2011 IEEE International Conference on Rehabilitation Robotics
Rehab Week Zurich, ETH Zurich Science City, Switzerland, June 29 - July 1, 2011

Building a Safe Care-Providing Robot

Leila Fotoohi
Automation Institute
University of Bremen
Bremen,Germany
fotoohi@iat.uni-bremen.de

Axel Gräser
Automation Institute
University of Bremen
Bremen,Germany
ag@iat.uni-bremen.de

Abstract—**A service robot especially a care-providing robot, works in the vicinity of a human body and is sometimes even in direct contact with it. Conventional safety methods and precautions in industrial robotics are not applicable to such robots. This paper presents a safety approach for designing the safe care-providing robot FRIEND. The approach is applied in each step of design iteratively to identify and assess the potential hazards during design. The steps are explained briefly in this work. The main contribution of this paper is verification of safety requirements using the Ramadge-Wonham (RW) framework. The greater complexity of the tasks the robot will perform, the more complex is the identification of safety requirements. Use of this framework led us to analyze the requirements and verify them formally, systematically and on a modular basis. In our approach human-robot interaction (HRI) is also modeled by a set of uncontrolled events that may happen any time during operation. Subsequently the safety requirements are modified to consider these interactions. As a result the safety module behaves like a controller, running in parallel with the system, which maintains the system safe and works according to the safety requirements by enabling the admissible sequences of events.**

Keywords-service robot, reliability, supervisory control theory

I. Introduction

Up to now most service robot development has concentrated only on realization of the system. Less attention is paid to the safety considerations. Although several designers of service robots have started to launch their innovations into the market, there still exist neither international guidelines nor regulations comprising safety aspects for a specific product of the service robot type.

Recently researchers have focused their attention to develop the standards for intelligent assist devices (IAD).Their work is basically adaptation of safety standards for industrial robot (e.g. ISO 10218) and machinery (e.g. ISO 13849). However their evolutions still ongoing and the established committees are still involved in this evolution [1]. While these standards are more focused on dynamic issues, the other current standards for safety-critical programmable devices can be still used for other general aspects of safety, such as design procedures (e.g. ISO 62262/IEC 61508). These standards set a generic approach for all safety life cycle [2] activities in the system design.

Here an approach for safety in a care-providing robot FRIEND is discussed. Based on general safety and methodology of IEC 61508 [2], our approach is developed to comprise systematic ways to do the safety analysis process according to the overall safety life cycle phases. In this approach, which is a stepwise process, we start from the analysis and continue the analysis path up to the verification and validation of the realized safety requirements. The steps are partially tool supported to ease the task as well as to enhance the reliability of the results. We explain these steps briefly, and finally give an example that with more detail about verification that is novel application of Ramadge-Wonham (RW) framework.

Hazard identification is the initial step, for which we use the common system engineering approaches for safety analysis to define possible credible hazards in the system. In these analyses we take the advantages of the software tool APIS-IQ to document the result systematically as well to impose version control in the development team. Hazard analysis on UML models of other robotic applications have successfully applied, such as robowalker MIRAS [3]. However the complexity of the analysis is different for each robotic application and depends on the complexity of the tasks that a robot can perform.

After definition of the requirements, necessary to avoid the identified hazards, we implement these requirements in the system. Depending on the task, the implementation can be performed either into the already existing software control structure or into the parallel safety system which monitors the robot behavior especially in critical situations and decide about the proper action based on these observations. Finally the realization is verified by applying Ramadge-Wonham (RW) framework. Since introduction of this theory many researchers have worked on its application in different fields of system engineering. Some research results have been reported on robot in manufacturing cells [4,5] and some in mobile robots applications [6]. In [7] the authors applied this method to a medical application, which is used for a patient support system of a MRI scanner. However most of the previous works do the analysis for the known controlled system. In other words the reverse engineering is done for a system, in which the requirements are already known. Furthermore the analysis is carried out on fully autonomous systems with limited user interactions, or even no interaction at all. But a care–providing robot, is a semi-autonomous system and still under development. The safety requirements are still open questions that need further investigation.

