

UL 1998 - Software in Programmable Components

*Manoj Desai, Underwriters Laboratories Inc., Research Triangle Park, North Carolina
Laura Elan; Underwriters Laboratories Inc., Northbrook, Illinois*

Abstract:

This paper provides an overview of the "Standard for Safety, UL 1998 –Software in Programmable Components". Since this is a companion paper to the tutorial session under the same heading, the paper is only intended to provide a general, high level overview. Besides focusing on the key concepts, it will also discuss practical application of the standard.

Background:

Microprocessor-based control of product functionality has added another dimension to compliance engineering. With its use, there is also a growing need for addressing safety in the design and implementation of microprocessor-based controls. The emergence of new standards are associated with the increased knowledge of potential failure modes and a need for maintaining and improving levels of safety. As a result, product designers now recognize that potential safety-related risks must be identified, and systematic design and validation measures applied from the very beginning of the product development cycle, continuing throughout installation, and use.

Introduction:

Development of the UL 1998 Standard began in 1988 with a review of existing large-scale, mission-critical system standards. The objective of this review was to identify relevant requirements that could be scaled for practical application to consumer and industrial products.

During the initial writing, it was noted that many of the standards available at the time relied principally on engineering process requirements. The process criteria provides a foundation in UL 1998, however, a key objective was to augment these criteria by further codifying fail-safe and fault tolerant design requirements for software controlling safety-related functions. It was important to use UL 1998 to push the state of software engineering requirements forward by identifying intrinsic criteria associated with the delivered code. This approach coincides with the traditional UL product safety approach. Additional input was obtained from lead architects in industry who had been practicing these techniques for years.

The scope of the UL 1998 Standard was limited to application-specific, non-networked software in a programmable component embedded in a product for which a failure may

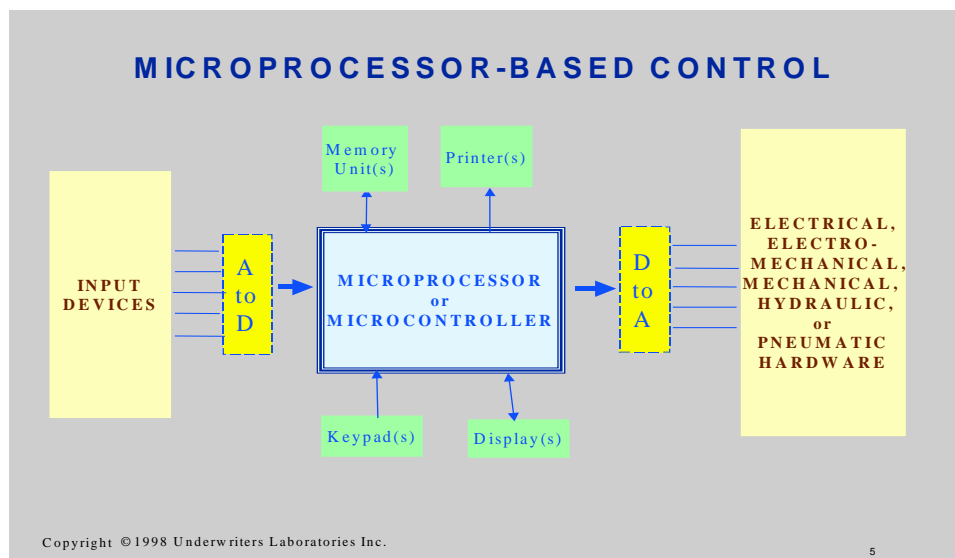
result in a risk of fire, electric shock or injury to persons. UL 1998 achieved the status of ANSI standard in February of 1999.

Approach:

Given the functional complexity inherent in software logic embedded in programmable components, the requirements in the UL 1998 Standard address safety management during development and maintenance by specifying both process criteria and design criteria. As such, it emphasizes the conduct of risk analysis activities; fail-safe and fault-tolerant criteria related to the design of safety-related software; consideration of provisions for hardware malfunctions; the application of analysis and test methods; documentation; handling of software changes; qualification of off-the-shelf software, and labeling that uniquely identifies the specifics of the product interface, the hardware platform, and the software configuration.

