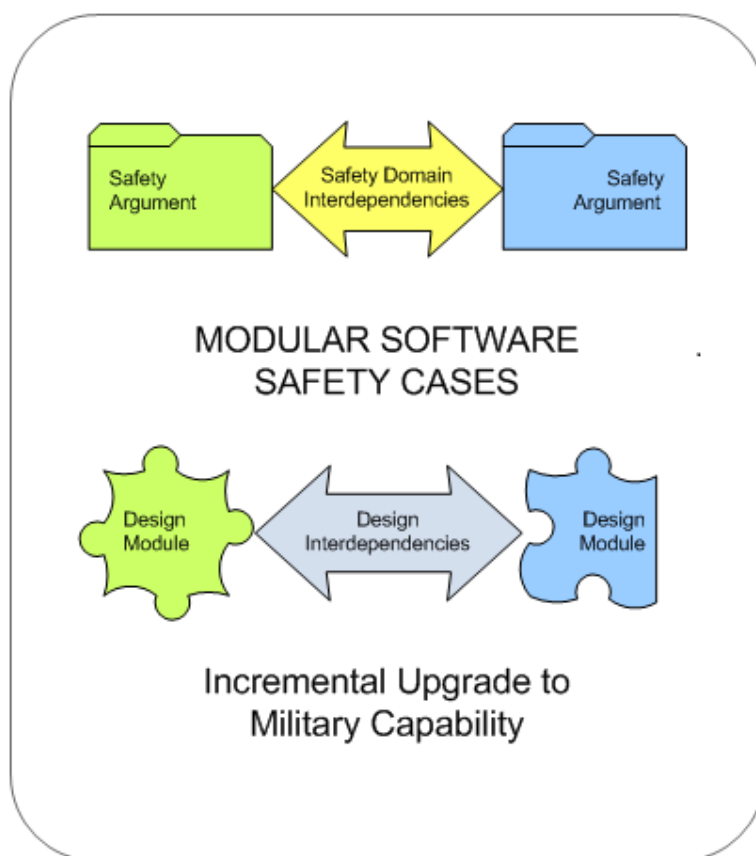


# Modular Software Safety Case Process GSN

Date: 19-Nov-2012



Copyright 2012 © AgustaWestland Limited, BAE SYSTEMS, GE Aviation, General Dynamics United Kingdom Limited, and SELEX Galileo Ltd. All rights reserved.

# Contents

Contents .....	ii
<b>1 Introduction.....</b>	<b>1</b>
1.1 Purpose .....	1
1.2 Scope .....	1
1.3 Document Overview.....	1
1.4 References .....	1
<b>2 Goal Structuring Notation .....</b>	<b>2</b>
<b>3 GSN Extensions for Modular Safety Cases.....</b>	<b>4</b>
3.1 Modular Extensions Defined by the GSN Standard.....	4
3.1.1 Symbols.....	4
3.1.2 Patterns .....	5
3.2 Modular Extensions Not in the Standard .....	6
3.2.1 Modular GSN Extensions Previously Proposed .....	6
3.2.2 Further Modular GSN Extensions Under Consideration .....	9

# 1 Introduction

## 1.1 Purpose

This document gives a summary of the modular extensions to Goal Structuring Notation (GSN). It also indicates how they are used to form a safety case architecture within the Modular Software Safety Case (MSSC) process. The current definition of GSN is given in ref [1].

## 1.2 Scope

The document is intended to allow a modular GSN diagram to be interpreted. It should be read in conjunction with [1], which provides an appendix which specifically discusses notation interpretation.

## 1.3 Document Overview

This document gives a summary of GSN elements then proceeds to explain each element in more detail.

