PESARO 2013

The Third International Conference on Performance, Safety and Robustness in Complex Systems and Applications

ISBN: 978-1-61208-268-4

April 21 - 26, 2013

Venice, Italy

PESARO 2013 Editors

Wolfgang Leister, Norsk Regnesentral (Norwegian Computing Center), Norway

Pascal Lorenz, University of Haute-Alsace, France

PESARO 2013

Foreword

The Third International Conference on Performance, Safety and Robustness in Complex Systems and Applications (PESARO 2013), held between April 21st-26th, 2013 in Venice, Italy, continued the inaugural event dedicated to fundamentals, techniques and experiments to specify, design, and deploy systems and applications under given constraints on performance, safety and robustness.

There is a relation between organizational, design and operational complexity of organization and systems and the degree of robustness and safety under given performance metrics. More complex systems and applications might not be necessarily more profitable, but are less robust. There are trade-offs involved in designing and deploying distributed systems. Some designing technologies have a positive influence on safety and robustness, even operational performance is not optimized. Under constantly changing system infrastructure and user behaviors and needs, there is a challenge in designing complex systems and applications with a required level of performance, safety and robustness.

We take here the opportunity to warmly thank all the members of the PESARO 2013 Technical Program Committee. The creation of such a high quality conference program would not have been possible without their involvement. We also kindly thank all the authors who dedicated much of their time and efforts to contribute to PESARO 2013. We truly believe that, thanks to all these efforts, the final conference program consisted of top quality contributions.

Also, this event could not have been a reality without the support of many individuals, organizations, and sponsors. We are grateful to the members of the PESARO 2013 organizing committee for their help in handling the logistics and for their work to make this professional meeting a success.

We hope that PESARO 2013 was a successful international forum for the exchange of ideas and results between academia and industry and for the promotion of progress in the field of performance, safety and robustness in complex systems and applications.

We are convinced that the participants found the event useful and communications very open. We also hope the attendees enjoyed the charm of Venice, Italy.

PESARO Advisory Committee:

Piotr Zwierzykowski, Poznan University of Technology, Poland
Wolfgang Leister, Norsk Regnesentral (Norwegian Computing Center), Norway
Yulei Wu, Chinese Academy of Sciences, China
Harold Liu, IBM Research, China

PESARO 2013

Committee

PESARO Advisory Committee

Piotr Zwierzykowski, Poznan University of Technology, Poland
Wolfgang Leister, Norsk Regnesentral (Norwegian Computing Center), Norway
Yulei Wu, Chinese Academy of Sciences, China
Harold Liu, IBM Research, China

PESARO 2013 Technical Program Committee

Amr Aissani, USTHB - Algiers, Algeria
Salimur Choudhury, Queen's University - Kingston, Canada
Juan Antonio Cordero, École Polytechnique / INRIA, France
John-Austen Francisco, Rutgers University - Piscataway, USA
Mina Giurguis, Texas State University - San Marcos, USA
Mesut Günes FU-Berlin, USA
Charles Kamhoua Kenmogne, Air Force Research Laboratory, USA
Wolfgang Leister, Norsk Regnesentral (Norwegian Computing Center), Norway
Harold Liu, IBM Research, China
Olaf Maennel, Loughborough University, UK
Kia Makki, Technological University of America (TUA), USA
Birgit Milius, Institut fuer Eisenbahnwesen und Verkehrssicherung (IfEV) /Technische Universitaet – Braunschweig, Germany
Liam Murphy, University College Dublin, Ireland
Asoke K. Nandi, The University of Liverpool, UK / University of Jyvaskyla, Finland / University of Calgary, Canada
Harald Øverby, NTNU, Norway
Maciej Piechowiak, Kazimierz Wielki University - Bydgoszcz, Poland
M. Zubair Rafique, King Saud University - Islamabad, Pakistan
Roger S. Rivett, Land Rover - Gaydon, UK
Dhananjay Singh, Electronics and Telecommunications Research Institute (ETRI), South Korea
Mukesh Singhal, University of Kentucky, USA
Young-Joo Suh, Pohang University of Science and Technology (POSTECH), South Korea
Batool Talha, University of Minnesota, USA
Yuejin Tan, National University of Defense and Technology - Changsha, China
Yang Wang, Georgia State University, USA
Yun Wang, Bradley University - Peoria, USA
Jun Wu, National University of Defense and Technology, China
Yanwei Wu, Western Oregon University, USA
Yulei Wu, Chinese Academy of Sciences, China
Gaoxi Xiao, Nanyang Technological University, Singapore
Bin Xie, InfoBeyond Technology LLC - Louisville, USA

Copyright Information

For your reference, this is the text governing the copyright release for material published by IARIA.

The copyright release is a transfer of publication rights, which allows IARIA and its partners to drive the dissemination of the published material. This allows IARIA to give articles increased visibility via distribution, inclusion in libraries, and arrangements for submission to indexes.

I, the undersigned, declare that the article is original, and that I represent the authors of this article in the copyright release matters. If this work has been done as work-for-hire, I have obtained all necessary clearances to execute a copyright release. I hereby irrevocably transfer exclusive copyright for this material to IARIA. I give IARIA permission or reproduce the work in any media format such as, but not limited to, print, digital, or electronic. I give IARIA permission to distribute the materials without restriction to any institutions or individuals. I give IARIA permission to submit the work for inclusion in article repositories as IARIA sees fit.

I, the undersigned, declare that to the best of my knowledge, the article is does not contain libelous or otherwise unlawful contents or invading the right of privacy or infringing on a proprietary right.

Following the copyright release, any circulated version of the article must bear the copyright notice and any header and footer information that IARIA applies to the published article.

IARIA grants royalty-free permission to the authors to disseminate the work, under the above provisions, for any academic, commercial, or industrial use. IARIA grants royalty-free permission to any individuals or institutions to make the article available electronically, online, or in print.

IARIA acknowledges that rights to any algorithm, process, procedure, apparatus, or articles of manufacture remain with the authors and their employers.

I, the undersigned, understand that IARIA will not be liable, in contract, tort (including, without limitation, negligence), pre-contract or other representations (other than fraudulent misrepresentations) or otherwise in connection with the publication of my work.

Exception to the above is made for work-for-hire performed while employed by the government. In that case, copyright to the material remains with the said government. The rightful owners (authors and government entity) grant unlimited and unrestricted permission to IARIA, IARIA's contractors, and IARIA's partners to further distribute the work.

Table of Contents

Developing Safe Control Systems using Patterns for Assurance

André Alexandersen Hauge
Institute for energy technology, Halden, Norway
University of Oslo, Norway
andre.hauge@hrp.no

Ketil Stølen
SINTEF ICT, Oslo, Norway
University of Oslo, Norway
ketil.stolen@sintef.no

***Abstract*—The Safe Control Systems (SaCS) method is a pattern-based method supporting the development of conceptual designs for safety critical systems. A pattern language offers support for the method by six different kinds of basic patterns, operators for combining patterns, and a graphical notation for visualising a pattern composition. Intended users of SaCS are system developers, safety engineers and HW/SW engineers. The method has so far been applied in two cases within different industrial domains. This paper demonstrates and presents experiences from the application of SaCS within the railway domain. We consider an interlocking system that controls the appliances of a railway station. We argue that SaCS effectively supports the establishment of requirements, a design satisfying the requirements, and an outline of a safety demonstration for the design.**

Keywords-conceptual design; pattern language; development processes; safety;

I. INTRODUCTION

This paper demonstrates and presents experiences from the use of a pattern-based method called Safe Control Systems (SaCS) to develop a conceptual design for a railway interlocking system.

SaCS has previously been tested out in the nuclear domain for the development of a reactor control system design [1].

