Safety Science 84 (2016) 225–237

Contents lists available at ScienceDirect

Safety Science

journal homepage: www.elsevier.com/locate/ssci

Hazard analysis of human–robot interactions with HAZOP–UML

Jérémie Guiochet

University of Toulouse, UPS, LAAS-CNRS, Toulouse, France

ARTICLE INFO

Article history:
Received 16 July 2015
Received in revised form 20 November 2015
Accepted 14 December 2015

Keywords:
Hazard identification
Risk analysis
Robot safety
HAZOP
UML

ABSTRACT

New safety critical systems are about to appear in our everyday life: advanced robots able to interact with humans and perform tasks at home, in hospitals, or at work. A hazardous behavior of those systems, induced by failures or extreme environment conditions, may lead to catastrophic consequences. Well-known risk analysis methods used in other critical domains (e.g., avionics, nuclear, medical, transportation), have to be extended or adapted due to the non-deterministic behavior of those systems, evolving in unstructured environments. One major challenge is thus to develop methods that can be applied at the very beginning of the development process, to identify hazards induced by robot tasks and their interactions with humans. In this paper we present a method which is based on an adaptation of a hazard identification technique, HAZOP (Hazard Operability), coupled with a system description notation, UML (Unified Modeling Language). This systematic approach has been applied successfully in research projects, and is now applied by robot manufacturers. Some results of those studies are presented and discussed to explain the benefits and limits of our method.

1. Introduction

Besides the developments of well-known safety critical systems in aeronautics or transportation, new systems are about to appear in our everyday life: robots at home, at work, or in the hospitals (Royakkers and van Est, 2015). Such systems, will interact with users, and execute tasks in the vicinity or even in physical contact with humans. Hence, a failure of such complex systems may lead to catastrophic consequences for users which is a major obstacle to their deployment in real life. Most safety analysis techniques coming from the dependability (Avižienis et al., 2004) or risk management (ISO31000, 2009) domains could be used for such systems, but some specificities of robots limit their efficiency. For instance, the fact that robots move in unstructured and unknown environments makes the verification and validation (mainly through testing) non sufficient (it is impossible to guarantee that all main scenarios have been tested); the presence of users and complex non deterministic software (with decisional mechanisms) limit the use of quantitative risk analysis techniques; classical hazard analysis techniques are also not adapted to the complexity of human–robot interactions. Little work has been done about risk analysis for such systems, although it is a major challenge for robot certification (Mitka et al., 2012). Many robotics studies about estimation and treatment of collision risks exist (many references presented by Haddadin (2014)), but few are on risk analysis methods (Dogramadzi et al., 2014). The safety community has rarely addressed this issue, whereas we have been working on this for a decade (Guiochet and Vilchis, 2002; Guiochet et al., 2004).

Some robot manufacturers use directives (2006/42/EC, 2006) or standards (ISO13849-1, 2006) dedicated to machines, but they are not completely applicable, particularly when there is a human–robot physical interaction. Generic standards like IEC61508-5 (2010), are also hardly applicable due to uncertainties in the robot behavior (in this standard, fault correction through artificial intelligence is not recommended for safety integrity level SIL2 to SIL4). More recently, the standard ISO10218-1 (2011) for industrial robots that might share their workspace with humans, has been completed by the ISO13482 (2014). It is also important to note that such standards, do not cover other application domain robots. For instance, in the medical field, there is no robotic-specific standard, and the robots are considered as active medical devices such as defined in the 93/42/EEC (1993), and covered by ISO/FDIS14971 (2006) for risk management. In all those standards, classic risk management and design recommendations are proposed, but no specific guidelines for risk analysis techniques are presented.

E-mail address: jeremie.guiochet@laas.fr
URL: http://homepages.laas.fr/guiochet

http://dx.doi.org/10.1016/j.ssci.2015.12.017
0925-7535/

To cope with the previous issues, we suggest a hazard identification technique with the following objectives:

1. applicable from the very beginning of the development process,
2. includes human activity as a source of hazard,
3. provides guidance for analysts with list of guide words,
4. focuses on operational hazards, i.e., hazards linked with the robot tasks and interactions.

Among risk analysis techniques, the most widely used are Preliminary Hazard Analysis (PHA), Hazard Operability Analysis (HAZOP), Fault Tree Analysis (FTA), and Failure Mode, Effects, and Criticality Analysis (FMECA). The two first may be applied as hazard analysis at the very early steps of a development process, whereas FTA and FMECA are more dedicated to advanced steps, focusing more on reliability aspects. Thus, we chose to base our method on HAZOP, and to combine it with the system modeling language UML (Unified Modeling Language). This method developed at LAAS (Guiochet et al., 2010, 2013; Martin-Guillerez et al., 2010), has been successfully applied in several French and European projects (PHRIENDS, 2006–2009; SAPHARI, 2011–2015; MIRAS, 2009–2013) in collaboration with robot manufacturers (KUKA Robotics, AIRBUS Group and Robosoft). This paper synthesizes for the first time our work on HAZOP–UML, and proposes an analysis of the applications in these projects.

The remainder of this paper is structured as follows. Section 2 provides background on UML and HAZOP. In Section 3, we present the HAZOP–UML method, and in Section 4, results of several experiments are analyzed and discussed. In Section 5, related work on model-based safety analysis is compared to our approach. We conclude in Section 6 by outlining the benefits and limits of HAZOP–UML, and listing some future directions.

2. Background

2.1. Unified Modeling Language

UML (Unified Modeling Language) is a graphical notation, widely used in software and system engineering domains to support early steps of the development process. Its specification is available on the Object Management Group UML page.[1] The current version (UML 2), has thirteen diagrams, that could be classified in static diagrams (e.g., class diagram) and dynamic diagrams (e.g., use case, sequence and state machine diagrams). UML is a language, and not a method, as it is not specified in which chronological order each diagram must be used. But, use cases and sequence diagrams are typically used at the beginning of any project development. State machine diagrams are also widely used in reactive systems as robot controllers. Hence, we will present those three diagrams, focusing only in the elements we will use for our approach. One main pitfall using this language is to mix different levels of details in the same diagram. For instance, mixing some high level specifications with implementation constraints on the same diagram is error prone and also not recommended for the safety analysis. This is why we also put forward in this paper some modeling rules to avoid this pitfall and to guide the analysts.

As a running example, we will use some models of the case study MIRAS (2009–2013), an assistive robot presented Fig. 1, for standing up, sitting down and walking, and also capable of health-state monitoring of the patients. It is designed to be used in elderly care centers by people suffering from gait and orientation problems where a classic wheeled walker (or "rollator"), is not sufficient for patient autonomy. The robotic rollator is composed of a mobile base and a moving handlebar.

[1] www.uml.org: accessed 2015-05-15.

Fig. 1. MIRAS robot prototype during clinical investigation.

2.1.1. Use case diagrams

This diagram is the basic requirement UML model, presenting the system to analyze, the actors communicating with it, and the objectives for the use of the system: the use cases. The example of Fig. 2 only presents a subset of the complete use case diagram (15 use cases), and the two involved actors. In this diagram, the proposed services are to help the patient to stand up (UC02), deambulate (UC01), and sit down (UC03). The system is also able to detect physiological issues and trigger an alarm (patient heartbeat and fatigue, in UC08). We also represent that the system offers the profile learning facility (UC10). In some projects using UML the mechanical part of a robot is represented as a UML actor, and the system boundary (the box around use cases) defines the robot controller (including software and hardware). We do not recommend using such an approach to perform the hazard identification, indeed, the complete system has to be studied as a whole.

This diagram provides an expressive and simple mean to communicate between developers, analysts and users. This graphical representation is always completed with a textual description as in Fig. 3. Important information such pre and post conditions, and non-functional requirements are included. Use case diagram only represents functional requirements. Textual description of the normal, alternative and exception flows may also be presented with sequence diagrams as presented hereafter.

In the UML OMG standard, some relations may exist between use cases (mainly the relations *extend* and *include*) but we recommend not to use them, as they often lead to misunderstandings and to an unclear application of the HAZOP–UML method. In order to prepare the HAZOP–UML study, an extract from the use case textual description should be done, with only the pre and post conditions, and also the invariants coming from safety properties in the "Non functional requirements" category. An example of such a table is given in Fig. 4 for the UC02 of the MIRAS running example.