## 1.4 References

- [1] GSN Community Standard: ( <http://www.goalstructuringnotation.info/>)  
Version 1; November 2011, (c) 2011 Origin Consulting (York) Limited  
PDF : [www.goalstructuringnotation.info/documents/GSN\\_Standard.pdf](http://www.goalstructuringnotation.info/documents/GSN_Standard.pdf)
- [2] *“Safety Case Composition Using Contracts -Refinements based on Feedback from an Industrial Case Study” - J Fenn, R Hawkins, T Kelly, P Williams - Proceedings of 15th Safety Critical Systems Symposium (SSS'07), February 2007 (Proceedings published by Springer)*

## 2 Goal Structuring Notation

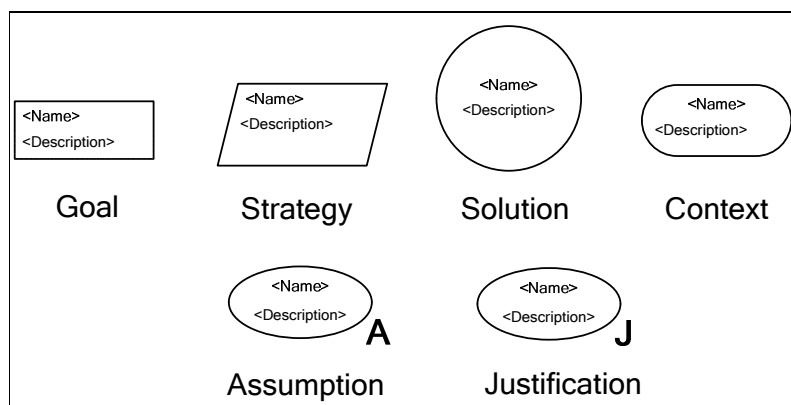
GSN is a graphical argumentation notation which can be used to explicitly document the elements of any argument (claims, evidence and context) and - perhaps more significantly - the relationships that exist between them (i.e. how claims (conclusions) are supported by other claims (premises), how claims are supported by evidence, and the assumed context which is defined for the argument).

GSN was developed at the University of York in the early 1990s.

The required safety properties of the system are expressed as goals, where the aim is to show that each of these Goals is met by the product. To demonstrate that a Goal is met, it is necessary to either provide direct evidence that the statement in the Goal is true or to decompose the Goal into sub-Goals, which embody statements that aggregate to the equivalent of the parent Goal. The argument is complete only when all of the Goals are populated with evidence.

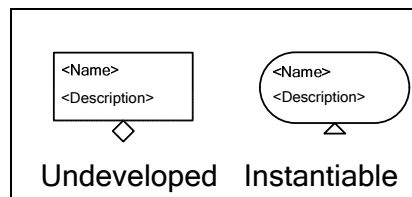
GSN also allows supporting information such as assumptions, constraints, justifications and context to be represented.

Examples of the GSN elements for Goals, Strategies, Solutions, Constraints, Justifications and Assumptions are shown in Figure 2-1.



**Figure 2-1: GSN Elements**

Modifiers may be applied to diagram elements as shown in Figure 2-2. A diamond identifies an element requiring further decomposition ('undeveloped entity'), and a triangle identifies elements that require instantiation.

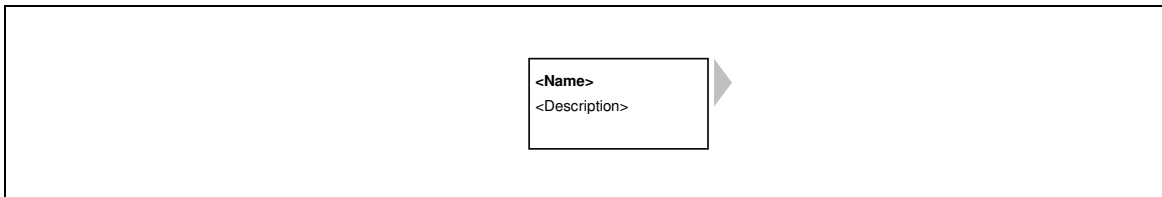


**Figure 2-2: GSN Element Modifiers**

The MSSC Process description assumes that users are familiar with the core concepts laid out in the GSN standard [1].

In practice, tools that support GSN may introduce additional notation elements beyond those in the standard. For instance the modifier symbol used in the Casemaker tool to identify

elements common to several diagrams, or used to implement off-page connections, is a grey triangle as shown in Figure 2-3.



