

# Rule-Based Ship Design

A White Paper

Process, Power & Marine, a division of Intergraph





# Table of Contents

<b>1.</b>	<b>Introduction .....</b>	<b>1</b>
<b>2.</b>	<b>Advantages to Rule-Based Design.....</b>	<b>2</b>
	2.1 Embedding Design Practices .....	2
	2.2 Automating Routine Procedures .....	3
	2.3 Reducing User Requirements.....	4
	2.4 Improving Consistency, Reducing Risk.....	6
	2.5 Ease of Model Modification .....	7
<b>3.</b>	<b>SmartMarine 3D Design Process .....</b>	<b>8</b>
	3.1 Early Design Stage.....	8
	3.2 Detailed Design Stage.....	9
	3.3 Production Design Stage.....	10
<b>4.</b>	<b>The Rules Process .....</b>	<b>11</b>
	4.1 Logical Connection Definition.....	11
	4.2 Assembly Connection Definition.....	12
	4.3 Feature Placement.....	13
	4.4 Parametric Selection .....	13
	4.5 Feeding the Deliverables.....	14
	4.6 Maintenance Over the Life Cycle .....	14
<b>5.</b>	<b>Conclusion.....</b>	<b>16</b>
<b>6.</b>	<b>Appendix .....</b>	<b>17</b>
	6.1 Author's biography .....	17

# 1. Introduction

The ability to drive a ship design with automated rules is critical to the future of computer-aided ship modeling. A rule-based system allows a shipyard to embed knowledge from many sources, resulting in benefits like reduction in modeling time, reduction in errors, and improved consistency.

SmartMarine 3D uses a rule-based approach at each stage of model development, across all disciplines, from early design to manufacturing output. By using the same properties and geometry throughout each stage of the design, the rules can make consistent selections, and can be used to automatically update those selections due to design modifications. To drive the rules system, SmartMarine 3D relies on a series of connections that are created between model objects in the early and detailed design stages. The connections drive user-customizable code, and the code uses model geometry, properties, and user input to make decisions about feature placement and manufacturing output.

As shipyards work to maintain their competitive advantage in the global marketplace, technology differentiators become a key to success. Today's shipyards demand more work from a reduced set of design experts. Because of this demand on the knowledge-base, the ability to drive a design with rules is not just a luxury; it becomes a necessity. If those rules are customizable for any shipyard, processes and knowledge can be built directly into the system, and rule-based decision making can be used in every stage of the design process. The goal of rule-based design is not necessarily to replace the users making the decisions, but to allow those users to accomplish more work with greater accuracy in a shorter period of time. The result is an improved process reflected in improved business results. This paper will highlight the key advantages to rule-based ship design, and will outline the process architecture that SmartMarine 3D has implemented to bring rules customization to the user.

## 2. Advantages to Rule-Based Design

Since shipyards are not high-volume assembly lines like car manufacturers, it is difficult to find processes that can be repeated over and over, thousands of times. For every different ship design, and for every variation to an owner's specifications, there are different scenarios that have to be considered during design and construction. However, most shipyards have clear best practices and pre-defined workflows that can be applied to every design. Capturing this information and turning it into rules can provide many advantages to a shipbuilder, throughout the design process, and from one ship to the next.

If a shipyard can easily customize its design software to capture its decision-making process, it will benefit in five key ways:

- Increased ability to embed standard shipyard practices
- Increased automation of routine procedures
- Reduction in user experience requirements
- Improved design consistency and reduced risk
- Ease of model modification

### 2.1 Embedding Design Practices

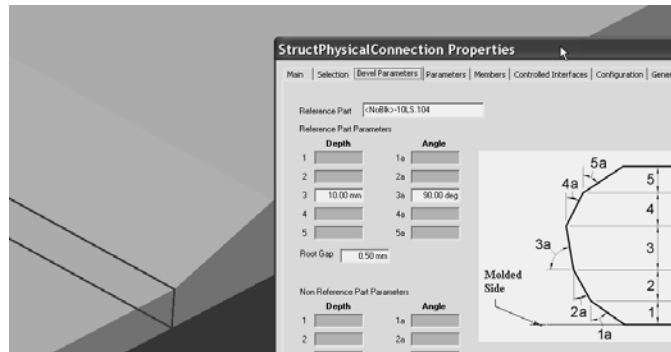
The collective knowledge of a shipyard is proven-in-practice, and constitutes part of an individual company's unique know-how and capabilities. This knowledge is often contained in manuals, or passed from worker to worker. In some cases it is never passed on, and has to be re-discovered by each new generation of employees. In addition to user knowledge, design requirements come from the ever-evolving classification rules. At the next stage of the model design, production requirements define how the model should be built. All of these information sources may have input in the final design, but they don't necessarily work together to get the design right the first time.