The six kinds of basic patterns offered by SaCS are categorised according to two development perspectives: *Process Assurance*; and *Product Assurance*. Both perspectives detail patterns according to three aspects: *Requirement*; *Solution*; and *Safety Case*. We distinguish between basic and composite patterns. Each basic pattern contains an instantiation rule that may be used to assess whether it is correctly instantiated.

The basic SaCS patterns captures design solutions as well as commonly accepted safety engineering practices, e.g., development processes and activities, risk assessment methods, and other methods for providing safety assurance as reflected in international safety standards and guidelines, e.g., [2], [3], [4], [5]. Basic SaCS patterns are defined in a format inspired by classical literature on patterns [6], [7], [8]. It differs with respect to its explicit definition of inputs, outputs, and the instantiation rules that defines the transition from input to output for each pattern. The explicitly defined parameters facilitate easy combination of several patterns. The instantiation rules facilitate validation of the result of pattern instantiation based on the pattern definition and the given inputs. The composite patterns are expressed graphically and its notation is inspired by languages for system modelling, e.g., the modelling of patterns by collaborations, or modularisation of a specification by decomposition in UML [9], and literature related to visualisation, e.g., related to risk analysis [10] and general literature on visualisation of complex data [11].

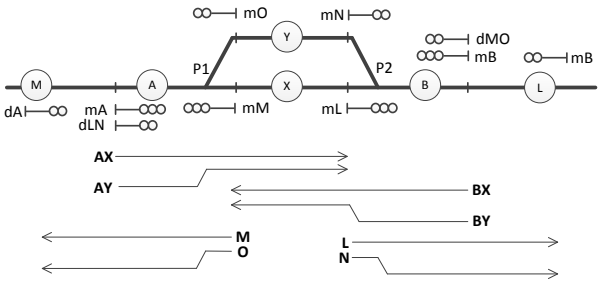

Figure 1. Railway Station Overview

The remainder of this article is structured as follows: Section II outlines the railway case on design of an interlocking system. Section III gives a short background on the SaCS method and its supporting pattern language. Section IV presents our hypothesis and main prediction. Section V to Section VIII exemplifies the stepwise application of the SaCS method and supporting pattern language in an example-driven manner for establishing a conceptual design for the railway interlocking system. Section IX outlines the results of applying SaCS. Section X presents related work, while Section XI concludes.

II. THE SYSTEM: RAILWAY INTERLOCKING

Fig. 1 illustrates the main appliances of a train station with two tracks.

The station in Fig. 1 is connected in both ends of the station area to neighbouring stations with a single track. An interlocking system controls the appliances associated with the station in order to safely control the movements of trains along defined train routes. The train station that is used as a case has a level crossing (Note: the level crossing is not depicted). The annotations in Fig. 1 denote the following:

- The circles with a letter inside, i.e., M, A, X, Y, B, and L denote the different track sections.

 ISBN: 978-1-61208-268-4

- All lines that end with two or three adjacent circles represent either a distant light signal, i.e., dA, dB, dLN, and dMO or a main light signal, i.e., mA, mB, mL, mM, mN, and mO.
- There are two points for switching traffic onto different tracks, these are identified as P1 and P2.
- The arrows illustrate the eight train routes, i.e., AX, AY, BX, BY, L, M, N, and O that are possible with the depicted track configuration.

The SaCS method was applied to develop a conceptual design for an interlocking system to control the appliances of a railway station with two tracks and a level crossing such that trains may move safely according to defined train routes. The interlocking system is one of many sub-systems, though the most critical to safety, in an overall system for controlling train movements. The conceptual design is intended to model a replacement of an existing system governing the interlocking rules only.

III. Background – SaCS

A. The SaCS Method

The method interleaves three main activities, each of which is divided into sub-activities:

S *Pattern Selection* – The purpose of this activity is to support the conception of a design by selecting: a) SaCS patterns for requirement elicitation; b) SaCS patterns for establishing design basis; c) SaCS patterns for establishing safety case.

C *Pattern Composition* – The purpose of this activity is to specify the use of the selected patterns by specifying: a) compositions of patterns; and b) instantiations of patterns.

I *Pattern Instantiation* – The purpose of this activity is to instantiate the composite pattern specification by: a) selecting pattern instantiation order; and b) conducting stepwise instantiation.

Pattern selection is supported by a selection map that is introduced in Section V-A. Pattern composition is supported by a pattern language outlined in Section III-B. Pattern instantiation is supported by instantiation rules defined for every basic pattern (defined in [12]).

B. The SaCS Pattern Language

Fig. 2 presents the main graphical elements used to illustrate the kind of patterns involved in a composite pattern.

The SaCS Pattern Language (SaCS PL) consists of patterns (basic SaCS patterns) of different types, and annotations for specifying how patterns are combined and applied in order to derive a conceptual design.

Every basic SaCS pattern is defined such that it may be used stand-alone. Every pattern is also defined such that it is easy to use several patterns together as every input and output parameter of a pattern is explicitly detailed.

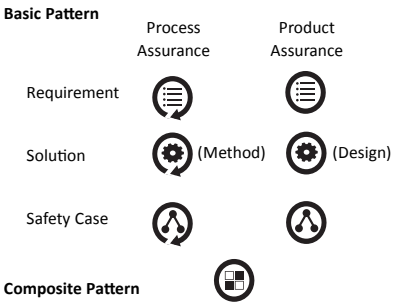

Figure 2. Icons for Visually Representing a Pattern

A composition of patterns (composite for short) may be expressed by, e.g., mapping an output parameter of a pattern to an input parameter of a second pattern and thereby defining a relationship between the patterns.

The different icons in Fig. 2 for representing a pattern reflect the different types of patterns in SaCS PL. Operators combine patterns. The operators will be introduced and explained by the examples provided in Section V to Section IX. A composite pattern may contain any type of SaCS pattern, i.e., basic patterns, composite patterns, or a combination of composite and basic patterns.

IV. Hypothesis

Success is evaluated based on the satisfaction of predictions. The hypothesis (H) and predictions (P) for the application of SaCS is defined below.

H: The SaCS method facilitates effective and efficient development of conceptual designs that are: 1) consistent; 2) complete; 3) correct; 4) comprehensible; 5) reusable; and 6) implementable.

Definition *A conceptual design is a triple consisting of a specification of requirements, a specification of design, and a specification of a safety case. The design characterises a system that satisfies the requirements. The safety case characterises a strategy for demonstrating that the design is safe with respect to safety requirements.*

We deduce the following prediction from the hypothesis with respect to the application of SaCS on the case described in Section II:

P: Application of the SaCS method on the railway case described in Section II results in a conceptual design that uniquely characterises the railway case and is easily instantiated from a composite SaCS pattern. Furthermore, the conceptual design is 1) consistent; 2) complete; 3) correct; 4) comprehensible; 5) reusable; 6) implementable.