**Figure 2-3: Example of Other Notation**

## 3 GSN Extensions for Modular Safety Cases

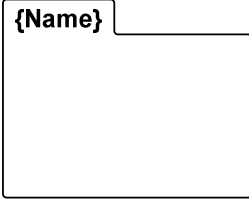
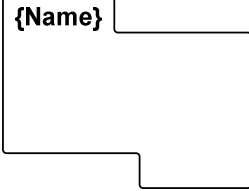

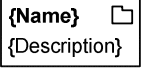
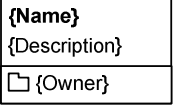
### 3.1 Modular Extensions Defined by the GSN Standard

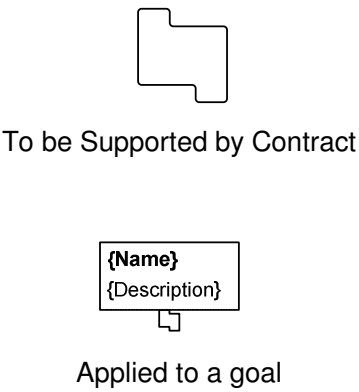
The following modular extensions are defined by the GSN Standard [1]. For a description of GSN symbols and their use please refer to the standard.

#### 3.1.1 Symbols

To assist readers some selected descriptions from the GSN standard's modular appendix are reproduced here alongside the same symbols. These symbols have been necessarily redrawn and include some minor stylistic differences.

**Table 1: New Entities added to the Core GSN Notation to support Modularity**

<p><b>Module</b> and <b>Contract Module</b> symbols are used in Module view to represent the referenced module of argument without displaying the content of the argument. The arguments represented by these symbols are not necessarily documented using GSN.</p>	 
<p><b>Public Indicator</b>, rendered as a miniature <i>module</i> symbol and superimposed on a <i>goal</i>, <i>solution</i> or <i>context</i> symbol at the top right. This indicates that the element is publicly visible to other <i>modules</i>, and can be referenced as an <i>away goal</i>, <i>away solution</i> or <i>away context</i>.</p>	 <p>Public Indicator Symbol</p>  <p>Example of use (goal)</p>
<p>An <b>away goal</b>, rendered as a rectangle with a bisecting line in the lower half of the rectangle. The area in the lower portion contains a miniature shaded <i>module</i> symbol. This repeats a claim presented in another argument <i>module</i> which is used to support the argument in the local <i>module</i>. The Module Identifier provides a reference to the module that presents the original claim.</p>	

<p><b><i>To be supported by contract:</i></b> This annotation, attached centrally immediately below the <i>goal</i> to which it relates, denotes that support for the claim presented by the attached <i>goal</i> is intended to be provided from an argument in another <i>module</i>, linked by an as-yet-undisclosed <i>contract</i>.</p> <p>At some later stage, the element may be updated to replace this annotation with support from a named <i>contract module</i>, or may be left as it is, with the necessary support defined in a higher-level argument abstraction.</p> <p>This annotation can only be applied to <i>goal</i> elements, and can be used in conjunction with the '<i>To be instantiated</i>' annotation, but is mutually exclusive with the '<i>To be developed</i>' annotation.</p>	 <p>To be Supported by Contract</p> <p>{Name} {Description}</p> <p>Applied to a goal</p>
--	---

Guidance on the use of these symbols is also provided by the standard.

### 3.1.2 Patterns

In order to represent patterns of argument rather than merely argument instances, GSN has been extended to support structural and entity abstraction in Annex A1 of the GSN Standard.

The extensions to core GSN presented in Annex A1 are intended for the representation of abstract argument patterns. In cases where the elements defined in these sections are used in the development of instantiations of the patterns to produce individual assurance arguments, it is important to ensure that they are all removed, or instantiated, in the final, delivered, version of the argument.

In Annex A1 of the GSN standard curly brackets are used to denote instantiable placeholders, thus {placeholder}.