2.1.2. Sequence diagrams

Fig. 5 shows a sequence diagram, describing a possible scenario, which is actually an instance of an UML use case. This diagram shows a nominal scenario for the UC02. Other scenarios are possible for the UC02, like alternative flow of events (e.g., the patient releases the handles while she is standing up). This second scenario will be represented with another sequence diagram (not presented here). The expressiveness of such diagram is well adapted to represent human–robot interactions, and have proven to be useful while discussing with other stakeholders who are not experts in this language (doctors, mechanical engineers, etc.). All messages

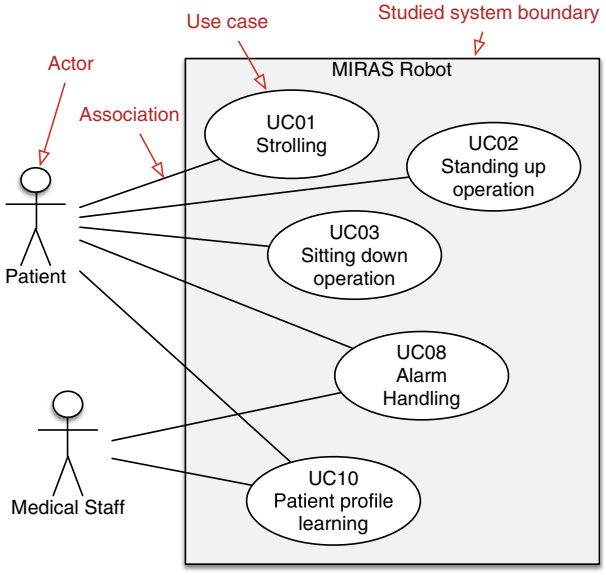

Fig. 2. Extract of MIRAS use case diagram from Guiochet et al. (2013).

exchanged between actors and the system are represented along their lifelines. In our case three types of messages are used:

- *indirect interaction* through robot teach pendant (hardware or software interfaces),
- *cognitive interaction*, e.g., gesture or voice/audio signals are exchanged,
- *physical interaction*, direct contact between physical structure of the robot and the user.

In the example of Fig. 5, the messages are all physical contacts, so we did not add this information which can be done using a UML annotation. In UML, a sequence diagram is a representation of an *Interaction*, where actors and the system (*Lifeline*), send some *Message* that might have *Arguments* and *Constraints*. Here the message *2:initiateStandingUp* is sent to the robot with a *force* exerced on the handles. As the time increases from top to bottom, each message has a *sending* and *receiving occurrence event*. It is also possible to represent on a message a *guard condition* for its execution (e.g., *[end of course]* of message 4).

We recommend not to use the UML2 *fragments* (loops, alternatives, etc.) but to rather use several diagrams to represent alternatives flows for instance. We also recommend to draw a *system sequence diagram*, i.e., representing only the actors and the system, and not the internal objects of the system.

2.1.3. State machines

These deterministic automata diagrams are based on the statecharts proposed by Harel (1987). A state machine is given for all the objects with a dynamic behavior. An example is given in Fig. 6 where the considered object is the MIRAS robot controller. A transition is represented with an arrow between a start state and a destination state, and can have the following facultative form of `event [guard]/ action()`, where:

- `event` is the trigger element of the transition, which could be:
  - *signal event*: asynchronous external event (e.g., button pressed, voice command)
  - *call event*: reception of an operation called by another object of the system
  - *change event*: a change of a boolean variable based on the estimation of a system variable
  - temporal event (*after* or *when*): expired duration *after(<duration>)*, or absolute time *when(date=⟨date⟩)*
- `guard` is a condition estimated only if the event occurs
- `action` is a list of actions performed instantly when the transition is triggered

In this method we use state diagrams to specify at the beginning of a project, the different operational modes of the robot. This diagram is also useful for the detailed design and implementation of the robot controller, which is out of the scope of this paper.

2.2. HAZOP

HAZOP (HAZard OPerability) is a collaborative hazard identification technique, developed in the 70's, and is widely used in the process industries. It is now standardized by the standard IEC61882 (2001). Its success mainly lies in its simplicity and the possibility to apply it at the very beginning of the development process. It is also adaptable to the formalism used to describe a system as presented in the standard DefStan00-58 (2000). HAZOP does not consider failure modes as FMECA, but potential deviations of the main parameters of the process. For each part of the system, the identification of the deviation is systematically done with the conjunction of:

- system parameters, e.g., in the case of an industrial process: `temperature`, `pressure`, `flow`, etc.,
- guide words like: `No`, `More`, `Less` or `Reverse`.

The role of the guide word is to stimulate imaginative ideas and initiate discussions. A proposed list of guide words is given in Fig. 7. For instance, we can have the following conjunctions (e.g., for a chemical process):

Use Case Name		[Name of the use case]
Actors		[An actor is a person or other entity external to the system being specified who interacts with the system and performs use cases to accomplish tasks]
Preconditions		[Activities that must take place, or any conditions that must be true, before the use case can be started]
Normal Flow	Description	[User actions and system responses that will take place during execution of the use case under normal, expected conditions.]
	Postconditions	[State of the system at the conclusion of the use case execution with a normal flow (nominal)]
Alternative flows and exceptions		[Major alternative flows or exceptions that may occur in the flow of event]
Non functional requirements		[All non-functional requirement: e.g., dependability (safety, reliability, etc.), performance, ergonomic]

Fig. 3. Use case textual description template.

Use case name	UC02. Standing up operation
Abstract	The patient stands up with the help of the robot
Precondition	The patient is sitting down The robot is waiting for the standing up operation Battery charge is sufficient to do this task and to help the patient to sit down The robot is in front of the patient
Postcondition	The patient is standing up The robot is in admittance mode
Invariant	The patient holds both handles of the robot The robot is in standing up mode Physiological parameters are acceptable

Fig. 4. UC02 use case textual description with pre, post conditions and invariant.

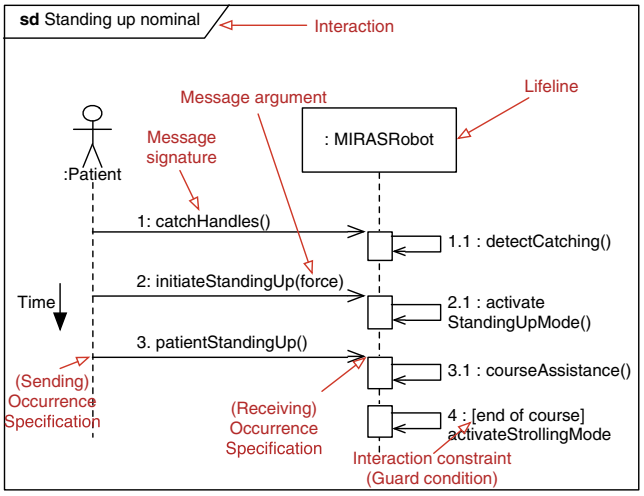

Fig. 5. Sequence diagram for the nominal scenario of *UC01: Standing up operation*.

- `Temperature` ⊗ `More` → `Temperature too high`,
- `Flow` ⊗ `Reverse` → `Product flow reversal`.

For each deviation, the procedure is then to investigate causes, consequences and protection, and produce document usually in a table form (similar to FMECA), with columns like: Guide word, Element, Deviation, Possible causes, Consequences, Safeguards, Comments, Actions required, etc.

Even though the HAZOP method has proved to be efficient, the results may be questionable when the boundary of the study is too vast or not well defined, or when the guide words are either too numerous or too limited for the analysis to be relevant. Another limitation is that there is no systematic method to adapt the guide words to the considered domain, so adaptation depends on the expertise of the initiators of the method. Additionally, the HAZOP method needs the allocation of human resources and suffers from combinatorial explosion when too many deviations are considered or when the analysts go into too much details. Hence, the success of a HAZOP study depends greatly on the ability of the analyst and the interactions between team members. The choice of the considered "system parameters", is of high importance, because all the study relies on it. The HAZOP–UML method proposed in this paper is aimed at providing more guidance to analysts to identify which parameters they have to consider.

3. HAZOP–UML

One main issue when applying HAZOP is to identify the system parameters. We propose to use UML to partition and describe the system. The considered parameters will be then some elements of the UML diagrams. In this section we will give guidelines to identify those parameters, and the associated guide words to identify possible deviations. This work is the result of several applications and refinement, and may also be completed or modified by the analysts. Even if our objective is to propose a systematic approach, it is important to note that HAZOP–UML does not identify all hazards. First because no single hazard identification technique is actually capable of finding all the hazards (Cantrell and Clemens, 2009), and also because we will focus on the identification of the operational hazards, i.e., hazards linked to the human–robot interactions, through dynamic models of the system.

