A Methodology to Upgrade Legacy Industrial Systems to Meet Safety Regulations

Doaa Soliman¹, Kleanthis Thramboulidis²,¹, Georg Frey¹
¹Chair of Automation, Saarland University, Saarbrücken, Germany
²Electrical & Computer Engineering, University of Patras, Greece
{doaa.soliman, georg.frey}@aut.uni-saarland.de, thrambo@ece.upatras.gr

Abstract— There is a need to upgrade legacy system in industry to conform with safety norms and regulations defined by recent standards. The great investment for the development of these systems is the main reason for the industry to look for approaches to upgrade legacy systems instead of adopting a redevelopment of the whole system. In this paper, we describe an approach to upgrade legacy industrial applications based on the IEC61131 function block model without the need to redesign the whole application. The approach that integrates the 3+1 SysML view model with safety engineering is adopted and is tailored to the needs of upgrading legacy applications. Challenges are identified and solutions are proposed towards the definition of the whole development process including the verification of the so generated safety application. A laboratory system is used as a case study in this paper to demonstrate the applicability of the proposed approach.

Keywords- 3+1 SysML; Safety applications; PLCopen; verification and validation; Model-checking

I. INTRODUCTION

The IEC 61131 [1] set of programming languages is widely used in industry and the majority of the legacy industrial automation systems are based on these languages. On the other hand social demands for safety have accelerated international standard organizations, such as ISO and IEC to release standards on safety. That is why safety issues of IEC 61131 have already been examined by the research community. In [2], where a safety related evaluation of programming language constructs is given, function block diagrams of IEC 61131 that are based on verified libraries are considered suitable for safety-related control applications that should meet safety requirements of different Safety Integrity Levels (SILs) according to IEC 61508 [3]. Applications that are developed on the basis of verified libraries, such as the safety library of PLCopen [4], are suitable to meet the requirements of the second upper Safety Integrity Level, i.e., SIL 3. As claimed in [5], the use of IEC 61131 function block paradigm even in the case of using safe libraries does not guarantee the safety level of the system. Safety analysis should be performed during development and the integration of the safety analysis with the traditional development process of IEC 61131 based industrial automation systems is not obvious. Hazards should be identified before and during the development process of the system and safety measures must be defined to reduce the risk imposed by the use of the system. During the last years the required solutions in technologies to ensure safety have been more complicated [6].

Manufacturing industries due to the increase in automation and the stronger demands on safety, which is imposed by safety standards, face the challenge of upgrading legacy industrial automation systems to conform with the norms and regulations imposed by these standards and certify that systems are safe for the human life and the environment. The alternative to throw away the legacy system and develop a new one from scratch to meet the requirements specification, if such a specification exists, is very expensive. In most of the cases, a) the reengineering of the legacy system is not an adopted solution due to several reasons [7], and b) the main reason for upgrading legacy industrial systems to be compliant with safety standards lies in the high value of software running on the existing system. In this case, the only alternative is to develop a safety system that will meet the safety requirements of the system and effectively integrate it with the legacy system. It is obvious that an approach like the one described in [5], which addresses the integration of the development process with safety engineering, may not be used since it addresses the development of a new system. This may be used only in the case we adopt a reengineering of the whole system. Therefore, this paper presents an approach for upgrading a legacy system to be compliant with safety regulations and norms defined by current standards, such as IEC 61508 [3] and IEC 61511 [8]. The proposed approach, which utilizes SysML, addresses the following activities that have to be performed by the engineer:

1. Definition of safety requirements for the upgraded system.
2. Definition of the requirements of the safety system.
3. Design of the safety system.
4. Verification of the safety application.
5. Integration with the legacy system.
6. Verification of the upgraded system.

We are not aware of any other published work that describes a methodology for upgrading a legacy system to meet safety requirements. However, there are several works that consider the integration of UML [9][10][11] and SysML [12] with safety engineering.

The main contribution of this paper is the definition of an approach (a methodology) to upgrade legacy industrial systems
to meet safety norms and regulations. The approach is based on the 3+1 SysML-view model and its integration with safety engineering [5]. The 3+1 SysML-view model is a SysML based realization of the Model Integrated Mechatronics paradigm where SysML is used as a modeling language on the mechatronics layer. The paper also briefly describes an approach for the verification of the safety application. A laboratory system is used as running example to illustrate the proposed approach.

The remainder of this paper is organized as follows: Section 2 briefly presents the approach that is adopted as basis for this work. In section 3, the XY coordinated table, i.e., the laboratory system used as a case study in the context of this work is described. In Section 4, the proposed approach is presented. A brief description of the verification process of the safety application is given in section 5 and the paper is concluded in the last section.