The novelty of our work is to apply this framework to carry out safety design verification, where an abstract model of a system is checked for desired behavioral properties. The choice of abstract model and the related safety specification depends on the task to be performed and may vary from a task to another. However we show a simple example here to illustrate how the RW framework is applied for the systematic safe design and verification of a care- providing robot with human-robot interaction (HRI).

This paper is organized as follows. In section 2, at first a short summary about FRIEND system components and software architecture is given. In section 3, the safety approach is shown with brief explanation about each process step. The next two sections, 4 and 5, deal with theory of discrete event systems and supervisory control. We apply the theory to FRIEND system, given a simple case study for verification of safety requirement in section 6. Finally we conclude the paper and point out open problems and further works.

II. CARE-PROVIDING ROBOT FRIEND

FRIEND[1] system is the third generation of a multi-actuator and multi-sensor service robot, under development within the IAT research group at the University of Bremen.

This robot is built around a robot arm, which is a 7 DOF manipulator mounted on a wheelchair. The robot can accomplish different types of scenarios, such as ADL (Activity Daily Life), workshop, library scenario for handicapped people and will help them to be independent from care personnel at least for 1.5 hours. Fig. 1 shows a side view of the system with a description of different parts.

There are different input devices which receive a command in higher abstraction level (e.g. "pour a drink") from a user. A TFT display mounted on a panning arm in front of the user serves a human machine interface (HMI) with a touch-sensitive surface. For users who lack function in both hands, other input devices can be used (e.g. chin joystick, eye tracker, speech control, …).

Different sensors are used for perception of environment, such as: an intelligent tray for the precise location of objects to be manipulated on the tray and a stereo camera system for objects and obstacles recognition. There are some other additional sensors used for remote controlling of appliances in the environment, e.g. an infra-red control unit for opening doors.

A software structure called MASSiVE (Multilayer Architecture for Semi-Autonomous Service Robots with Verified Task Execution) is developed for task planning and execution of tasks in FRIEND system [8].

The task execution is as follows. The task is first selected by the user (e.g. pouring a drink), then initial monitoring is done with the stereo camera system mounted on top of the wheelchair. The user interacts with the system to complete the task. He is able to handle some erroneous situations that can be resolved by the user; for example he can support to locate the target to be grasped, if it cannot be recognized by machine vision. More details about software architecture and tasks' execution can be found in other literatures [8,9].

[1] FRIEND : **F**unctional **R**obot with dexterous arm and user- fr**IEN**dly interface for **D**isabled people

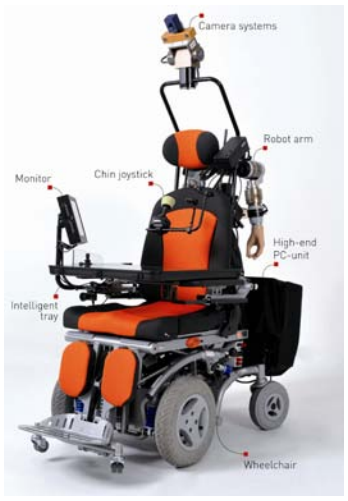

Figure 1. FRIEND system side view

The environment is not formally modelled in the practical sense, therefore any feedback signals transmitted from the environment are only semi-qualified. So the riskless operation cannot be guaranteed at all times. Therefore additional sensors are necessary which observe the manipulation and send feedback to the system in case of observing unsafe situations during manipulation.

III. SAFETY APPROACH FROM ANALYSIS TO VALIDATION

In this section we explain briefly all the steps of our approach for the safety of care-providing robot. Fig. 2 shows these steps starting from analysis to verification and validation of the safety requirements. At the initial step for safety analysis the possible hazards in the system have to be determined. Here we apply the common methods in system engineering to the FRIEND system [10]. One of these methods is Failure Mode and Effect Analysis (FMEA). This method proceeds from known causes to unknown effects; thus it is an inductive technique and it is usually done after HAZard and OPerability analysis (HAZOP) [11]. It can be applied in the intermediate phase of development when the components of the system are known. We apply APIS-IQ software for systematic hazard analysis as well as for final methodical documentation. In this software application a tree structure of the system must first be extracted showing each component, together with its subcomponents having an important role in proper functionality of the component, and finally the entire system. As more than fifty percent of the development focuses on software development we break the system down into two major parts: Hardware and Software. For the software part, we break down the software component according to the software control architecture MASSiVE. To reduce complexity of the task and also for a feasible analysis of the software we consider only the interface functions used between different software modules. (IDL CORBA functions)