As already presented, we propose to focus on the three main dynamic UML diagrams: use case, sequence and state diagrams. For those diagrams, some generic deviations are presented in Section 3.1. The whole process is then introduced in Sections 3.2, and 3.3 presents a prototype of a tool for HAZOP–UML.

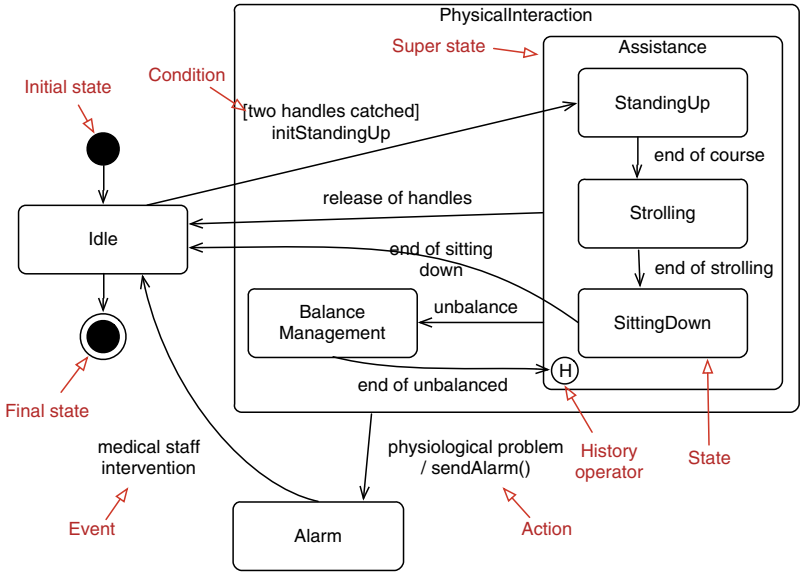

Fig. 6. Simplified version of MIRAS state machine.

Guideword	Interpretation
No/None	Complete negation of the design intention / No part of the intention is achieved and nothing else happens
More	Quantitative increase
Less	Quantitative decrease
As Well As	All the design intention is achieved together with additions
Part of	Only some of the design intention is achieved
Reverse	The logical opposite of the design intention is achieved
Other than	Complete substitution, where no part of the original intention is achieved but something quite different happens
Early	Something happens earlier than expected relative to clock time
Late	Something happens later than expected relative to clock time
Before	Something happens before it is expected, relating to order or sequence
After	After Something happens after it is expected, relating to order or sequence

Fig. 7. Guide words list adapted from IEC61882 (2001).

3.1. Guide words

Instead of using the term "parameter" usually used in HAZOP studies, entities and attributes of UML elements are introduced in this section. Then for each element, a generic interpretation for a deviation is proposed. This analysis is based on the UML metamodel (OMG-UML2, 2007). The selected UML entities are: use case, message, state machine.

3.1.1. Guide words for use cases

Fig. 8 presents an extract from the UML metamodel, focusing on a use case. The UML class diagram notation is used to represent this metamodel. This diagram specifies that a use case may be composed of 0 to several (noted as "*") *Behaviors*. Indeed, a use case is usually composed of a nominal behavior (or nominal scenario), and several exceptions. Each Behavior may have 0 to several *Constraints*, which are pre and post conditions. As introduced in Section 2.1, we add to this metamodel one constraint to the *Behavior* of a *UseCase*: the invariant. Indeed, when an analyst studies all possible deviations, we would argue that the non-functional requirements, which may be safety invariants (e.g., robot velocity should not exceed 20 cm/s) have to be taken into account. We should then consider that the attributes of a use case are: preconditions, postconditions, and invariants, which are all UML *Constraints*. For this reason, we apply the classical HAZOP guide words to the concept of constraint in a generic way and formulate an interpretation to guide the analyst. The result of this work is given in Table 1. Only six guide words were interpreted, we also remove many redundancies in the interpretation. Let consider the example of use case UC02 ("standing up operation") described in Fig. 4. The precondition "The robot is in front of the patient" combined with the guide word "No", leads to the following scenario: the patient tries to standup while the robot is not properly positioned. This might induce excessive effort for the patient and a fall which is catastrophic in our case study. If we consider this use case, with 9 conditions and 6 guide words, this leads to 54 possible deviations. Moreover, the interpretation of a guide word may change from an analyst to another. Nevertheless, the objective is to eventually identify all hazards, and the original guide word used for the identification is of no real importance.

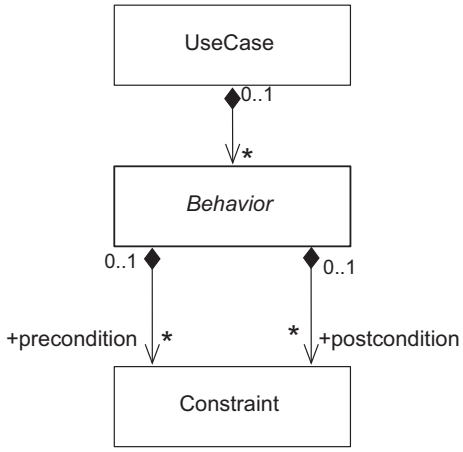

Fig. 8. Reduced concepts for specification of use cases.

Table 1
Guide words list and generic interpretation for use cases.

Entity = Use Case		
Attribute	Guideword	Interpretation
Preconditions/postconditions/ invariants	No/none	The condition is not evaluated and can have any value
	Other than	The condition is evaluated true whereas it is false, or vice versa
	As well as	The condition is correctly evaluated but other unexpected conditions are true
	Part of	The condition is partially evaluated Some conditions are missing
	Early	The condition is evaluated earlier than required for correct synchronization with the environment
	Late	The condition is evaluated later than required for correct synchronization with the environment

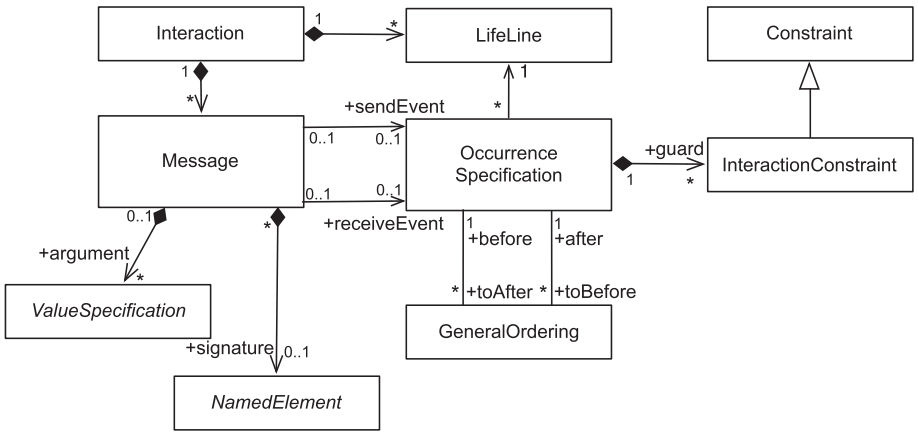

Fig. 9. Reduced metamodel for interactions in UML (sequence diagrams) extracted from OMG-UML2 (2007).

3.1.2. Guide words for sequence diagrams

Sequence diagrams are one of the graphical representation of the *Interaction* UML concept. It is composed of *Lifelines* exchanging *Messages*. This is represented in the simplified metamodel in Fig. 9. This metamodel extracted from OMG-UML2 (2007) has very little differences with the version (OMG-UML2, 2011), so we kept this representation which is simpler, and expressive enough for its use in HAZOP–UML. Based on this metamodel, we define five attributes for the Message:

1. General Ordering: the general order of the messages within the interaction.
2. Send/receive event timing: event related to the clock time.
3. Lifelines: send and receiving lifelines of a message.
4. Interaction Constraint: guard condition on a message.
5. Message argument: parameters of a message.

Other elements of the metamodel have not been considered, as we did not find any possible deviation or we intentionally avoid to consider them because they would have produced redundant possible deviations (interested reader may find more about UML *interaction fragments in* OMG-UML2 (2011). The resulting table for the generic deviations and their interpretation is given in Table 2. In tOMG-UML2 (2011) the following explanation is given: "A *GeneralOrdering* represents a binary relation between two *OccurrenceSpecifications*, to describe that one *OccurrenceSpecification* must occur before the other in a valid trace. This mechanism provides the ability to define partial orders of *OccurrenceSpecifications* that may otherwise not have a specified order." This could be interpreted as the fact that in some diagrams a *GeneralOrdering* relation can be added as a constraint. But in a sequence diagram, the physical position of the message already specifies an order for a valid trace. Hence, in our approach, we will interpret a sequence diagram as a valid trace, i.e., with a valid specified ordering of the message. This trace is descriptive (and not prescriptive like the state machine), but changing the ordering may lead to hazardous interactions.