II. INTEGRATING SAFETY ENGINEERING WITH THE DEVELOPMENT PROCESS: THE CASE OF THE 3+1 SYSML-VIEW MODEL

The approach that we have adopted in the context of this work adopts the Preliminary Hazard Analysis (PHA) and applies it using as source information the Mechatronic System (MTS) requirements, as shown in Fig. 1, where a draft model of the integration of safety engineering with the development process of the 3+1 SysML-view model is presented. PHA and Operations Hazard Analysis are methodologies that are usually applied at this stage of development and their objective is to identify hazards, determine their causes, and plan their elimination or mitigation.

According to the above approach, the first activity of the safety engineer is to apply hazard analysis based on the MTS requirements which are composed of SysML requirement diagrams and essential use cases. During this process a list of hazards and their causes are identified. These artifacts are next used for the risk analysis, which is part of the PHA that is executed for each hazard. The result of this activity is the identification of the severity and the probability for each hazard and then the estimation of the associated risk. The process of safety measure definition complements this solution-independent phase of safety analysis and has as result an updated MTS requirements specification that is consistent with the safety requirements of the system.

The MTS requirements specification is next used to define the system’s architecture, i.e., the MTS architecture. A second phase of safety analysis, a solution-dependent hazard analysis that results among others in the definition of required SILs for the system components and their assigned functions is performed at this stage. Then, hazard analysis is applied to every Mechatronic Component (MTC) iteratively down to the primitive MTC level. The hazard analysis for each MTC is composed of a PHA followed by a solution dependent Hazard Analysis. A detailed discussion on the solution dependent safety analysis is given in [12].

III. THE CASE STUDY

The system used as a case study in this paper is the XY coordinated table of the Automation laboratory at Saarland University, shown in Fig. 2. This type of precision-controlled automated movement is broadly used in mechanical processes and applications including material handling and pharmaceutical. The system consists of an automatically moved XY drawing table with manually adjustable marking pen mounted above an Aluminum base. The two axes are operated with Siemens synchronous motors and precision linear guides from a Rexroth Bosch company. The drives are controlled through Sinamics S120 servo drives. The system is controlled using a soft-PLC WinLC-T from Siemens equipped with S7-technology. The user interface is achieved using Wincc HMI software that allows the operator to perform the required motion tasks through simple graphical screens. The control system equipments, i.e., PLC and Servo drivers, are connected through a DP Profibus protocol. For safety reasons the system is protected by a glassy cell with two doors on the top of the cell.
IV. THE PROPOSED APPROACH

Among the challenges we have to address in upgrading a legacy system to meet safety standards we discriminate: the requirements specification for the safety system, its design, its integration with the legacy system and its verification. For the requirements specification a three steps hazard analysis is performed consisting of a preliminary hazard analysis, a solution dependent and a component Hazard Analysis. SysML is used to create the requirements model of the system and capture modeling artifacts such as hazards, causes (any kind of causes, i.e., systemFault, systemMisBehavior, ActorMisBehavior), safety measures of both types, i.e., DevelopmentTimeMeasures and OperationTimeMeasures, but also safety functions and safety constraints. These modeling constructs have been defined to represent in the model the corresponding safety analysis concepts to facilitate the integration of the development process with safety analysis. A SysML profile to capture the safety related key concepts and facilitate the application of the proposed approach is under development. Fig. 3 presents part of this profile. The design is also expressed in SysML and will be consequently used to automatically generate the Function Block Diagrams (FBDs) that constitute the design diagrams for the safety application. Block definition and internal block diagrams may be used to capture the structure of the application, while sequence or activity diagrams may be used to capture the dynamics of interactions between the application components [13].

A. Preliminary Hazard Analysis

Even though we are dealing with a legacy system, we adopt the approach of preliminary hazard analysis on the use case level as defined in [5]. It is more effective first to have a safety analysis of the system performed on an abstract level of system description, i.e., the one that captures the system as a black box and emphasizes its interactions with its actors. In this case, hazards that emanate from the actor-system interaction and are caused by actors and/or system misbehaviors will be identified, as shown in the activity diagram of Fig. 4, that models the proposed PHA process. Essential use cases are used since they provide an effective mechanism for specification of human machine interactions at this level of abstraction. Essential use cases may be easily derived from the system use cases, if these already exist for the system; otherwise these may be constructed using the user’s guide of the system, assuming that this captures all the interactions of the system with its actors. Based on these essential use cases, hazards should be identified and the use cases should be modified to avoid these hazardous situations. It is the objective of PHA: a) to concentrate on the actor machine interactions and identify the system and actor misbehaviors that may cause hazards, and b) to plan mitigation strategies for those hazards that have unacceptable risk [5].