One of the key advantages of rule-based design is the ability to capture knowledge from various sources and combine it to drive the design. Gathering and standardizing the shipyard's knowledge results in the ability to see patterns in decision-making. In return, embedding these repetitive patterns can help to automate the design process.

To embed the standard practices, data is built up into a catalog that can be shared across designs, or customized for each design. This is not limited to features that are placed from ship to ship; it also includes rules that can be applied as each feature is placed.

An example of this embedded knowledge is weld data. Weld decisions can be captured in a set of rules that drives the selection of every bevel in the ship. In the most basic form, the weld assignment could be made by the experienced user as part of a manual design process. Once the user has chosen a bevel, the bevel data can be populated based on the thickness of the two parts. Using the manual approach, the user must manually modify the weld to account for any design changes.

A better alternative is for a rule to automate this placement, taking into account the attributes of the two objects being welded together. The rules can use properties like material, thickness, orientation, and location in the ship to make decisions about the type of weld that is needed. Figure 1 shows a plate to plate connection with a thickness difference between the two parts. The rule has computed the difference and automatically applied a chamfer in addition to the bevel. If a modification is made to the geometry or properties of these objects, the rules automatically verify that the selected weld is still valid. If it is not, the rules pick a different weld.



*Figure 1: Result of automatic weld selection*

In the best-case scenario, the rule also checks the location where the two parts will be assembled, and assigns a bevel based on the equipment available in that workshop. Additional production planning inputs can be used to drive the bevel selection and orientation, including the workcenter where the two parts will be welded together, and the upside of the two parts as they are welded.

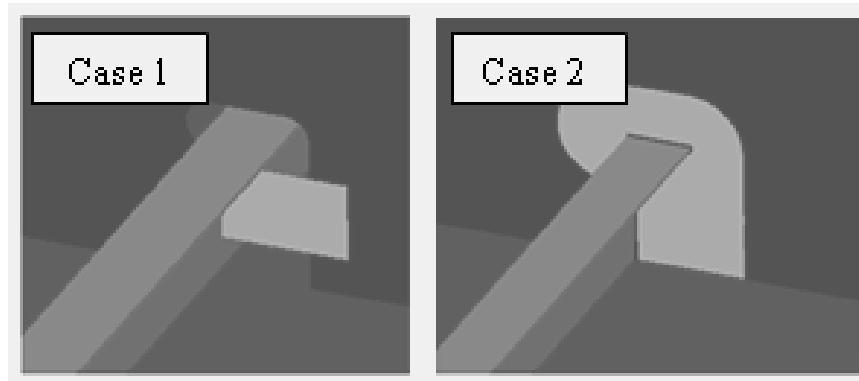
The best-case scenario illustrates that by embedding the user's knowledge, the type and geometry of the parts, and the production specifications, the weld selection can be completely rule-driven. This same concept can be applied to other types of detailed features, resulting in over 90% of some types of features being placed automatically.

## 2.2 Automating Routine Procedures

Once the shipyard knowledge has been embedded into the software, rules can be used to make decisions about routine procedures. Instead of manually applying design details throughout the ship model, rules can be run to place known features automatically.

An example of this automation is the creation of the connection between a profile and a plate. When a stiffener ends at a bulkhead, it will be welded to the bulkhead. When the stiffener penetrates through a bulkhead, a standard slot opening for that cross section type will be cut in the plate, and welds may be applied to connect the plate to the stiffener. Additional structural reinforcement may be needed in the form of a clip. Because these are standard practices for a shipyard, there is no need for any user interaction in this process. A rule-based decision can be made, and all necessary features can be placed automatically.

If the penetrated plate is made watertight, the clip will also need to be watertight. The user does not have to detect this and do something about it; the rule will fire again and make a new decision. Case 1 of Figure 2 shows the profile penetrating through a non-tight plate. Case 2 of Figure 2 shows the same automated connection after the penetrated plate has been made watertight.