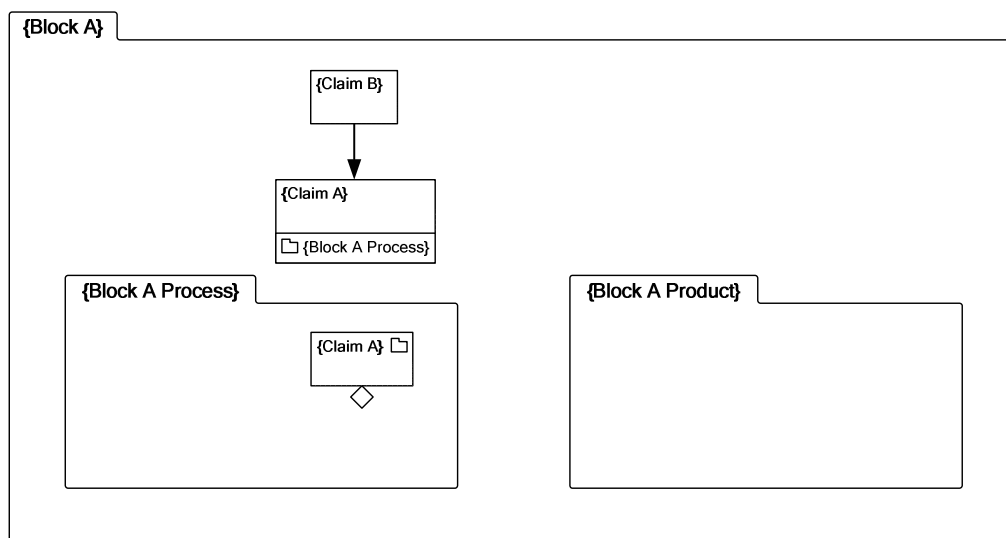
## 3.2 Modular Extensions Not in the Standard

This section contains modular extensions that are not in the GSN standard, along with some rationale for them.

### 3.2.1 Modular GSN Extensions Previously Proposed

The following extensions and symbols have been proposed through public dissemination or represent combinations of modular and 'pattern language' extensions to GSN which is not explicitly offered within the standard, but constitutes typical usage of pattern language with the modular notation. As such, IAWG believe these are 'low risk' for adopters of MSSC to use whilst they are considered for inclusion in a future version of the GSN standard.

#### 3.2.1.1 Containment



**Figure 3-1: Illustrating Containment**

Containment is a mechanism by which the visibility of claims, context and solutions in GSN may be restricted. This is illustrated in Figure 3-1, in which the scope of *Goal: {Claim A}* is limited to *SC Module: {Block A}*.

Containment has previously been identified as useful for presenting a clearer top level architecture view of the system.

Within MSSC it has been employed for this, for example;

- Separating a process argument from the product argument it supports within a safety case module for a component of a system, whilst hiding that process argument from other SC modules.
- To clarify which Public Claims, Context and Solutions from a set of supplied SC modules are those that are intended to be used outside the set.

It has also been used to;

- Hide the interactions between SC Modules and any components they cover to facilitate the supplier ownership of Intellectual Property Rights for their product.

The basic principles of containment are that every SC module which is created must have a containing SC module declared for it, and can only have **one** containing SC module. The



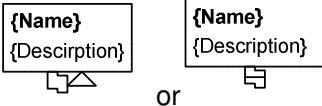

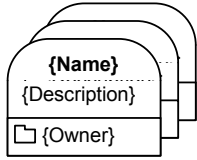
containing SC module defines the *scope* of the SC module (only SC modules declared to have the same containing SC module are of the same scope). Public Claims, Context or Solutions in a SC module are not available to be used from outside of the containing SC module (i.e. only available to SC modules of the same scope). This means that, for example, a SC module reference cannot be made to a SC module which has different scope from the referencing SC module. Similarly, an away goal reference cannot be made to a goal provided by a SC module of different scope.

By default, or where no other containing SC module is declared, the containing SC module is the *global* SC module, which is an abstract SC module containing all possible SC modules. It is not possible for another SC module to contain the global SC module. In practice, it is generally preferable that instead of using the global SC module, a top-level containing SC module be created which defines the scope of the entire safety case.

The other rules for the use of SC modules are not affected by the introduction of the containment rules described above.

