3.5. Software Contribution Safety Argument Pattern with Grouping

Software Contribution Safety Argument Pattern with Grouping			
Author	Richard Hawkins		
Created	07/12/10	**Last modified**	07/12/10

INTENT

This pattern is an extension of the Software Contribution Safety Argument Pattern. It provides the option of grouping the argument to reflect natural requirements groupings in the software design. For example, for an instantiation of the Software Contribution Safety Argument Pattern at the software architecture level, it may be desirable to create groupings in the argument which reflect each of the individual architectural design elements.

MOTIVATION

Grouping aspects of the Software Contribution Safety Argument Pattern can help to manage the safety argument where there exist a large number of claims at a particular tier of decomposition by splitting the argument into manageable chunks.

STRUCTURE

The structure of this argument pattern is shown in Figure 7 below.

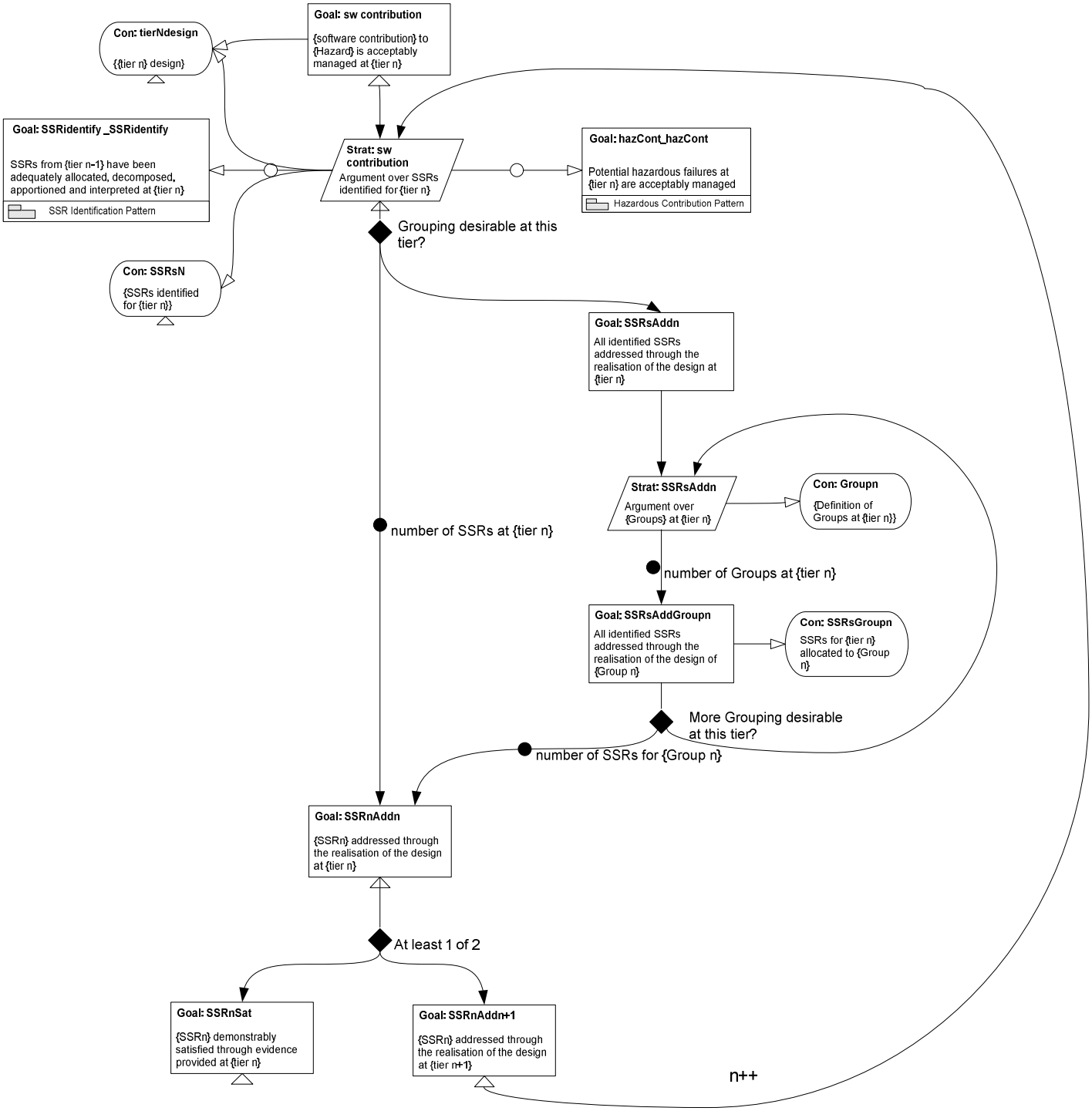

Figure 7 – Software Contribution Safety Argument Pattern with Grouping Structure

PARTICIPANTS

In this section the participants of the Software Contribution Safety Argument Pattern are not restated. These participants are documented fully in the Software Contribution Safety Argument Pattern. Just the participants of the grouping addition are described here.

Goal: SSRsAddn

An instance of this goal is created if grouping is desirable at the tier of instantiation. This goal claims that all the SSRs that have been identified for the current tier of design have been realised through the design.

Strat: SSRsAddn

The strategy adopted is to provide an argument over a number of groups of design elements at {tier n}

Con: Groupn

This context defines what the groups are over which the argument will be structured.

Goal: SSRsAddGroupn

An instance of this goal is created for each group over which the argument will be made at {tier n}. {Group n} should be instantiated with the name of the design grouping which is being considered.

Con: SSRsGroupn

This context defines which of the SSRs that were identified for {tier n} have been allocated to {Group n}. All SSRs must be allocated to a group.

Goal: SSRnAddn

An instance of this goal is created for each SSR allocated to {Group n} (represented as SSRn). There is an option for how this goal is supported. It can be supported by either, or both of goals 'SSRnSat' and 'SSRnAddn+1'. It may be necessary to justify such a decision by providing an argument. The Argument justification software safety argument pattern may be used to provide such an argument.

APPLICABILITY

This pattern should be applied whenever it is desirable to group the structure of the argument at a particular tier to reflect natural requirements groupings in the software design.

IMPLEMENTATION

The key implementation decision is when to create groupings in the argument. The option to argue over a group of SSRs could be implemented for any large components in the software design (e.g logical partitions) in order to split the argument into manageable chunks. There is however no obligation to create any groupings. It should be noted that arguing over a group of SSRs is simply an organising principle to make the argument more easily managed and the decision to group (or not) will *not* affect the argument that is ultimately provided as to how those SSRs have been satisfied.

POSSIBLE PITFALLS

The option of grouping the argument as defined in this pattern should only be used to group together SSRs at an existing level of decomposition. It should not be used when decomposing a design, or deriving additional SSRs. In this case the 'normal' decomposition option in the pattern (as defined in the Software contribution safety argument pattern) should always be used.

RELATED PATTERNS

This pattern extends the Software contribution safety argument pattern.

4. Conclusions

This document has presented a catalogue of software safety argument patterns. The catalogue contains a number of patterns which may be used together in order to construct a compelling software safety argument for the system under consideration.

The software safety argument patterns describe the nature of the argument and safety claims that would be expected for *any* software safety case. The way the argument is supported may be different for each system but the 'core elements' of the argument (as defined by the patterns) remain.

The effectiveness of the software safety argument patterns has been demonstrated through application to a number of case studies. The pattern catalogue is also included as part of the SSEI standard of best practice for software in the context of defence standard 00-56.

The authors actively seek feedback from users of the patterns presented in this document, and will update the contents of the catalogue, where required, based on user experiences.