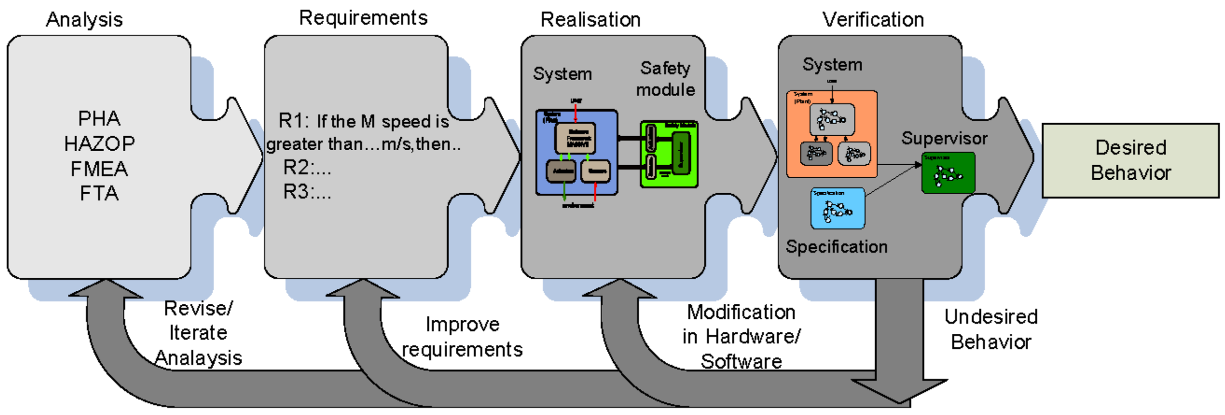

Figure 2. Safety approach process steps

As the software development in this project is done in rhapsody environment, failure modes for the software are extracted from UML models of each related object or module. For the hardware part we do the analysis as it is common in system engineering. The reader is referred to the other literature for more details and examples about methods [10, 11].

All the failure modes, which would result finally in unsafe behaviour of the system, are filtered out in the next step. By determining the remedies for these hazards the safety requirements are achieved and are listed in natural language at this step.

The next step is the realisation of safety requirements. These requirements are realised in a safety module run in parallel to the system [12]. The idea of the safety module for FRIEND system is to add a monitoring system equipped with additional sensors for observing the important parameters during manipulation, which play a major role in safety of the system. Then based on this observation the module decides for activation or deactivation of the event or series of events which would finally results in bringing the system to a hazardous situation. In other words the module acts as a supervisor to determine the next state, using a practical and efficient way, from the trajectory of states in the implementation. This decision is made by enabling or disabling admissible events. This step will be further discusses in section 6 by an example. The actions are carried out either by existing actuators (e.g. by direct controlling the manipulator) or by extra actuators, which are relevant only for safety purposes (e.g. watch dog system). Fig. 3 shows the schematic diagram for safety module for the system.The boundary of the safety module shows only a part of actuators and sensors. The reason, as discussed above, is that observation and actions by the safety module can be performed either by specific sensors and actuators only for safety purposes or by the already reliable existing sensors and actuators used for manipulation.

As the user is always in interaction with the service-robot during manipulation, his contribution might enhance availability of the system in some abnormal situations. Depending on the hazard categories a warning message on human machine user interface is given to the user for correct actions to keep the system safe and operational as well. Also depending on the user observation, his command might change the system manipulation process such that the safety is ensured.

Finally for safety design's verification a formal method based on RW framework is developed. The safety requirements are expressed with the natural language which can be formalized in automata that show the desired behaviour of the system. Taking the advantage of supervisory control theory, the maximal permissive behaviour of the system (considering uncontrollable events) guaranteeing these requirements can be automatically calculated and the system is correct by construction. However since the verification consider only abstract models and the results depend on the correctness of the models itself, the final realisation has to be validated against safety requirements. Theses can be done by a series of real time testing and human in the loop simulations. These methods are beyond scope of work of this work and the next sections more focus on verification, starting by a brief introduction to the related theory.