Definition *A conceptual design instantiates a SaCS composite pattern if each element of the triple can be instantiated from the SaCS composite pattern according to the instantiation rules of the individual patterns and according to the rules for composition.*

 ISBN: 978-1-61208-268-4

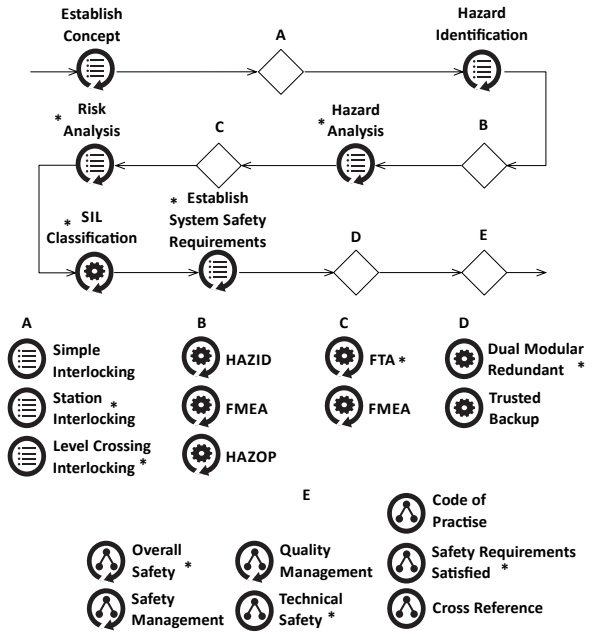

Figure 3. Pattern Selection

V. Elicit Functional Requirements

A. Pattern Selection

Fig. 3 provides an overview of the patterns considered in the railway interlocking case, organised into a pattern selection map based on defined relationships between patterns. Due to space restrictions the pattern definitions are not provided (described in [12]). The following abbrevations are used in Fig. 3: HAZID – HAZard IDentification; FMEA – Failure Modes and Effects Analysis; HAZOP – HAZard and OPerability Studies; FTA – Fault Tree Analysis; SIL – Safety Integrity Level.

The selection process starts at the pattern referenced in the upper left corner of Fig. 3 and follows the direction of the arrows. Pattern selection ends when all patterns have been considered. The diamond represents a choice; more than one pattern may be selected. The letter above a choice is a reference to a correspondingly named group of patterns in the lower part of Fig. 3. The symbol "*" is used to identify the patterns that are used in this article. Each pattern definition clearly describes the problem addressed by the pattern and its intended application, thus the pattern definitions may be conferred for support in the selection process. The application of the SaCS method and the selection process as exemplified in this article assures that relevant patterns may be selected at each stage of development. The rationale given by a user at each selection step on the selection of patterns should give assurance for the correct set of patterns being selected.

We assume in this article that the information provided in Section II sufficiently details the development objectives such that the pattern *Establish Concept* (supports clarifying objectives) may be passed in the selection process.

The patterns *Station Interlocking* and *Level Crossing Interlocking* associated with choice A in Fig. 3 capture the problem of eliciting functional requirements for a system that shall control the appliances available in a station with several tracks and a level crossing, respectively. They were chosen as support.

B. Pattern Instantiation

A pattern may have multiple input and output parameters. The instantiation of the input parameters define the context for interpreting the pattern, while the instantiation of the output parameters define the result of pattern instantiation. The pattern definition describes the transition from input to output. The instantiation of the pattern *Station Interlocking* produced a set of requirements. The following is one of these requirements.

"FR.1: a train route AX may be locked (secured for train movements) when: a) the train routes AY, M, O, N, BX and BY are in the state not locked; and b) point P1 is aligned; and c) track sections A, X and B are in the state vacant."

C. Pattern Composition

Fig. 4 specifies a composite pattern. Everything above the horizontal line may be thought of as a kind of preamble. The icon for a composite occurs in the upper left corner underneath the name of the composite, which is *Functional Requirements*. The inputs and outputs of the composite are tagged by an arrow pointing either towards (indicating input) or from (indicating output) a list of parameters. The parameter list is visualised on the form [<parameter list>]. Everything below the horizontal line describes the actual composition. The input and output of the patterns occurring within the composite are distinguished in the same manner as input and output in the preamble. The input parameter Mch of the composite is used by both contained patterns.

The patterns *Station Interlocking* and *Level Crossing Interlocking* are related by the *combines* operator (symbolised by two overlapping circles). Hence, the icon decorating the line connecting the parameter lists in Fig. 4 symbolises a combines relationship.

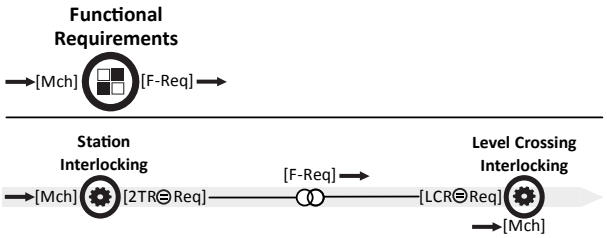

Figure 4. Functional Requirements – Composite

The annotation [2TR=Req] (the symbol = enclosed by a circle represents an *alias* operator) found in Fig. 4 defines an alias 2TR for the output parameter Req. The *combines* operator creates an output parameter list named F-Req that consists of 2TR and LCR.

The grey wide arrow in the background indicates the recommended pattern instantiation order and gives guidance to the process of applying the patterns.

VI. Elicit Safety Requirements

A. Pattern Selection

Once the main functional requirements have been elicited based on selected patterns from choice A of Fig. 3, further traversal leads to the pattern *Hazard Identification*. This pattern defines the process of identifying potential hazards. In choice B, patterns describing methods for hazard identification are offered. We assume that the hazards associated with the operation of the interlocking system, e.g., collision train-train, collision train-object, and level crossing accident, are identified such that the *Hazard Identification* pattern as well as the patterns in choice B may be passed in the selection process.

The *Hazard Analysis* pattern however is selected as it provides guidance on the process of deriving the potential causes of hazards. In choice C of Fig. 3, different process solution patterns (representing methods) supporting hazard analysis may be selected. The *FTA* was selected as support for *Hazard Analysis* under the assumption that a top-down fault tree analysis is an acceptable and effective method for identifying potential causes of failure.

Further traversal of Fig. 3 leads to the pattern *Risk Analysis*. The pattern provides guidance on how to address identified hazards and to establish a notion of risk. The *SIL Classification* pattern was selected as it defines the method for classifying railway systems and their components with respect to criticality.

The pattern *Establish System Safety Requirements* describes the process of establishing safety requirements based on inputs from risk assessment. It was regarded as relevant for the case and selected as support.

B. Pattern Instantiation

Safety requirements are defined on the basis of risk assessment. The process requirement patterns selected in Section VI-A support the process of eliciting safety requirements and may be applied subsequently in the following order:

1. *Hazard Analysis* – used to identify potential causes of hazards based on input on applicable hazards.
2. *Risk Analysis* – used for addressing hazards with respect to their severity and likelihood of occurring combined into a notion of risk.
3. *Establish System Safety Requirements* – used for defining requirements on the basis of identified risks.

When *Establish System Safety Requirements* was instantiated on the basis of inputs provided by the instantiation of its successors, the following safety requirement was among those identified: *"SR.1: A main signal belonging to a train route may only signal a proceed aspect if the train route is locked"*.

Other results from the instantiation of the mentioned patterns were a hazard log that traces identified hazards to potential causes of these hazards, a fault tree analysis, and a qualitative risk assessment. In the following we only refer to the safety requirement exemplified above.

C. Pattern Composition

Fig. 5 presents the *Safety Requirements* composite. The pattern has two input parameters namely ToA (short for Target of Assessment) and Haz (short for Hazards) and one output parameter S-Req (short for Safety Requirements).

The hazards associated with the parameter Haz, e.g., collision train-train and collision train-object, consisted of a set of relevant generic top events. The hazards were assessed with respect to the intended operation of interlocking system by the use of the *Hazard Analysis* pattern, supported by the *FTA* pattern. The result of pattern instantiation is the output HzLg (short for hazard log) containing an overview over hazards and their potential causes.

The output HzLg from *Hazard Analysis* is an input of the *Risk Analysis* pattern (illustrated by the *assigns* operator that is drawn as an arrow from HzLg to Haz). The *SIL Classification* pattern was used as support for *Risk Analysis*. The output Risks from *Risk Analysis* is used as input to *Establish System Safety Requirements*.

The output S-Req of *Establish System Safety Requirements* (S-Req is defined as an alias for the output parameter Req) is an output of the composite.

