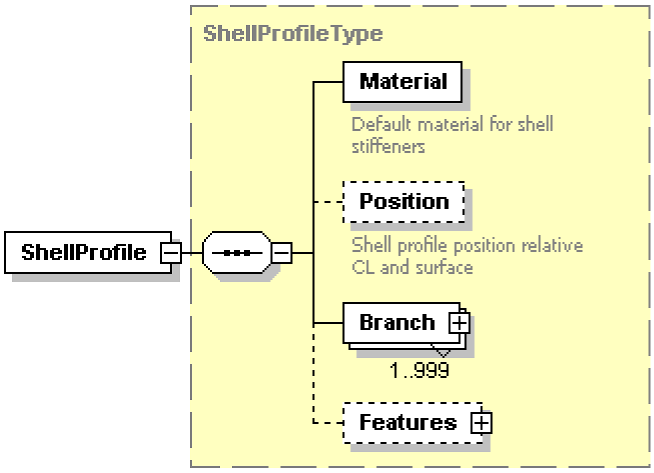
  In the graphical presentation there are symbols showing how the elements relate to each other. Looking at the graph representing the Ship element, we can see that it may have an infinite number of child element but there must be at least 1. Each of the elements may be any of the child elements in the graph: “Defaults”, “Seam”, “ShellProfile”, “ShellStiffener”, “HullCurve”, “PrincipalPlane or “Plane”.

  The symbol:

  indicates “Choice”, i.e. you may select any of the child elements. The choice symbol can be combined with number restrictions, in this case the minimum number of choices is 1 and there is no upper limit. A single number like “3”, indicates that there must be exactly three elements.

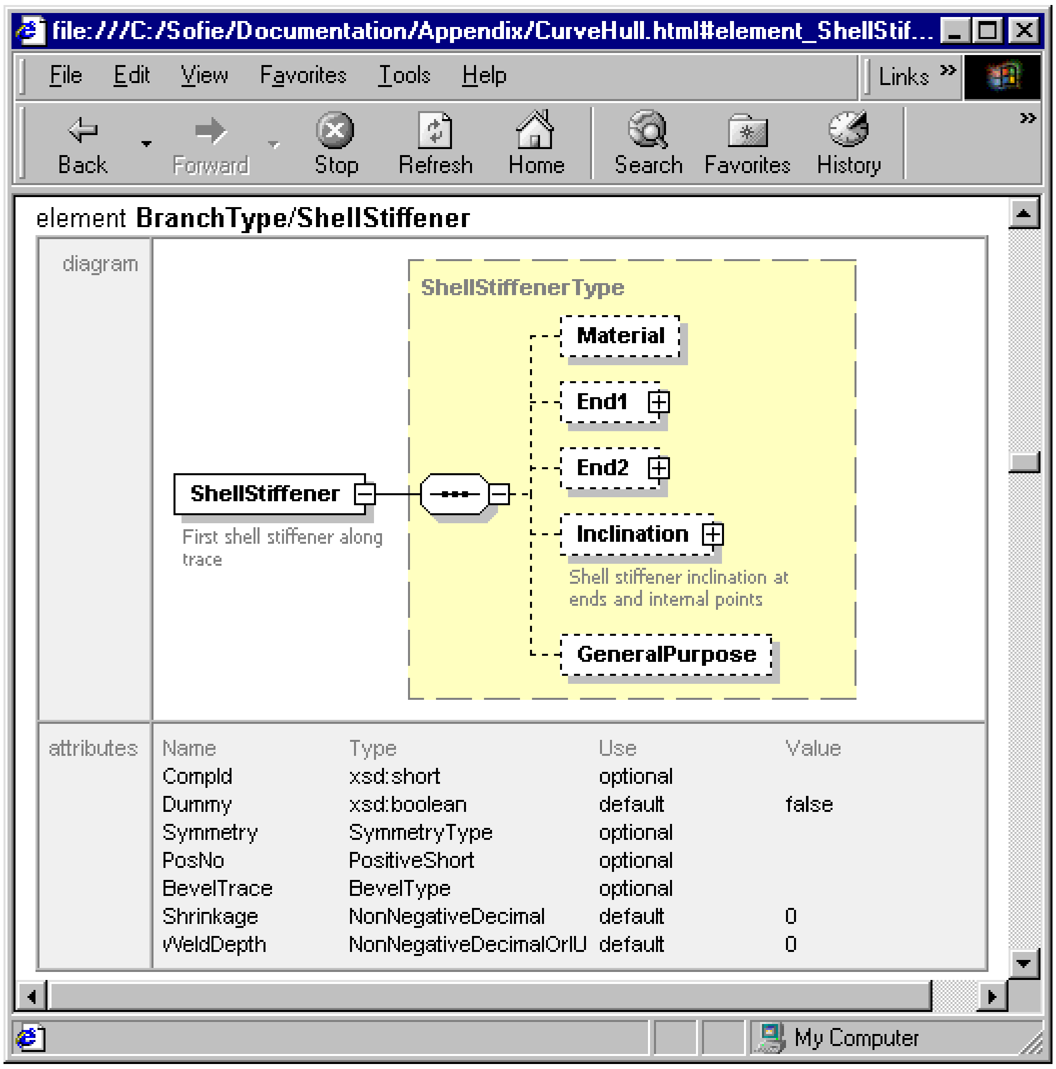
  The “ShellProfile” graph illustrates another common relation, a “sequence”:


ShellProfileType
ShellProfile
Material
Default material for shell stiffeners
Position
Shell profile position relative CL and surface
Branch
1..999
Features

This means that the “ShellProfile” element consists of all the listed child elements. They have to appear in the same order as in the graph. However, if the child element is drawn with a dashed line it is an *optional* element. In this example the “Material” element must be given in the input, while the “Position” and “Features” elements can left out. The ShellProfile element must have at least one “Branch” element but no more than 999.

- **Attributes**

  In the appendix you will also find the attributes of the elements.


file:///C:/Sofie/Documentation/Appendix/CurveHull.html#element_ShellStif...
File Edit View Favorites Tools Help
Links
Back Forward Stop Refresh Home Search Favorites History
diagram
ShellStiffenerType
Material
End1
End2
Inclination
Shell stiffener inclination at ends and internal points
GeneralPurpose
ShellStiffener
First shell stiffener along trace
My Computer

element **BranchType/ShellStiffener**

attributes

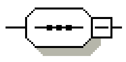
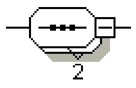
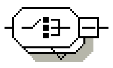
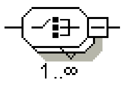
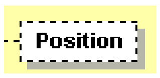
Name	Type	Use	Value
CompId	xsd:short	optional	
Dummy	xsd:boolean	default	false
Symmetry	SymmetryType	optional	
PosNo	PositiveShort	optional	
BevelTrace	BevelType	optional	
Shrinkage	NonNegativeDecimal	default	0
WeldDepth	NonNegativeDecimalOrIU	default	0

Below the graph is a listing of all the attributes that belong to the current element (the ShellStiffener in this example), in this case the “ShellStiffener” element. The listing will show:

- the name of the attribute
- the type of the attribute
- whether the attribute is required or optional
- default value for the attribute, if any

- **Summary**

  :

	Indicates a “sequence”
	Sequence with number condition. (Here exactly two)
	Indicates “choice”
	Choice with number condition. (Here at least one and no upper limit.)
	Optional element is marked with a dashed line.

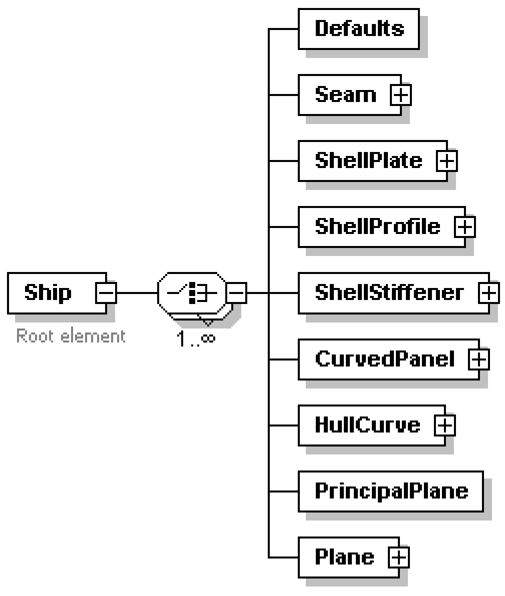
10.5.2 Basic Syntax of the Input File

The input file has one root element, <Ship>. This element must have a reference to the schema file, “CurvedHull.xsd”:

```
<Ship xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\install directory\bin\xml\CurvedHull.xsd">
```

(Section *Schema File for the Input Language* for details about the location of “CurvedHull.xsd.)

The root element, “Ship”, may have an infinite number of child elements, each element (except “Defaults”) representing a curved hull model object:



As shown in the picture, the input language currently supports: stored planes (principal plane or other plane), seams and hull curves, shell profiles and “stand-alone” shell stiffeners, shell plates and curved panels. They will all be documented in the following sections.

10.5.3 Defaults Element

The Defaults element corresponds to the “defaults” that you may set in the interactive curved hull. You may select a default surface and a default limit box. The surface and box will be used when not set specifically for a curved hull object (curves and seams for instance).

The limit box is defined with six attributes: XMin, XMax, YMin, YMax, ZMin and ZMax. Naturally, they define the minimum and maximum value along the x-, y- and z-axes. If an attribute is omitted this is interpreted as an “unlimited” value. Thus, an empty Defaults element, “<Defaults/>”, will be interpreted as a box, unlimited in all directions. The element “<Defaults YMin="0"/>” sets a minimum value along y-axis, but in all other directions the box is unlimited.

The Defaults element may appear any number of times in the file. A Defaults element is modal, i.e. it is valid until the next occurrence of a Defaults element. A new instance of the Defaults element overrides the previous one completely. Example:

```
<Defaults Surface="SPHULL" YMin="0" XMin="FR40" XMax="FR100"/>
   <Seam ObjId="SPS900">
      <ByPrincipalPlane Z="2000"/>
   </Seam>
   <Seam ObjId="SPS901">
      <ByPrincipalPlane Z="4000"/>
   </Seam>
<Defaults ZMax="12000"/>
   <Seam ObjId="SPS902">
      <ByPrincipalPlane Z="FR50" Surface="SPHULL"/>
```

```
    </Seam>
```

The first Defaults element sets a restriction along the y-axis and x-axis. This box is valid when the seams “SPS900” and “SPS901” are generated.

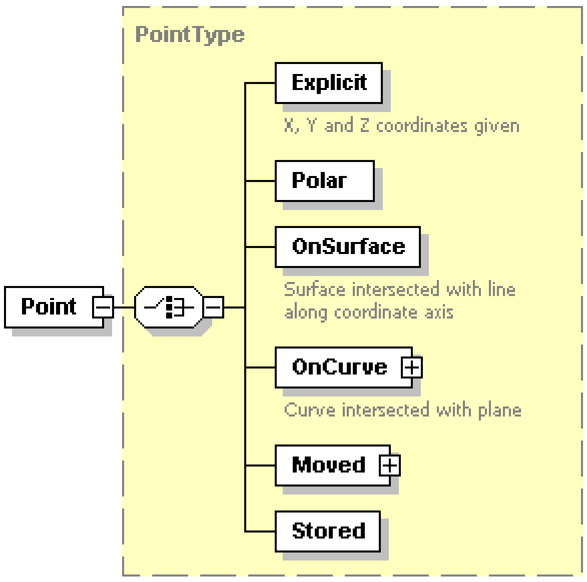
The second Defaults element sets a restriction along the z-axis (maximum value). Since this Defaults element completely overrides the previous one, the box is now unlimited along the x- and y-axes. Also, there is no default surface selected.

10.5.4 Point Element

The **Point** element defines a CHM “point”. As in Interactive Curved Hull it can be defined in a number of ways:

- Explicit point by three coordinate values
- Polar coordinates
- Point on surface
- Point on curve
- Point moved along a curve

The Point element has the following child elements:


PointType
Explicit
X, Y and Z coordinates given
Polar
OnSurface
Surface intersected with line along coordinate axis
OnCurve
Curve intersected with plane
Moved
Stored
Point

The point types “**Explicit**”, “**Polar**”, “**OnSurface**”, “**OnCurve**” and “**Moved**” will be explained in the following section. Please see *Explicit point*, *Polar Point*, *Point on Surface*, *Point on*

Curve and *Moved Point*. The "**Stored**" element is a reference to a point stored in the data bank. The Stored element has one attribute:

ObjId	The name of the referenced point object.
Refl	When set to "true" this flag indicates the resulting point will a reflected (in CL) version of the given point specification. For instance: `<Point Refl="true">` `<Explicit X="FR50", Y="3000", Z="10000"/>` `</Point>` will result in a point where X=FR50, Y=**-3000** and Z=10000. This is an optional attribute, the default value is "false".

- ### Explicit point

  This element has three attributes giving the coordinate values of the point: X, Y and Z. The attribute value is a single coordinate in the traditional AVEVA Marine format.