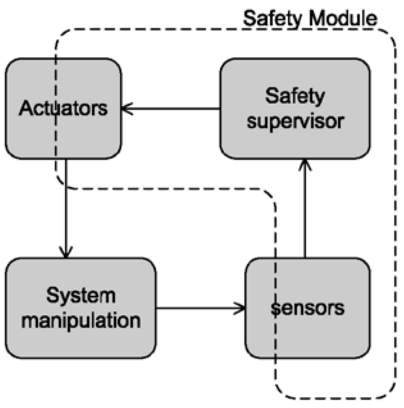

Figure 3. Safety module schematic diagram

IV. Discrete-Event Behavoiur of FRIEND

In a discrete-event system (DES) the state space of the system is discrete and state transitions are event-driven, which can be seen at different point of time. A service robot's behavior is viewed as a collection of events and states. For example all states that robot can reach during implementation of a daily life task can be interpreted as states of the system. As an example consider task scenario of pouring a drink. The robot manipulator can reach the following states: the gripper is opened, manipulator is in the bottle position, the manipulator is grasping bottle, the manipulator with grasped bottle is in glass position, and manipulator is in the state of pouring water into the glass.

The transitions can be interpreted as commands which are given to the manipulator at different points of time through the program: open gripper, move manipulator to bottle position, grasp bottle, move manipulator and grasped bottle to glass position, pour water into glass. These can be extracted from the process abstract structure (PSA) for each task [9].

A discrete-event system can be formally modeled by an automaton. An automaton is usually presented by 5-tuple set [13,14]:

$$G= \{Q, \Sigma, \delta, q_0, Q_m\}. \quad (1)$$

Here Q is a finite set of states, Σ is a finite set of events σ, which make up the alphabet of the system, δ is the transition (partial) function $\delta(q, \sigma)=q'$, in which q' is the next state, q_0 is the initial state and Q_m is the subset of final or marked states. Actually G models the physical system and is called a generator. It generates sequences of the events of the system.

The robot system like any other DES has a set of events associated with it. This set of event is thought as the alphabet of the system. Each scenario for robot tasks can be written as a sequence, or string, over this alphabet which finally builds the language of the system, defined below:

$$L(G) := \{s \in \Sigma^* : f(x_0,s) \text{ is defined}\}. \quad (2)$$

The language generated by automaton G is defined as:

$$L_m(G) := \{ s \in L(G): f(x_0,s) \in Q_m \}. \quad (3)$$

For example for the eating scenario, looking into PSAs for eating scenario, the marked language would be a complete sequence of events starting from initializing the task up to the final events, which is eating and the task is finished.

Once the language of the system and automaton which generates this language is known, other safety properties of the system can also be analyzed. For example it can be determined if from a given state 'x' another state 'y' is reachable, or if in a given automaton a substring is possible or does not exist.

For modular design, the automaton model of different subsystems of the system can be combined together to build a larger system. These combinations can be accomplished by suitable binary operations on the automaton models. The most usual operation, which synchronizes two automata on common events, is called synchronous product. In this composition the common events in two subsystems are allowed to change the state of the system, provided they are included in the current active event sets of both subsystems.

There is also a complete synchronization which only lets the common events to be triggered, called product composition (or 'meet'). This is used when two specifications for subsystems need to be combined to find the complete specification for the two subsystems. These two or more specifications must be completely synchronized together.

V. Supervisory Control Theory

Supervisory control theory known as the Ramadge-Wonham (RW) framework was pioneered by them in the early 1980's [13, 14]. Since then many other researchers have also made contributed to its development. In this theory the plant is first modeled as a finite state automaton. Then using the automaton model of the specification for the plant, the latter will be controlled in a feedback loop. The most challenging step is constructing these two models for the real problem. A formal approach for modeling plant and specification is still an open area for research; owing to the computation complexity (especially for large systems) and model interpretation the application of RW (as with any theory) pose a challenge to the practitioners.