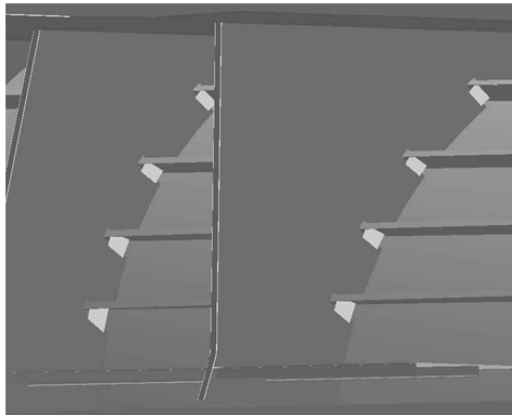


*Figure 2: Non-tight vs. tight collar selection*

## 2.3 Reducing User Requirements

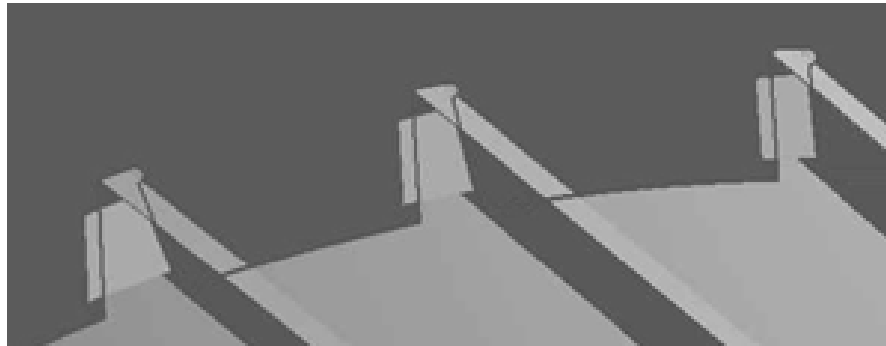
As design standards are embedded into the customized software, the rules can make more and more of the decisions. This takes the burden away from the user, and it reduces the user's required level of design experience. The individual user no longer has to be an expert, and the real design experts can focus on the more complicated design issues.

Figure 3 illustrates how a rule-based decision can alleviate the burden on the user. In this case, shell profiles are penetrating through a transverse bulkhead, and slots will be applied.



*Figure 3: Shell profiles penetrating a bulkhead*

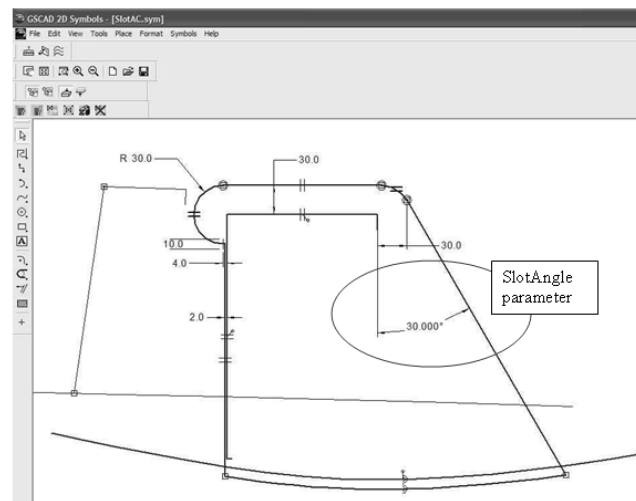
What the design user may not know is how these parts will be assembled. In this case, the production planning user has already determined that the assembly will be oriented based on the shell plates. The shell profiles will be welded to the hull plates first, and then the transverse bulkhead plates will be dropped vertically onto the profiles and the hull plates. The transverse bulkhead plates must fit over the profiles as they are dropped vertically, and so the slots should be opened to provide the necessary clearance. Because each shell profile has a different orientation relative to the assembly's shop floor orientation, each slot must open to a slightly different clearance angle, as shown in Figure 4.



**Figure 4:** Slots opening to different angles

To place the slot features manually, the user would first need to know the assembly orientation and the order of construction. To know the angle for each slot, the user would also have to know the assembly orientation angle, and the angle of each profile relative to the assembly orientation. Finally, the user would have to make the calculation and input the angle value into the slot feature. In the meantime, the production planner may change the orientation of the assembly by a few degrees, and the manually calculated slots will no longer fit over the profiles when the bulkhead plates are dropped.

The solution is to drive this decision process with a rule. A parametric slot symbol is created in 2D, and it allows the slot to be opened to some angle. The 2D symbol is shown in Figure 5.



**Figure 5:** 2D slot symbol

As the slot symbol is placed, a rule runs to find the assembly where the transverse bulkheads plates will meet the shell profiles. If the rule finds that the plates will slide onto the profiles, there will be no need to open the slot, and the angle value will not be calculated. If the rule finds that the plates will be dropped onto the profiles, the rule will calculate the assembly orientation angle. Figure 6 shows the Visual Basic code that the user can define to make these decisions.