Example:

```
<Point>
      <Explicit X="FR50-125" Y="4000" Z="5000"/>
</Point>
```

- ### Polar Point

  A point can be defined with polar coordinates i.e. with an angle value and a the length of the radius. In curved hull the angle and the radius is always applied in a frame plane, specified by giving an X coordinate value. The attributes of the "Polar" element are:

X	An X-coordinate value which defines the frame plane. Required attribute.
Angle	The angle value. Required attribute.
Radius	The radius. Required attribute.

Example:

```
<Point>
      <Polar X="FR30" Radius="10000" Angle="40"/>
</Point>
```

- **Point on Surface**

This element that represents a point on a surface, has five attributes:

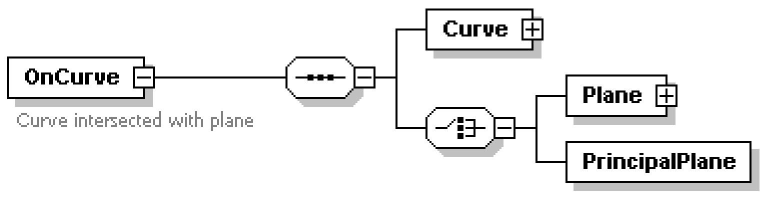
Surface	The name of the surface, optional attribute. If omitted the surface in the current Defaults element will be used. If there is no default surface then an error will be signalled
Approx	This attribute is used when you want to define an approximate coordinate. It indicates which one of the coordinates x, y or z that should be interpreted as the approximate one. Possible values are “X”, “Y” or “Z” (in capitals).
X,Y,Z	The coordinate values. A single coordinate value in AVEVA Marine format. At least two of them must be given.

Example:

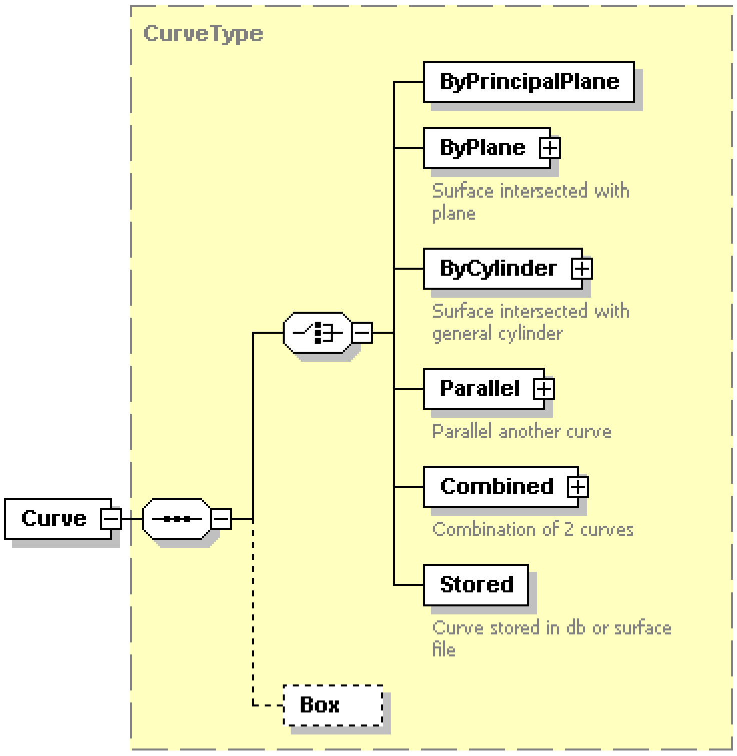
```
<Point>
      <OnSurface Surface="SPHULL" Approx="Z" X="55000" Y="5000" Z="8000"/>
</Point>
```

- **Point on Curve**

In this case the point is defined by an intersection between a curve and a plane:



The plane can be a principal plane or any of the other plane types. The principal plane element is described in section *PrincipalPlane/ByPrincipalPlane Element* (however, the “ObjId” attribute is not applicable here). The Plane element is documented is section *Plane/ ByPlane Element*. The curve can be a stored curve or a complete curve definition:



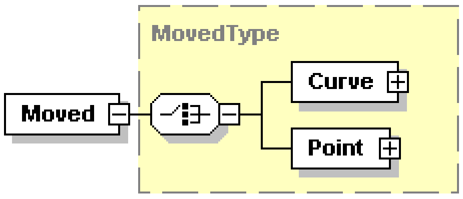
Curve will be explained later in the document, see section *Seam Element*.

Example:

```
<Point>
   <OnCurve>
      <Curve>
         <Stored ObjId="SPZ950"/>
      </Curve>
      <PrincipalPlane X="FR50"/>
   </OnCurve>
</Point>
```

- ### Moved Point

This point is moved along a curved or moved from another point:

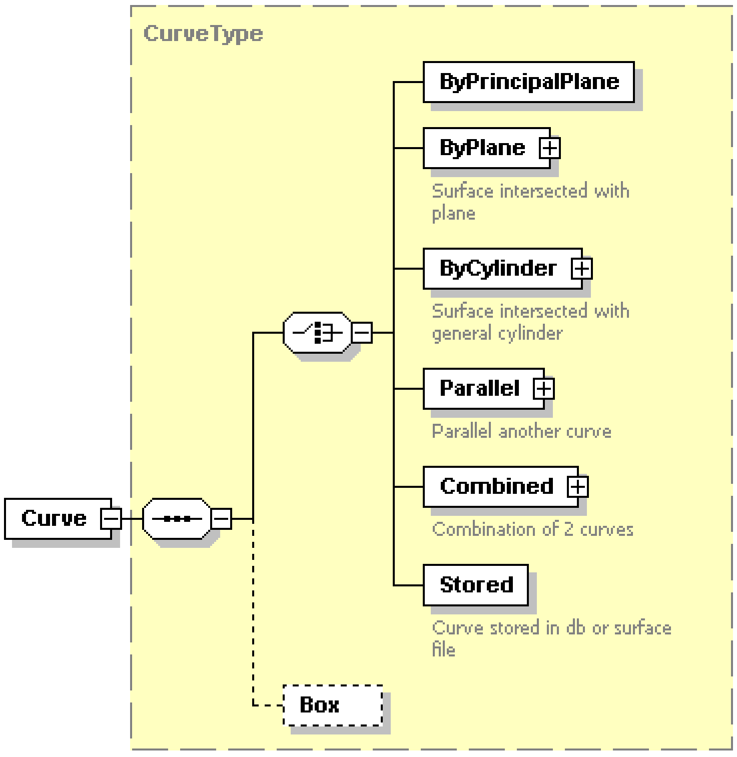


In the “Moved” element you give the distance which the point will be moved:

Distance	The distance along the curve. Required attribute

- **Point Moved Along a Curve**

When the point is moved along a curved it will be moved from one of the end points of the curve. The curve can be a stored curve/seam or any other curve:

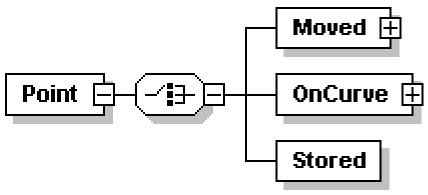


In the “Curve” element you select from which end point the point will be moved:

FromEnd	Indicates from which end of the curved the point will be moved. Possible values are the main directions of the ship: “For”, “Aft”, “Top”, “Bottom”, “PortSide”, “Starboard”. If FromEnd=“Aft” is given, the point will be moved from the aftmost end point of the curve. Optional attribute, if omitted the system will select the start point of the curve.

- **:Point Moved From Another Point**

When the point is moved from the point, this point must be “Moved”, “OnCurve” or “Stored” :



The stored point must be of a type that in some way refers to a curve, i.e. “Moved” or “OnCurve”.

In the “Point” element the “Direction” attribute selects in what direction the point will be moved:

Direction	Indicates from which point will be moved. Possible values are the main directions of the ship: “For”, “Aft”, “Top”, “Bottom”, “PortSide”, “Starboard”. If Direction=“Aft” is given, the point will be moved from the aft direction along the curve. Optional attribute, if omitted the system will move the point in the positive direction of the curve.

Example:

```
<Point ObjId="MOVED_PT1">
   <Moved Distance="1500">
      <Curve FromEnd="Aft">
         <ByPrincipalPlane Z="8500"/>
      </Curve>
   </Moved>
</Point>
```

Example:

```
<Point ObjId="MOVED_PT2">
   <Moved Distance="4000">
      <Point Direction="For">
         <Stored ObjId="MOVED_PT1"/>
      </Point>
   </Moved>
</Point>
```

10.5.5 PrincipalPlane/ByPrincipalPlane Element

This element defines a principal plane that will be stored in the data bank. It has two attributes:

ObjId	The name of the plane object, required attribute.
Refl	When set to “true” this flag indicates the resulting plane will a reflected (in CL) version of the given plane specification. This is an optional attribute, the default value is “false”.

X, Y or Z	You select one of the attribute X, Y or Z. The value is the coordinate value. It can be a single coordinate value given in the traditional AVEVA Marine format, for instance:
	
	X=”FR20”
	
	Y=”LP10+200”
	
	Z=”FR10+2F1I (Imperial units are also allowed.)

Example:

```
<PrincipalPlane ObjId="PLANE2" Z="LP20-350"/>
```

Example:

```
<PrincipalPlane ObjId="PLANE1" X="FR20"/>
```

The “**ByPrincipalPlane**” element is almost identical to the “PrincipalPlane” element, but it does not have the “ObjId” attribute. ByPrincipalPlane will not generate a stored plane, it is used when defining a seam or hull curve by a principal plane. The **ByPrincipalPlane** has an attribute, “Surface”:

Surface	The surface which will be intersection by the plane, optional attribute. If omitted the surface in the current Defaults element will be used. If there is no default surface then an error will be signalled

Example:

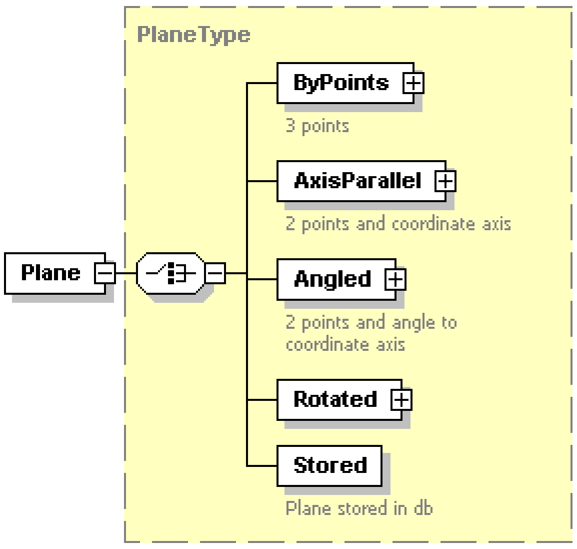
```
<Seam ObjId="SPS958">
   <ByPrincipalPlane Z ="LP38" Surface="ESHULL"/>
</Seam>
```

10.5.6 Plane/ByPlane Element

The “**Plane**” element models a plane to be stored in the data bank. As in interactive curved hull, the plane can defined in a number of ways:

- By three points
- By two points and a coordinate axis
- By two points and angle(s)
- As a "rotated" plane

The Plane element has the following child elements:



The plane types "**ByPoints**", "**AxisParallel**" and "**Angle**" will be explained in the following section. Please see *Plane by Three Points*, *Plane by Two Points and Axis*, *Plane by Two Points and Angle(s)* and *Rotated Plane*. The "**Stored**" element is a reference to a plane stored in the data bank. The Stored element has one attribute:

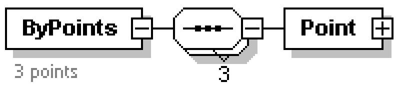
ObjId	The name of the referenced plane object.
Refl	When set to "true" this flag indicates the resulting plane will a reflected (in CL) version of the given plane specification. This is an optional attribute, the default value is "false"

The "**ByPlane**" element is almost identical to the "Plane" element, but it does not have the "ObjId" attribute. ByPlane will not generate a stored plane, it is used when defining a seam or hull curve by a principal plane. The **ByPlane** has an attribute, "Surface":

Surface	The surface which will be intersection by the plane, optional attribute. If omitted the surface in the current Defaults element will be used. If there is no default surface then an error will be signalled

- **Plane by Three Points**

This plane is defined by three point.



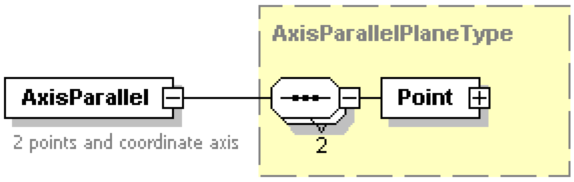
Each point can be one of the point types “Explicit”, “OnSurface” and “OnCurve“. Please see the section *Point Element* for a detailed description.