In this framework a possible set of enabled events is called control pattern. The supervisor is any function in the form:

$$S: L(G) \rightarrow 2^{\Sigma}. \quad (4)$$

The controlled behavior is defined as a language L(S/G), which is a subset of L(G) and has followings properties:

1. $\varepsilon \in L(S/G)$; (ε is an empty sting)
2. $(s \in L(S/G) \wedge \sigma \in S(s) \wedge s\sigma \in L(G)) \Rightarrow s\sigma \in L(S/G)$;
3. No other strings belong to L (S/G).

The marked language of the supervisor is (typically):

$$L_m(S/G) = L(S/G) \cap L_m(G) \quad (5)$$

The events in the system are divided to controllable and uncontrollable events.

$$\Sigma = \Sigma_c \cup \Sigma_u \quad (6)$$

The controllable events can be prevented from occurring by the supervisor; for instance by commands to an actuator. On the other hand, the supervisor cannot directly prevent the occurrence of uncontrollable events, such event triggered from environment, humans, by actuators (like finishing a task), data read from sensors or faults occurring in the system hardware.

A specification represented by language K which is a subset of marked language of system is said to be controllable if:

$$\overline{K}\Sigma_u \cap L(G) \subseteq \overline{K} \quad (7)$$

, in which $\overline{K}$ is the prefix closure of language K.

Then a marked non-blocking supervisor exists such that

$$L(S/G) = \overline{K} \quad (8)$$

For more detail about discrete-event systems and supervisory control theory the reader is referred to the related literatures.[13–15]

VI. Implementation of Approach

Case Study- Robot and Wheelchair interaction

Here we show the verification approach for one safety requirement in the system. The user can select different input devices to give commands to the robot arm to perform some tasks for him. He can also give commands through the human-machine interface to control the wheelchair movement. After FMEA and HAZOP analysis for the robot arm, one of the outputs of the analysis is: "owing to the possibility of collision with the environment and bringing the user to an unsafe situation, the synchronous movement of robot and wheelchair must be prevented". In other words, once the robot is manipulating, the wheelchair cannot move. While this requirement is already implemented in the software structure, it also needs to be added to the safety module and monitoring system. The remedy for this hazard is monitoring robot movement and wheelchair movement with separate external sensors and sending feedback about their readings.

For this problem we can write down the safety requirement in natural language: "Wheelchair cannot move while the robot arm is moving, and vice versa".

For the realization, the internal sensors of the robot arm are used to measure its movement. Also a motion sensor is mounted on the wheels to detect wheelchair movement. Based on these observations the safety module then decides when it is safe to start movement of the robot or the wheelchair. In other words it checks the preconditions necessary for the safe robot or wheelchair movement.

To build such a controller formally, we first model the subsystems, involved in this requirement, in automata. The robot arm can reach numerous numbers of states in each task; however the abstract model exhibiting its relevance for control design is enough. To this end it can be modeled as a 2-state automaton modeling receiving a start command from the controller and terminating its motion when the assigned task is finished. The wheelchair model is also one automaton with two states with two events: start moving and stop moving.

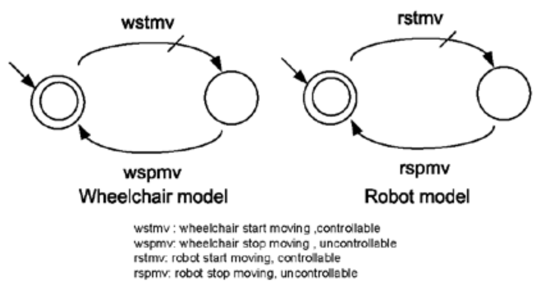
wstmv
rstmv
wspmv
Wheelchair model
rspmv
Robot model
wstmv : wheelchair start moving ,controllable
wspmv: wheelchair stop moving , uncontrollable
rstmv: robot start moving, controllable
rspmv: robot stop moving, uncontrollable

Figure 4. Robot's and wheelchair's movement model

In order to interpret the above mentioned safety requirement in automata we need to bring in two specifications. For the first specification let the start command to the robot arm, be issued only when the wheelchair is not moving. In the same way the second specification restricts the wheelchair movement while the robot is moving by prohibiting it from starting. These are shown in Fig. 5.