3.1.3. Guide words for state machines

The same approach was used for the state machines. This diagram can also be used for detailed system design, which may lead to a combinatory explosion for the HAZOP analysis. Hence, we reduced the number of concepts to a very simple version as presented in Fig. 10. Note that we replaced in this model the original class *Behavior* by *Action*. Actually, in UML an action is the fundamental unit of behavior specification, which can be associated to a state or a transition. We only consider in this method the action on transitions, which is sufficient to express relevant behavior. Of course, our proposal could be extended to the complete state machine metamodel, to identify all possible deviation at design time, but this is out of the scope of our method.

According to this metamodel, the resulting table for possible deviations is given in Table 3. In order to provide more guidance, we also point out in this table if the transition is triggered or not for some deviations.

3.2. HAZOP–UML process and outputs

According to the previous tables, the process to perform HAZOP–UML is the following procedure: for each entity, for each attribute, for each guide words, identify one or several possible deviations and analyze it (them). A graphical view is given in Fig. 11. The analysis of the deviation may include the identification of possible causes and consequences. Depending on the project, it is also possible to evaluate the risk (consequence of the deviation effect, and likelihood of the considered deviation). Nevertheless, this information is usually too complex or impossible to obtain. On the contrary, such analysis always includes identification of recommendations to treat the deviation or its causes or it consequences (prevention and protection means). To establish such a study, the columns of a table as in Fig. 12 are given hereafter:

1. Entity: the UML element on which the deviation is applied (here UC02 is the same for all the table so it is in the head of the table).
2. Line number: for traceability (UCx.line_number).
3. Attribute: the considered attribute (e.g., a use case precondition).
4. Guide word: the applied guide word.
5. Deviation: the deviation resulting from the combination of the entity attribute and the guide word based on Tables 1–3.
6. Use Case Effect: effect at the use case level.
7. Real World Effect: possible effect in the real world.
8. Severity: rating of effect of the worst case scenario in the real world.
9. Possible Causes: possible causes of the deviation (software, hardware, human, etc.).
10. Safety Recommendations for prevention or protection.
11. Remarks: explanation of analysis, additional recommendations, etc.
12. Hazard Numbers: real world effects are identified as hazards and assigned a number, helping the users to navigate between results of the study and the HAZOP–UML tables.

Table 2
Guide words list and generic interpretation for sequence diagram messages.

Entity = Message		
Attribute	Guideword	Interpretation
General ordering	No	Message is not sent
	Other than	Unexpected message is sent
	As well as	Message is sent as well as another message
	More than	Message sent more often than intended
	Less than	Message sent less often than intended
	Before	Message sent before intended
	After	Message sent after intended
	Part of	Only a part of a set of messages is sent
	Reverse	Reverse order of expected messages
Send/receive event timing	As well as	Message sent at correct time and also at incorrect time
	Early	Message sent earlier than intended time
	Later	Message sent later than intended time
Lifelines (receiving and sending objects)	No	Message sent to but never received by intended object
	Other than	Message sent to wrong object
	As well as	Message sent to correct object and also an incorrect object
	Reverse	Source and destination objects are reversed
	More	Message sent to more objects than intended
	Less	Message sent to fewer objects than intended
Interaction constraint (message guard condition)	No/none	The condition is not evaluated and can have any value
	Other than	The condition is evaluated true whereas it is false, or vice versa
	As well as	The condition is well evaluated but other unexpected conditions are true
	Part of	Only a part of condition is correctly evaluated
	Late	The condition is evaluated later than correct synchronization with the environment
Message arguments (parameters)	No/None	Expected parameters are never set/ returned
	More	Parameters values are higher than intended
	Less	Parameters values are lower than intended
	As Well As	Parameters are also transmitted with unexpected ones
	Part of	Only some parameters are transmitted
		Some parameters are missing
	Other than	Parameter type/ number are different from those expected by the receiver

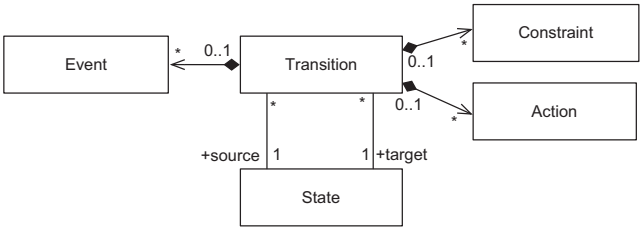

Fig. 10. Adapted UML metamodel of state machine.

In Fig. 12 given example, a precondition of UC02 (previously presented in Fig. 4) is analyzed using the guide words No and Other than. It leads to identify the hazard HN6 (Fall of the patient due to imbalance caused by the robot).

The resulting documents are the tables as the raw artefacts, but also:

- a concatenated list of identified hazards,
- a list of hypotheses made to perform the analysis, which need to be confirmed by domain experts to validate the study,
- a list of safety recommendations.

All those documents reference each others using numbered labels for lines, hazards (HN), recommendations (Rec), and hypothesis. Examples of a hazard table and recommendation list are given in Figs. 13 and 14. As an example, recommendation Rec2 from Fig. 14, covers hazards HN6 (fall of the patient), and has been formulated in the HAZOP table UC02 line 15 (UC02.15).

3.3. A tool for HAZOP–UML

To ease the analysis of complex systems, we developed a prototype of a tool to support the method. It helps to manage the combinatorial aspects of the HAZOP method by maintaining consistency between UML models and HAZOP tables and by providing document generation and management features. The tool is built as an Eclipse plugin (www.eclipse.org) using the Graphical Modelling Framework (GMF). In this tool presented in Fig. 15, the analyst can draw UML use case and sequence diagrams. Using guide word templates, HAZOP tables are automatically generated, ready to be filled out by the analyst using choice lists.

The list of guide words, the list of columns and the list of severities are editable using the main project view. Using the template, the analyst can add a line in the table by selecting a message, and then select applicable deviations and fill in the corresponding columns. When completing the table, the recommendation list and corresponding hazards are automatically generated in the project view. The toolbox of the HAZOP guide words allows deviations to be added (for example, several deviations for the same keyword). Finally a report in HTML can be generated consisting of HAZOP tables, UML diagrams, and hazards, recommendations and hypotheses lists.

4. Experiments and results

This section provides results of the experimentation of HAZOP–UML on three robotic applications developed within the following projects:

- ANR-MIRAS (Multimodal Interactive Robot of Assistance in Strolling) (MIRAS, 2009–2013) an assistive robot for standing up, sitting down and strolling already presented in Section 2.1.
- FP6-PHRIENDS (Physical Human–Robot Interaction: depENDability and Safety) (PHRIENDS, 2006–2009). The system is a mobile robot with a manipulator arm. The considered environments are workshops and factories with human workers.

Table 3
Guide words list and generic interpretation for state machines.