Example:

```
<Plane>
   <ByPoints>
      <Point>
         <Explicit X="FR40" Y="0" Z="LP25"/>
      </Point>
      <Point>
         <OnSurface Approx="Y" X="FR40" Y="LP18" Z="LP30"/>
      </Point>
      <Point>
         <Explicit X="FR60" Y="LP12" Z="LP38"/>
      </Point>
   </ByPoints>
</Plane>
```

- **Plane by Two Points and Axis**

This defines a plane by two points and the condition that the plane is parallel to one of the coordinate axes



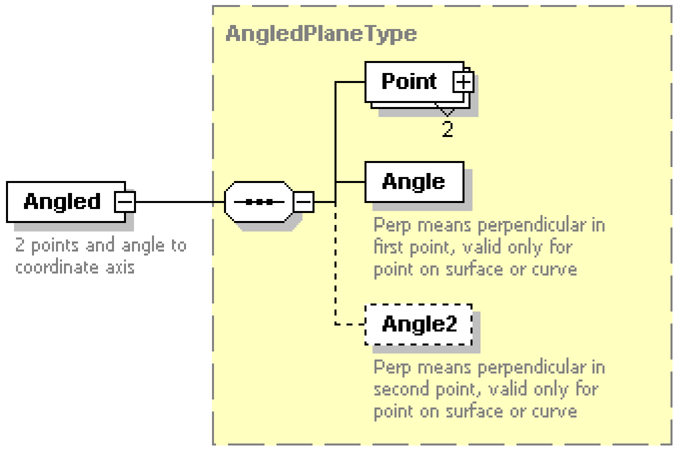
Each one of the points can be any of the types “explicit”, “on surface or “on curve”. Please see the section *Point Element* for a detailed description.

The actual axis is selected by the “Axis” attribute in the “AxisParallel” element:

Axis	Possible values are “X”, “Y” or “Z”, required attribute.

- **Plane by Two Points and Angle(s)**

This defines a plane by two points and one or two “angle conditions”:


AngledPlaneType
Point
2
Angle
Perp means perpendicular in first point, valid only for point on surface or curve
Angle2
Perp means perpendicular in second point, valid only for point on surface or curve
Angled
2 points and angle to coordinate axis

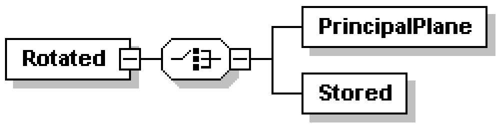
Each one of the points can be any of the types “explicit”, “on surface or “on curve”. Please see the section *Point Element* for a detailed description. An “angled” plane must have at least on “angle condition”, given in the “Angle” element. The “Angle” element has two attributes:

Axis	Coordinate axis, “X”, Y” or “Z”. Required attribute
Angle	The angle value in the interval -360 to 360 degrees. Required attribute.

A second “angle condition” may be given in the element “Angle2”. Please note that child elements of and angle plane must be given in the order: Point(1) - Point(2) - Angle - Angle2.

- ### Rotated Plane

This plane is defined by rotating a principal plane (the “Base Plane”) a specified angle. The rotation axis is defined by giving a second principal plane (the “Rotation Plane”). The intersection line between these two planes is the rotation axis:



The “Rotated” element has the following attributes:

Axis, Coord	“Axis” and “Coord” define the rotation plane. Axis is “X”, “Y” or “Z”. Coord is a coordinate value. Both attributes are required.
Angle	The coordinate values. A single coordinate value in AVEVA Marine format. At least two of them must be given.

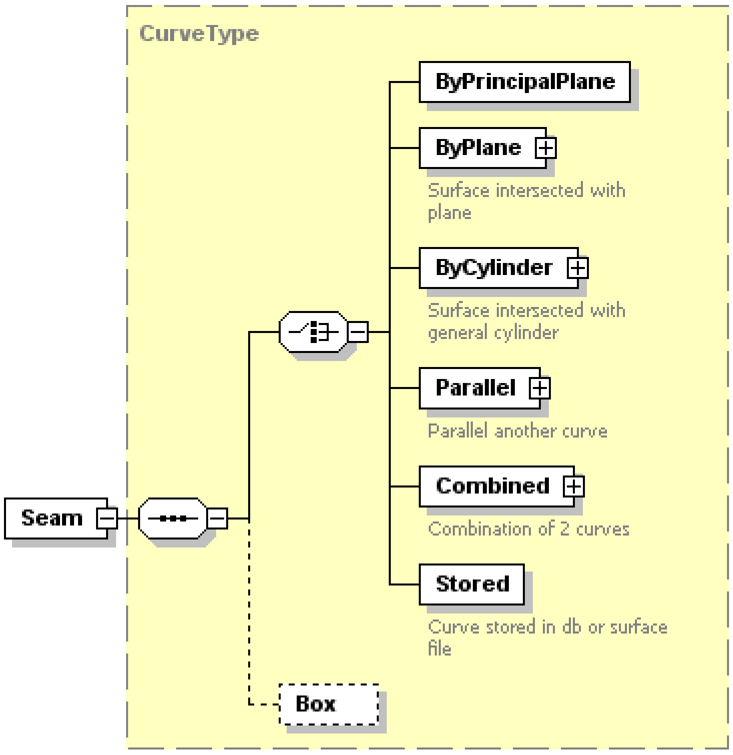
The “base plane” is defined by a “PrincipalPlane” element or it can be a reference to a stored plane. The stored plane must be a principal plane. The “PrincipalPlane” is documented in *PrincipalPlane/ByPrincipalPlane Element*.

Example:

```
<Plane ObjId="SOLPL2">
   <Rotated Angle="45" Axis="Y" Coord="0.0">
      <PrincipalPlane X="FR50"/>
   </Rotated>
</Plane>
```

10.5.7 Seam Element

The Seam element models a shell seam. It has a curve definition and a limit box:



The Box element is optional. If the box is omitted the box defined in the current Defaults element will be used. If there is no Defaults element, the curve will unrestricted along all coordinate axes.

The Seam element has three attributes:

ObjId	The name of the seam, required. The attribute value must apply to the general AVEVA Marine rules for naming objects.
Refl	When set to “true” this flag indicates the resulting seam will be a reflected (in CL) version of the given seam specification. This is an optional attribute, the default value is “false”.

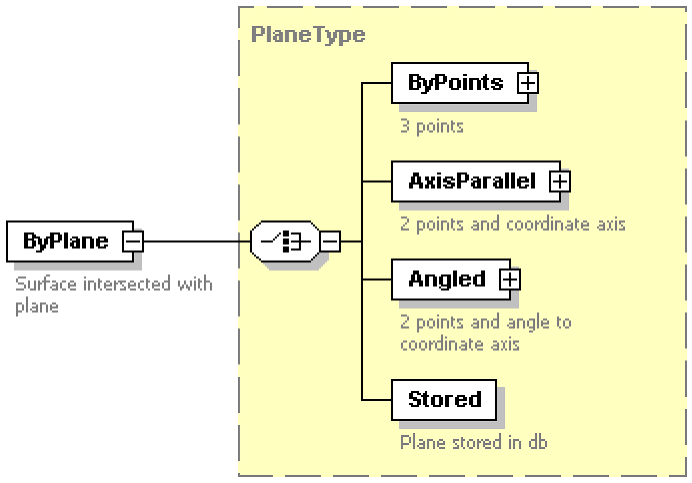
Symmetry	The symmetry of the seam, possible values are “Auto”, “Symmetric”, “SB” (valid SB only), “PS” (valid PS only) and “CL” (over/in CL). The value “Auto” means that the symmetry will be selected automatically: a seam which is completely on the PS side will be valid for both SB and PS, a seam on the SB side will be valid only SB and a seam over CL or in CL will be “over/in CL”. The attribute is optional, the default value is “Auto”
BlockLimit	Indicated is the seam is a block limit, optional. Possible values are “true” or “false”, the default value is “false”.

As shown in the picture above, the curve can be defined in a number of ways all recognized from interactive curved hull:

- by intersection between a surface and a plane.
- by intersection between a surface and a general cylinder
- parallel another curve
- as a combination of two other curves
- from an existing hull curve or from a curve in an external surface system

• Seam by Plane

The plane can be a principal plane or one of the other plane types, “ByPoints”, “Angled”, “AxisParallel” or “Stored”. The plane is defined by a “ByPrincipalPLane” or a “ByPlane” element:


PlaneType
ByPoints
3 points
AxisParallel
2 points and coordinate axis
Angled
2 points and angle to coordinate axis
Stored
Plane stored in db
ByPlane
Surface intersected with plane

The “ByPrincipalPlane” is documented in *PrincipalPlane/ByPrincipalPlane Element* and the “ByPlane” element in *Plane/ByPlane Element*

Example:

1 - Seams by principal plane

```
1 - Seams by principal plane

<Seam ObjId="SPS950" Symmetry="Symmetric" BlockLimit="true">
   <ByPrincipalPlane X="FR40"/>
<Box XMin="FR20" YMin="0" XMax="FR100" YMax="30000"/>
</Seam>

<Seam ObjId="SPS952" Symmetry="PS">
   <ByPrincipalPlane X="FR50"/>
</Seam>

<Seam ObjId="SPS953" Symmetry="SB">
   <ByPrincipalPlane Y="LP10"/>
</Seam>

<Seam ObjId="SPS955" Symmetry="CL">
   <ByPrincipalPlane Y="LP15"/>
</Seam>

<Seam ObjId="SPS958" Symmetry="Symmetric">
   <ByPrincipalPlane Z="LP38"/>
</Seam>
```

Example:

2 - Planar seam by three points

```
<Seam ObjId="SPS970">
   <ByPlane Surface="SPHULL">
      <ByPoints>
         <Point>
            <Explicit X="18000" Y="2000" Z="10000"/>
         </Point>
         <Point>
            <Explicit X="10000" Y="2000" Z="10000"/>
         </Point>
         <Point>
            <OnSurface Approx="Z" X="55000" Y="5000" Z="8000"/>
         </Point>
      </ByPoints>
   </ByPlane>
</Seam>
```

Example:

3 - Planar seam by axis parallel plane

```
<Seam ObjId="SPS953">
   <ByPlane>
      <AxisParallel Axis="Y">
         <Point>
            <Explicit X="FR20" Y="LP15" Z="LP34"/>
         </Point>
         <Point>
            <Explicit X="FR28" Y="LP15" Z="LP24"/>
         </Point>
      </AxisParallel>
   </ByPlane>
</Seam>
```

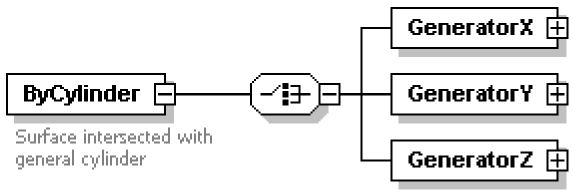
Example:

4 - Planar seam by angled plane

```
<Seam ObjId="SPS972">
   <ByPlane>
      <Angled>
         <Point>
            <Explicit X="30000" Y="4000" Z="7000"/>
         </Point>
         <Point>
            <Explicit X="15000" Y="3000" Z="6000"/>
         </Point>
         <Angle Axis="X" Angle="10"/>
      </Angled>
   </ByPlane>
</Seam>
```

- ### Seam by General Cylinder

  A seam created by intersecting the surface with a general cylinder, has a child element called “ByCylinder”. This element have an child element “GeneratorX”, “GeneratorY” or “GeneratorZ”:

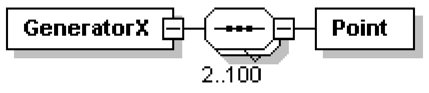


The “ByCylinder” element has three attributes:

Surface	The name of the surface to be intersected, optional. If the attribute is omitted then the surface in current Defaults element will be used. If there is no default surface then an error will be signalled.
Angle1	Controls the angle of the directrix in the start point. The angle is calculated against the u-axis.
Angle2	Controls the angle of the directrix in the end point

By selecting one of the “Generator” element you select the generator axis for the cylinder. GeneratorX means that the generator axis is along the X-axis. GeneratorY and GeneratorZ indicates a generator along the Y- and Z-axis respectively.

The Generator (X, Y or Z) element also defines the directrix curve by a sequence number of point elements, at least two and no more than 100:



The Generator element has two attributes controlling the generator axis:

Min	Minimum value for the generator along the selected axis. Optional attribute.
Max	Maximum value for the generator. Optional attribute

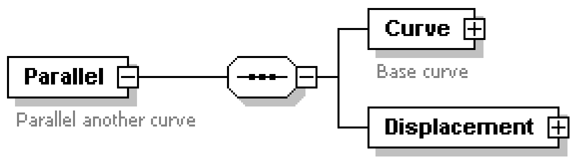
The “Point” element has two attributes giving the coordinate values of the point. For “GeneratorX” you give Y- and Z-coordinates, for “GeneratorY” you give X- and Z-coordinates and for “GeneratorZ” you give X- and Y-coordinates.