HazardAnalysis, RiskAnalysis and SafetyMeasureDefinition are the main subprocesses of the PHA. HazardsToMitigate will be associated in SysML design diagram with the related safety measures and will be used along with the AcceptedHazards in the system’s verification process.

It is the objective of PHA: a) to concentrate on the actor machine interactions and identify the system and actor misbehaviors that may cause hazards, and b) to plan mitigation strategies for those hazards that have unacceptable risk [5]. System faults are also considered at this stage at a very high level of abstraction. System faults introduced by the specific implementation will be considered later during the solution dependent hazard analysis. Essential use cases are used since they provide an effective mechanism for specification of human machine interactions at this level of abstraction. Essential use cases may be easily derived from the system use cases, if these already exist for the system; otherwise these may be constructed using the user’s guide of the system, assuming that this captures all the interactions of the system with its actors. Based on these essential use cases, hazards should be identified and the use cases should be modified to avoid these hazardous situations. This will introduce additional functionality for the system but probably also for the user. The intention is to identify possible hazards that may be generated from this level of description. These hazards may lead to accidents or harm the system’s environment. For each hazard, we have to identify its causes and the severity level and calculate the risk of it. The risk is defined as the product of the probability of its occurrence (likelihood of occurrence) and its severity, i.e., Riskhazard = Probabilityhazard x Severityhazard.

The legacy system of our case study has three main use cases: system initialization, system setup and perform task. In the system initialization use case, the operator is able to define the operating speed, the home position and select one of the pre-defined tasks to be performed by the system, e.g., drawing a straight line on a work piece. In the system setup use case, the machine is prepared for automatic operation by moving to the home position. The main system operation, i.e., the execution of the selected task with the defined operating speed is performed through the perform task use case. Fig. 5 presents the perform task essential use case.
Since safety issues were not considered for the legacy application, except from the protective cell, which is not monitored, neither controlled by the system, the operator may access the machine while operating. This may lead to human injury due to the high speed movement of the drawing table.

A possible hazard for the perform task use case is the “protective cell door is open.” Analyzing the Actor behavior in the context of this use case, we identify the following possible misbehavior of the actor: “the actor does not close the doors.” It is obvious that this ActorMisbehavior may cause a hazard during the system operation. This hazard, if combined with EnvironmentalConditions or OperationalConditions, such as the operator access the machine with his hand, may lead to an accident. All this information is then used to perform the risk analysis that is executed per each hazard. Since the probability and the severity of this hazard are high, the risk of the hazard is also high which means that it should be defined as not accepted and actions should be taken to mitigate the risk. Safety measures, i.e., recommendations, should be proposed to mitigate or eliminate the hazard. Example of such a safety measure is to have the system “check the status of doors” before starting its behavior but also during its operation. So, the safety related function “confirm doors closed and locks applied” should be considered as prerequisite to the perform task use case to avoid hazards.

Taking into account the proposed safety measures we modify the use case by adding extra, safety related, functionality to its description, so as to define a safe actor-system interaction. Fig. 6 presents the updated use case, i.e., the one that mitigates the risk of the identified hazard. This results in a list of functional and non-functional requirements (required SILs) for the safety application. The safety application has to “lock the doors” and prevent unlocking until the end of the use case, i.e., until the drawing table reaches the end position, which mean that the motors are in a “safe stop” mode. In this mode the system has to wait for the operator’s “unlock command”.

The exception, “doors open during system operation” has to be identified for this use case and the behavior of the system for this condition should be defined. This has also been captured as hazard in the hazard analysis phase. This hazard may be caused either by SystemMisBehavior (fail in locking the doors) or by ActorMisBehavior (break of the locking mechanism). An activity diagram may be used to capture in a semi-formal notation the behavior described by the updated use case. The notation of SysML for the interruptible area maybe used to capture the behavior of the system for this exception that would be an emergent stop of the system. This is an OperationTimeSafetyMeasure (see Fig. 3), since it enables an operator to handle during operation-time a system state that may lead to a hazard or accident. On the other side, a DevelopmentTimeMeasure defines, during development time, the proper system behavior that is required to prevent hazards. For example, the definition of a high required SIL, during design time, for the safety application components, e.g., using a guard locking interlock switch for handling the doors of the protective cell, is considered as DevelopmentTimeMeasure.