In the next step of the process, the rule will compare this assembly angle to the angle of the shell profile, and determine how far the slot should open to allow the plate to be dropped. The calculated angle will be applied to the 2D symbol, which will be used to cut the 3D plate part.

This type of rule can be defined by experienced users, limiting the amount of knowledge required for the typical designer.

```

Public Sub ParameterRuleLogic(pPLM As IJDParameterLogic)
    On Error GoTo ErrorHandler

    'Get the answer to the assembly method question
    Dim sAssyMethod As String
    sAssyMethod = pPLM.SelectorAnswer("SlotRules.SlotRootSel", "AssyMethodInSlot")

    Select Case sAssyMethod
        Case "Slide", "Default value"
            'do nothing to set the angle
            pPLM.Add "SlotAngle", 0.0001

        Case "Drop", "Drop at angle", "Vertical drop"
            'first, get the First Meet assembly
            Dim oAssembly As IUAssembly
            Dim oObject1 As Object
            Dim oObject2 As Object

            Dim oSlot As New StructDetailObjects.Slot
            Set oSlot.object = pPLM.SmartOccurrence
            Set oObject1 = oSlot.Penetrated
            Set oObject2 = oSlot.Penetrating

            Dim pCHelper As New StructDetailObjects.Helper
            Set oAssembly = pCHelper.FirstMeet(oObject1, oObject2)

            Dim oPlanWrapper As New PlanningObjects.FlnAssembly
            Set oPlanWrapper.object = oAssembly

            'then, get the angle
            Dim dAssyAngle As Double
            dAssyAngle = oPlanWrapper.SlotOpenAngle(pPLM.SmartOccurrence)

            If dAssyAngle < 0.0001 And dAssyAngle > -0.0001 Then
                dAssyAngle = 0.00001
            ElseIf dAssyAngle >= 1.57079633 Then
                'the slot angle should be set to zero
                dAssyAngle = 0.00001
            ElseIf dAssyAngle > 0.5235987756 Then
                dAssyAngle = 0.5235987756
            End If

            pPLM.Add "SlotAngle", dAssyAngle
    End Select
End Sub

```

Check assembly construction type

Calculate the assembly angle

Push an angle value to the parametric 2D symbol

I

Figure 6: Slot angle definition code

## 2.4 Improving Consistency, Reducing Risk

The fourth key benefit to rule-based design is the consistency that comes from having a single source making the decisions. When the decisions are made by the rules, it is not necessary to validate every output from the model; the data can be trusted. In areas where manual selection is still required, additional checking rules can be written to verify that the choice made by the user is valid. This improved accuracy results in time savings, as well as reductions in scrap and rework. Most importantly, the increase in reliability means a reduction in risk.

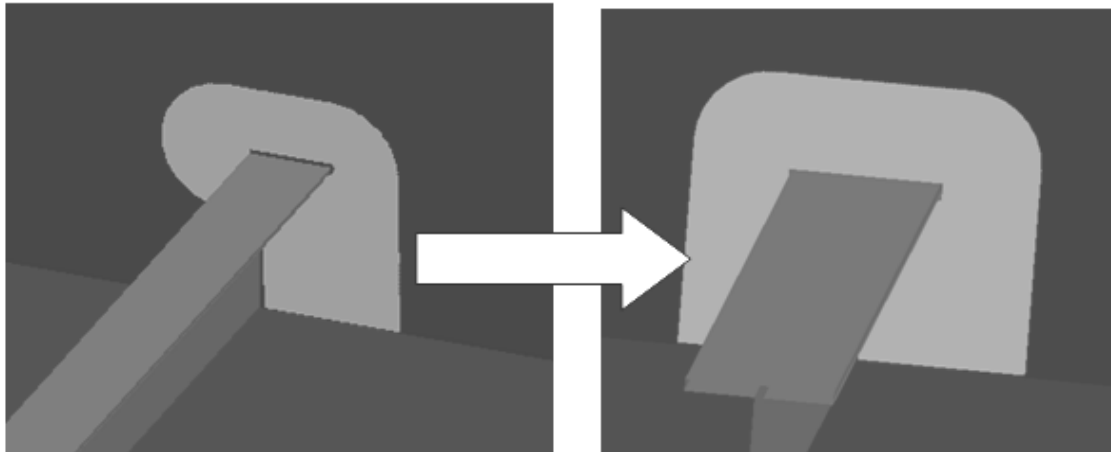
An example of rule-based checking is SmartMarine 3D's Check Manufacturability tool. SmartMarine 3D provides a set of user-definable checking rules, written in Visual Basic code, that validate the combination of the detailing data and the production data. A series of checks can be run against the design in various disciplines, and in different design stages. For example, a check can be run against detailed structural weld data to ensure that an assembly can be constructed in the selected workcenter. Another check can be run against pipe bending to ensure that the design data is valid for the assigned pipe bending machine requirements.