Fig. 6 defines the *Requirements* composite. The composite is defined in order to make later illustrations simpler and consists of the composite for establishing functional requirements (defined in Fig. 4) and the composite for establishing

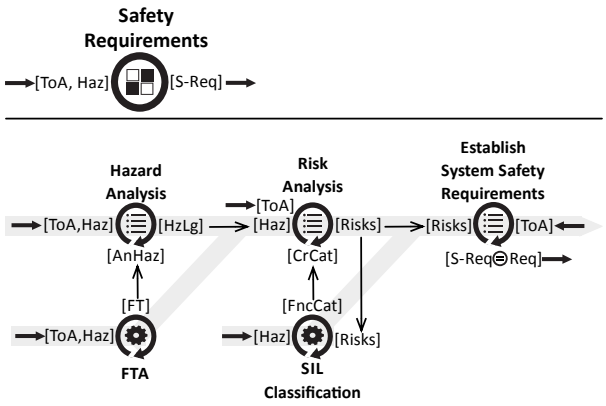

Figure 5. Safety Requirements – Composite

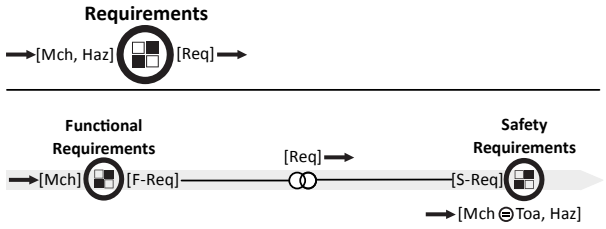

Figure 6. Requirements – Composite

safety requirements (defined in Fig. 5).

VII. Establish Design Basis

A. Pattern Selection

At choice D of Fig. 3, there are two available design patterns that may be selected.

The *Trusted Backup* pattern describes a system concept where an adaptable controller may operate freely in a delimited operational state space. Safety is assured by a redundant non-adaptable controller that operates in a broader state space and in parallel with the adaptable controller. A control delegator grants control privileges to the most suitable controller at any given time on the basis of switching rules and information from safety monitoring.

The *Dual Modular Redundant* pattern describes a system concept where two similar controllers operate in parallel. The parallel operating redundant controllers and a voting unit provides mitigations against random error.

The *Dual Modular Redundant* pattern was selected as guidance for establishing the design on the basis of an evaluation of the strengths and weaknesses of the two design patterns with respect to the requirements.

B. Pattern Instantiation

Fig. 7 represents an excerpt (simplified) of the result, fully described in [12], of applying the *Dual Modular Redundant* pattern. The pattern was instantiated according to its instantiation rule and the design was defined to comply with the requirements identified in Section V and Section VI.

Fig. 7 illustrates the main parts of our interlocking system. A component identified as *Cmd* is responsible for the communication towards the operators of the system. Dual controllers, identified as *Ctrl1* and *Ctrl2*, are responsible for providing interlocking functionality, e.g., lock a train route upon request from an operator. A component identified as *IO* is responsible for communicating with e.g., points, lights, and track sections as presented in Fig. 1 in order to communicate their states to the dual controllers. The *IO* component is also responsible for safe application of the output from the dual controllers (the *IO* component contains a voter).

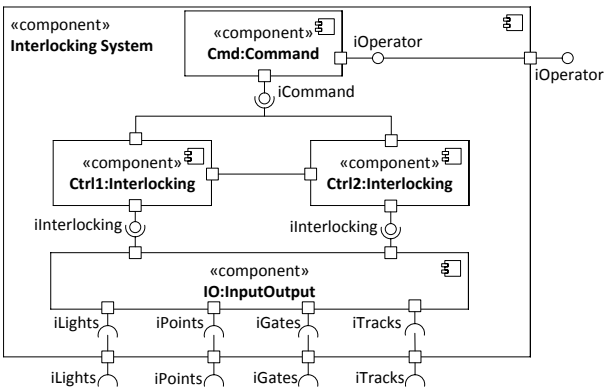

Figure 7. UML Component Diagram

Besides the presented excerpt, results from pattern instantiation include a description of: the interaction between the interlocking system and other systems; the functionality of the internal components of the interlocking system; and the interaction between the components within the interlocking system. In the following when we refer to the system design we mean the full design of the interlocking system. The fulfilment of identified requirements, e.g., FR.1 defined in Section V-B and SR.1 defined in Section VI-B, is manifested in different decomposed models of the full design.

C. Pattern Composition

In Fig. 8, the output parameter S of *Dual Modular Redundant* represents the abstract system design described by the pattern, the instantiation of which is represented by the design outlined in Section VII-B.

Fig. 8 specifies that the instantiation of the output parameter S of the pattern *Dual Modular Redundant* shall satisfy the instantiation of the output parameter Req of the pattern *Requirements*. The relationship is captured by a *satisfies* operator where the bullet is associated with the output that describes what should be satisfied and the check mark is associated with the output that is required to satisfy.

VIII. Establish Safety Case

A. Pattern Selection

The *Overall Safety* pattern associated with choice E in Fig. 3 was selected as a means to argue that the system is

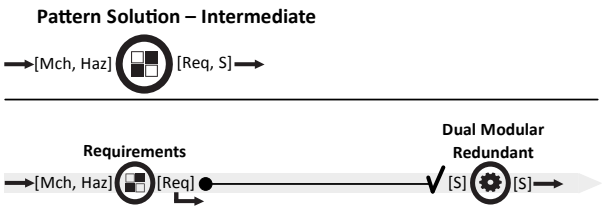

Figure 8. Intermediate Solution – Composite

 ISBN: 978-1-61208-268-4

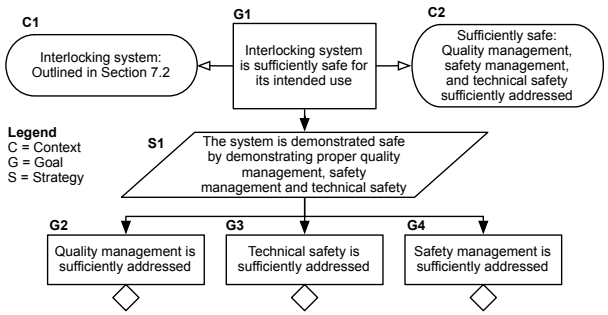

Figure 9. GSN Safety Case (excerpt)

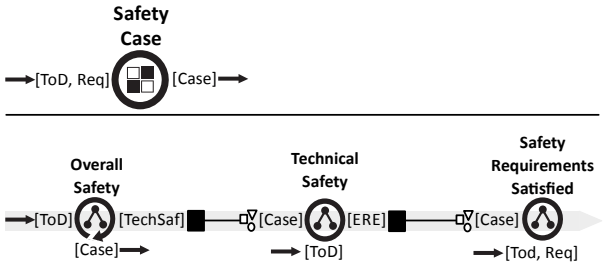

Figure 10. Safety Case – Composite

sufficiently safe based on satisfactory quality management, safety management and technical safety.

The *Technical Safety* pattern was selected for arguing satisfactory technical safety. A strategy defined by the pattern is to explicitly address all risks associated with the system (contrary to an implicit safety demonstration). The explicit risk demonstration strategy is here intended to be detailed by the *Safety Requirements Satisfied* pattern that may be used to structure the argument that all requirements are satisfied.

B. Pattern Instantiation

Fig. 9 presents an excerpt expressed in GSN [13] of the safety case provided upon instantiation of *Overall Safety*, *Technical Safety* and *Safety Requirements Satisfied* according to their instantiation rules.