### 3.2.1.2 Symbols

**Table 2: Modular GSN Extensions Previously Proposed by IAWG**

<p>An uninstantiated goal that will be supported through a safety case contract. A combination of <i>instantiable</i> and <i>solved by contract</i> ornaments requires licence be taken because they overlap.</p>	
<p>An uninstantiated safety case module. (The use of the uninstantiated link ornament on SC Modules is not ruled out in the GSN Standard and its guidance, nor is there any explicit mention of it)</p>	
<p>Collective context which is inherited by goal in the safety case contract.</p> <p>This is a notational/presentational convenience, a substitute for having a large number of context bubbles on a diagram which is semantically the same.</p> <p>Symbol defined and usage proposed in Ref [2].</p>	

### 3.2.1.3 Patterns and Templates

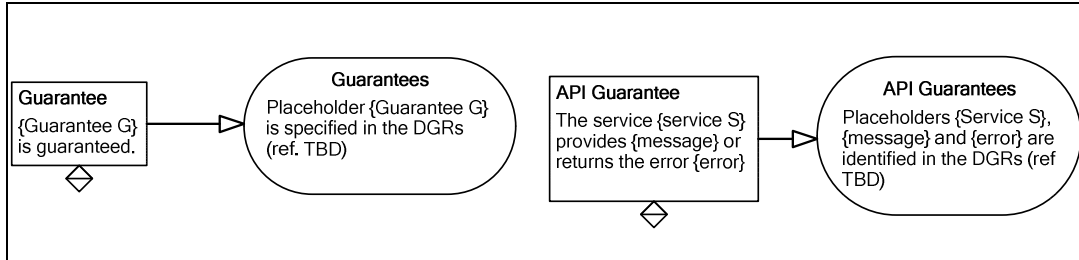
In MSSC modular GSN distinction is made between patterns and templates.

- **Patterns:** provide guidance for the GSN author and must be tailored by them, they are not present in the final argument. MSSC process provides guidance in the form of GSN patterns.

- **Templates:** contain symbols or text that require instantiation, and must be accompanied in the final argument by that instantiation data. Templates allow a concise representation of many similar arguments to be presented.

Instantiable text in templates should be identified by placing it in ornate brackets, and instantiation data for this should be furnished in context.

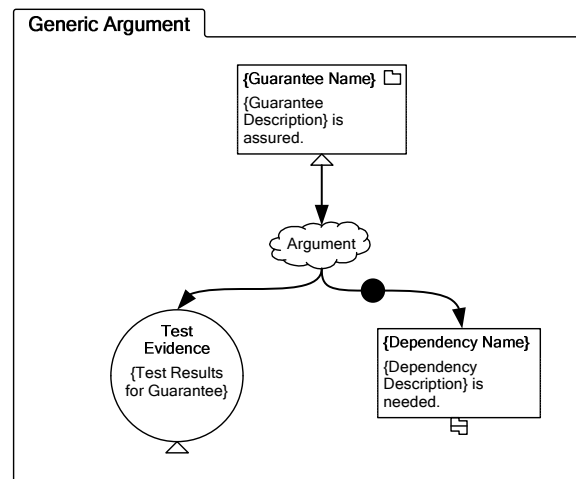
Figure 3-2 shows two slightly different approaches to specifying instantiable placeholders in an argument template, one in which a single placeholder is used in a claim, and one in which there are several.



**Figure 3-2: Use of Placeholders in Template Instantiation**

To correctly derive an instance of an argument all of the literal values of the instantiable placeholders in the generic argument must be identified collectively, not individually. This is illustrated in Figure 3-3 in which a goal named *Speed* is supported by **two** dependencies, *Distance* and *Time* and some test evidence.

In a second instance of the argument, *Area* is provided, and crucially this claim does not depend on either *Distance* or on *Time*.



#### Corresponding Instantiation Data

	{Guarantee Name}	{Guarantee Description}	{Dependency Name}	{Dependency Description}	{Test Results for Guarantee}
1	Speed	Output of Speed is assured.	Distance	Input of Distance is...	Test Spec 5.1
			Time	Input of Time is...	Test Spec 5.2
2	Area		Length		
			Width		

**Figure 3-3: Example of Template Instantiation**

### 3.2.2 Further Modular GSN Extensions Under Consideration

The following symbols are currently being considered and trialled by IAWG for use as part of MSSC. They should be considered and are offered for readers' consideration as 'work in progress'. As such, they would not necessarily be recognised by the broader GSN community and have not currently been formally submitted for approval to the GSN Standardisation Committee. Whilst they may offer useful alternatives for representing certain aspects of modular safety arguments, adopters should also consider whether their technical authorities or independent safety assessors are likely to look favourably on their usage until such time as they might be adopted into the GSN standard.

Users of the MSSC process should clearly state the GSN definition used; whether it is by reference to an external standard such as [1], or another scheme such as the one presented here. This assures a common understanding between safety case author and reader.