As it becomes more and more necessary to sub-contract work to other shipyards, rule-based design is even more valuable to ensure design consistency. The primary shipyard can develop the rules that should be used throughout the design, and then distribute the rules to sub-contractors.

## 2.5 Ease of Model Modification

If a design is rule-based and driven by the relationships between objects, modification becomes very easy. There is no need for the user to react to design changes; the rules will automatically propagate the effects of modifications. The benefit of automatic updates is less work for the user.

In the case of a watertight slot, a matching watertight collar is selected automatically based on the cross section type. As shown in Figure 7, if the cross section type changes, the rules cause the slot to automatically recompute, and a different collar is selected.



*Figure 7: Collar selection based on cross section*

### 3. SmartMarine 3D Design Process

The SmartMarine 3D design process follows a path from early-stage design to manufacturing output, but the path is not restricted by data being thrown over the wall at the conclusion of each stage. SmartMarine 3D builds in seamless interaction between the design stages. Data that is assigned to a SmartMarine 3D object follows that object throughout its life-cycle, and this concept facilitates modification. Modifications made in each stage are reflected immediately to the other stages. This concept also contributes to a system of rules that can be run at various levels of the design.

#### 3.1 Early Design Stage

The first stage in the design process is the early design stage. The tasks in this stage include definition of surfaces, called systems, which will later be used to drive detailing and manufacturing data. At this stage, plate and profile systems are connected together and split, and properties are assigned. Rules are used in the first design stage to drive the placement of objects such as tripping brackets and stiffeners, and to run services, such as symmetry assignment, that automatically set properties for the plates and stiffeners.

Tripping brackets and stiffeners can be rule-based and parametric. Based on the number of objects selected as boundaries for the tripping bracket, the rule will automatically select the correct type of bracket. The rule is also used to ensure that the parametric tripping bracket will also be sized correctly to the objects that are bounding it. Rules for tripping stiffeners can be customized to define the connection angle, and they can control what type of connection will be applied to the end of the stiffener. For example, the rules decide if the end of the tripping stiffener should be lapped to its boundary, or connected to the flange with a tee weld.

It is difficult to drive the placement of major systems through rule-based automation, and so the user is most involved at the early design level. But, everything they do will contribute to the design automation in later stages.

As the surfaces of the model are being defined, production planning begins. The hull model can be split into blocks, and parts that result from splitting large systems can be assigned to assemblies at this level.

The final component of the early design stage is drawing extraction. The early stage drawings are driven by the systems that have been defined at this level.

## 3.2 Detailed Design Stage

As portions of the early design model are completed, they pass to the detailed design stage. At this point, the first set of completely automated rules is run. These rules rely on the geometry and properties defined in the earlier stage, along with data gathered from planning.

In addition to thickening and trimming the detailed parts, features are placed during detailed design. A feature is any detail that is applied either manually or automatically to the plate or profile parts. Features either drive the final geometry of the part, or hold properties that will be used to drive manufacturing output data. Rule-based features include:

- Slots
- Collars
- Web cuts
- Flange cuts
- Welds
- Corner features
- Edge features
- Brackets

In this stage, less work is required from the user, and more of the work can be done by the rules. Each shipyard can control how much they want to automate. If the shipyard chooses to embed their knowledge and limit manual modification, it is possible to derive 90% or more of the detailed output directly from the rules. In particular, slots and welds can be highly automated.

As the detailed design progresses, data defined in the early design stage may be changing. Because the detailed design re-uses the same model objects, the objects and their relationships to other objects update automatically. At the same time, the rules are re-fired to make any necessary changes to the feature selection.

Production planning continues at this stage as assemblies are defined further. When planning data drives the rules, features are updated automatically to account for changes to the planning data.

Drawing extraction is also continuing throughout the detailed design stage, and the output drawings from this stage will show the detailed parts and their features.

### 3.3 Production Design Stage

The final stage in the process is the production design stage. At the beginning of this stage, the data defined in the detailed design stage passes through another set of rules to define the manufacturing output. The manufacturing rules rely on the model geometry and the properties defined at the detailing level, in addition to user settings.