The safety case provided upon pattern instantiation describes a decomposable safety demonstration, annotated in GSN [13], arguing that the system design is sufficiently safe for its intended purpose. One of the decomposed parts of the safety case put forward a claim that the system is safe given that the safety requirements are correct, suitable and satisfied. Further, compliance to identified safety requirements is shown by referring to the properties of the design as defined by the design models. The design models act as supporting evidences to claims put forward in the safety case. The safety case contains the claims that must be supported by evidences and that once shown provides assurance that the design is safe.

C. Pattern Composition

Fig. 10 defines the composite *Safety Case*. It specifies that the input to the composite is ToD (short for Target of Demonstration) and Req (short for Requirements). The output of the composite is identified as Case and represents the output provided when the pattern *Overall Safety* is instantiated.

A part identified as TechSaf (short for Technical Safety) of the *Overall Safety* patterns is detailed (specified by the *details* operator, the "black box" represents the output that is detailed and the small icons represent the output that details) by the *Technical Safety Pattern*. A part identified as ERE (short for Explicit Risk Estimation) of *Technical Safety* is detailed by *Safety Requirements Satisfied*.

IX. Composite Pattern Solution

A. Pattern Composition

Fig. 11 defines the composite *Pattern Solution* that combines all patterns applied in the case.

Using the *satisfies* operator *Pattern Solution* specifies that the instantiation of output parameter S of the *Dual Modular Redundant* shall satisfy the instantiation of Req of the composite *Requirements* (defined in Fig. 6). Further, the *demonstrates* operator constrains the instantiation of the output parameter Case of *Safety Case* to be a safety demonstration for S of *Dual Modular Redundant*. It is also specified that the instantiation of parameter S-Req (representing an element of the parameter set named Req, see Fig. 6) of the composite *Requirements* is assigned to the input parameter Req of *Safety Case*. The instantiation order of the patterns is defined by the grey arrow in the background that indicates that the *Requirement* composite shall be instantiated first, then the *Dual Modular Redundant* and the *Safety Case* patterns may be instantiated in parallel.

B. Pattern Instantiation

The composite *Pattern Solution* described in Fig. 11 is a pattern and may thus be applied on several cases, e.g., for developing an interlocking system for a station similar

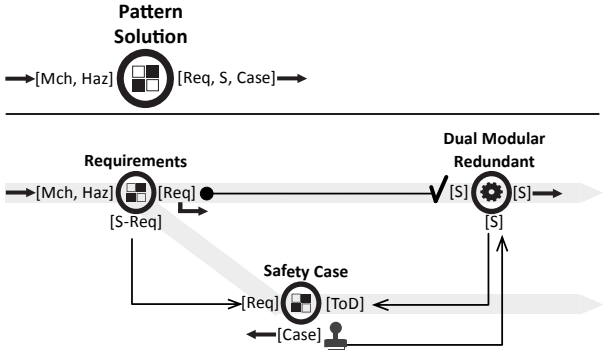

Figure 11. Pattern Solution – Composite

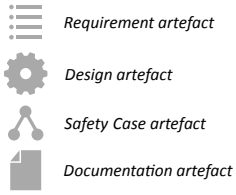

Figure 12. Icons for denoting different types of artefacts

to the one described in this article. In order to describe a specific application of a composite pattern, annotations for specifying the instantiation of parameters may be added to the composite pattern diagram.

Fig. 12 illustrates the different icons and the respective type of artefacts that they represent that are used to specify parameter instantiation.

Fig. 13 is identical to Fig. 11 with the addition of annotations specifying the instantiations of parameters. An icon symbolising the type of artefact that is referred to and a string identifying the referred artefact illustrates an artefact reference. A dotted line connecting an artefact reference to a parameter specifies that the referred artefact instantiates the parameter.

The instantiations illustrated in Fig. 13 refer to the artefacts that are described in this article rather than the full version of these artefacts as provided in [12].

When every composite pattern diagram that is applied during development is annotated with their instantiations, the traceability between the input and output of every pattern and between patterns are provided.

X. Related Work

To the best of our knowledge, there exists no other pattern-based method that combines diverse kinds of patterns into compositions like SaCS. SaCS has been conceived to facilitate efficient development by the support of patterns, separation of concerns in the style of pattern languages, a clearly defined process and application of acceptable development practices as required by safety standards, and at the same time documentation and visualisation in the manner of system modelling.

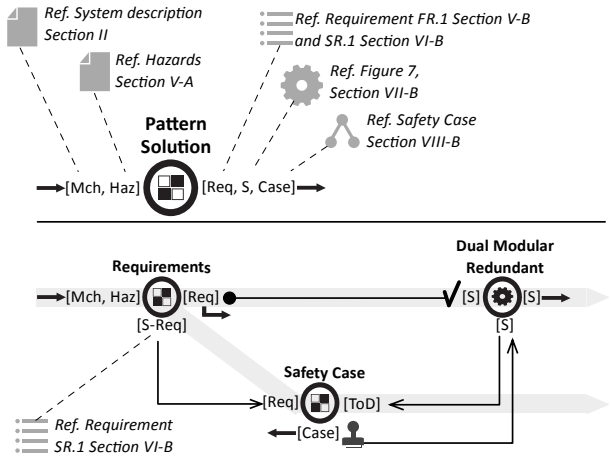

Figure 13. Pattern Solution – Composite (annotated with instantiations)

The concept of systematically applying a set of patterns is inspired by the work of Alexander et. al [6] on architecture of buildings. Important sources of inspiration applicable to development of software based systems are pattern approaches for: requirements elicitation [14], [15], software design [7], [8], [16], and safety demonstration [17], [13]. Two challenges associated with the referenced pattern approaches are that: the integration of patterns is detailed informally; each pattern approach only reflects on one perspective important when developing critical systems, e.g., software design [8]. SaCS offer the ability to combine different kinds of patterns and detail the combination.

The notation for detailing the application of patterns and the focus on establishing a complementing set of development artefacts offered by SaCS are inspired by safety domain needs on providing assurance. International safety standards, e.g., [2], [4] express requirements related to assurance. While safety standards to a large degree describe what are the required practices to be performed during development, SaCS provides guidance on applying accepted safety engineering practices. The European railway regulation [18] and the associated guideline [19] on common safety methods for risk evaluation and assessment has influenced the work on defining SaCS patterns. While the guideline [19] addresses the railway domain, SaCS may be applied in different domains by selecting among the available patterns those that are accepted within a given domain.

XI. Conclusions

We have demonstrated how the conceptual design is instantiated from several SaCS basic patterns within a specific SaCS composite (Fig. 13). Each constituent basic pattern of the composite has clearly defined inputs and outputs and provides guidance on instantiation through defined instantiation rules (these are detailed in [12]). Operators for composition define the combination of instantiation results from several patterns. The composite pattern details: the identifier and type of every pattern applied in order to derive the conceptual design; the pattern instantiation order; and the flow of data through the network of patterns giving traceability between development artefacts.

The conceptual design is built systematically in manageable steps (exemplified in Sections V to IX) by a clearly defined process for pattern selection, instantiation and merging of results (by pattern composition). The conceptual design (fully described in [12]) is consistent (see Section IV). The required triple is provided by the instantiation of the parameters *Req, S*, and *Case* of the composite pattern

 ISBN: 978-1-61208-268-4

illustrated in Fig. 13. In [12] we argue that the triple completely specifies the required function. We also argue that the design (instantiation of S in Fig. 13) correctly specifies the fulfilment of requirements (instantiation of Req in Fig. 13). The conceptual design is expressed in a form that we think is easy to understand (textual descriptions, UML diagrams, and GSN [13] diagrams) and that is easy to detail or reuse. We also claim that the conceptual design may be easily detailed into an implementable system.

XII. Acknowledgments

This work has been conducted and funded within the OECD Halden Reactor Project, Institute for energy technology (IFE), Halden, Norway.