Key Terms:

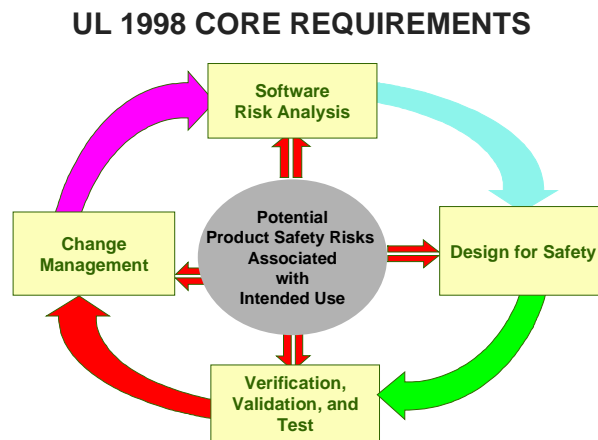
UL 1998 defines a programmable component as any microelectronic hardware that can be programmed in the design center, the factory, or in the field. Here the term "programmable" is taken to be "any manner in which one can alter the software wherein the behavior of the component can be altered.



The above figure illustrates a basic configuration of the microelectronic hardware and embedded software that may be present. A more general configuration may also include, the operating system or executive software, communication software, microcontroller, network, input/output hardware, any generic software libraries, database management, and user interface software.

Risk Analysis

UL 1998 requirements are intended to focus on risks that occur in software or in the process used to develop and maintain software. They are also designed to carefully examine different phases of the software development cycle, including requirements, design, and implementation and testing steps. The standard requires that a risk analysis should be conducted to determine the set of risks. The analysis should identify the states and transitions that are capable of resulting in a risk. Please refer to section 3, “Risk Analysis” and section 12.3, “Risk analysis approach and results” for the detailed requirements.



Copyright ©1998 Underwriters Laboratories Inc.

6

In practical terms, UL 1998 requirements are designed to provide identification of risks and traceability of actions taken to mitigate those risks throughout the software development cycle.

The standard also recognizes that other components like microelectronics hardware, for example, memory also contribute to the overall risk. Any embedded programmable component must therefore have both hardware and software components designed and validated to address safety requirements.

Design:

Product designs using programmable components usually are more complex designs because they provide greater functionality than the electromechanical designs they replace. The design complexity results from the use of software and microelectronic logic to provide additional features including codifying safety decisions and integrating safety functions with performance functions. The increased complexity leads to an increased potential for design inadequacies --- hence incomplete or incorrect implementation of safety-related functionality may occur. As a result the product designer faces additional challenges in achieving and validating designs for compliance with safety requirements.

UL 1998 requirements address the design challenges of identifying safety dependence, specifying and verifying process and data integrity, accurately representing the physical state of the product on a continuous basis, handling external events such as interrupts in the proper order, and providing the appropriate extent of self-monitoring.

UL 1998 recognizes the importance of the good design and need for the design verification. Therefore those activities are addressed in a number of places in the standard. For example, it discusses “Qualification of Design, Implementation, and Verification Tools” in Section 5, identifies issues related to “Software Design” in section 6, and recommended measures to address “Microelectronic Hardware Failure Modes” in section 8 of the standard.

Partitioning:

The focus of UL 1998 is on identification and mitigation of safety related risk. The concept of “Partitioning” is discussed in the standard. It is the separation of safety-related components from other (non-safety-related) components. This is a mechanism to protect and preserve integrity of data and instructions used in the safety related section. Section 7 “Critical and Supervisory Sections of Software” contains specific requirements. In practical terms, it provides a way to effectively design, develop, test, and maintain safety-critical components. It helps both the manufacturer and any independent third party auditor who is verifying compliance to a standard similar to UL 1998.