Entity = State machine		
Attribute	Guideword	Interpretation
Destination state	Other than	The transition leads to another state than expected
Transition	No/none	The transition is not triggered when intended
	Never	The transition is not triggered because the event never occurs or the condition is never met
Event	No/none	The transition is triggered while the event does not occur
	Other than	**transition not triggered** : the transition is not triggered when the event occurs **transition triggered** : the transition is triggered when another event occurs
Condition	No/none	The condition is not evaluated and can have any value, the transition is triggered
	Other than	**transition not triggered** : the condition is evaluated false whereas it is true, the transition is not triggered **transition triggered** : the condition is evaluated true whereas it is false, the transition is triggered
	As well as	The condition is well evaluated but other unexpected conditions are true, the transition is triggered
	Part of	Only a part of condition is correctly evaluated, the transition is triggered
	Early	The condition is evaluated sooner than required, the transition is triggered
	Late	The condition is evaluated later than required, the transition is triggered
Action	No/none	The transition is not triggered, there is no action
	Other than	The transition is triggered but an action other than intended takes place
	As well as	The transition is triggered, the action as well as an unexpected action take place
	Part of	The transition is triggered but only a part of action takes place
	Early	The transition is triggered but the action takes place sooner than correct synchronization with the environment
	Late	The transition is triggered but the action takes place later than correct synchronization with the environment
	More	The transitions is triggered but the result of the action, if quantifiable, is too high
	Less	The transitions is triggered but the result of the action, if quantifiable, is too low

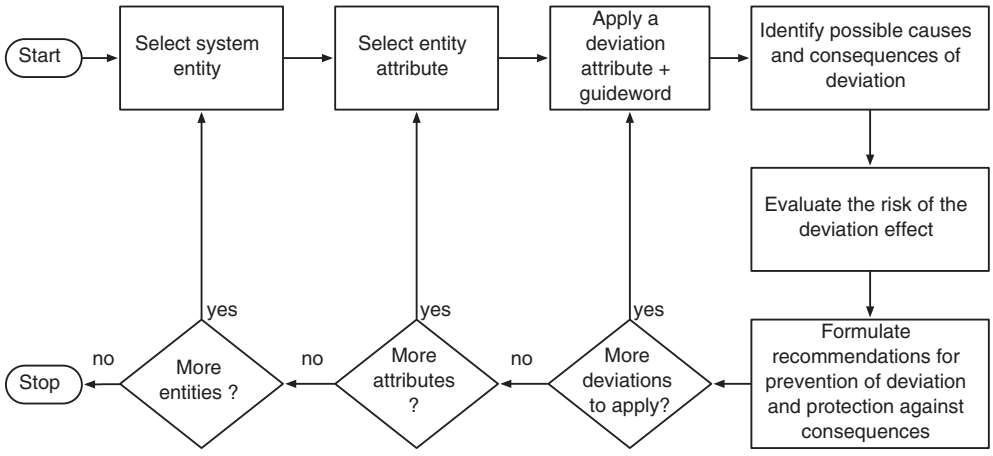

Fig. 11. HAZOP–UML process.

Project: MIRAS **Date:** 04/08/2009
HAZOP table number: UC02 **Prepared by:** DMG
Entity: UC02.Standing up operation **Revised by:** JG

Line Number	Attribute	Guide word	Deviation	Use Case Effect	Real World Effect	Severity	Possible Causes	Safety Recommandation	Remarks	Hazard Num.
UC02.15	Battery charge is sufficient to do this task and to help the patient to sit down (precond)	No/ none	Battery charge is too low but the robot starts the standing up operation	The robot interrupts its movement (standing up or walking)	Loss of balance or fall of the patient	Serious	HW/SW Failure Specification error	Worst-case electrical consumption must be evaluated beforehand. Take the lower bound of the battery charge estimation	If the robot stops during standing operation, the most probable scenario is that the patient will fall back on the seat.	HN6
UC02.16		Other than	Battery charge is high enough but the robot thinks otherwise	Robot refuses to start stand up operation	Patient is confused	None	HW/SW Failure Specification error	None		

Fig. 12. HAZOP–UML table extract.

Num.	Hazard	Severity	References
HN4	Fall of the patient without alarm or with a late alarm	Severe	UC13.SD01.29
HN5	Physiological problem of the patient without alarm or with a late alarm	Severe	UC03.SD02.57
HN6	Fall of the patient due to imbalance caused by the robot	Severe	UC12.SD01.19,30
HN7	Failure to switch to safe mode when a problem is detected. The robot keeps moving	Severe	UC12.SD01.62,89
HN1	Incorrect position of the patient during robot use	Serious	UC13.SD01.1,2,3

Fig. 13. Hazard list extract.

Num.	Safety recommandation	Hazard Num.	References
Rec1	The standing-up profile should be validated by a human operator	HN8, HN12	UC03.SD02.91,96
Rec2	Worst-case electrical consumption must be evaluated beforehand (and display of the mean battery time left by the robot)	HN6	UC02.15
Rec22	Send regularly a network heartbeat from the robot to the medical staff control panel. Launch alarm on time-out.	HN6	UC01.SD1.15,24
Rec31	Safety margins should determined for maximum and minimum height of the robot (monitoring is required)	HN8	UC03.SD02.91

Fig. 14. Recommendation list extract.

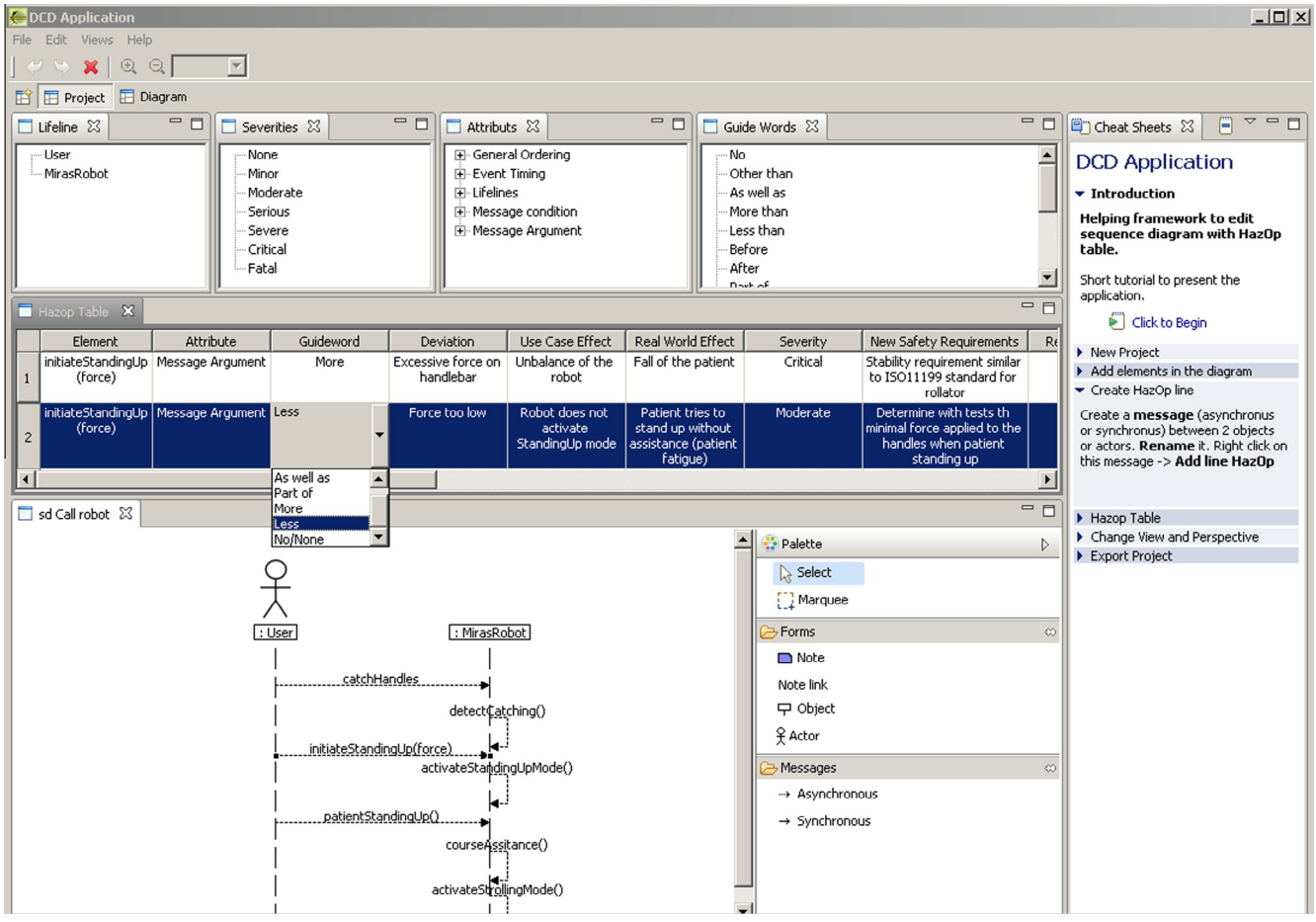

Fig. 15. Main view of the tool to support the HAZOP–UML method.

Collaborative work between a human and a robot is possible (e.g., the robot can give an object to the human). The arm is the KUKA Light Weight Robot (LWR), a seven degrees of freedom arm which contains torque and motor position sensors. The mobile base is the KUKA omnirob product.

- FP7-SAPHARI (Safe and Autonomous Physical Human-Aware Robot Interaction) (SAPHARI, 2011–2015). As in PHRIENDS, an Industrial coworker operates in a manufacturing setting accessible to human workers. The mobile manipulator may encounter humans while moving between the different workstations because the operation area is freely accessible to human workers. It takes and places part boxes on shelves, work stations, or on the robot base in order to convey them. The robot navigates autonomously in its operation area. When the robot encounters unexpected or difficult situations the worker might intervene and help by giving the robot direct haptic instructions.