The user-defined manufacturing rules control how the output will be processed and marked, and in what format the output will be delivered. Like the detailing rules, the rules in this stage provide the user with a set of options. The options have default selections, but the user may override them by choosing from a pre-defined set of answers. Rule-based manufacturing data includes:

- Margin
- Shrinkage
- Templates
- Pin Jigs
- Plate processing and marking
- Profile processing and marking

Like the detailed design stage, this stage relies less on the user. Most decisions can be driven by the rules. As changes are made in the earlier design stages, the manufacturing rules can automatically update the output data.

The rule-driven drawings produced in this stage are the plate manufacturing drawing, template and pin jig drawings, and the profile sketches.

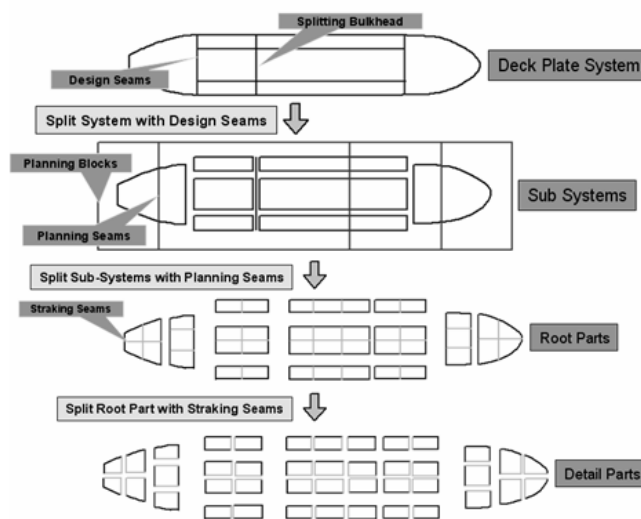
## 4. The Rules Process

SmartMarine 3D offers rule-based decision-making at every step of the design process. Customers can choose to utilize the delivered default rules, or they can customize the rules at any level to meet their needs. Customization can be done over time, allowing each shipyard to prioritize its own high-value automation. Where decisions can't be driven by rules, manual placement can still be used; but to achieve the real productivity potential of the system, automation should be used wherever possible.

SmartMarine 3D rule-based automation is driven by a concept of connections, or relationships, between objects. As objects are modeled, relationships are established and connections automatically created in any case where a boundary or penetration is created. These connections are used to drive the rules, and to control updates when modifications are made.

### 4.1 Logical Connection Definition

To drive the rule-based design, SmartMarine 3D relies on the relationships and connections between objects. Relationships are created at different stages of the design process. Figure 8 shows the progression of connections from early stage design through detailed design.



*Figure 8: Connection definition*

In the early design process, the user creates plate and profile systems to define thickened surfaces in the Molded Forms environment. The next step in the process is to break up the systems by placing design seams to define splits, or by splitting automatically at intersecting systems. Throughout this early design process, production planning can be ongoing, and the major block splits can be established. Additional seams can be defined automatically where the structure crosses block boundaries.

Rules and automation are also used to drive the decisions made at the planning level. Parts are automatically assigned to blocks based on user-definable rules. Parts can be combined to form assemblies, and this process can be driven automatically based on rules defined by the user.

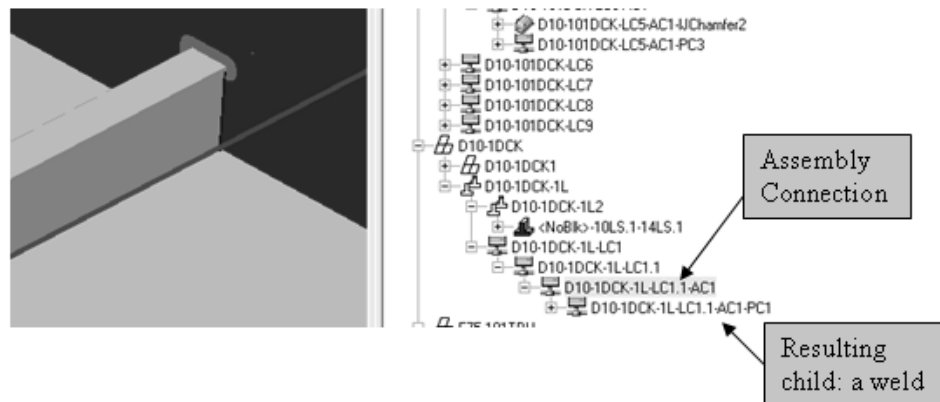
The result of the early stage design is a series of systems that are connected through relationships called logical connections. Logical connections are created as objects in the database. They maintain relationships throughout the design process, and when modifications occur, they trigger updates to the rules.