#### 3.2.2.1 Away Goal Scheme B: "Symmetric"

An alternative symbology for away goals is needed but remains to be determined.

- Scheme A: "Original"  
Is defined by the contents of sections 3.1 and 3.2.1.
- Scheme B: "Symmetric"  
Is the same as scheme A but with the following some symbols overridden (as described in the Table 4 below).

Scheme B is needed for two reasons

1) to allow an integration to provide support to its Block SC Modules using argument other than following the strategy within a SC Contract Module. In MSSC this need is exemplified by the integration argument supporting the Blocks *Goal: Not Prevented*, but there are likely to be other situations.

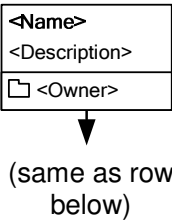
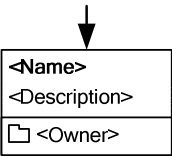
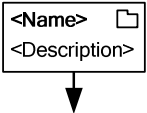
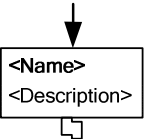
2) To allow a Block SC Module to express a need for support without specifying to the integrator how they must provide it. In Scheme A a Block must always tell the integrator to use a SC Contract to support its needs.

For goals which are in a safety case module public interface Scheme B will provide symbols that can distinguish in a consistent way between any combination of the following three attributes;

**Table 3: A consistent Approach to Attributes of Goals in Scheme B**

<b>Scheme B Attribute Name</b>	<b>Scheme B Attribute Question</b>	<b>Scheme B Alteration to Goal Symbol</b>
In Owner	Is the goal drawn in its owning safety case module or an away reference?	Symbol alteration approach is not yet determined
Provider or Consumer	Does the goal provide support to other safety case modules, or does it require support from them?	Symbol alteration approach is not yet determined
Integration Advice	Is the goal intended to be used in a safety case contract when integrated?	Symbol alteration approach is not yet determined

**Table 4: Override Symbols Needed by Away Goal Scheme B: “Symmetric”**

Scheme A Symbol	Description	Scheme B Symbol
 <p>(same as row below)</p>	<ul style="list-style-type: none"> <li>Away Reference (drawn outside owner)</li> <li>support is provided to the other safety case module (to the owning safety case module)</li> </ul> <p>A goal that is an away reference to one that the owning safety case module requires support for. This is depicted outside that owner, optionally instantiable</p>	Symbol is not yet determined
<p>(same as row above)</p> 	<ul style="list-style-type: none"> <li>Away Reference (drawn outside owner)</li> <li>support is required from the other safety case module (from the owning safety case module)</li> </ul> <p>A goal that is an away reference to one that the owning safety case module supports. This is depicted outside its owning safety case module, optionally instantiable.</p>	NO CHANGE
	<ul style="list-style-type: none"> <li>In Owing safety case module</li> <li>support is provided to other safety case modules</li> </ul> <p>As drawn in owning safety case module, Public goal providing support (supported in owner), optionally instantiable.</p>	NO CHANGE
	<ul style="list-style-type: none"> <li>In Owing safety case module</li> <li>support is required from other safety case modules</li> </ul> <p>As drawn in owning safety case module Public goal requiring support (not supported in owner), optionally instantiable.</p> <p>The need for the integrator to support this with a safety case contract is not specified by the new symbol.</p>	Symbol is not yet determined

In summary Scheme B will have the following benefits;

- allows visual distinction to be made between goals providing and requiring support.
- prevents inappropriate specification of “intended to be integrated by contract”. ‘Scheme B’ was developed in consideration that it is inappropriate for a safety case module developer to specify how a safety case integrator will perform integration, i.e. using a safety case contract as in “solved by contract”.
- should also avoid the problem of overlapping ornaments, i.e. scheme A’s *Instantiable* and *solved by contract*. does not occur under scheme B..

...and the following disadvantages;

- a) The title of the goal is not always at the top of the symbol.