However these specifications consider only a fully autonomous system. We have to take into account the human interactins with the system. He can interact with system through human machine interface and activate commands for moving robot or those for moving wheelchair anytime.

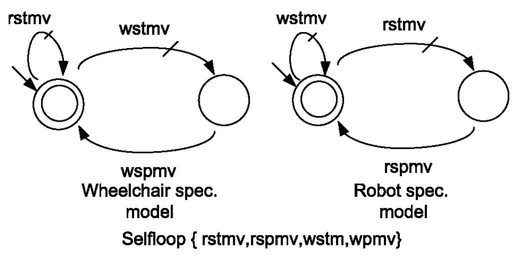

Figure 5. Robot's and wheelchair's movement safety specification

This can be abstractly modeled by two uncontrollable events enabled at each state of the system model. In other words the model of HRI would be a single automaton with two uncontrollable events 'crstmv','cwstmv' (due to the space limit we skip to draw this model that is only a single automaton). Also the admissible behavior in case of HRI must be added to the specifications. This is modeled by mutual exclusion of executing commands for the movement of wheelchair and robot issued by user; i.e. When the command for moving robot (wheelchair) is given, the other command moving wheelchair (robot) is ignored (Fig. 6). The self-loop of the other unrelated events must be considered in the models. The other important point is considering the uncontrollable events to avoid the uncontrollability in the final specification.

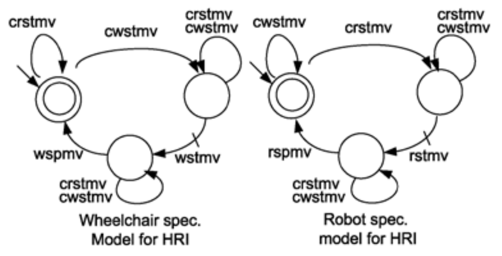

Figure 6. Robot's and wheelchair's movement safety specification for HRI

The specification for the whole system would be the product composition (meet) of specifications; the resulting automaton is shown in Fig. 7. However looking into the language of the specification we can see an undesired behavior; namely when two commands are received at the same time for moving wheelchair and moving robot, they can be executed in any order.

L (K) ={ε,'crstmv''cwstmv''wstmv',

'cwstmv''crstmv''rstmv',….}

The specification must be changed to remember this situation, namely the order of given commands. For this reason the states S2 must be spitted into two states. (It is not depicted here, due to the space limit). After this modification in specification the final calculated supervisor shown on Fig. 8 fulfills all above mentioned requirements.

Although what is given here as safety requirement is a simplistic example of verification, it serves to demonstrate how behavior of robot can be modeled formally and verified according to the safety requirements systematically, based on RW framework. The safety requirements are not always as

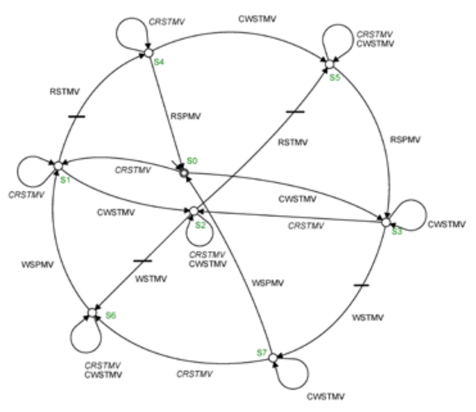

Figure 7. System specification

simple as given example and maybe more preconditions need to be verified before a safety action can happen (e.g. handling the situation when robot moves toward user while pouring a hot drink).

The application of supervisory control theory, apart from safety matters (correct execution order of events), can also be advantageous when the blocking problem becomes an issue to be solved. Blocking usually happens in circumstances when two tasks need one resource to be executed at the same time, for example, in FRIEND, when the user demands from the robot the execution of different tasks simultaneously.