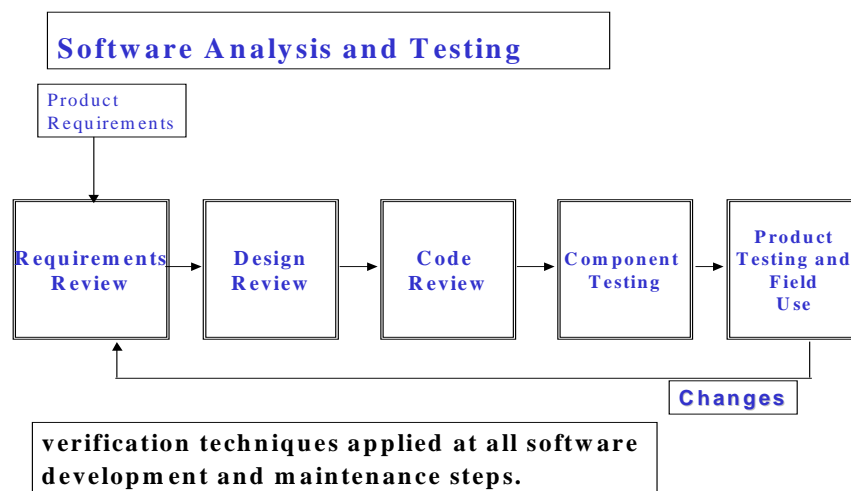
Off-The-Shelf Software

We are all aware that new technology is being introduced at a rapid pace and that to take advantage of that technology, hardware and software are often developed by a third party. When a product makes use of “Off-The-Shelf” (OTS) software, its use should be reviewed. It is important to clearly understand the purpose for which it is used, how it is used and how it interfaces with other components. More details can be found in the Section 13 of the standard, “Off-the-Shelf (OTS) Software”. It provides a practical way to assure that the product remains safe when a developer decides to use OTS software.

Software Analysis / Testing

Section 11 of UL 1998 emphasizes the role of software analysis, and the importance of testing throughout the software development cycle. It requires that software design and code analysis be conducted to demonstrate correctness and completeness of the safety critical functions. Besides commonly used testing that includes development and post-release testing, there are requirements for failure mode and stress testing. When it comes to certification, the reviewer relies on the test plans and test results as supporting evidence to satisfy him/her that the product complies with requirements of the standard.

The following figure illustrates typical software analysis and test activities in different phases of a software lifecycle.



Please note that any change resulting from product testing and/or the field use should also go through the same review and verification steps to fully understand the impact of that change on the safety of the product.

Documentation

In order to demonstrate compliance with the requirements of the standard, one must provide sufficient documentary evidence. Section 12 of the standard provides a detailed list of the documentation requirements. The essentials of the requirements are to keep a detailed account of all your plans (e.g software development, test, configuration management), designs, analysis (e.g. risk analysis), major decisions and rationale (e.g. OTS), walkthroughs, reviews, inspections, and testing (e.g. failure mode test) activities. It is also required to control and document all the changes to software so that it does not negatively impact safety related functioning of the product. The requirements related to “Software Changes and Document Control” are contained in section 14 of the standard.

The following figure illustrates one of the techniques used to demonstrate that appropriate actions were taken in the product development cycle to mitigate the risks identified in the risk analysis.

TRACEABILITY MATRIX

Identified Risk	Safety Requirement	Design Verification Records			
		Requirements Reviews	Design Reviews	Code Reviews	Tests

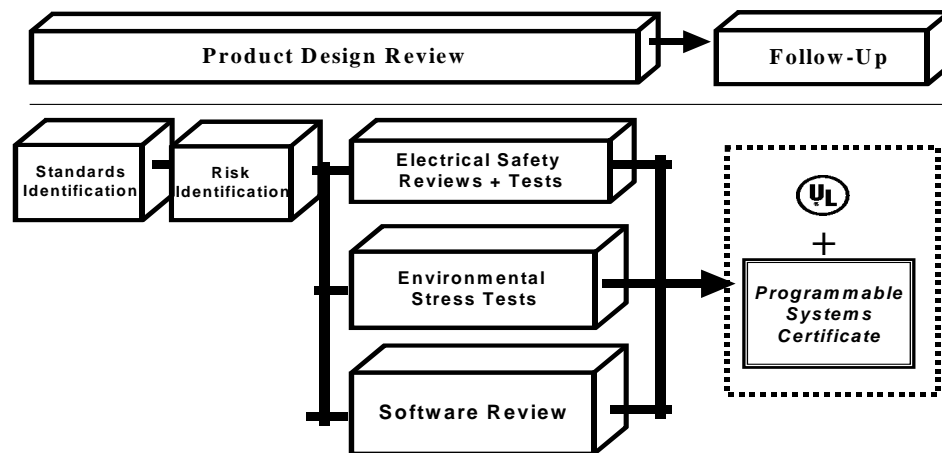
A means to demonstrate verification coverage of identified risks.