Example:

```
<Seam ObjId="SPS900">
   <ByCylinder>
      <GeneratorY Min="0" Max="50000">
         <Point X="FR10" Z="LP35"/>
         <Point X="FR15" Z="LP33"/>
         <Point X="FR20" Z="LP31"/>
         <Point X="FR25" Z="LP29"/>
         <Point X="FR30" Z="LP27"/>
      </GeneratorY>
   </ByCylinder>
   <Box XMax="FR40" XMin="FR5" YMax="40000" YMin="0" ZMax="40000"/>
</Seam>
```

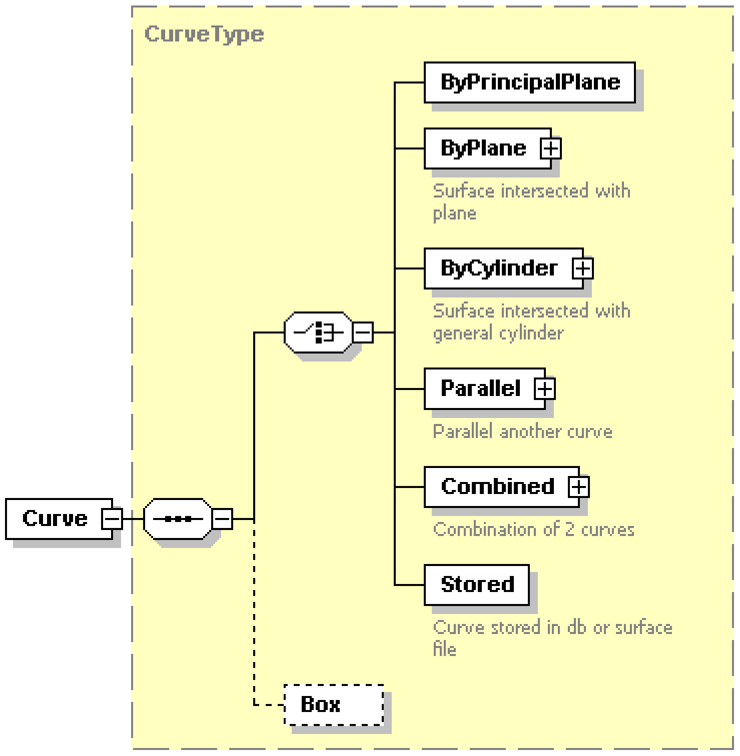
- ### Seam Parallel to Another Curve

A seam can be created parallel another curve by a child element called “Parallel”:

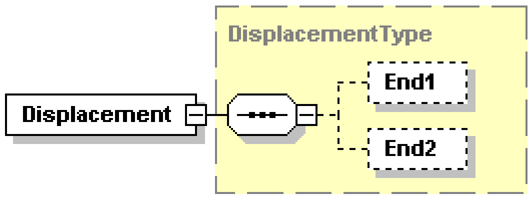


The Parallel element consist of a “Curve” element defining the base curve and a “Displacement” element holding details about the displacement like distance and direction.

The Curve element can be a reference to a stored curve or a complete curve definition:



The “Displacement” element has two child elements:



The attributes of “Displacement” are:

Side	Indicated in what direction to move the curve. Required attribute, possible values are: “For”, “Aft”, “PS”, “SB”, “Top” and “Bot”.

Method	Method for calculating the displaced curve. Possible values are: “X”, “Y”, “Z” and “Perp”. Perp is the default value. X, Y or Z means that the displacement is made along the curves created by intersecting the surface with principal planes of the given type. Perp means that the curves are created by the intersection of the surface and planes that are perpendicular to the base curve in a certain point. The attribute is optional, the default is “Perp”.
Direction	Used to define direction in which End1 and End2 are given. Optional attribute, possible values are: “For”, “Aft”, “PS”, “SB”, “Top” and “Bot”. If omitted the End1 and End2 will be the end1 and end 2 of the base curve.

The “End1” and “End2” element holds displacement data for each end of the base curve:

Distance	The distance between the base curve and the new curve in the current end. If the distance should be equal along the whole base curve, then you only have to give it once (in End1 or End2)
Excess	Excess in the current end. The system will add this measure to the base curve before it starts to calculate the new curve.

Example:

1 (The Curve element is a reference to an existing curve)

```
<Seam ObjId="SPS954">
   <Parallel>
      <Curve>
         <Stored ObjId="SPZ953"/>
      </Curve>
      <Displacement Side="Top" Method="Perp">
         <End1 Distance="1000"/>
         <End2 Distance="1200"/>
      </Displacement>
   </Parallel>
   <Box XMax="FR50" XMin="FR5" YMax="40000" YMin="0" ZMax="40000"/>
</Seam>
```

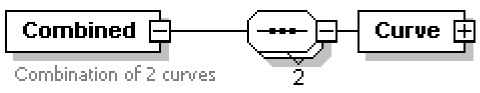
Example:

2 (The Curve element is complete curve definition)

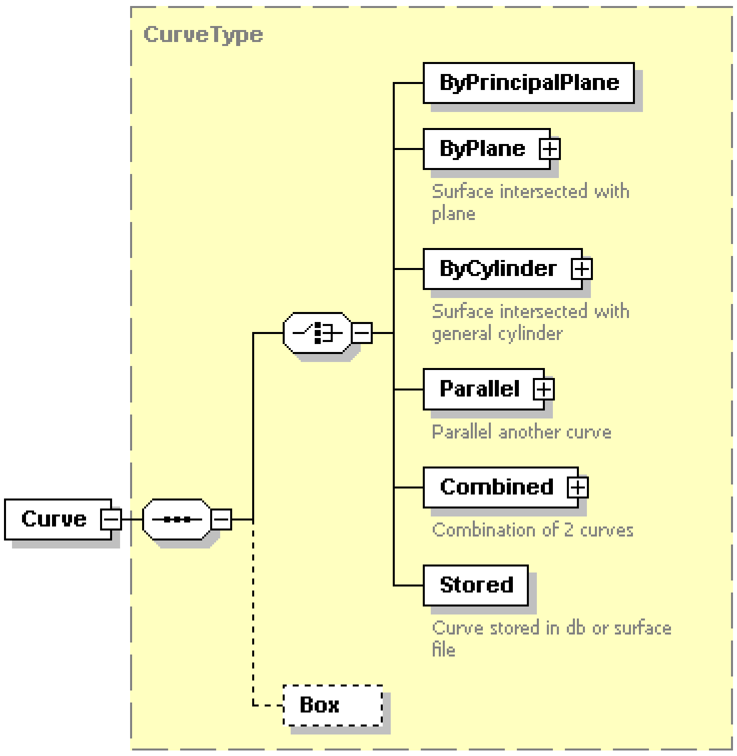
```
<Seam ObjId="SPS954">
   <Parallel>
      <Curve>
         <ByPlane Surface="SPHULL">
            <ByPoints>
               <Point>
                  <Explicit X="18000" Y="2000" Z="10000"/>
               </Point>
               <Point>
                  <Explicit X="10000" Y="2000" Z="10000"/>
               </Point>
               <Point>
                  <OnSurface Approx="Z" X="55000" Y="5000" Z="8000"/>
               </Point>
            </ByPoints>
         </ByPlane>
         <Box XMin="FR40" YMin="0" XMax="FR80" YMax="30000" ZMax="30000"/>
      </Curve>
      <Displacement Side="Top" Method="Perp">
         <End1 Distance="1000"/>
      </Displacement>
   </Parallel>
/Seam>
```

- ### Seam as a Combination of other Curves

A seam can be created as a combination of two other curves by a child element called “Combined”. The Combined element consists of two curve definitions:



Each curve can be e reference to an existing curved or a complete curve definition:



Each Curve element has an attribute, “Side”:

Side	Selects which part of the curve that should be used in the combination. Possible values are: “First”, “Last”, “For”, “Aft”, “PS”, “SB”, “Top” and “Bot”. Required attribute.

Example:

```
<Seam ObjId="SPS963">
   <Combined>
      <Curve Side="Aft">
         <Stored ObjId="SPS961"/>
      </Curve>
      <Curve Side="For">
         <Stored ObjId="SPS962"/>
      </Curve>
   </Combined>
   <Box XMin="FR40" YMin="0" XMax="FR100" YMax="30000" ZMax="40000"/>
</Seam>
```

- **Seam from an Existing/External Curve**

A seam can also be defined from an existing hull curve or from a curve in an external surface system by using the child element “Stored”. It has the following attributes:

ObjId	The name of the existing hull curve or the name of the external curve.
Surface	The name of the surface in the external surface system. If this attribute is omitted, “ObjId” will be interpreted as a hull curve in the data bank.

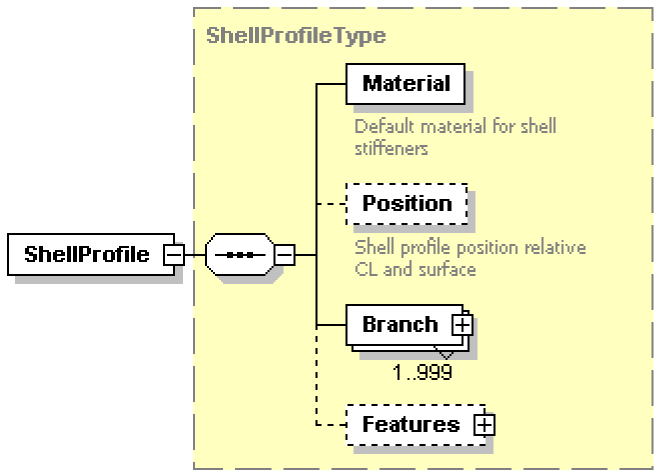
10.5.8 HullCurve Element

The <HullCurve> element is almost identical to the Seam element. You may take any seam definition and make it into a “HullCurve” definition by just replacing the “Seam” element with a “HullCurve” element. However the attributes “BlockLimit” and “Symmetry” are not applicable to a hull curve. (You should of course regard the naming conventions for seams and hull curves as well, and change the “ObjId” attribute if necessary.)

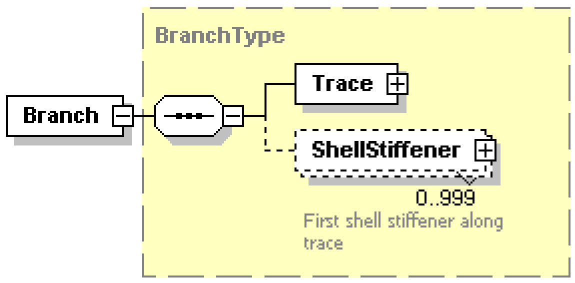
10.5.9 ShellProfile Element

- **Branches and Trace Curves**

The shell profile element models a complete longitudinal or transversal. The shell profile may have one or several branches:


ShellProfileType
Material
Default material for shell stiffeners
Position
Shell profile position relative CL and surface
ShellProfile
Branch
1..999
Features

Each branch has a curve definition and a number of shell stiffeners:



The trace curve is defined as any other curve, as in the "Seam" or "HullCurve" element. Please see section *Seam Element*.

The name of the shell profile is given in the "ObjId" element of ShellProfile element itself:

ObjId	The complete name of the shell profile (including the long/trans group name), e.g "SPT10", "PROFILE30".
SubType	The type of shell profile. Possible values are 'Long' for longitudinal and 'Trans' for transversal. Required attribute.

Other attributes in the shell profile element are:

Delete	If this attribute is "true" then the shell profile will be deleted before it is regenerated. Optional attribute, the default value is "false".
Skip	If Skip = "true" then the progam will ignore this object description. The combination Delete = "true" and Skip = "true" will delete the shell profile but not generate a new one. Optional attribute, the default value is "false".

- **Split Points**

The shell profile may be divided into a series of shell stiffener by a number of "split points" along the branch curve. The split point may be a principal plane or another object like a plane panel, a seam, a curve or another shell profile. The system will split the shell profile in these split points, one by one in the order they are given in input.

Important: The split points must be given in the order they appear along the trace of the shell profile. It does not matter of you arrange them in the same direction as the trace curve or in the reverse direction. However, there must be at least two split points otherwise AVEVA Marine cannot detect that the split points are given in reverse order. Consequently, if there are less than two split points, AVEVA Marine will assume that they are given in the direction of the trace curve.

The split points must be given in "End2" element in the "Connection" element of the shell stiffener. The split point in the first ShellStiffener element will be the split point between the first and the second stiffener. The split point in the second ShellStiffener element will be the split point between the second and the third stiffener, etc.

The split point in End2 of the last Shell Stiffener element is special. This will not be interpreted as a split point, but as a limit for end 2 of the very last shell profile. In the same way you can set a limit for end 1 of the first shell stiffener by defining data in End1/ Connection element of the first ShellStiffener element of the branch.