For all three experiments, we followed the same procedure. We recruited analysts (an engineer for PHRIENDS, a postdoctoral for MIRAS, and a Phd student for SAPHARI), who were trained in our laboratory to HAZOP–UML. As a first step, they were in charge of modeling the UML diagrams, and validate them with robotic and domain experts (for instance in MIRAS, validation was also performed by doctors from the hospitals of the project). A second step was the deviation analysis performed only by the recruited analyst, followed by a revision by another member of our laboratory already trained to HAZOP–UML. Then, the resulting hazard and recommendation lists were discussed and validated by the robotic and domain experts. Quantitative data (e.g., working time or numbers of deviations) and qualitative data (e.g., traceability or modifiability) coming from these experiments are presented in this section, and structured according to the following properties:

- Applicability: we estimated the resources needed for the application of HAZOP–UML.
- Guide words relevance: this is a critical point of the method as all the results will depend on the ability of those guide words to guide the analyst.
- Validity: we compared results from a Preliminary Hazard Analysis to HAZOP–UML to assess its validity.
- Usability: some benefits and limits of HAZOP–UML while using it.

4.1. HAZOP–UML applicability

Classic HAZOP is usually applied in collaborative workshops, involving many partners to maximize the chances of study completeness. On the contrary, HAZOP–UML can be applied by a single analyst and then validated by experts. This comes from the fact that the study is always based on a UML model, which has been done in collaboration with stakeholders (e.g., robotic engineers or medical staff). The fact that their knowledge has been captured by UML models, makes the safety analyst task more independent from domain experts. Of course, during the analysis several questions arise, and hypotheses need to be made to carry out the analysis. They need then to be validated by the experts (this is why we propose to produce a hypotheses list).

Considering that a single analyst can perform most of the work, we also evaluate the effort to perform the complete analysis. Numbers are given in Table 4 for the three robotic projects. The state-machine version of HAZOP–UML has only been applied to MIRAS and statistics are presented in Table 5.

For the three projects, the complexity was nearly the same (between 39 and 54 use case conditions, and 91 and 122 messages in sequence diagrams). For each project one analyst has been recruited. Those three analysts were a post-doctoral, an engineer, and a Dr-engineer. "Analyzed deviations" stands for the number of deviations the analyst has considered, but only a part of them leads to an 'Interpreted deviations".

Table 4
Statistics for the application of HAZOP–UML for the three projects.

	PHRIENDS	MIRAS	SAPHARI
Use cases	9	11	15
Conditions	39	45	54
Analyzed deviations	297	317	324
Interpreted deviations	179	134	65
Interpreted deviations with recommendation	120	72	50
Sequence diagrams	9	12	16
Messages	91	52	122
Analyzed deviations	1397	676	2196
Interpreted deviations	589	163	87
Interpreted deviations with recommendation	274	85	36
Number of hazards	21	16	28

Table 5
Statistics for the application of HAZOP–UML State-machine only to MIRAS.

	MIRAS
State machine diagram	1
States	9
Transitions	19
Analyzed deviations	215
Interpreted deviations with recommandation	161

The resulting numbers show that no combinatory explosion happened, and less than 0.5 man-month was necessary for each study. Few iterations for table updates were needed (between 2 and 3). The presented tool in Section 3.3 was under development during those three projects, so we used a classic spreadsheet software with templates and macros. The cross checking between HAZOP tables and UML diagrams was then done by hand, which is clearly a limit that we want to reduce with our tool. Same conclusions were drawn for the state machine study, which was only applied to the MIRAS project (Table 5). However, those three projects were successful regarding the applicability of our method.

4.2. HAZOP–UML guide words relevance

For all projects, statistics of guide word usage have been made. The results of PHRIENDS project are presented in Tables 6 and 7. A first remark is that most of the guide words have been used by the analyst except in some special cases. The lifeline attribute is particularly useful when the robotic system is communicating with different actors (e.g., other robots), which was not the case in our project. The PHRIENDS UML diagrams also did not include any constraint on the messages, so the "Interaction constraints" guide words weren't used either in our case study. The guide word "Less than" (Message sent less often than intended) was also not used, as no constraint on frequency for messages was specified in the UML diagrams. The analyst also considered that "Part of" (only a part of a set of message is sent) was not relevant, because the level of description of UML diagram did not allow to consider parts of a message (as it may be the case with complex message sending with long protocol). Nevertheless, we chose to keep these guide words as in some special cases they would be applicable.

Another result, which is not presented here, is the redundancy of the hazards found, with different guide words. This is actually not an issue, because our main objective is to find a list of hazards, whatever guide word used to identify it. To determine if the guide

Table 6
Sequence diagram guide words utility in PHRIENDS.

Message attributes	Guidewords	Deviations	Interpretations
1. General Ordering	No	91	75
	Other than	97	25
	As well as	91	13
	More than	91	7
	Less than	0	0
	Before	92	32
	After	91	15
	Part of	0	0
	Reverse	91	43
2. Message timing	Early	91	28
	Later	91	28
3. Lifelines	Not applicable in our case study, which considers only a single robot (and a single human)		
4. Interaction Constraint	No constraint were specified in the UML models		
5. Message arguments	No/none	91	59
	More	91	52
	Less	91	62
	As well as	71	2
	Part of	95	31
	Other than	112	98

Table 7
Use case guide words utility in PHRIENDS.

Use case attributes	Guidewords	Deviations	Interpretation
Conditions (39) (pre/post/inv)	No/none	42	39
	Other than	95	95
	As well as	41	23
	Part of	40	10
	Early	40	9
	Late	39	3

words list is not limiting, we only rely on the results of the application on the three projects. A formal demonstration is actually impossible, and as already discussed, no single hazard identification technique is actually capable of finding all the hazards. We thus consider that in order to propose a systematic approach, the selected guide words are sufficient to identify all the major hazards.

4.3. HAZOP–UML validity

Table 8 presents two results for validity. First, this study shows that all hazards found during the PHA (Preliminary Hazard Analysis), done by collaborative workshop between a safety analyst and robotic experts, were also identified during HAZOP–UML (performed by the analyst), and that new hazards were also found. The fact that all scenarios of use were modeled in UML significantly improves the analysis. For instance, the hazard HN11 (Disturbance of medical staff during an intervention), was only identified during use case analysis, and never mentioned during the PHA, whereas it is highly relevant in case of emergency intervention.

The second analysis presented in this Table shows that use cases (UC) and messages (Seq) analysis are complementary, whereas state machine analysis has a redundant contribution for hazard identification. For instance, HN4 identified 11 and 13 times during use case and sequence diagrams analyses, has been identified 32 more times during state machine analysis. Nevertheless, we believe that state machine analysis is also interesting to identify more sources of deviations that could be used in other risk analysis methods, and also provide safety recommendations which are different from use cases and messages ones.

4.4. HAZOP–UML usability

A major advantage of HAZOP–UML lies in its simplicity. Indeed, UML models have been simplified to be easily understandable by non experts without reducing its expressiveness. HAZOP is also an intuitive method. Several engineers from different domains (electronics, computer science or risk management) have been trained to the method in few days.

HAZOP–UML is completely integrated and consistent with the development process. Indeed, same UML diagrams were used in the projects, to define the scenarios. This helped us for each iteration in the development process to easily update the HAZOP tables. This traceability is an important issue in safety analysis methods, which are usually applied once due to the cost to apply them.

Among HAZOP–UML limitations, we remind that HAZOP–UML is focusing on operational hazards (linked with the robot tasks). We thus do not consider "machine" hazards already defined in many standards, like electrocution, explosion, etc. As already mentioned, this method should be completed by other hazard analysis techniques. A second limitation is the fact that the UML models and HAZOP tables do not explicitly mention the environment conditions of execution. For instance, a similar scenario but with high or low level of light might change the deviations and their consequences. It is still an open issue and an integration in the UML models would be an interesting direction. Last but not least, the HAZOP–UML has the same drawback as other risk analysis methods, which is a difficult determination and expression of the hazard because of the fuzziness of a hazard definition ("potential source of

Table 8
Hazard list and occurrences in PHA and HAZOP–UML in MIRAS.