In a similar way we examine the rest of the use cases of the system. For example, for the system setup use case we have the following reasoning. In the setup mode it is assumed that the operator should have access to the machine while operating. So, the motors’ movement is not allowed to exceed a predefined safe speed. The hazard “motors speeds exceed the safe speed” can be handled as an exception in the system setup use case. To avoid this hazard the speed of the motors is checked during the setup mode time period. If the speed exceeds the safe one, it is the responsibility of the safety application to bring the system to a safe state and also prevent the manual access to the machine until it reaches a safe state.

B. Design of the safety system and solution dependent hazard analysis

The safety system has to be designed so as to meet the requirements defined by the PHA. For each modified use case and more specifically for each extra functionality, we have to propose the system components required to realize the specific functionality. For example, for the realization of the Unlock doors functionality, two Guard locking Interlock switches, one push button and a software component are required. The result of this process will be the definition of the system’s architecture. The SysML internal block definition (ibd) diagram of Fig. 7 presents the proposed hardware architecture for the safety system of the case study. An analogous ibd diagram is constructed for the software application and the «allocate» association of SysML is used to define its deployment on the hardware architecture. The ibd provides a better way to capture the architecture of the system compared to the bdd, since it captures not only the composition or aggregation as is the case for bdd but also captures the interactions among system components as well as interfaces which are crucial for the system architecture. Of course bdd is used in a previous step of the development process.

The proposed architectural design has to pass from safety assessment to identify possible hazards that may be caused by the specific solution. Faults introduced by the specific collaboration of system’s MTCs may lead to hazards. Sequence as well as collaboration diagrams are used as source...
information in the solution specific hazard identification process. For example, the hazard “emergency stop not activated” may be caused by a fail in the communication link between the EmergencyStopButton and the PLC (see Fig. 7) or to a fault of EmergencyStopButton component or the corresponding software component. In order to eliminate or mitigate the faults of these components, low-level redundancy is added to the model. For example, a second communication link is introduced in the model to link the emergency stop button with the PLC. At this stage the required SIL for the hardware and software components is defined. The use of traditional hardware components, e.g. PLC, Emergency Stop Buttons and so on, introduces a high probability for hazards and results in high risk regarding safety. High SIL is required for the safety system components, so as to minimize the probability of the hazard that emanates from the corresponding solution. This is why safety certified hardware components are selected. This is also the case for the software components, so the safety application is realized using the Safety Function Block (SFB) library of PLCopen [4].

The SysML design specification is next used to get the IEC 61131 FB design diagrams for the software part of the safety system. This process may be automated so as to have the SysML specification to be automatically transformed, using the appropriate model-to-model transformers, to IEC 61131 FBD diagrams. A description of this approach is given in [13].

The software part of the safety system was implemented using KW-Software tool, but any other PLC programming tool can be used assuming that: a) it supports the IEC 61131-3 FBD programming language, b) it provides support for the PLCopen SFB Library, and c) it may export the FBD specification in PLCopen XML format. These features will allow the automatic transformation of the safety application to a specification that is readable by a model checking tool for verification. The implementation of the safety application in FBD and its transformation to UPPAAL XML format is presented in [14] and [15].

The FB diagram of the developed safety application, shown in Fig. 8, implements all the safety functionality related to the handling of doors. The first SFB, i.e., the SF_GuardMonitoring, is to monitor the status (open/closed) of the doors based on the guard switches mounted on these. The second SFB, i.e., the SF_GuardLocking, is to handle the lock and unlock commands.

Figure 7. Internal block definition diagram for the hardware part of the safety system.

Figure 8. Part of the FBD design diagram of the safety application.

C. System Integration

While integrating the safety related system with the legacy system, the responsibilities of every system must be clearly defined. SysML sequence diagrams are used to semi-formally capture the interactions between the legacy system and the safety one. An example sequence diagram related with the safe perform task use case of the example application is shown in Fig. 9. It clearly shows the communication dialog between safety-related system and the upgraded legacy system. For example, adjusting the mode selector switch to Automatic position is operated by the safety-related system. Then, a signal is sent from the safety-related system to the upgraded legacy system to inform an automatic mode request. However, a normal operation of the machine is only possible when the doors are closed and locked. So, the upgraded legacy system enters wait states until an enabling signal is received from the safety system.

Safety-related issues, such as monitoring doors status and, locking and unlocking doors, are handled by the safety system. So, safety-related devices are chosen for this purpose. These devices are certified to be complaint with safety standards and have specified SIL; in our case study the chose switches have SIL equal to 3. However, non-safety-related devices such as Reset push button can be also handled by the safety system. To
logically recognize safety and standard inputs of the safety system, they are given different declaration types, e.g. SAFEBOOL and BOOL.