## 4.2 Assembly Connection Definition

The detailed design begins when the early-stage design goes through the detailing process. The detailing process trims the parts to their final 3D shape, and creates one or more assembly connections for every logical connection. The process started in Figure 8 is continued throughout the detailed design. Further relationships can be established at the detailing stage by splitting the parts with straking splits. In case of a straking split, a new assembly connection is created.

The key to this part of the process is the creation of the assembly connection. Assembly connections are the brains of the entire rule-based modeling process. They analyze the type of relationship between two detailed objects (bounded or penetrating), and start additional rules to decide what child features are needed in each case. Those child features may place additional children of their own.

In the case shown in Figure 9, the logical connection is defining the relationship between the profile and the plate. When the two parts are detailed, the assembly connection is created and its rules determine that the profile should be welded to the plate. The result of this decision is a child, which is the weld.



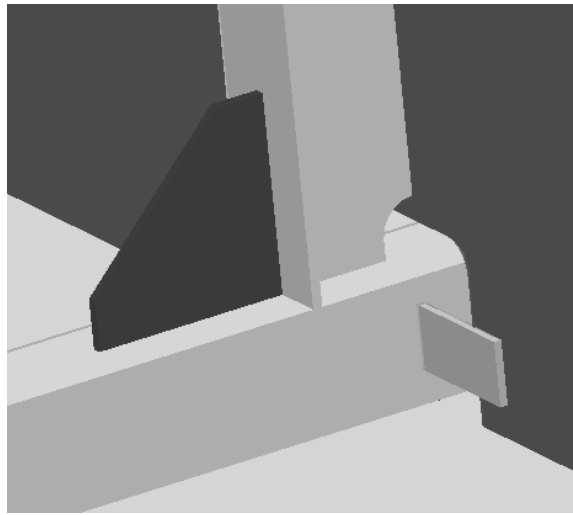
*Figure 9: Assembly connection object*

The decisions made by the rules are not limited to geometry and property settings. Information can also be gathered directly from the user, in the form of questions. A question is a customizable property that can have multiple solutions, depending on the case. These properties provide input to drive the selection that is made by the rule. The question property always has a default answer, and that answer can be calculated based on model properties such that it never needs to be over-ridden. But in some cases, the user needs more control over the selection, and so they can choose from a

pre-defined set of answers. Whenever the user chooses a new question answer, the rule will re-fire to utilize the user's answer.

### 4.3 Feature Placement

The product of the rules selection can be additional objects in the model, like a bracket placed automatically where a profile is bounded by the flange of another profile. The rules selection can also result in a cutting curve that removes material from objects in the model, as in the case of a radius that is placed to avoid a slot. In Figure 10, a single assembly connection has automatically placed a slot, a collar, a bracket, an end connection with a radius, and all necessary welds. If the profile or plates coming together in this assembly connection are modified, the rules will run through the selection process again.



*Figure 10: Assembly connection children*

### 4.4 Parametric Selection

In addition to choosing features, the rules are used to control the size and shape of the feature. A single parametric symbol can be used for any size of corner radius, or different corner radius symbols can be selected based on the geometry case. In the previous example, a corner radius was placed automatically because the vertical profile was bounded to the flange of another profile. The rules also detected the slot, calculated the gap, and placed a long scallop feature that would provide 25 mm clearance above the slot.

In another example, an angle profile that is bounded to the web of another angle profile will get an end connection automatically. If the angle is bounded to web right of the second profile, an end cut will be applied to cut around the flange of the bounding profile, as shown in Case 1 of Figure 11.

If the same profile is applied to web left, the flange is no longer in the way, and an end cut with flange clearance is no longer needed. If the rule finds that the bounded profile is shorter than the bounding profile, a straight welded cut is applied. If the profiles are the same height, or if the bounded profile is taller, a snipe cut is applied, as shown in Case 2 of Figure 11.



*Figure 11: Automated end cut selection*

The key to the rule-based decision making is that the user does not have to make any of these decisions, either during creation or modification.

## 4.5 Feeding the Deliverables

Once the detailed design is complete, SmartMarine 3D provides output in the form of drawings and manufacturing data. These deliverables are also rule-based to provide automation capabilities to the customer. Drawing resymbolization is based on a series of object filters and logic. The user can customize the drawing resymbolization logic to choose the output that meets the shipyard's standards.