Num	Description	PHA	HAZOP–UML		
			UC	Seq.	State machine
HN1	Incorrect posture of the patient during robot use	2	4	3	4
HN2	Fall of patient due to imbalance not caused by the robot		29	27	30
HN3	Robot shutdown during its use	1	2		5
HN4	Patient falls without alarm or with a late alarm		11	13	32
HN5	Physiological problem of the patient without alarm or with a late alarm		15	10	
HN6	Fall of the patient due to imbalance caused by the robot	10	51	37	10
HN7	Failure to switch to safe mode when a problem is detected. The robot keeps on moving		8		
HN8	Robot parts catching patient or clothes	3	5	4	
HN9	Collision between the robot (or robot part) and the patient	2	14	14	
HN10	Collision between the robot and a person other than the patient		5	14	2
HN11	Disturbance of medical staff during an intervention		1		
HN12	Patient loses his/her balance due to the robot (without falling)	11	1	70	1
HN13	Robot manipulation causes patient fatigue	12	1	53	21
HN14	Injuries of the patient due to robot sudden movements while carrying the patient on its seat			3	
HN15	Fall of the patient from the robot seat	2	10	12	
HN16	Frequent false positive alarms (false alarm)			3	

harm", from ISO/IEC-Guide51 (1999)) which may designate both a cause or a consequence. Three columns in the HAZOP table can represent a hazard: deviation, use case effect, real word effect. In many tables, we found that some real word effects were already mentioned as use case effects in other HAZOP table lines. We chose to reduce the number of hazards, taking into account only the "real word effect" as a hazard, but for some cases where it was obvious that the treatment would be completely different, we also took into account the deviation and use case effect. For instance, in Table 8, the hazard HN2 (Fall of patient due to imbalance not caused by the robot) and HN6 (Fall of the patient due to imbalance caused by the robot), lead both to the fall of the patient, but have been differentiated. Even if we provide a well guided method, extraction and formulation of hazards list require a high level of expertise from the safety analyst, in order to choose the right level of description of a hazard.

5. Related work on model-based hazard identification, tools and methods

This section presents related work, focusing on model-based safety analysis, and more particularly those using UML. The concept of "model-based" refers to the fact that a safety analysis technique (e.g., FTA) is based on an abstract representation of the studied system. This was already done at the very first hours of the risk analysis techniques using for instance block diagrams, or had-hoc representations. The quite recent model-based term, usually refers to the use of standardized models (like UML) and the possibility to have tools assisting analysts to produce automatic, or semi-automatic safety analysis based on a system model. Generally, model-based safety analyses focus on the following issues (Blanquart, 2010):

1. Fault propagation analysis:
   (a) *bottom-up*: a fault effect on the system,
   (b) *top-down*: induction of faults inducing an unwanted effect,
2. Dependability (or safety) properties verification.
3. Quantification of probability of unwanted events.

Many high-level modeling languages for safety analyses have been defined to cover those points. Just to cite some of them, HIPS-HOPS (*Hierarchically Performed Hazard Origin and Propagation Studies*) and its associated tool developed at Hull university,[2] automatically generates fault trees and FMECA tables starting from system models (e.g., Simulink models). For each component, fault annotations are given, and the tool propagates those faults to build safety models (e.g., Fault trees). Altarica (Boiteau et al., 2006; Lipaczewski et al., 2015) provides means for fault tree generation or properties verification from system and reliability models. Additionally, many European research projects addressed model-based safety analysis: ESACS (2001–2003)[3] in transportation domain, followed by ISAAC (2004–2007)[4] in avionics, then CESAR (2009–2012)[5] followed by CRYSTAL (2013–2017)[6] for embedded systems. Previous techniques and works, usually rely on a precise description of the system behavior, which is usually not available at the beginning of a human–robot project.

The method put forward in this paper falls within the scope of fault propagation analysis, and can be described as a "middle-up approach", as we do not start from "faults" but from deviations. Our objective is then to identify hazards (and hazardous situations) during human–robot interaction. A very close work is advanced by Leveson (2011), with a method called STPA (System Theoretic Process Analysis), which provides guidance to users combining guide words (like in HAZOP) and fault models, applied to models, based on a process/controller/actuator/sensor representation. Many recent applications of STPA can be found, e.g., in robotics (Alemzadeh et al., 2015), space (Ishimatsu et al., 2010), railway (Thomas and Leveson, 2011) or automotive (Sulaman et al., 2014). One difference with our approach is that scenarios are actually not modeled in this approach. Users are represented as "controllers", which is not clear while describing human–robot interactions. STPA objective is also different in the way that it really focuses on the identification of cause-consequence chain, which is not the objective of HAZOP–UML (only find the hazards and hazardous situations). We also propose to use UML which is not the case in STPA. On the contrary, the work done in the CORAS project (CORAS, 2014; Bjørn Axel Gran and Thunem, 2004), is based on UML to analyze security. Even if we focus on safety, our objectives are the same. A major difference is that we strongly interconnect UML models and the risk analysis technique HAZOP, which was not addressed in CORAS.

Our risk analysis approach is based on a re-interpretation of HAZOP guidewords in the context of some UML diagrams. A similar approach has been followed in some previous studies considering UML structural diagrams (Hansen et al., 2004; Gorski and Jarzebowicz, 2005; Jarzebowicz and Górski, 2006) and dynamic diagrams (Johannessen et al., 2001; Allenby and Kelly, 2001; Arlow et al., 2006; Iwu et al., 2007; Srivatanakul, 2005). In all those papers, the guide words were quite reduced (e.g., only omission and commission) or the link with UML language elements was not fully explored. We actually extended the results of those studies, focusing only on use case, sequence and state machine diagrams, in order to explore deviations during operational life. We also paid a particular attention to the human errors expression and analysis in this method, which was absent from the previous papers.

6. Conclusion

We set forth a new method for the safety analysis of human–robot interaction called HAZOP–UML. To build this method we used the UML metamodel to identify the basic elements of three dynamic models. We then proposed three guide words tables for use cases, messages of sequence diagrams, and state machines. Those guide words tables help the safety analyst to imagine possible deviations for every elements of those dynamic models. Those deviations are then reported in HAZOP tables, where causes, consequences, and recommendations are formulated. This process produces lists of hazards, recommendations, and hypotheses.

This method has been applied successfully on several projects, and we present in this paper a general analysis of the benefits and the limits of the method. We particularly focus on the applicability and validity of the approach. Main advantages of HAZOP–UML are:

- simple (training and application),
- applicable at the first step of the development process,
- limits the combinatory explosion,
- consistent with system models, and inherits of system modeling benefits: traceability and modifiability,
- easily supported by a computer assisting tool.

Even if the models and HAZOP tables can be easily achieved, the main limit lies in the necessity of a high expertise to formulate hazards from HAZOP tables. It is up to the safety analyst to determine the right level of detail for the hazard identification.

[2] http://hip-hops.eu (accessed 2015-05-15).
[3] www.transport-research.info/web/projects/project_details.cfm?ID=2658.
[4] http://ec.europa.eu/research/transport/projects/items/isaac_en.htm.
[5] www.cesarproject.eu.
[6] www.crystal-artemis.eu.

Additionally to the three projects presented in this paper, HAZOP–UML has also been used as a first step of a method to build independent safety monitors in the context of autonomous robots (Machin et al., 2014), and we also plan to use it as an entry point for defining virtual words for testing mobile robots in simulation. A future direction is the complete transfer to industry, which is already started in the project CPSELabs (2015–2018).

Acknowledgments

This work was partially supported by the MIRAS project, funded under ANR-TecSan 2009 framework, and the PHRIENDS and SAPHARI Projects, funded under the 6th and the 7th Framework Programme of the European Community.

References

2006/42/EC, 2006. Council Directive on Machinery. Official Journal of the European Union L157.

93/42/EEC, 1993. Council directive of the 14th of june 1993 concerning medical devices. Official Journal of the European Union.

Alemzadeh, H., Chen, D., Lewis, A., Kalbarczyk, Z., Iyer, R., 2015. Systems-theoretic safety assessment of robotic telesurgical system. In: 34th International Conference on Computer Safety, Reliability and Security.

Allenby, K., Kelly, T., 2001. Deriving safety requirements using scenarios. In: Fifth IEEE International Symposium on Requirements Engineering, pp. 228–235.

Arlow, A., Duffy, C., McDermid, J., 2006. Safety specification of the active traffic management control system for English motorways. In: The First Institution of Engineering and Technology International Conference on System Safety.