As a consequence, the number created of shell stiffener model objects is always equal to the number ShellStiffener elements given in the input file. This ShellProfile element will create a transversal with one single shell stiffener. The second end of the stiffener will end at the panel “JUMBO-PLF7000”:

```
<ShellProfile ObjId="SPT900" SubType="Long">
   ...
   <Branch>
      <Trace>
         <ByPrincipalPlane Z="7500"/>
      </Trace>
      <ShellStiffener>
         <End2>
            <Connection>
               <Stored ObjType="PlanePanel" ObjId="JUMBO-PLF7000"/>
            </Connection>
         </End2>
      </ShellStiffener>
   </Branch>
</ShellProfile>
```

This ShellProfile element will create a transversal with two shell stiffeners, one on each side of the panel “JUMBO-PLF7000”:

```
<ShellProfile ObjId="SPT900" SubType="Long">
   ...
   <Branch>
      <Trace>
         <ByPrincipalPlane Z="7500"/>
      </Trace>
      <ShellStiffener>
         <End2>
            <Connection>
               <Stored ObjType="PlanePanel" ObjId="JUMBO-PLF7000"/>
            </Connection>
         </End2>
      </ShellStiffener>
      <ShellStiffener>
      </ShellStiffener>
   </Branch>
</ShellProfile>
```

- **Multiple Intersection Points**

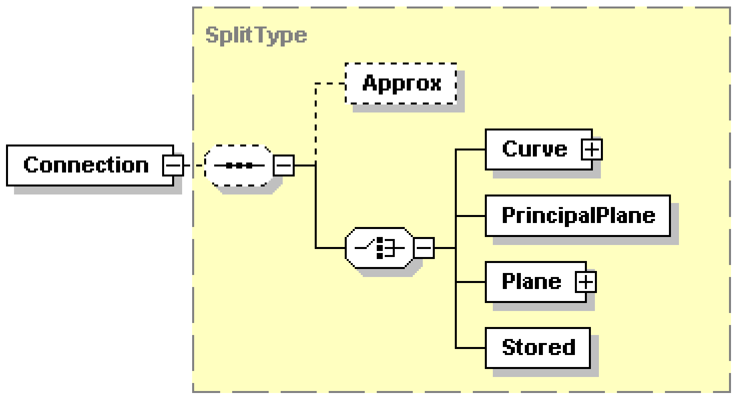
When the shell profile is split by a plane or another objects, it may of course result in many intersection points. However, the Hull XML generator can only handle one intersection point at a time. For that purpose it is possible to give an approximate point. In case of multiple intersections the Hull XML generator will select the one that is closest to the approximated point.

If no approximated point is given in the input, then the first intersection point along the trace line of the shell profile will be selected.

The details about how to give approximated points is explained in the next section, *Split Point Details*

- **Split Point Details**

A split point or an end limit is defined in the "Connection" element in "End1" or "End2" element of a "ShellStiffener". As shown below, the split point can be defined by a complete curve definition, a plane or by an existing object. In case the split results in multiple intersection points, an approximated "point" can be given in the "Approx" element:



The attributes of the Connection element are:

Type	A valid AVEVA Marine connection code, optional element.
Clearance	The clearance between the "true" stiffener end point and the end point of the trace curve. The clearance will be calculated against the plane select by the "Plane" attribute, see below. Note that any value given by "Clearance" will override the connection code. Optional element.
Plane	The plane against which the clearance will be calculated. Possible values: **Cutting** The endcut will be parallel to the plane of the splitting object. **Frame** The endcut will be parallel to the X plane. **Buttock** The endcut will be parallel to the Y plane. **Waterline** The endcut will be parallel to the Z plane. Optional attribute, default value is "Cutting".

An approximated "point" can be given in the "Approx" element. The "point" is defined by three attributes: "X", "Y" and "Z". You don't have to give all, one of them is enough:

X	X-coordinate value, optional attribute.
Y	Y-coordinate value, optional attribute.
Z	Z-coordinate value, optional attribute.

The "Curve" element is the same as described in *Seam Element*.

The "PrincipalPlane" element has one attribute:

X, Y or Z	A single coordinates value in AVEVA Marine format. Examples: "FR10+100", "LP10-250", "14000".

The "Plane" element is documented in *Plane/ByPlane Element*.

The "Stored" element should be used when the split point is an existing model object. It has the following attributes:

ObjType	The type of object, possible values are "HullCurve", "Plane", "PlanePanel", "Seam" and "ShellProfile". Required attribute.
ObjId	The name of the object. Required attribute.
Refl	Indicated whether the object should be used in its normal or reflected position. Possible values are "true" and "false". Optional attribute, default value is "false".

- **Shell Profile Properties**

A shell profile has a number of attributes defining profile properties like profile type and dimensions, profile side and material side. The attributes are divided into two elements, "Material" and "Position". The Material element is a required element and it has attributes defining material properties:

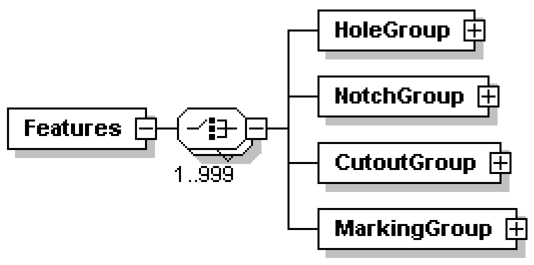
Type	The profile type, required attribute.
Parameters	The profile dimensions, optional attribute. The values must be separated with one or more blanks, "100 10" for instance. Please note that "100,10" is NOT a valid parameter string.
Grade	Material grade (also denoted as "quality" in AVEVA Marine). Optional attribute.

The Position element is optional and has the following attributes:

Symmetry	The symmetry of the shell profile, possible values are "Symmetric", "SB" (valid SB only), "PS" (valid PS only) and "CL" (over/in CL). The attribute is optional, the default value is "Symmetric".

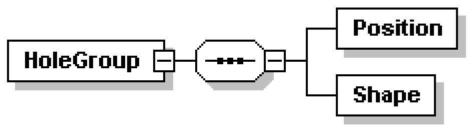
MaterialSide	The direction of the profile material relative to the trace line. Optional attribute. Possible values are “For”, “Aft”, “Top”, “Bot”, “CL” or “Side”.
ProfileSide	Indicates on what side of the surface the shell profile will be located. Possible values are “In” and “Out”. Optional attribute, default value is “In”.

The shell profile may also have features: holes, notches, cutouts and markings which are defined in the “Features” element:



- **Holes**

A group of holes may be defined in a “HoleGroup” element:



HoleGroup has two required elements, “Position” defining the location for the holes and “Shape” defining the type of hole. The Position element must precede the Shape element. HoleGroup has one attribute:

Height	Distance form the trace line to the centre of the hole.

The holes can be positioned where a principal plane intersects the trace line. The Position element have one attribute which is “X”, “Y” or Z”:

X, Y or Z	Hole positions.The string may contain multiple coordinate values separated with blank(s). Repetition terms may also be used. Examples: “FR10(5)FR100”, “1000 2000 3000”, “FR10+50 FR20()25 83000”

The Shape element has the following attributes:

Type	The hole type. Either a standard type like “D”, “HE” etc. or the name of an arbitrary hole geometry. Required attribute.
Parameters	The hole parameters for standard holes. The values must be separated with on or more blanks. Examples: “50”, “100 50”.

Inclination	The angle of the hole geometry.
Mirror	Indicated whether a hole should have its normal appearance or be mirrored about its V-axis.

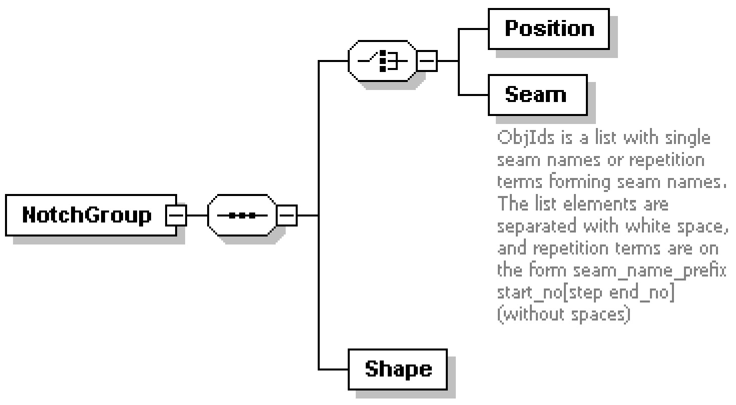
Example:

HoleGroup Example

```
<HoleGroup Height="100">
   <Position Z="800(200)1800  2200 2200"/>
   <Shape Type="D" Parameters="50"/>
</HoleGroup>
```

- **Notches**

A group of notches can be defined by a “NotchGroup” element:


Position
Seam
ObjIds is a list with single seam names or repetition terms forming seam names. The list elements are separated with white space, and repetition terms are on the form seam_name_prefix start_no[step end_no] (without spaces)
NotchGroup
Shape

The position of the notches may be given as a number of principal planes in a “Position” element. As an alternative, a number of seam references may be given in the “Seam” element. The shape of the notch is defined in the “Shape” element.

The NotchGroup element has one attribute:

TopSide	Possible values are “true” and “false”. If “true” the notch will be placed on top of the profile instead of trace edge. Optional attribute, the default value is “false”.

The attributes of the Position element are:

X, Y or Z	Hole positions.Use one of the attributes. The string values may contain multiple coordinate values separated with a blank(s). Repetition terms may also be used. Examples: “FR10(5)FR100”, “1000 2000 3000”, “FR10+50 FR20()25 83000”

The attributes of the “Seam” element are:

ObjIds	A string containing one or several seams names separated with blank(s). Repetition terms can also be used. Required attribute. Examples: “SPS100 SPS101 SPS132”, “SPS100(5)150”, “SPS100(5)150 SPS138”.
Refl	Indicates whether seams should be used in their normal or reflected position. Possible values are “true” or “false”. Required attribute. Default value is “false”

The attributes of the Shape element are:

Type	The notch type. Either a standard notch code or the name of an arbitrary hole geometry. Required attribute.
Parameters	The hole parameters for standard holes. The values must be separated with on or more blanks. Examples: “50”, “100 50”.
Mirror	Indicated whether a notch should have its normal appearance or be mirrored about its V-axis. Possible values are “true” or “false. Optional attribute, the default values is “false”.

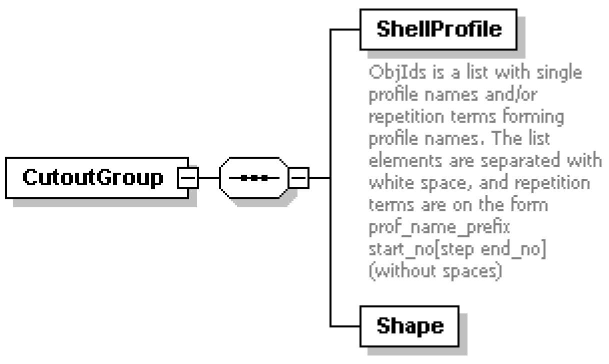
Example:

NotchGroup Example

```
<NotchGroup>
   <Seam ObjIds="SPS900()904 SPS908"/>
   <Shape Type="R" Parameters="50"/>
</NotchGroup>
```

- Cutouts

A group of cutouts can be defined in a “CutoutGroup” element:


ShellProfile
ObjIds is a list with single profile names and/or repetition terms forming profile names. The list elements are separated with white space, and repetition terms are on the form prof_name_prefix start_no[step end_no] (without spaces)
CutoutGroup
Shape

The “Shell Profile” element has attributes forming references to intersecting shell profiles:

ObjIds	A string containing one or several shell profile names separated with blank(s). Repetition terms can also be used. Required attribute. Examples: “SPT10 SPT11 SPT12”, “SPT10()20”, “SPT10()20 SPT22 SPT30(2)38 SPT41”
Refl	Indicates whether the profiles should be used in their normal or reflected position. Possible values are “true” or “false”. Required attribute. Default value is “false”

The Position element has attributes controlling the shape of the cutout:

Type	The AVEVA Marine cutout code. Required attribute.
Parameters	Additional cutout parameters separated by blanks. Optional attribute.

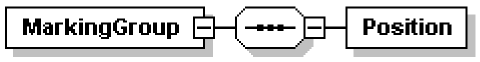
Example:

CutoutGroup Example:

```
<CutoutGroup>
   <ShellProfile ObjIds="SPL910 SPL915"/>
   <Shape Type="10"/>
</CutoutGroup>
```

- **Markings**

A group of marking lines be defined in a “MarkingGroup” element:



MarkingGroup attributes:

Length	The length of the marking line(s). Length may also have the values “ProfileHeight” which indicates that the length of the marking line should match the profile height.
Inclination	The angle between the marking line(s) and the trace.Inclination may also have the values: **Perp** - The marking lines will be perpendicular to the profile trace line **Plane** - The marking lines will be in the plane given in the “Position” attribute Optional attribute, default value is “Perp”.

Text	The marking text.
Symmetric	Indicates whether the marking lines should be on both sides of the shell profile. Possible values are “true” and “false” (“true” indicates both sides). Optional attribute, default value is “false”.