References

[1] A. A. Hauge and K. Stølen, "A pattern-based method for safe control systems exemplified within nuclear power production," in *Lecture Notes in Computer Science*, vol. 7612, September 2012, pp. 13–24.

[2] IEC 62278, "Railway Applications – Specification and Demonstration of Reliability, Availability, Maintainability and Safety (RAMS)," IEC 62278, ed. 1.0, International Electrotechnical Commission, 2002.

[3] IEC 62279, "Railway Applications – Communication, Signalling and Processing Systems – Software for Railway Control and Protection Systems," IEC 62279, ed. 1.0, International Electrotechnical Commission, 2002.

[4] IEC 62425, "Railway Applications – Communication, Signalling and Processing Systems – Safety Related Electronic Systems for Signalling," IEC 62425, ed. 1.0, International Electrotechnical Commission, 2007.

[5] IEC 61025, "Fault Tree Analysis (FTA)," IEC 61025, ed. 2.0, International Electrotechnical Commission, 2006.

[6] C. Alexander, S. Ishikawa, M. Silverstein, M. Jacobson, I. Fiksdahl-King, and S. Angel, *A Pattern Language: Towns, Buildings, Construction*. New York: Oxford University Press, 1977.

[7] F. Buschmann, K. Henney, and D. Schmidt, *Pattern-Oriented Software Architecture: On Patterns and Pattern Languages, Vol. 5*. Wiley, 2007.

[8] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, 1995.

[9] OMG, "Unified Modeling Language Specification, version 2.4.1," Object Management Group, 2011.

[10] M. S. Lund, B. Solhaug, and K. Stølen, *Model-Driven Risk Analysis: The CORAS Approach*, 1st ed. Springer, 2010.

[11] E. R. Tufte, *The Visual Display of Quantitative Information*, 2nd ed. Graphics Press, 5 2001.

[12] A. A. Hauge and K. Stølen, "A Pattern Based Method for Safe Control Conceptualisation – Exemplified Within Railway," Institute for energy technology, OECD Halden Reactor Project, Halden, Norway, Tech. Rep. HWR-1037, 2013.

[13] GSN Working Group, "GSN Community Standard Version 1," York, England, 2011.

[14] E. Hull, K. Jackson, and J. Dick, *Requirements Engineering*. Springer, 2004.

[15] M. Jackson, *Problem Frames: Analyzing and Structuring Software Development Problems*. Addison-Wesley, 2001.

[16] R. S. Hanmer, *Patterns for Fault Tolerant Software*. Wiley, 2007.

[17] R. Alexander, T. Kelly, Z. Kurd, and J. McDermid, "Safety Cases for Advanced Control Software: Safety Case Patterns," University of York, York, UK, 2007.

[18] European Union, "Commission regulation (EC) No 352/2009 on the adoption of a common safety method on risk evaluation and assessment as referred to in Article 6(3)(a) of Directive 2004/49/EC of the European Parliament and of the Council," Official journal of the European Union, 2009.

[19] ERA, "Guide for the application of the Commission Regulation on the adoption of a common safety method on risk evaluation and assessment as referred to in Article 6(3)(a) of the Railway Safety Directive," European Railway Agency, 2009.

 ISBN: 978-1-61208-268-4

Patterns in Safety System Development

Jari Rauhamäki, Timo Vepsäläinen and Seppo Kuikka
Department of Automation Science and Engineering
Tampere University of Technology
Tampere, Finland
jari.rauhamaki|timo.vepsalainen|seppo.kuikka@tut.fi

***Abstract*—Development of safety systems for modern industrial control applications is challenged on the one hand by ever growing systems and on the other hand by increasing cost pressures. That is, design process efficiency is a crucial aspect. How to efficiently utilize existing engineering knowledge and document suitable approaches to the common problems of the domain? Design patterns provide a design process with solutions. Design patterns can represent existing knowledge from past projects or illustrate solution blueprints inspired indirectly, e.g., by safety standards. Thus, they provide a designer with support for design decisions during a development process.**

Keywords-design pattern; safety; control system; engineering

I. INTRODUCTION

Safety awareness is constantly increasing across engineering disciplines, regulative governing bodies as well as customers. This trend results in an increasing demand for higher safety integrity levels as well as broadens the product spectrum in which safety systems are deployed. On the other hand, safety system engineering has a constantly increasing need to make the design process efficient in terms of schedule and cost. These issues lead to pressure to increase the efficiency of the safety system development process.

Engineering industry produces vast amounts of tacit and explicit knowledge during customer and R&D projects. This knowledge is a valuable resource that can be used to increase efficiency when available in a suitable format. Explicit project knowledge is typically left as is, i.e., produced knowledge is archived, but it is not indexed or otherwise edited to be easily accessible. Engineers can access the information, but they need to know exactly the project id, subsystem, diagram etc. to locate the existing solution to the problem they are working with. In the context of safety system development explicit existing knowledge could be, for example, a solution to arrange communication between safety-critical and non-safety-critical subsystems according to a safety standard. Tacit knowledge is another source of valuable engineering knowledge. Tacit knowledge is knowledge of individuals or organizations, not available in explicit documented format. In a context of safety system development tacit knowledge could be for instance a solution model of an engineer to a certain problem.

Development of a safety system is bureaucratic and costly, typically regulated by legislation, regulations and standards, which set requirements for the development process. Typical requirements are sets of certain safety functions that need to be implemented in the system and collections of methods and techniques that need to be utilized to achieve sufficient safety integrity levels of the safety functions to reduce risks into a tolerable level. The standards, legislation and regulations require various matters, but give little to no solutions on how these requirements can be fulfilled not to mention guidance for practical safety function implementation.

Our proposed solution for the problems above is application of design patterns in the field of safety system development. Design patterns document solutions to problems commonly encountered and they have proven their value in engineering disciplines such as software engineering [1]. In software engineering large amounts of patterns have been identified and documented.

The contribution of this article is to show how design patterns could benefit the engineering process also the in domain of safety system development. We indicate the rationale to use design patterns in the safety system engineering domain, which is not similar to traditional software engineering though some reasons of use are obviously the same. Problematic issues related to patterns in context of safety system engineering are discussed to provide a broader viewpoint. This also provides a premise and rationale for further studies considering the topic.

The article is organized as follows. In section II we provide background information on design patterns and the domain. Section III presents related work and positions the research. In section IV we present a generalized model of a development process in which safety related aspects are involved and illustrate pattern usage in such a process. In section V the justification for the usage of design patterns in context of the safety system development is discussed in detail. In section VI, the challenging issues of design pattern usage in the domain are pointed out. Sections VII and VIII discuss future work and conclude the article respectively.

 ISBN: 978-1-61208-268-4

II. BACKGROUND

A. Patterns in engineering

The concept of design patterns originates from Christopher Alexander's book: A Pattern Language: Towns, Buildings, Construction [2]. The book illustrates 253 patterns considering architecture, urban design and community habitability. Thus, the roots of design patterns are originated in a domain that has been studied and used for hundreds of years. This illustrates the original nature of design patterns, which is to document solutions identified from real world applications.

Alexander defines design patterns as abstracted solutions to recurring design problems in a given context [2]. This definition is also adopted by the Design Patterns: Elements of Reusable Object-Oriented Software [3], which considers design patterns in the domain of software engineering. The definition includes the three main elements of a design pattern: context, problem and solution. Patterns illustrate solutions to problems that can be applied, in a suitable context, many times but never end up with completely identical solutions.