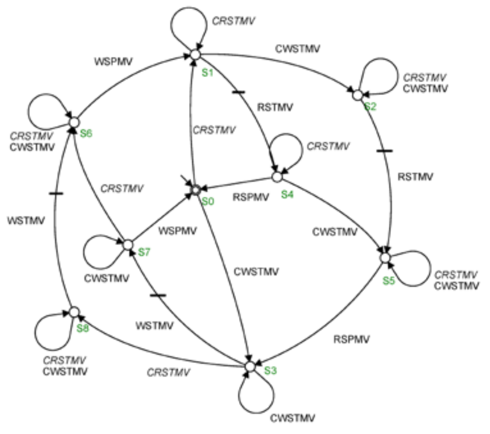

Figure 8. Supervisor

VII. Conclusion and Further Work

In this paper we discussed a stepwise approach for safety of or a care-providing robot system. This approach complies with the standards for safety of programmable automation systems. The novelty of the approach lies in the systematic verification of safety requirements for such a complex system that works in dynamic environment and operates semi-autonomously. In these systems the specification and safety requirements cannot be established straightforwardly as for industrial robotic applications.

We used a model-based verification, based on the RW framework, for the safety requirement's verification, in order to be sure about the correctness of the implemented safety controller. Taking advantage of supervisory control theory, we can confirm that the controller is correct by construction. For the case study we showed the implementation for a simple example in FRIEND. The techniques accompany all system development phases to ensure safety of the final designed system. They are applied iteratively parallel to the modifications in the system and design revisions, until the final design is achieved.

Acknowledgment

The authors would like to thank Professor W. M. Wonham for his helpful suggestions, encouragement and advice during preparation of this work.

References

[1] Antonio Bicchi, Michael A. Peshkin, J. Edward Colgate "Safety for physical robot-uuman interaction" in B.Siciliano, O. Khatib, Handbook of Robotics, page 1335-1348.

[2] IEC 61508 "Functinal safety of electrical/electronic/programmable electronic safety-related systems".

[3] Jeremie Guiochet, Damien Martin-Guillerez, David Powell,"Experience with a model-based user-centered risk assessment for service robots" , IEEE 12th International Symposium on High Assurance Systems Engineering, 2010.

[4] B. A. Brandin, "The real-time supervisory control of an experimental manufacturing cell", IEEE Transactions on Robotics and Automation, Vol. 12, No.1, February 1996.

[5] R. J. Leduc and W. M. Wonham, "Discrete event systems modeling and control of a manufacturing testbed", Canadian Conference on Electrical and Computer Engineering, Vol. 2, 1995.

[6] Jing Lui and Houshang Darabi "Ramadge-Wonham supervisory control of mobile robots: lessons from practice", IEEE International Conference on Robotics 8 Automation, 2002.

[7] Theunissen, RJM; Schiffelers, RRH; Beek, van, DA (Bert); Rooda, JE "Supervisory control synthesis for a patient support system", Eindhoven University of Technology, 2008.

[8] Martens, O. Prenzel, J. Feuser, and A. Gräser: "MASSiVE: multi-layer architecture for semi-autonomous service-robots with verified task execution"; Proceedings of 10th Int. Conf. on Optimization of Electrical and Electronic Equipments , vol 3, pp. 107-112; 2006; May, Brasov, Romania, ISBN 973-653-705-8.

[9] Oliver Prenzel: "Process model for the development of semi-autonomous service robots"; PhD Dissertation, University of Bremen; Shaker, 2009.

[10] L.Fotoohi: "Safety Analysis and Methods in Safety Critical Systems: Application in FRIENDII"; 2006; 28th Colloquium of Automation; Salzhausen; November,unpublished.

[11] Felix Redmill, Morris Chuddleigh, James Catmur, "System safety: HAZOP and software HAZOP" , John wiley & Sons ,1999.

[12] Leila Fotoohi, Axel Gräser "A supervisory control approach for safe behavior of service robot, case study: FRIEND"; 25th ACM Symposium on Applied Computing , March 22-26,2010, Sierre, Switzerland.

[13] W.M.Wonham,"Supervisory control of discrete systems", 2010-11, University of Toronto (available at http://www.control.utoronto.ca/DES).

[14] P. J. Ramadge and W. M. Wonham, "Supervisory control of a class of discrete event processes," SIAM J. Contr. Optim., vol. 25, no. 1, pp. 206-230, Jan. 1987.

[15] Christos G. Cassandrass, Stephane Lafortune, "Introduction to discrete event systems",Springer,1999.