The marking lines can be positioned where a principal plane intersects the trace line. The Position element have one attribute which is “X”, “Y” or Z”:

X, Y or Z	Positions for the marking lines. The string may contain multiple coordinate values separated with blank(s). Repetition terms may also be used. Examples: “FR10(5)FR100”, “1000 2000 3000”, “FR10+50 FR20()25 83000”

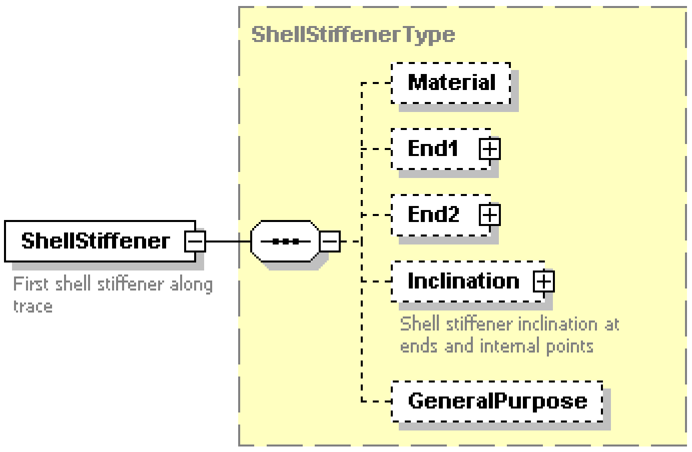
Example:

MarkingGroup Example

```
<MarkingGroup Length="ProfileHeight" Inclination="Perp"
              Text="MARKING_TEXT" Symmetric="true">
   <Position Z="4500 5500"/>
```
</MarkingGroup>

• Shell Stiffener Properties

Each shell stiffener element may also set a number of properties for the stiffener; bevel, endcut, connection codes, etc. The shell stiffener has a great number of attributes organized in several child elements:


ShellStiffenerType
Material
End1
End2
Inclination
Shell stiffener inclination at ends and internal points
GeneralPurpose
ShellStiffener
First shell stiffener along trace

The attributes of the ShellStiffener element:

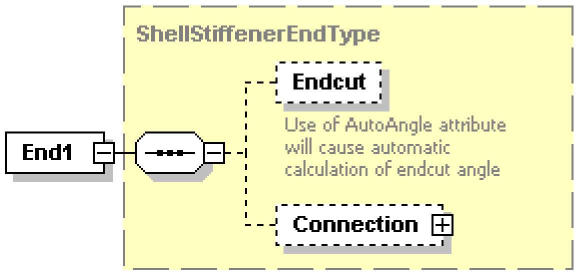
Symmetry	The symmetry for the shell stiffener.
Dummy	Flag indicating whether this is a true shell stiffener or a dummy interval. Possible value are “true” or “false”. Optional attribute, default value is “false”.
Posno	The position number.
BevelTrace	The bevel code for the bevel applied along the trace of the shell stiffener.
Shrinkage	Shrinkage, optional attribute.
WeldDepth	Fillet weld depth, optional attribute

The “Material” element has the same attributes as the Material element of the shell profile. However, here it is an optional element. It omitted the shell stiffener will inherit these properties from the “Material” element from the shell profile.

- Endcut, Bevel and Connection Data

In End1 and End2 you may set attributes for each stiffener end, like endcut, connection code, bevel codes, etc. Please note that end1 and end2 refers to the direction the shell stiffeners are given in the input file, not the true end1/end2 of the shell stiffener model object in the data bank. In other words, data given in the End1 one element will set properties in end2 of the stiffener object if you given the ShellStiffener elements in reverse direction compared to the trace curve.

End1 and End2 has two child element “Endcut” and “Connection”:


ShellStiffenerEndType
Endcut
Use of AutoAngle attribute will cause automatic calculation of endcut angle
End1
Connection

End1/End2 attributes

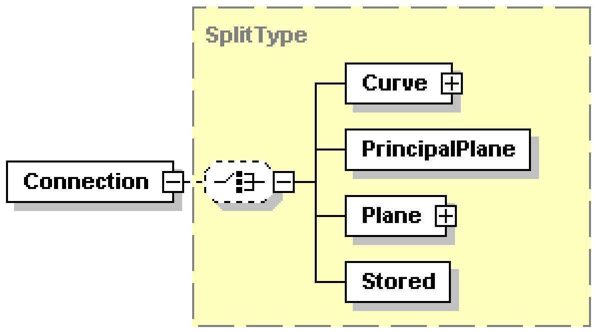
Excess	Excess value, optional attribute.
BevelWeb	Bevel on the profile web, optional attribute.
BevelFlange	Bevel on the profile flange, optional attribute.

Endcut attributes:

Type	AVEVA Marine endcut code according to *Hull Standards*. Optional attribute.

Parameters	Additional endcut parameters. Optional attribute.
AutoAngle	Indicates how the endcut angle should be calculated, optional attribute. Possible values: **Cutting** - The endcut will be parallel to the plane of the splitting object. **Frame** - The endcut will be parallel to the X plane. **Buttock** - The endcut will be parallel to the Y plane. **Waterline** - The endcut will be parallel to the Z plane.

In the Connection element the "split points" and end limits can be defined. This is described in the section *Branches and Trace Curves*.

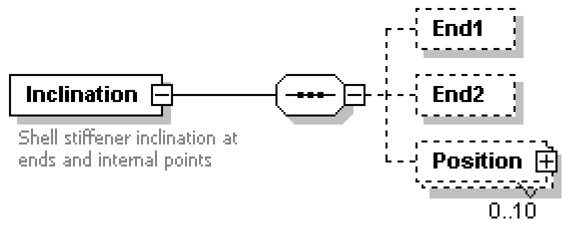


Connection attributes:

Type	Connection code according to *Hull Standards*. Optional attribute.
Clearance	The clearance between the stiffener end and the plane selected by the "Plane" attribute, see below. Optional attribute
Plane	Indicates how the clearance should be calculated. Possible values: **Cutting** - The clearance will be calculated perpendicular to the plane of the object used to split the shell profile. **Frame** - The clearance will be perpendicular to the X plane. **Buttock** - The clearance will be perpendicular to the Y plane. **Waterline -** The clearance will be perpendicular to the Z plane. Optional attribute, the default value is "Cutting".

- **Inclination Data**

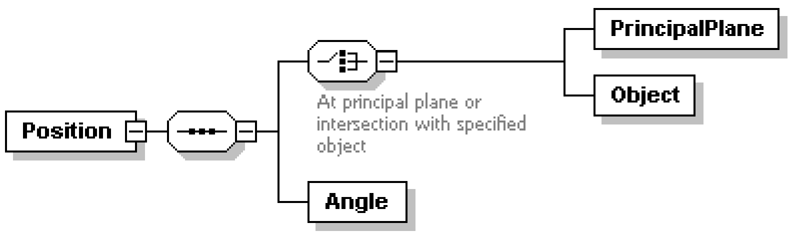
The "Inclination" element there are data describing the inclination between the web of the shell profile and the surface. The inclination can be defined in each end of the profile and in a number of points along the trace of the shell stiffener:



End1/End2 attributes:

Axis	Possible values are: **Perp -** The shell stiffener should be perpendicular to the surface in this end **X** - The angle should be measure against the X-axis (in the XY-plane) **Y** - The angle should be measure against the Y-axis (in the YZ-plane) **Z** - The angle should be measure against the Z-axis (in the XZ-plane) Required attribute.
Angle	The inclination angle. Only relevant if Axis is “X”, “Y” or “Z”.

The stiffener inclination may also be controlled in a number of points along the trace of the stiffener, each point given in a “Position” element:



The location of the “inclination point” can be define by a principal plane or where another object intersects the shell stiffener.

PrincipalPlane has one attribute:

X, Y or Z	You select one of the attributes X, Y or Z. The value is a single coordinate value. It can be a single coordinate value given in the traditional AVEVA Marine format, for instance:
	X=”FR20”
	Y=”LP10+200”
	Z=”14500”

Object attributes:

ObjType	The type of object, possible values are "HullCurve", "Plane", "PlanePanel", "Seam" and "ShellProfile". Required attribute.
ObjId	The name of the object. Required attribute.
Refl	Indicated whether the object should be used in its normal or reflected position. Possible values are "true" and "false". Optional attribute, default value is "false".

The "Angle" element has two attributes: "Axis" and "Angle". They are the same as in the "Inclination/End1" element.

- **General Purpose Data**

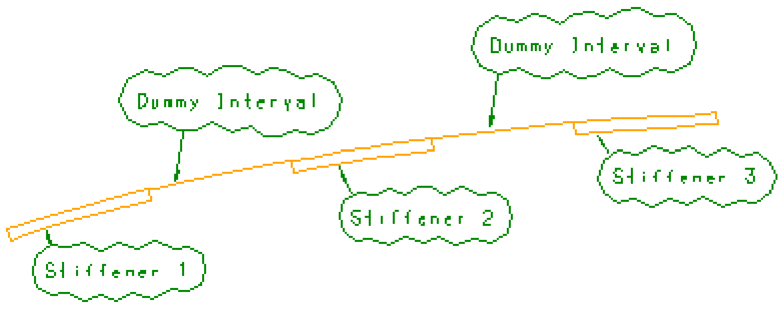
In the "ShellStiffener/GeneralPurpose" element there are attributes defining production data:ShellStiffener element.

GPS1	General purpose string 1
GPS2	General purpose string 2
GPS3	General purpose string 3
GPS4	General purpose string 4
LocationCode	Location code
SurfTreat	Surface treatment
Dest	Destination
PartsList	Parts list name

We have already seen a "ShellStiffener" element that can be used within the "ShellProfile/ Branch" element. This "ShellStiffener" element is similar but can appear directly under the "Ship" root element.

While the "ShellProfile" models a complete shell profile along its full trace line (or trace lines in case of multiple branches), the ShellStiffener element will generate a single shell stiffener in the profile.

This alternative way to generate a shell profile "stiffener by stiffener" is primarily implemented to support conversion from the PROFGEN[1] input files to the new XML format. However the resulting shell profile object will still apply to the same storing conventions as when generating the profile by the "ShellProfile" element. For example: a branch of the profile always has a continuous chain of shell stiffeners without any gaps. AVEVA Marine will achieve this by adding dummy intervals to cover possible gaps between stiffeners:



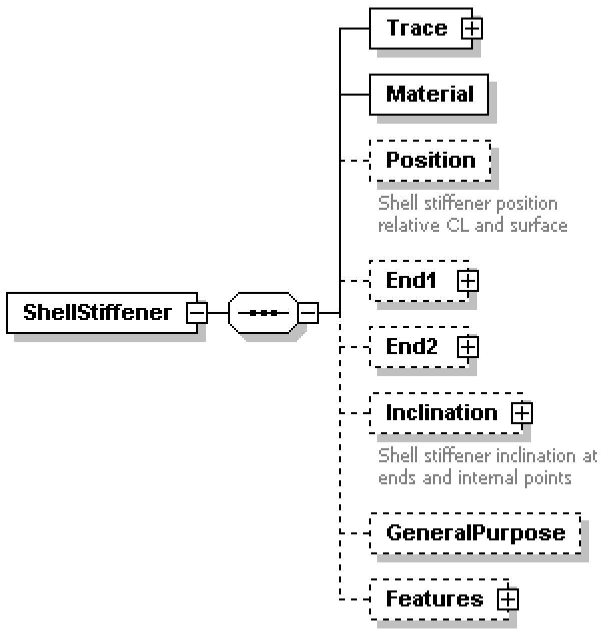
(1) PROFGEN is a program in Tribon 5 that generates shell profiles in batch via an input file. It is not available in AVEVA.

When generating a shell profile "stiffener by stiffener" AVEVA Marine will try to "cleanup" among the dummy intervals to avoid fragmentation in many small intervals. This means that AVEVA Marine will:

- minimize the number of dummy intervals, by joining neighbouring intervals into one.
- remove dummy intervals in the ends of a branch
- remove a branch if it only consists of dummy intervals

Conclusion: If you want to have your own user-defined dummy intervals, you may run into problem when generating the profile "stiffener by stiffener". You should then consider to use the "ShellProfile" element instead.

The ShellStiffener element looks like this:


ShellStiffener
Trace
Material
Position
Shell stiffener position relative CL and surface
End1
End2
Inclination
Shell stiffener inclination at ends and internal points
GeneralPurpose
Features

The attributes of the ShellStiffener element are:

ObjId	The name of the shell stiffener.
ShellProfileId	The name of the shell profile to which this stiffener will belong
SubType	The type of shell profile to which this stiffener will belong. Possible values are ‘Long’ for longitudinal and ‘Trans’ for transversal. Required attribute.
DeleteProfile	If this attribute is “true” then the ***shell profile*** will be deleted before it is regenerated. Please note that it is the whole longitudinal/transversal that will be deleted! This is an optional attribute, the default value is “false”.
Skip	If Skip = ”true” then this the program will ignore this object description. The combination Delete = ”true” and Skip = ”true” will delete the shell profile but no new shell stiffener will be generated. Optional attribute, the default value is “false”.
Dummy	Flag indicating whether this is a true shell stiffener or a dummy interval. Possible value are “true” or “false”. Optional attribute, default value is “false”. However, keep in mind that AVEVA Marine might reorganize the dummy intervals, see above.
Posno	The position number.
BevelTrace	The bevel code for the bevel applied along the trace of the shell stiffener.
Shrinkage	Shrinkage, optional attribute.

This ShellStiffener element is quite similar to the “ShellProfile/Branch/ShellStiffener” element. For instance, the “Material”, “End1”, “End2”, “Inclination” and the “GeneralPurpose” elements are exactly the same, please see *Shell Stiffener Properties*.

The “Position” element is the same as ShellProfile/Position, please see *ShellProfile Element*.

Naturally, the Trace element defines the trace curve of the shell stiffener. It can be defined in (almost) the same way as the trace curve for a shell profile, please see *Branches and Trace Curves* for shell profiles.