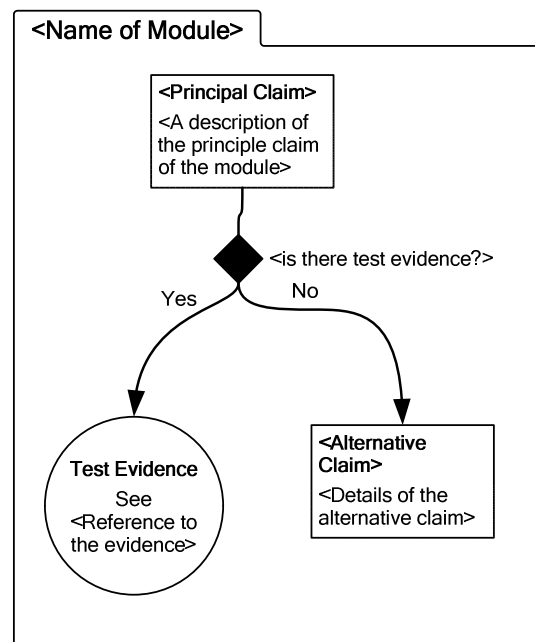
To try to ensure that the SC is as easily navigable as possible when read top down it is suggested that, just as in scheme A the development of away goals was constrained to Safety Case Contract Modules, in Scheme B as applied to MSSC the development of away goals may be constrained to Integration Safety Case Modules.

### 3.2.2.2 Patterns and Templates

Structural abstraction as described in the standard should be clarified by text associated with the relevant structural symbols on the GSN diagram describing the abstraction or choice to be made *and* this text should contain an abstract placeholder that is instantiable.

Either patterns or templates need to be altered to meet the objective of distinguishing them, and this unfortunately leads to a deviation from the GSN standard.

Since pattern arguments must not exist in the final SC, but template arguments do, the continued use of curly brackets for templates will allow the appearance of the final SC to remain as per the existing GSN standard. The objective is to distinguish pattern placeholders, however, and so this proposal is that they should change to using angled brackets, see Figure 3-4 for an example.




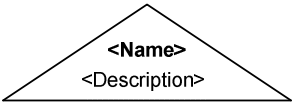
**Figure 3-4: Example use of Angled Brackets in a Pattern**

Some patterns are patterns for templates, and contain text in both angled brackets and ornate brackets {<thus>}; or even <{thus}> if the pattern, when used, may or may not result in a template.

The instantiable symbol ornament (a triangle at the bottom of the main symbol) must be used for template instantiable entities. A pattern, particularly patterns for templates are more clearly presented if the instantiable symbol ornament is not used below pattern instantiable entities, but only below template instantiable entities.

### 3.2.2.3 Instantiation Table(s)

When outside an enclosing document it may be considered insufficiently clear by some readers that the GSN as a whole is a template argument, or which of the perhaps many GSN Context symbols contain or refer to the instantiation data for the argument.

Existing Symbol	Description	Additional Symbol
	Instantiation Table(s) for a GSN Template	

**Figure 3-5: Instantiation Tables**

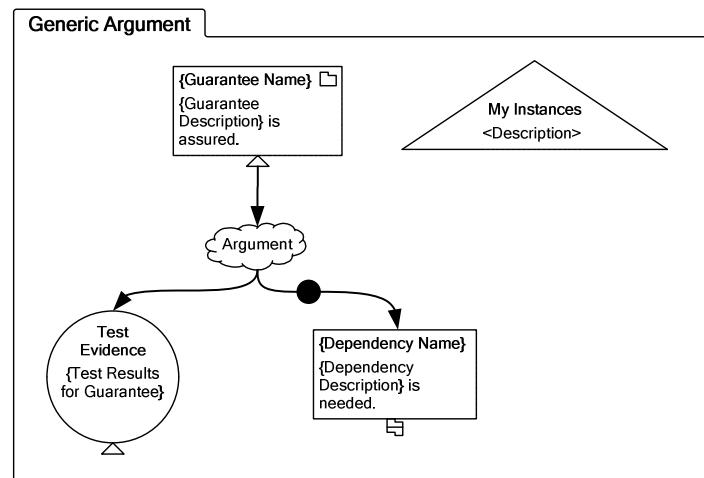
Instantiation data describes the system and may also be presented in GSN context symbols or the artefacts they refer to.

Since the instantiation table has a table-based or textual presentation and the GSN is diagrammatic and symbolic, they are often necessarily treated separately during safety case developments.

