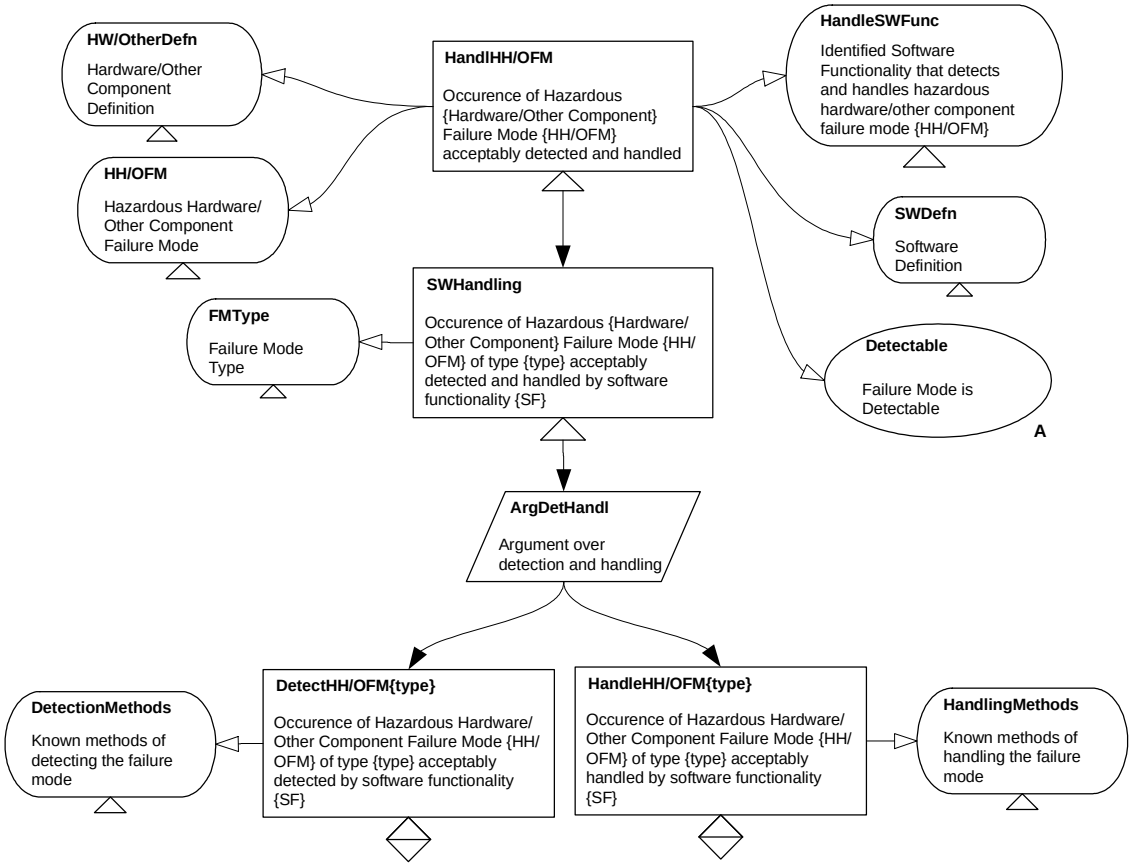
Handling of Hardware/Other Component Failure Mode

Author(s)	Rob Weaver, John McDermid, Tim Kelly		
Created	18/09/00	**Last Modified**	20/04/04

Intent	The intent of this pattern is to develop an argument that the software functionality can handle failures by hardware or other components.
Also Known As	
Motivation	The motivation for this pattern is to be identify the ways in which failure modes are detected and handled by the software, depending upon the type of the failure mode.