As a first step of upgrading the legacy system, the firmware of both Sinamics S120 drives (see Fig. 10) is updated to allow extended integrated safety functions, i.e., SS1, SLS. The control of these extended safety functions is realized using a TM54F terminal expansion module. The safety-relevant logical pre-processing of the input signals is handled in a safety PLC. The safety PLC accepts input signals from safety-related devices (Emergency Stop button, Mode Selector switch and Guard switches), standard devices (Reset button, Unlock Guard button), standard PLC (Microbox PLC) via IO modules or upgraded drive systems via TM54F.

Fig. 11 presents an abstract view of the integration of the legacy system with the safety one. This integration is compliant with the PLCopen recommendations. To recognize the corresponding signals easier, in the graphical interface overview, the input signals get sorted in correspondence to the safety functions and get assigned directly to the safety application regardless if they belong to the functional or safety application. Most of these standard signals (1) do not get manipulated by the functional application anyway. Some systems are able to address the standard inputs/outputs directly from the safety application; others provide these signals through the functional application. The signals (2), which are processed by the functional application and which do not belong to specific safety function such as diagnosis, are simply illustrated at the interface between the two applications. However, hidden interface between safety PLC and drive systems are not shown in Fig. 11, since the drive internal state machine is complex and vendor specific.

As an example of handling integrated safety functions, the safely limited speed (SLS), which is an integrated safety function offered by many drive systems, is initiated and monitored by the safety system. However, actuating and controlling the motors speeds is still under the responsibility of the legacy system via the drive systems. So, if the operator selects the setup mode by adjusting the Auto/Manual selector switch to Manual position, the SLS safety function is activated via TM54F. As a response to this activation, the motors start moving with safely limited speed and an acknowledge signal is send from drives via TM54F to the safety system. In that case, a safe state is indicated, i.e., using green light led, and the operator is allowed to access the XY table. If the acknowledge signal is lost for any reason, the unsafe state is assumed. Therefore, actions such as alarm generation are taken to prevent hazardous situations.

V. Verification of the Safety Application

The safety application, which was implemented in FBD language and handled as a project by the IEC 61131 tool, is exported into PLCopen XML format. This format is the one to be transformed to UPPAAL XML format to be verified against the safety functionalities, as shown in Fig. 12. For this process to be automated, we have modeled the FBs of the PLCopen SFB library as timed automata (TA) using the UPPAAL graphical editor tool. Every validated SFB was then verified against specified properties using the UPPAAL verifier. To test the functionality of the verified TA SFBs, an approach is suggested to build up and verify a TA safety
application using the TA SFBs. This approach was tested using several safety application examples presented in [14].

Based on the verified PLCopen SFB library [15], which is described in UPPAAL XML format, and a defined UPPAAL XML template, the PLCopen XML safety application is transformed into UPPAAL XML format.

The main actions of the model transformer are:

1. Identify FBs instances (local variables) in PLCopen XML project.
2. For each FB instance: a) insert a timed automaton XML model in the List of models part of UPPAAL XML and b) insert its corresponding input/output variables declarations in Global declaration part of UPPAAL XML template.
3. The variables declared in the External variables part of PLCopen XML are inserted in the Global declaration part of the UPPAAL XML specification.
4. From the FBs positions defined in the FBD body of the PLCopen project, the execution flow of the FBs is extracted and transformed to system priorities depicted in the System part of the UPPAAL XML specification.
5. The internal and external connections described in PLCopen XML are converted to connections of TA models and inserted in the List of models part in UPPAAL XML.

After applying the transformation actions, the produced UPPAAL XML project can be used by the UPPAAL model checker tool to perform the verification.

CONCLUSION

A methodology to upgrade a legacy industrial system to be compliant with rules and regulations imposed by safety standards has been presented in this paper. PHA is applied to a high level description of the system the one expressed by essential use cases, to identify the safety requirements and the corresponding safety measures to mitigate the risk imposed by them. This results to the definition of the requirements of the safety system. SysML is used to create the requirements model of the system and express the system architecture. The application of the solution dependent hazard analysis on the proposed architectural diagram is the next step of the safety analysis process. The safety system is designed, implemented and verified to meet the requirements generated by the application of PHA. The UPPAAL model checking tool was used to verify the safety application that is expressed in the form of IEC 61131 function block diagrams.

We are currently working: a) on the automation of the transformation of the safety application specification in PLCopen XML format into a readable format by the UPPAAL model checker tool, and b) on the model transformation from SysML to IEC 61131.

REFERENCES