Application:

Let's briefly discuss how the UL 1998 is applied in practice. It is a reference standard and is used in conjunction with product safety standards that contain safety requirements for the programmable component and for the product hardware.

The following chart illustrates the UL certification process for one scenario in which a software review is one of the necessary reviews that a product undergoes before it becomes eligible to receive the UL Listing mark.

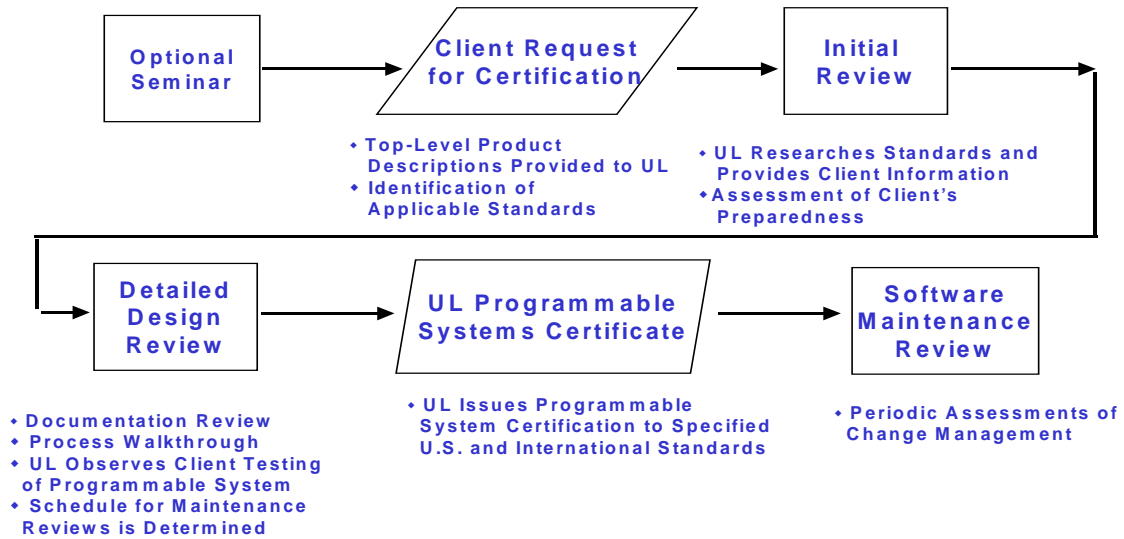
PROGRAMMABLE SYSTEMS SERVICE



UL 1998 review is requested in two other scenarios as well. 1) A client sees a benefit of such a review 2) UL engineer determines that it is needed because of the product design and use of new technology performing safety related functions.

Next, the obvious question is how the software review is conducted. The following chart illustrates the different steps involved in a software review.

SOFTWARE REVIEW STEPS



Other Standards:

Besides UL 1998, there are other standards which address the safety of programmable components. The paper can not be complete unless a brief mention of those standards is not included.

One of these is IEC 61508. This standard is targeted for complex systems and addresses the safety of a system based on a top down approach. It is based on the safety integrity level of the system and components that make up the system. There is lot of similarity between IEC 61508 and ANSI/UL 1998 as both uses a risk-based approach.

The other noteworthy standard is IEC 601-1-4, which addresses the safety of programmable electronics in medical devices. Again, this standard uses a risk-based approach and relies on a good software lifecycle model and traceability of actions taken to mitigate risks.

Author:

Mr. Manoj Desai has 20 years experience in the various aspect of software development. He has M.S, in Computer Science from the University of Nebraska at Lincoln and B.S. in Chemical Engineering from Indian Institute of Technology, Kanpur, India..

Before joining UL, Mr. Desai worked in various technical and management positions at IBM with responsibility for software design, development, testing and assurance. His work included participation on IBM corporate task forces with Mike Fagan on software quality and the product lifecycle.

At UL, Mr Desai is associate managing engineer with responsibility for certification and research in the area of safety related software.