Structure



Participants		
	HandlHH/OFM	The overall objective of the argument - to provide sufficient support for the claim that that a hardware or other component failure mode can be handled by another component.
	HW/OtherDefn	This Hardware or Other Component Definition should give a clear description of the system hardware/other component. From the model it should be possible to determine how the failure mode effects the software and what type it is.
	HH/OFM	This context identifies the Hazardous Hardware or Other Component Failure Mode, for which this pattern develops the handling argument.
	HandlSWFunc	This context describes the software functionality that can detect and handle the occurrence of the failure mode.
	SWDefn	This Software Definition should give a clear description of the system software. From the model it should be possible to determine how the software functionality can handle the failure of the hardware or other component
	Detectable	This argument assumes that the software failure mode is detectable. A handling argument cannot be generated for an undetectable failure mode.
	SWHandling	This claim asserts that the failure mode of a particular type can be handled by the software functionality
	FMType	This context identifies the failure mode type which can be one of Omission, Commission, Early, Late and Value. The definitions of these failure modes are: Omission: *The service is never delivered* Commission: *The service is delivered when not required* Early: *The service occurs earlier than intended* Late: *The service occurs later than intended* Value: *The information (data) delivered has the wrong value*

	ArgDetHandl	This strategy describes the argument approach – decomposing across the detection and the handling of the failure mode.
	DetectHH/OFM {type}	This claim asserts that the failure mode can be detected by the software functionality
	DetectionMethods	This context provides the possible detection methods based upon the type of the failure mode. Detection methods include: Omission: Detection on Time (infinite threshold) Commission: Detection on Time (early) and/or unexpected input Early: Detection on Time (Early) Late: Detection on Time (Late) Coarse Value: Detection on out of safe bounds (e.g. range, rate of change) Subtle value failures can be detected if redundancy is employed.
	HandleHH/OFM {type}	This claim asserts that the failure mode can be handled by the software functionality
	HandlingMethods	This context provides the possible handling methods based upon the type of the failure mode and whether redundancy is employed.
Collaborations	• **HW/ODefn**, **HH/OFM** should be suitable for identifying the type of the failure mode identified in **FMType**. • **SWDefn**, **ContribSWFunc** should be suitable for identifying the handling software functionality identified in **SWHandling**. • **DetectionMethods** should be suitable for identifying the argument below **DetectHH/OFM{type}**. • **HandlingMethods** should be suitable for identifying the argument below **HandleHH/OFM{type}**.	
Applicability	This pattern identifies the claims about handling a hardware or other component failure mode by a piece of software functionality. It assumes that the failure mode has been identified. It also assumes that the type of the failure mode can be determined and the software functionality that can handle the software can be identified.	

	The pattern is only applicable to failure modes that can be detected. Undetectable failure modes cannot be argued about using this pattern.
Consequences	After instantiating this pattern the following undeveloped goals will remain: • **DetectHH/OFM{type}** and **HandleHH/OFM{type}** To satisfy the decomposition of **HandlHH/OFM** these goals need to be decomposed.
Implementation	This pattern should be instantiated in a Top Down fashion. All goals and contexts should be instantiated before continuing to a lower level in the pattern. It should be determined whether the failure mode is detectable before trying to decompose the argument. The type of the failure mode must be determined based upon the definitions provided in the Participants section. **Possible Pitfalls** • Not correctly determining whether the failure mode is detectable or undetectable can lead to an argument being generated that does not cover the failure mode in all possible contexts. • Incorrectly identifying the software functionality that can handle the failure mode. • Not correctly identifying the type of the failure mode can lead to an incorrect argument being developed.
Examples	None provided at this stage.
Known Uses	
Related Patterns	*Effects of Other Components* – This pattern has undeveloped goals which can be the overall objective of *Handling of Software Failure Mode.* This pattern forms part of a software safety argument pattern catalogue, which includes the following patterns: *Component Contributions to System Hazards* *Hazardous Software Failure Mode Decomposition* *Hazardous Software Failure Mode Classification* *Software Safety Argument Approach* *Absence of Omission Hazardous Failure Mode*

	Absence of Commission Hazardous Failure Mode *Absence of Early Hazardous Failure Mode* *Absence of Late Hazardous Failure Mode* *Absence of Value Hazardous Failure Mode* *Effects of Other Components* *Handling of Software Failure Mode* *Handling of Hardware/Other Component Failure Mode*