Important: The trace curve for a shell stiffener is currently restricted to a planar curve or a along a named curve.

You may also create feature like holes, notches, cutouts and marking lines. The features are defined in a “Features” element, please see *Holes*, *Notches*, *Cutouts* and *Markings* in *Shell Profile Properties*

.

Example:

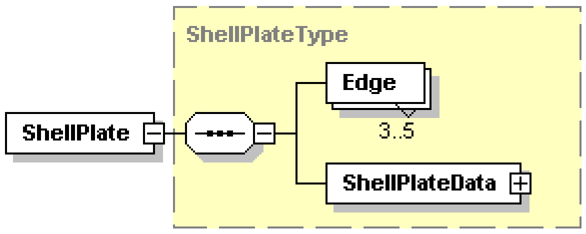
```
<ShellStiffener ObjId="SPL950-S1" ShellProfileId="SPL950" SubType="Long">
   <Trace>
      <ByPrincipalPlane Z="5000"/>
   </Trace>
   <Material Type="10" Parameters="300 25" Grade="A36"/>
   <Position ProfileSide="In" MaterialSide="For"/>
   <End1>
      <Connection Type="70" Clearance="8" Plane="Cutting">
         <PrincipalPlane X="FR20"/>
      </Connection>
   </End1>
   <End2>
      <Connection Type="70" Clearance="10" Plane="Cutting">
         <PrincipalPlane X="FR25"/>
      </Connection>
   </End2>
   <Inclination PerpWhole="true"/>
   <GeneralPurpose GPS1="GPS1" GPS2="GPS2" GPS3="GPS3" GPS4="GPS4"/>
   <Features>
      <HoleGroup Height="100">
         <Position X="FR21(1)23"/>
         <Shape Type="D" Parameters="50"/>
      </HoleGroup>
      <MarkingGroup Length="ProfileHeight" Inclination="Perp"
                    Text="MARKING_TEXT" Symmetric="true">
         <Position X="FR24"/>
      </MarkingGroup>
   </Features>
</ShellStiffener>
```

10.5.10 ShellPlate Element

The “ShellPlate” element is implemented in Tribon M2 SP2 or later.

Important: Generating shell plates and curved panels only works with special settings in the environment! Please see *Naming Rules for Parts of a Curved Panel*

The “ShellPlate” element defines a shell plate in curved hull:



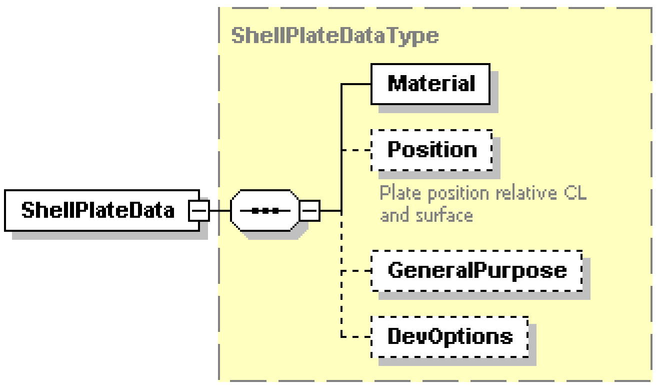
The ShellPlate element may have these attributes:

ObjId	The name of the plate. The name should reflect the symmetry of the of the plate by having a proper suffix: empty suffix for a symmetric plate, “P” for portside specific, “S” for starboard specific and “SP” for plates extending over/in CL. However, if the suffix is omitted the system will automatically add the correct suffix depending on the value of the “Symmetry” attribute. ObjId is a required attribute
PosNo	The position number of the plate. Optional attribute

You may select 3-5 seams that will defines the outer contour of the plate. The seams should be given in a clockwise order (looking outwards from the inside of the surface) and starting by the aftmost. Each seam is given in a “Edge” element, which may also contain details such as bevel, excess, compensation, etc. This is the complete list of attributes in the “Edge” element:

ObjId	The name seam used to limit the boundaries of the plate.
Refl	**false**: The seam defined by “ObjId” should be used in it normal position. **true**: The seam defined by “ObjId” should be used in it reflected position.
Bevel	Bevel code
Excess	Excess value
ExcessType	Excess type
Compensation	Compensation value
CompensationCha nge	Compensation change value

A shell plate may also have a set of properties given in the “ShellPlateData” element:



- **Material Element**

Material is a required element, holding the following attributes:

Thickness	The plate thickness. Optional attribute with default value 10.
Grade	Grade, also mentioned “plate quality”.
LaminateThis	Laminate code for “this” side. This /other side is defined by “MaterialSide”, see below.
LaminateOther	Laminate code for “other” side.

- **Position Element**

The “Position” element is optional:

Symmetry	The symmetry of the plate, possible values are “Symmetric”, “SB” (valid SB only), “PS” (valid PS only) and “CL” (over/in CL). The attribute is optional and the default value is “Symmetric”
MaterialSide	The material side: “In” or “Out”. Optional attribute with default value “Out”. Optional attribute with default value “Out”.
Offset	Controls how much of the plate thickness that will be on the inside and outside of the mould surface respectively. Offset is always in the opposite direction of the “MaterialSide”. Optional attribute, the default value is 0. Example: Thickness is 15, MaterialSide is “Out” and Offset=0. The whole plate thickness will be on the outside of the mould surface.

- **GeneralPurpose Element**

The “GeneralPurpose” element is optional, since it holds only optional attributes:

GPS1	Identification string to be defined by the user.
GPS2	Identification string to be defined by the user.
GPS3	Identification string to be defined by the user.
GPS4	Identification string to be defined by the user.

SurfTreat	Identification string to be defined by the user.
Dest	Identification string to be defined by the user.
RawPlate	Name of raw plate to be used.

- **DevOptions Element**

The optional element “DevOptions” contains data controlling how to develop the shell plate. If the whole element “DevOptions” is omitted then default for the project will be used. All the attributes are optional:

RollAxes	Selects whether rollaxes should be calculated or not. **true** - calculate rollaxes **false** - no calculation of rollaxes. By default, rollaxes will be calculated.
WorkshopMethod	This attribute selects the workshop method used to form the curvature of the plate. The system will adjust the size of the plate for the selected method. Possible values are “**Contraction**“, “**Expansion**“ or “**NoDeformation**”. When “NoDeformation” is selected then the system will make no adjustment of the size of the plate. “NoDeformation” can only be used for plates that are single curved, or close to single curved. The default value is “Contraction”.
RawPlateMargin	The least rectangle circumscribing the developed plate will be enlarged with this margin along edges.
StripDirection	This attribute may be used for plates, where the normal development fails. StripDirection = “X” means for instance that frame sections will be used for the strips. (Strips are sections used in the development process.) Possible values are “**X**”, “**Y**”, “**Z**” or “**Auto**”. Auto is the default.
NumStrips	Number of strips and triangles above and below the baseline used in the development process. If not given, this number will be selected automatically depending on the curvature of the plate.
SplineTolerance	Used by the spline function when creating the plate edges. Default value is 1.

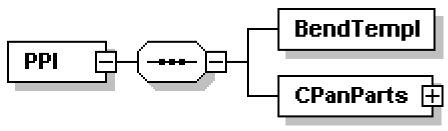
Example:

```
<ShellPlate ObjId="SPSHP-100" PosNo="1">
   <Edge ObjId="SPS502" Bevel="203" Excess="5"/>
   <Edge ObjId="SPS602" Bevel="201" Excess="4"/>
   <Edge ObjId="SPS501" Bevel="202" Excess="5"/>
   <Edge ObjId="SPS601" Bevel="203" Excess="4" />
   <ShellPlateData>
      <Material Grade="A" Thickness="10"/>
      <Position MaterialSide="Out" Symmetry="Symmetric" Offset="5"/>
      <GeneralPurpose GPS1="A1-3" GPS2="A1-2" GPS3="A1-3" GPS4="A1-4" />
   </ShellPlateData>
</ShellPlate>
```

- **Production Program Interface Data (PPI)**

This element contains data used by some of the production programs. Currently you may set data that will be used by the “bending template” and the “Curved Part Generation” programs. The PPI data is optional, if not given it will be controlled by default values.

The PPI data are organized into two sub parts, one for bending templates and one for “CPanParts” settings:



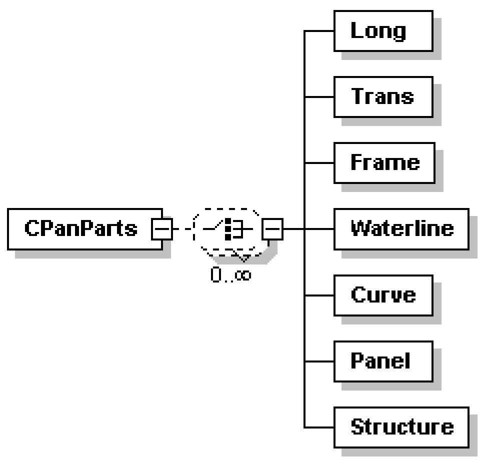
Bending Template Data

The attributes in this element will control where the bending templates will be placed.

ObjId	The name of the seam (limit) along which the templates will be placed.
Side	The side of the plate where the templates will be placed. Possible values are: **In -** the inside of the plate **Out -** the outside of the plate **Auto -** The system will automatically select the side.
Distances	A number of distance values, selecting the position of the templates.

CPanParts Data

This part contains data controlling the marking of the plate. The data is optional, if omitted the marking will be controlled by default values: **MARK_FR**, **MARK_WL**, **MARK_LONG**, etc.



The attributes of the "CPanParts" element are flags that will override the IP values for this plate:

MarkTemplate	This attribute will override the default value **MARK_TEMPL** for this plate. Possible values are: "true" - Mark all the template curves "false" - Do not mark any template curves. If the attribute is omitted, the current setting default value will control the marking of template curves.
MarkPanel	Similar to the "**MarkTemplate**" attribute, "**MarkPanel**" will override the default value **MARK_PLATE** for this plate (controlling whether abutting panels will be marked or not).
MarkTrans	do. for the default value "**MARK_TRANS**".
MarkLong	do. for the default value "**MARK_LONG**".
MarkFr	do. for the default value "**MARK_FR**".
MarkWl	do. for the default value "**MARK_WL**".
MarkStruct	For future use (currently there is no default value controlling the marking of structures)

You may also select a number of objects to be specifically marked. You may give a list of object names in the elements "Long", "Trans", "Frame", "Waterline", "Curve", "Panel" and "Structure". Each element has an "ObjIds" attribute that is a list of object names.

Note: If you give a list of object names for a group, i.e. some transversals as ObjIds="SPT901 SPT902" then only these transversals will be marked and no other transversals.

In the example below the following will apply to the plate:

- All template curved will be marked.
- No longitudinals will be marked
- The marking of frames and waterlines will be controlled by the default values "**MARK_FR**" and "**MARK_WL**".
- The abutting panel "SP241-50SP" will be marked, but no other panels.

- Transversals “SPT900” AND “SPT901” will be marked, but no other transversals.
- The curves “SPY200” AND “SPY201” will be marked.

Example:

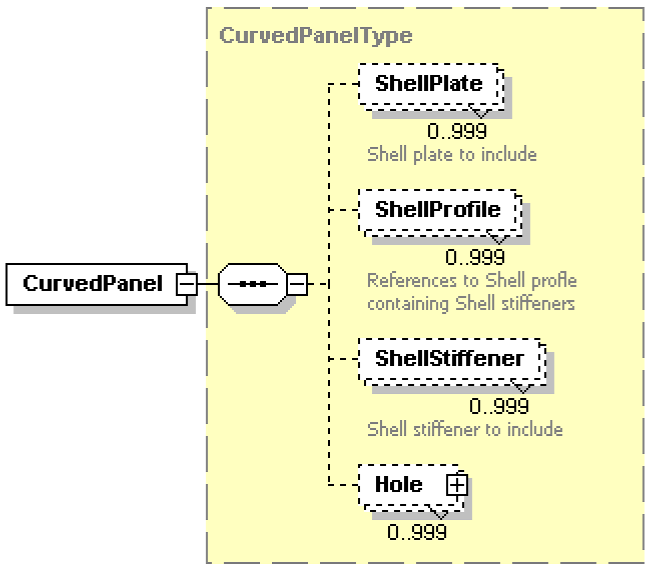
```
<ShellPlate ObjId="SPSHP-100" PosNo="1032">
   ...
   <ShellPlateData>
      ...
      <PPI>
         <BendTempl ObjId="SPS502" Side="Auto"
                    Distances="1000 1250 1450"/>
         <CPanParts MarkTemplate="true" MarkLong="false" >
            <Panel ObjIds="SP241-50SP"/>
            <Trans ObjIds="SPT900 SPT901"/>
            <Curve ObjIds="SPY200 SPY201"/>
         </CPanParts>
      </PPI>
   </ShellPlateData>
</ShellPlate>
```

10.5.11 CurvedPanel Element

The “CurvedPanel” element is implemented in Tribon M2 SP2 or later.