An analogy for pattern solution application can be found in interior furnishing. An apartment building may have dozens of apartments with the same floor plan, but none of the apartments is similar in interior decoration. When residents move in an apartment, they furnish it, i.e., let us assume they apply an imaginary "Furnish for habitability" pattern. The context of the pattern is an unfurnished and empty apartment, the problem is the low habitability of an unfurnished apartment and the solution is to furnish the apartment with furniture, textiles and other decoration elements to improve habitability. As none of the apartments have identical furnishing the "Furnish for habitability" pattern has been applied multiple times but ending up with a distinct outcome each time.

Process patterns illustrate processes used to complete a task. The purpose is to divide the execution of a task into steps and provide instructions how to execute the steps to complete the whole task. [4]. Process patterns also represent the context, problem, solution paradigm.

B. Two kinds of control systems

Safety systems often, though not always, co-exist and sometimes also co-operate with ordinary control systems. A control system is a system consisting of sensor(s), logic(s) and actuator(s). In this sense, a control system is similar to an E/E/PE (Electrical/Electronic/Programmable Electronic) safety system.

The purpose of a control system is, however, different from the purpose of a safety system. The main purpose of a control system is to control a machine or a process to produce a desired output, e.g., rolls of paper or printed circuit boards. The purpose of a safety system is to ensure the safety of humans, environment and machinery itself, i.e., the main concern of a safety system is to prevent the realization of hazards.

The problem is that the tasks of these two systems differentiate in purpose. A safety system tries to retain the system in a safe operation state whereas a control system tries to maximize the output of the system. To carry out the tasks the same process variables need to be considered. The situation is even worse as the systems often have opposite preferences considering the state of the system.

C. Some patterns for functional safety system development

In our recent research projects, we have focused on safety system principles and architectures. It was noted that patterns for safety systems are not available although patterns for the related domains are considered (see section III). However, we see potential in patterns in the domain of safety system development. During the recent projects, we have identified and developed patterns for the development of safety systems. Table I summarizes the patterns published in VikingPlop'12 [5].

The patterns consider various aspects of safety systems. The Separated safety and the Productive safety patterns consider the co-existence and distribution of liabilities between the safety and main control systems. The Separated override, De-energized override and Safety limiter patterns illustrate approaches to override the main control system with a safety system. The approaches have distinct redeeming features and downsides. For instance, the Separated override pattern emphasizes separation between the systems whereas the Safety limiter pattern allows cooperation between the systems and reduces the amount of needed hardware. The Hardwired safety pattern proposes usage of a hardwired safety system instead of a software based solution in a suitable context.

TABLE I. FUNCTIONAL SAFETY SYSTEM PATTERNS [5]

Pattern	Description
Separated safety	Development of a complete system according to safety regulations is a bureaucratic and slow process. Therefore, divide the system into basic control and safety systems and develop only the safety system according to safety regulations.
Productive safety	A control system utilizes advanced and complex corrective functions to keep the controlled process in the operational state. These functions are very hard to implement in a safety system. Therefore, implement the corrective functions in a basic control system and use simple(st) approach for the safety system.
Separated override	A safety system must be able to override a basic control system whenever systems control same process quantities. Therefore, provide the safety system with a separate actuator to obtain a safe state.
De-energized override	A safety system must be able to override a basic control system whenever systems control same process quantities. Therefore, let the safety system use de-energization of the basic control system's actuator(s) to obtain a safe state.
Safety limiter	A safety system must be able to override basic control system whenever systems control same process quantities. Therefore, disengage the basic control system completely from the actuator and let the safety system control the actuator. Route the output of the basic control system to the safety system and let the safety system treat the control value so that safe operation is ensured.
Hardwired safety	Development of safety-related application software for simple safety function is bureaucratic, time consuming and costly. Therefore, instead of a software-based solution, use a hardware-based safety system.

 ISBN: 978-1-61208-268-4

III. RELATED WORK

Design patterns have been studied and documented in the field of software engineering extensively covering for example object-oriented software [3] and [6], Pattern-oriented architecture [7] and [8], enterprise applications [9], [10], and service-oriented architecture [11]. These books concentrate on software engineering for desktop and server-side applications and architectures. Though these patterns may be usable in safety system development they are not focused on safety aspects.

Fault tolerance is a part of safety system design as safety systems should preferably be fault-tolerant to be able to operate under fault conditions and ensure safety. However, fault-tolerance is not a sufficient condition for safety. Fault-tolerant software can be hazardous in a safety system if the functionality of the software is hazardous, e.g., due to erroneously set requirements. Design patterns for fault-tolerant software systems have been introduced, for example, by Hanmer [12].

E/E/PE safety systems include both hardware and software components. Armoush [13] and Douglass [14] introduce design patterns covering software and hardware aspects of safety systems. The presented patterns are focused on redundancy, which, again, is an approach to increase reliability and fault-tolerance of a system.

Eloranta, Koskinen, Leppänen and Reijonen [15] have studied distributed machine control systems and documented patterns for the design of such systems. Some of the patterns are also related to functional safety aspects. The application domain of the above patterns is closely related to our design patterns considering safety system development and architecture [5].

Koskinen, Vuori and Katara have studied and developed process patterns for the application of the IEC 61508-3 standard. In their article [4] they stated that process patterns can speed up the training of inexperienced engineers and remove ambiguities typically related to safety standard application. This provides additional support for the usage of patterns in the domain of safety systems.

Riehle [1] properly points out three main usage areas of design patterns in current software industry practice. These areas are communication, implementation and documentation. In this article, we consider how these usage areas are transferable into safety system engineering.

IV. GENERALIZED SAFETY SYSTEM DEVELOPMENT PROCESS

In this section, a generalized process for the development of safety-related E/E/PE systems is illustrated. The purpose is to provide an idea about how pattern usage can relate to such a process. The illustrated process is inspired by the IEC 61508-1 overall safety lifecycle [16] and eight steps to safety [14].

Development of a safety system begins with the definition of the overall *scope* of the EUC (Equipment Under Control) and the concept of the system. In this phase understanding about the system and its environment is built. In this context process patterns can be used to identify EUC related aspects (e.g., what are typical characteristics of, for example, bending machines) and typical machinery concepts.

Hazard and risk analysis follows the scope definition. Hazard and risk analysis forms a significant part of the development process as the results directly impact on the coverage of the safety system and selection of safety measures and safety integrity levels. Patterns can be used to identify typical hazards related to specific systems (machinery type) and processes (operations executed by the machinery). Process patterns can be used to describe and interpret the phases of the hazard and risk analysis as required by the followed standard.

When the risks are defined, the *requirements* for the risk mitigation methods are documented in the requirement specification phase. This includes the definition of the risk mitigation methods, safety functions, and the non-functional requirements and safety integrity levels related to them. Patterns can be used to document typical approaches to mitigate risks with the positive and negative effects related to the approaches thus providing support for decision making. The requirement specification phase can also be supported with process patterns. For instance, the Software Safety Requirements Specification pattern in [4] illustrates a requirement specification process mined from the IEC 61508 to provide help and document the sub phases of this development phase.

As the requirements for safety measures and functions are defined the process can continue on to the *realization* of safety system. The phase consists of design and implementation of the safety system. In this phase of development process, patterns have value as the level of abstraction suits well to describe solutions to design and implementation problems. The patterns provide designers with documented solutions to commonly encountered safety design problems. However, the patterns also provide information about consequences related to application of them. This enables an engineer to select the most suitable solution by justifying the consequences. For instance, the three override patterns described in Table I illustrate different approaches to a design problem where a safety system should be able to override a control system. Each of the solutions has their own consequences and the designer can choose the one that is the best fit for the system under development. Process patterns can support the realization process by, e.g., providing support to carry out the recurring phases of development such as the modification or architectural design of the software [4].

The implementation part of the *realization* phase can be supported with design patterns as in this phase engineers encounter a large number of common problems where design patterns are able to provide solutions. The patterns applied in the implementation phase often represent a lower level of abstraction and provide focused solution models to lower level implementation problems.