Manufacturing data output is driven by user-defined visual basic rules. As each manufacturing plate or profile is extracted from the detailed part, rules control how the object is processed and marked. Figure 12 shows the process settings for a flat plate part. The plate's upside is taken directly from the rule. A different set of options controls the marking for the plate.

The manufactured part relies heavily on the result of the rules that were run at the detailing level. Bevel data is taken directly from the physical connections, which were created automatically as children of the assembly connections. If a design modification forces a change to the bevel data, the manufacturing part is notified that a change was made, and it is marked as out-of-date. The user can approve the update to the manufacturing data, and send the update to the nesting system.

## 4.6 Maintenance Over the Life Cycle

After the ship leaves the yard, there is still potential for a life-cycle's worth of modifications to the 3D CAD model. Activities such as maintenance, repair, and overhaul can be planned using the model, and a rule-based associative system can assist in this process. A routine modification can result in many affected systems; but the SmartMarine 3D rules can automatically run at any time in the design process, analyzing the result of the change, and suggesting where the design should be updated to account for the change.

As an example, extra structural support may be needed for a deck that has received an upgraded (and heavier) piece of equipment. Using a manual modification approach, the user would have to calculate the effect of the heavier equipment, decide that a different foundation was needed, and make the change to the model. Using the rule-based approach, the equipment could be switched out in the model, and the foundation would automatically be adjusted to account for the added weight.

### Struct Mfg Plate Part Properties

Main **Process** Marking Relationship Configuration General Notes

User Answers

Results --> FlatPlateProcess Valid

Parameter	Value	Rule-Based	Catalog Value ▲
PlateNeutralAxis	Calculated	<input checked="" type="checkbox"/>	Fixec
PlateUpside	MoldedSide	<input checked="" type="checkbox"/>	semblyOrientatior
PlateUnwrapAlgo	Flat	<input checked="" type="checkbox"/>	
PlateWeldTab	Ignore	<input checked="" type="checkbox"/>	Apply
PlateFeatureTab	Ignore	<input checked="" type="checkbox"/>	Apply
PlateBevel	Fixed	<input checked="" type="checkbox"/>	
PlateBridge	Ignore		
PlateBaseLine	Ignore	<input checked="" type="checkbox"/>	Butt
PlateWLUncfold	Ignore	<input checked="" type="checkbox"/>	
PlateSurface	Ignore	<input checked="" type="checkbox"/>	
PlateFeature	Ignore		
PlateRollLines	Ignore		
PlateFSFORE	Ignore	<input checked="" type="checkbox"/>	

Figure 12: Manufacturing process rules

## 5. Conclusion

Rule-based design is a necessity for shipyards that intend to lead the industry. In addition to embedding the shipyard's knowledge and reducing mistakes and rework, rules can speed up the design process, enabling the shipyard to accomplish more in less time. The result goes beyond accuracy and time-savings; the business benefits of reduced risk and increased competitive advantage go hand in hand with these engineering benefits. SmartMarine 3D has a built-in rule-based capability that can be customized minimally or extensively to match each shipyard's workflow, or to drive a new and improved process workflow. The SmartMarine 3D rule-based system is in production in some of the most demanding shipbuilding processes in the world, proving that the SmartMarine 3D architecture is setting the standard for design automation.

## 6. Appendix

### 6.1 Author's biography

**Kristin Cochran** holds the current position of support consultant at Intergraph Corporation. She is responsible for user customization and reference data support in the areas of early-stage design, detailed design, planning, and drawings. Her background includes a B.S.E. in Naval Architecture and Marine Engineering from The University of Michigan, an M.S. in Ocean Engineering from Virginia Tech, and 10 years working with Intergraph's rule-based ship design tools.

**Headquarters**  
**Intergraph Corporation**  
170 Graphics Drive  
Madison, AL 35758

For more information about Intergraph,  
visit our Web site at [www.intergraph.com](http://www.intergraph.com).

Intergraph and the Intergraph logo are registered trademarks and SmartMarine is a trademark of Intergraph Corporation. Windows is a registered trademark of Microsoft Corporation. Other brands and product names are trademarks of their respective owners. Intergraph believes that the information in this publication is accurate as of its publication date. Such information is subject to change without notice. Intergraph is not responsible for inadvertent errors. ©2007 Intergraph Corporation, Huntsville, AL 35824-0001. All Rights Reserved.

11/07 **PPM110A0**