Important: Generating shell plates and curved panels only works with special settings in the environment! Please see *Naming Rules for Parts of a Curved Panel*

The CurvedPanel element can be used to create a curved panel. Basically you select the shell plates and shell stiffeners that you want to be included in the panel. You can also create holes:



In the CurvedPanel element you may give these attributes:

ObjId	The name of the panel. The name should reflect the symmetry of the symmetry of the panel by having a proper suffix: empty suffix for a symmetric panel, "P" for portside specific, "S" for starboard specific and "SP" for panels extending over/in CL. However, if the suffix is omitted the system will automatically add the correct suffix depending on the value of the "Symmetry" attribute. ObjId is a required attribute
Symmetry	The symmetry of the panel, possible values are "Symmetric", "SB" (valid SB only), "PS" (valid PS only) and "CL" (over/in CL). The attribute is optional and the default value is "Symmetric"
Block	The block to which the curved panel will belong. Required attribute.

The "ShellPlate", and the "ShellStiffener" elements have one single attribute "**ObjId**" which is the name of the plate/profile/stiffener that you want to include.

As an alternative to the "ShellStiffener" element, you may give the name of the whole shell profile in a "ShellProfile" element. This means that the system will automatically select the stiffeners of the longitudinal/transversal that are within the curved panel. The stiffener is accepted if the circumscribed box of the stiffener is inside the circumscribed box of the panel. The attributes of the "ShellProfile" element are:

ObjId	The name of the shell profile, required attribute.
Tolerance	This is a measure that the tolerance of the selection. The system will increase (Tolerance > 0) or decrease (Tolerance < 0) the circumscribed box of the panel before the comparison.

.

Example:

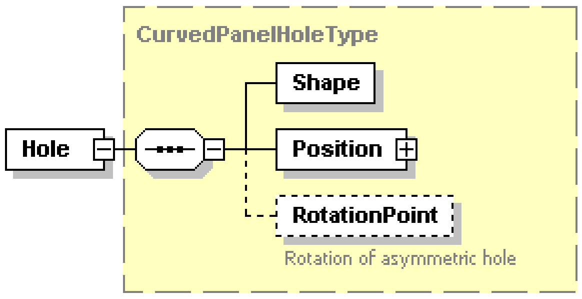
```
<CurvedPanel ObjId="SP241-130" Block="SP241" Symmetry="Symmetric">
   <ShellPlate ObjId="SPSHP-100"/>
   <ShellPlate ObjId="SPSHP-101"/>
   <ShellPlate ObjId="SPSHP-102"/>
   <ShellPlate ObjId="SPSHP-103"/>
   <ShellStiffener ObjId="SPL902-S2"/>
   <ShellStiffener ObjId="SPL902-S3"/>
   <ShellStiffener ObjId="SPL903-S2"/>
   <ShellStiffener ObjId="SPL903-S3"/>
</CurvedPanel>
```

Note: Please note that the order of the “ShellStiffener” and the “ShellPlate” element may be important. The order affects the “sequence number” that is assigned to the stiffener/plate when added to the panel. The sequence number is a number within the panel scope and there is one series for shell stiffeners and another series for the shell plates. The sequence number is used (for instance) when referencing the stiffener/plate from an assembly and it also forms the name of the part in the SB_PLDB/SBH_PROFDB data banks.

If you keep the same order, the sequence number will always be the same each time you rerun the XML input file.

- ### Holes in Curved Panel

The curved panel may also have holes:



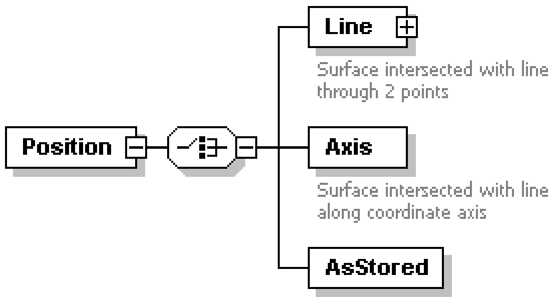
The attributes of the Hole element are

Symmetry	Selects the symmetry of the hole. Possible values are “AsPanel”, “PS” or “SB”. Optional attribute, the default value is “AsPanel”. “PS” and “SB” are relevant for symmetric panels to indicate that this hole is valid for portsida or starboard only. (For backward compatibility reasons the value “Symmetric” can still be used, it has the same effect as “AsPanel”.)
Marked	Possible values are “true” or “false”. If “true” the hole will only be marked, not burnt. Optional, default is “false”.
Developed	Possible values are “true” or “false”. “True” specifies that the hole should be developed. Optional, default is “true”.
Bevel	The bevel code, optional.
MarkOption	This attribute selects how the hole will he marked: • “Hole” Only hole to be marked (default). • “Cross” Only crossmark to be marked. • “Both” Crossmark and hole Optional attribute, the default value is “Hole”.
MarkType	This attribute selects type of crossmark: • “Big” Big cross over hole • “Small” Small cross • “Special” Special “4”-shaped cross. Optional attribute.
MarkLen	The length of the marking. Optional attribute

The **Shape element** holds attribute selecting the shape of the whole:

Type	The type the whole. It can be an standard hole like “D”, “HO”, “HE”, etc. or and arbitrary hole contour stored as a curve. In case of arbitrary hole, “Type” is the name of the curve. Required attribute.
Parameters	Parameters controlling hole measures. Required for standard holes, irrelevant for arbitrary hole. Please note that the parameters must be given with one or several blanks as delimiter: “50”, “250 50”, etc.

In the **Position element there are data defining the origin of the whole:**


Line
Surface intersected with line through 2 points
Position
Axis
Surface intersected with line along coordinate axis
AsStored

The origin of the hole is calculated as the intersection between a line and the surface. The line in question can be defined in three ways:

1) Limited Line

The line is defined by a start point and an end point. Both points are given by three coordinates.

Example:

```
<CurvedPanel ObjId="SP241-130" Block="SP241"Symmetry="Symmetric">
   ...
   <Hole>
      <Shape Type="HO" Parameters="75 35"/>
      <Position>
         <Line>
            <Point X="FR76+200" Y="5000" Z="7500"/>
            <Point X="FR76+250" Y="20000" Z="7500"/>
         </Line>
      </Position>
   </Hole>
</CurvedPanel>
```

2) Unlimited, Axis-parallel Line

The line is unlimited and parallel to one of the coordinate axes. Since an intersection between an unlimited line and the surface may result in multiple intersection points, you must supply an approximate coordinate (along the “parallel” axis). The **Axis** element has four attributes:

Approx	Selects the axis to which the line will be parallel.
X,Y,Z	Two coordinates values define the line and the third one (indicated by “Approx”) is an approximate coordinate. Example: Approx = “Y”, X=”FR78”, Y=”10000”, Z=”5500”. The line will be parallel to the Y-axis. The line is the intersection of the two principal planes X=”FR78” and Z=”5500”. In case of multiple intersections the system will select the point closest to the principal plane Y=”10000”.

Example:

```
<CurvedPanel ObjId="SP241-130" Block="SP241" Symmetry="Symmetric">
   ...

   <Hole>
      <Shape Type="D" Parameters="100"/>
      <Position>
         <Axis Approx="Y" X="FR78+200" Y="10000" Z="7700"/>
      </Position>
   </Hole>
</CurvedPanel>
```

3) As Stored

This option is applicable only when the hole is an arbitrary shape defined by a curve object. This curve contour is defined i a specific plane (for instance a view plane in X=FR50). The contour will be projected into the plate/panel along the normal of the plane.

The “AsStored” element may contain an approximate coordinate value in case of multiple intersections with the surface:

X,Y or Z	You give one of these attribute to define an approximate coordinate.

In case of an asymmetric hole the rotation must be defined. You may select a point or a vector defining the direction of the of the U-axis of the local hole coordinate system.

The **RotationPoint** element defines a point or a vector by giving three coordinates **X, Y** and **Z**. The system will interpret these values an a vector if the length is < 1. In the case a point is given the system calculates a vector from the origin of the hole to the point. In both cases the resulting vector is projected into the tangent plane.

- **Curved Panel a Complete Example**

Example:

```
<CurvedPanel ObjId="SP241-130" Block="SP241" Symmetry="Symmetric">
   <ShellPlate ObjId="SPSHP-100"/>
   <ShellPlate ObjId="SPSHP-101"/>
   <ShellPlate ObjId="SPSHP-102"/>
   <ShellPlate ObjId="SPSHP-103"/>
   <ShellStiffener ObjId="SPL902-S2"/>
   <ShellStiffener ObjId="SPL902-S3"/>
   <ShellStiffener ObjId="SPL903-S2"/>
   <ShellStiffener ObjId="SPL903-S3"/>
   <Hole>
      <Shape Type="HO" Parameters="75 35"/>
      <Position>
         <Line>
            <Point X="FR76+200" Y="5000" Z="7500"/>
            <Point X="FR76+250" Y="20000" Z="7500"/>
         </Line>
      </Position>
      <RotationPoint X="0.0" Y="0.2" Z="0.5"/>
   </Hole>
   <Hole>
      <Shape Type="HO" Parameters="75 35"/>
      <Position>
         <Axis Approx="Y" X="FR78+200" Y="10000" Z="7700"/>
      </Position>
      <RotationPoint X="0.0" Y="0.0" Z="0.5"/>
   </Hole>
</CurvedPanel>
```

10.5.12 Naming Rules for Parts of a Curved Panel

Traditionally, shell plates and shell stiffeners have been renamed when included in a curved panel (to include the name of the panel they belong to).

This handling has turned out to have some drawbacks, especially when implementing batch mode in curved hull. For this reason, Tribon M2 SP2 or later introduces a new way to handle the naming of shell plates and shell stiffeners. They will no longer be renamed when added to the curved panel.

In Tribon M2SP2 or later M2 versions the *new* naming rules will be applied when a special flag, “SBH_CPAN_RENAME_PARTS” is set to a value other than *YES* (case sensitive!).

In 12.0 the *new* naming rules is the default. The *old* naming rules will be applied only if “SBH_CPAN_RENAME_PARTS” is set to *YES* (case sensitive!).

Important: The generation of shell plates and curved panels in Batch Curved Hull depends on the new naming rules. It will NOT work properly with old naming rules.

The new naming rules will have two major impacts:

1. Shell plates and shell stiffeners *will not be renamed* when added to a curved panel.
2. When parts are transferred to the production data banks (SB_PLDB/SBH_PROFDB), they will be named according to new principles. See details in *Naming of Parts on the Production Data Banks* below.

Important: The flag "**SBH_CPAN_RENAME_PARTS**" must be used with caution! Once you have started to generate curved panels with the new/old naming rules, you should stay in this mode. It is NOT recommended to jump back and forth between new and old naming rules (at least not within the same project).

- **Naming of Parts on the Production Data Banks**

Although the new naming rules will not rename the plates/stiffeners in the model data bank (SB_OGDB) they will be renamed when transferred to the production data banks (SB_PLDB/SBH_PROFDB). The name in the production data bank will be based on the panel name and a sequence number to form a unique AVEVA Marine name for the part. The "sequence number" is just a unique number within the panel (one series for plates and one series for stiffeners) and carries no specific model information (in contrast to position number). The sequence number will be assigned automatically when adding plates and stiffeners to the curved panel.

Examples of what the names may look like are given in the table below:

Curved panel name: SP241-30

Component ID within the panel (sequence number)	Name in SB_OGDB	Name in SB_PLDB/SBH_PROFDB
1 (Plate)	SPSHP-200	SP241-30-**1**S, SP241-30-**1**P
2 (Plate)	SPSHP-201	SP241-30-**2**S, SP241-30-**2**P
1 (Stiffener)	SPL960-S3	SP241-30/S**1**S, SP241-30/S**1**P
2 (Stiffener)	SPL970-S10	SP241-30-S**2**S, SP241-30-S**2**P

10.5.13 Important Restrictions

It is important to understand that when a model object is generated from an input file it will be calculated "from scratch" every time you run the input file. For example: if a shell profile already exists in the data bank the old profile will be deleted before the new one is generated. (However the if the input file fails then the old shell profile and all its shell stiffener will remain in the data bank.)

This has an impact in some situations explained below:

Switching Between Batch and Interactive Generation

Consider the following sequence of operations:

1. Create shell profile via XML input.
2. Modify the profile in interactive Curved Hull.
3. Run the XML input file again. The modifications made in interactive Curved Hull are now lost!

Connections to Curved Panels

When generating shell profiles from an input file, the shell stiffeners will get auto generated names. This means that *connections to curved panels will be lost since* this connection is based on the name of the shell stiffeners.

Note: This restriction only applies to Tribon version M2 SP1 or earlier. In Tribon M2 SP2 or later this problem is solved.

Connections to the Assembly Data Bank

When generating shell profiles from an input file *connections to the assembly data bank will be lost*.

Note: This restriction only applies to Tribon M2 SP1 or earlier. In Tribon M2 SP2 or later this problem is solved.

10.5.14 Samples files

- Hull curves, seams and stored planes: SeamsCurves.xml
- Shell profiles: Profiles.xml
- Shell stiffeners: Stiffeners.xml
- Shell plates and curved panels: PlatesPanels.xml

These samples files can be viewed by clicking on the following link:

Schema_Curved_Modelling.chm