Avižienis, A., Laprie, J.C., Randell, B., Landwehr, C., 2004. Basic concepts and taxonomy of dependable and secure computing. IEEE Trans. Dependable Secure Comput. 1 (1), 11–33.

Bjørn Axel Gran, R.F., Thunem, A.P.-J., 2004. An approach for model-based risk assessment. In: 23rd International Conference, SAFECOMP 2004, Potsdam, Germany. Springer, Berlin/Heidelberg, pp. 311–324.

Blanquart, J.-P., 2010. Survey of State of the Art and of the Practice in Safety and Diagnosability. Tech. Rep. D_SP1_R5.8_M2, EADS Astrium Satellites, CESAR European Project.

Boiteau, M., Dutuit, Y., Rauzy, A., Signoret, J.-P., 2006. The AltaRica data-flow language in use: modeling of production availability of a multi-state system. Reliab Eng Syst Saf, 0951-8320 91 (7), 747–755.

Cantrell, S., Clemens, P., 2009. Finding all the hazards how do we know we are done? Prof. Saf. 54 (11), 45. American Society of Safety Engineers.

CORAS, 2014. A Platform for Risk Analysis of Security Critical Systems. <coras.sourceforge.net> (accessed 2015.05.17).

CPSELabs, 2015–2018. Cyber-Physical Systems Engineering Labs, Project funded by the European Union, Horizon2020 Programme. <www.cpse-labs.eu> (accessed 2015.05.17).

DefStan00-58, 2000. HAZOP Studies on Systems Containing Programmable Electronics. Defence Standard, Ministry of Defence, UK, part 1 and 2.

Dogramadzi, S., Giannaccini, M., Harper, C., Sobhani, M., Woodman, R., Choung, J., 2014. Environmental hazard analysis – a variant of preliminary hazard analysis for autonomous mobile robots. J. Intell. Robot Syst., 0921-0296 76 (1), 73–117. http://dx.doi.org/10.1007/s10846-013-0020-7.

Gorski, J., Jarzebowicz, A., 2005. Development and validation of a HAZOP-based inspection of UML models. In: 3rd World Congress for Software Quality, Munich, Germany.

Guiochet, J., Vilchis, A., 2002. Safety analysis of a medical robot for tele-echography. In: Proc. of the 2nd IARP IEEE/RAS Joint Workshop on Technical Challenge for Dependable Robots in Human Environments, Toulouse, France, pp. 217–227.

Guiochet, J., Motet, G., Baron, C., Boy, G., 2004. Toward a human-centered UML for risk analysis – application to a medical robot. In: Johnson, C., Palanque, P. (Eds.), Proc. of the 18th IFIP World Computer Congress (WCC), Human Error, Safety and Systems Development (HESSD04). Kluwer Academic Publisher, pp. 177–191.

Guiochet, J., Martin-Guillerez, D., Powell, D., 2010. Experience with model-based user-centered risk assessment for service robots. In: IEEE International Symposium on High-Assurance Systems Engineering (HASE2010). IEEE Computer Society, San Jose, CA, USA, pp. 104–113.

Guiochet, J., Do Hoang, Q.A., Kaaniche, M., Powell, D., 2013. Model-based safety analysis of human–robot interactions: the MIRAS walking assistance robot. In: 2013 IEEE International Conference on Rehabilitation Robotics (ICORR), pp. 1–7.

Haddadin, S., 2014. Towards safe robots, approaching Asimovs 1st law. Springer Tracts in Advanced Robotics, vol. 90. Springer.

Hansen, K.M., Wells, L., Maier, T., 2004. HAZOP analysis of UML-based software architecture descriptions of safety-critical systems. In: Nordic Workshop on UML and Software Modeling (NWUML04).

Harel, D., 1987. Statecharts: a visual formalism for complex systems. Sci. Comput. Program., 0167-6423 8 (3), 231–274.

IEC61508-5, 2010. Functional safety of electrical/electronic/programmable electronic safety-related systems: Part 5: Examples of methods for the determination of safety integrity level. International Electrotechnical Commission.

IEC61882, 2001. Hazard and Operability Studies (HAZOP Studies) – Application Guide. International Electrotechnical Commission.

Ishimatsu, T., Leveson, N., Thomas, J., Katahira, M., Miyamoto, Y., Nakao, H., 2010. Modeling and hazard analysis using STPA. In: 4th Conference of the International Association for the Advancement of Space Safety (IAASS).

ISO/FDIS14971, 2006. Medical devices – Application of risk management to medical devices. International Standard Organisation.

ISO/IEC-Guide51, 1999. Safety Aspects – Guidelines for their Inclusion in Standards. International Organization for Standardization.

ISO10218-1, 2011. Robots for industrial environments – safety requirements – Part 1: Robot. International Organization for Standardization.

ISO13482, 2014. Robots and robotic devices – safety requirements for personal care robots. International Organization for Standardization.

ISO13849-1, 2006. Safety of machinery – safety-related parts of control systems – Part 1: General principles for design. International Organization for Standardization.

ISO31000, 2009. Risk Management – Principles and Guidelines, International Organization for Standardization.

Iwu, F., Galloway, A., Mcdermid, J., Ian, T., 2007. Integrating safety and formal analyses using UML and PFS. Reliab. Eng. Syst. Saf. 92 (2), 156–170.

Jarzebowicz, A., Górski, J., 2006. Empirical evaluation of reading techniques for UML models inspection. ITSSA 1 (2), 103–110.

Johannessen, P., Grante, C., Alminger, A., Eklund, U., Torin, J., 2001. Hazard analysis in object oriented design of dependable systems. In: 2001 International Conference on Dependable Systems and Networks, Göteborg, Sweden, pp. 507–512.

Leveson, N.G., 2011. Engineering a safer world. Systems Thinking Applied to Safety. The MIT Press.

Lipaczewski, M., Ortmeier, F., Prosvirnova, T., Rauzy, A., Struck, S., 2015. Comparison of modeling formalisms for safety analyses: SAML and AltaRica. Reliab Eng Syst Saf, 0951-8320.

Machin, M., Dufoss, F., Blanquart, J.-P., Guiochet, J., Powell, D., Waeselynck, H., 2014. Specifying safety monitors for autonomous systems using model-checking. In: Bondavalli, A., Di Giandomenico, F. (Eds.), Computer Safety, Reliability, and Security, Lect. Notes Comput. Sci., vol. 8666. Springer International Publishing, pp. 262–277.

Martin-Guillerez, D., Guiochet, J., Powell, D., Zanon, C., 2010. UML-based method for risk analysis of human–robot interaction. In: International Workshop on Software Engineering for Resilient Systems (SERENE2010), London, UK, 2010.

MIRAS, 2009–2013. Multimodal Interactive Robot for Assistance in Strolling, Project supported by the French ANR (National Research Agency) under the TecSan (Healthcare Technologies) Program (ANR-08-TECS-009-04). <www.miraswalker.com/index.php/en> (accessed 2015.05.17).

Mitka, E., Gasteratos, A., Kyriakoulis, N., Mouroutsos, S.G., 2012. Safety certification requirements for domestic robots. Saf. Sci., 0925-7535 50 (9), 1888–1897.

OMG-UML2, 2007. Unified Modeling Language (UML), Superstructure, V2.1.2, formal/2007-11-02. Object Management Group.

OMG-UML2, 2011. Unified Modeling Language (UML), Superstructure, V2.4.1, formal/2011-08-06. Object Management Group, 2011.

PHRIENDS, 2006–2009. Physical Human–Robot Interaction: Dependability and Safety, Project supported by the European Commission under the 6th Framework Programme (STReP IST-045359). <www.phriends.eu> (accessed: 2015.04.30).

Royakkers, L., van Est, R., 2015. A literature review on new robotics: automation from love to war. Int. J. Soc. Robot., 1–22

SAPHARI, 2011–2015. Safe and Autonomous Physical Human-Aware Robot Interaction, Project supported by the European Commission under the 7th Framework Programme. <www.saphari.eu> (accessed 2015.05.17).

Srivatanakul, T., 2005. Security Analysis with Deviational Techniques (Ph.D. thesis). University of York.

Sulaman, S.M., Abbas, T., Wnuk, K., Hö st, M., 2014. Hazard Analysis of Collision Avoidance System using STPA. In: 11th International Conference on Information Systems for Crisis Response and Management (ISCRAM).

Thomas, J., Leveson, N.G., 2011. Performing hazard analysis on complex, software and human-intensive systems. In: 29th ISSC Conference about System Safety.