The instantiation data, however, is meta-data that describes the form of the argument as well as its content, and is in this respect fundamentally different from contextual information about the system (or other subject of the argument). It captures the valid combinations between different possibly values of the template placeholders. There being only one such GSN symbol per template encourages this.

When readers see an instantiation table it is clear that the GSN is a GSN template, even if the instantiation table is not yet visible (see Figure 3-6); and this helps clarify the difference between patterns, templates and patterns for templates.

The presence of the symbol in a pattern argument also clarifies that it is intended to be tailored for presentation of a generic instantiable template argument in the final safety case.



**Figure 3-6: Instantiation Table Symbol in Use**

In most practical cases a single table “in the narrative for” or referenced by the Instantiation Table symbol will be sufficient to represent the instantiation of the argument.

### 3.2.2.4 Solved by Contract Link Ornament

In order to concisely represent the safety case architecture diagram, and as done in the example safety case in the appendix of 0201, safety case contract modules have not been drawn out in full, but rather depicted using a link ornament as shown in Figure 3-7.

The safety case contract module should be depicted in full elsewhere.

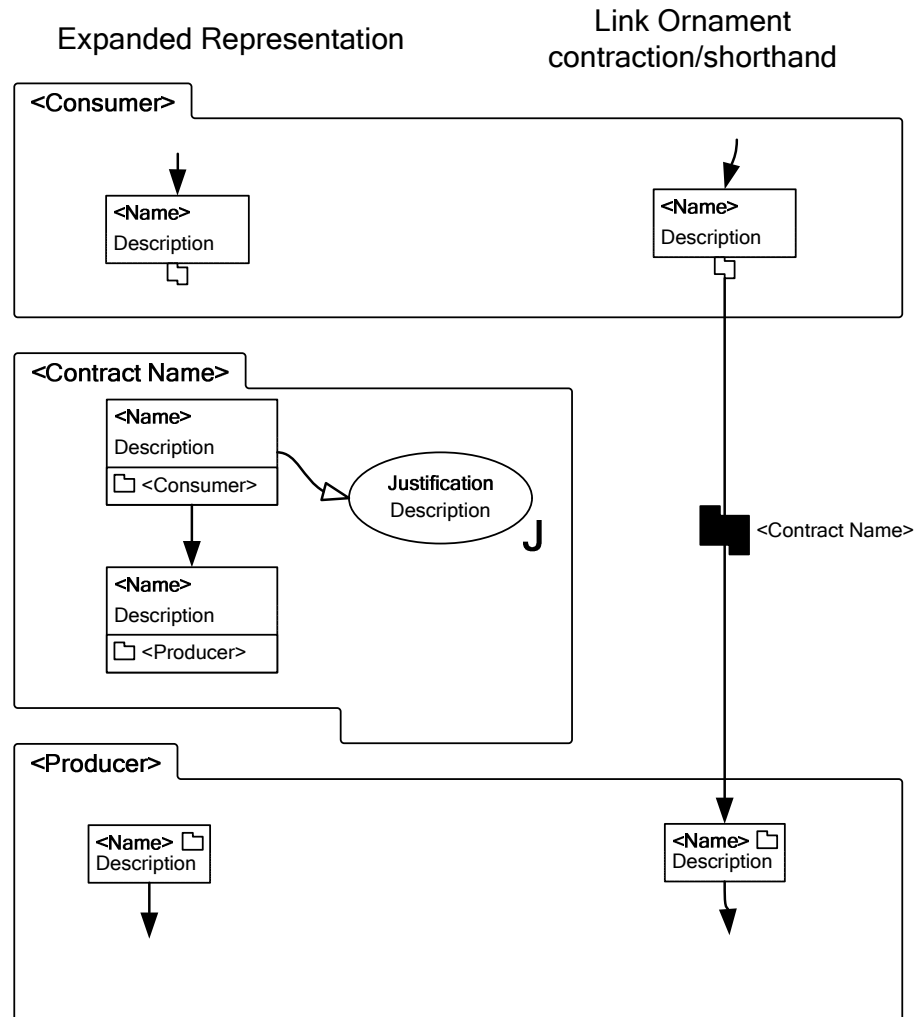


Figure 3-7: A fully expanded SC contract alongside the SC contract link ornament