The rest of the development process relate to *validation, verification, testing, installation and maintenance* aspects. Process patterns for validation and verification document and help to follow the processes. For instance, patterns for

 ISBN: 978-1-61208-268-4

validation and verification in context of the IEC 61508 are provided in [4]. Maintenance of long life cycle systems benefits from the usage of design patterns as known solution models are used.

V. RATIONALE FOR DESIGN PATTERN USAGE IN SAFETY SYSTEM ENGINEERING

As illustrated in Section II/A design patterns have redeeming features in context of the software engineering domain. However, the software engineering domain, at least desktop software engineering, is different from safety system engineering. Of course, software systems are a part of modern safety systems, but the nature of a pure software system is distinct from a safety system. In the following subsections rationale for the usage of design patterns in safety system engineering is discussed.

A. Ability to avoid physical damage

Normal application/desktop software, run on a personal computer, mobile device, server or similar device, has limited possibilities to interact with its environment. Potential physical risks associated with such devices and applications are, for example, overheating, electric shock, battery malfunctions and fans none of which are directly controllable by the application software ran in the system. That is not to say that application software cannot be critical. For example, a failure of banking, insurance or other large scale business system may inflict massive losses to its owners in form of revenue or work contribution losses and is thus considered critical. However, no (direct) human, environmental or machinery related hazards exist in such cases.

Systems in which safety systems are deployed are able to cause hazardous situations for humans, environment and hardware by their nature (if not, no safety control system would be required). Industrial and machinery control systems operate actuators (e.g., fans, valves, and heaters) process devices (e.g., conveyors, robots, and guillotines) and substances (e.g., toxic chemicals or hot fluids) that are hazardous for humans, environment and the systems itself. As the safety systems are dedicated to mitigate risk related to such machinery they are expected and required to have certain level integrity to carry out the safety functions.

Design patterns document good approaches, practices and solutions common in safety system development. This provides designers with tried solutions to problems as well as removes the need to reinvent the wheel thus resulting in a more productive development process as well as solutions with a justified approach. The development burden is decreased and the designers can focus on details as patterns describe the main solution model.

B. Experience as a valuable resource in safety system development

In the field of safety system engineering, well-tried solutions are welcome as they have additional empirical data to back up applicability. By identifying patterns from existing projects and designs and making the solutions explicit in patterns, experience can be transferred from one engineer to another. Design patterns support the illustration of experience in explicit format by requiring the pattern writer to consider different aspects of the solution. This work is carried out in consideration on the context in which the solution can be used, consequences and the resulting context related to the solution. Patterns document (or at least they should document) also negative consequences, preconditions and assumptions related to pattern application. This provides engineers with a foundation to use or not to use certain solutions and compare them against each other to select the best approach for the problem under consideration.

A good approach is to document the proven solutions of past projects into patterns to be used in forthcoming projects. In this way, the patterns are directly related to the domain, they can be written to solve a dedicated problem and the consequences are known. That is not to say one should limit to such patterns only. Third-party patterns may provide fruitful insight into other kinds of solution models and open new kinds of approach possibilities to solve a certain kind of problem with more desirable consequences.

Experience illustrated in format of patterns, also provides a name for the solutions and approaches. This enables the usage of patterns as a part of communication [1], but requires that the patterns have reached awareness of the engineering community using them. When this point is achieved, patterns can be used in communication to illustrate the solutions and approaches described in design patterns. For example, safety system engineers could discuss about how to override a control system with a safety system: "I think separated override [5] would be a good approach in this situation.", "I disagree; I find separated override an excessive action as it would require an additional safety actuator. Maybe we should consider de-energized override [5] instead", "That is true, de-energized override is a more cost-effective approach in this case."

C. Alleviating bureaucracy

Development of safety systems is regulated by directives, legislation and standards such as [17], [18], [19]. Such documents are written partly from a legislative point of view, are too generic to cover various applications and domains, and do not (want to) strictly enforce a certain approach. These aspects restrict the documents from providing solution models. Rather such documents require various techniques, methods, and processes to be used in the development of safety systems, but give minor importance on examples or other guidelines for any specific implementation. In addition, the documents are massive, often hundreds of pages long, which makes finding solutions difficult. This does not mean standards etc. are useless; they just have a different view to safety systems compared with patterns. The standards provide a framework that is applied in a certain way to develop the system. The framework describes methods and techniques to develop safety systems and, e.g., define what to verify and validate when a safety system is being developed. This is certainly a valuable aspect in safety system development.

The purpose of patterns in this context is to supplement the standards and document the solution models and

 ISBN: 978-1-61208-268-4

approaches compliant with the given requirements. The IEC 61508-3 [20], for instance, illustrates a number of techniques and measures to be used in the development of safety-critical software. However, little information on how these techniques shall be used and what kind of solution models they (may) produce is given. Especially safety-related standards can prove hard for a person with limited experience in the development of safety systems [4]. With design patterns, solutions and approaches to implement techniques and measures required in standards can be documented, which illustrates the usage of design patterns as a source of implementation [1]. Process patterns can be used to capture the recurring tasks in the development of a safety system [4].

In context of safety system development the value of patterns is fully established when patterns are mined from a system that has already found compliant with a safety standard. This adds confidence in the solution model validity in context of the considered safety standard. Such patterns can increase development process efficiency as the solution model can be used in other systems with a fairly good confidence as long as the context described in the pattern matches the context in which the pattern is applied. The solution, approach or method once approved in a certification process or assessment for standard compliance, for instance, is useful as it provides at least the main solution framework for the problem under consideration.

Development of a safety system also requires extensive documentation. This is required, e.g., to illustrate compliance with a standard considering development of a safety system or an informal document illustrating the safety foundation of a system, which can be used as a part of safety assessment of a system. Design patterns can be used for documentation purposes [1]. The applied patterns and roles of the patterns can be marked in a document (e.g., in a diagram). For an experienced pattern user this quickly indicates the type of solution used (described by the pattern). The need for reading textual representations decreases as the reader can obtain the information on the roles of the system elements directly from the diagram. In an informal supplementary documentation usage of well-known safety related patterns can be justified. The reader is able to identify the patterns applied and assess their suitability in context of the safety system under consideration. However, in context of the legal safety system documentation, the usage of patterns in documentation does not remove the need for textual representations as the usage of pattern notation in the documentation does not cover the whole functionality and all the aspects of the applied solution.

D. Co-existence of control and safety systems

A safety system often co-exists with a main control system as stated in section II. Although safety and control systems are designed to be separated, they often need to be connected some way (e.g., to share state and operation information). This aspect further increases the amount of work needed to design an operational entity consisting of safety and control systems.

Integration of safety and main control systems is sometimes, especially in context of larger processes, a unique design. The operation and responsibilities of the safety and control systems need to be defined and fitted to operate in harmony. If such a system is repeatedly designed from scratch, a great amount of design work needs to be redone. In such situations the design process may greatly benefit from the reuse of templates [21], model libraries and similar ways of reusing existing designs developed in a specific development environment. However, templates and library solutions as such are not a good fit to document solution models and approaches on a generic level. This is due to the fact that solutions are bound to the implementation environment: the solutions are described in terms of the implementation environment/tool. Such an approach complicates the understanding about the solution on a higher level of abstraction.

Contrarily, patterns provide a format to document solutions on a platform independent level. This enables the documentation of solutions, which can be used in different implementation environments as long as the context and other prerequisites are considered. The benefit of a pattern approach is that one is able to take the idea from a pattern and adapt the principle of the solution to solve the problem in hand, thus increasing the efficiency of the design process.

1) A case for pattern usage in design of safety and control system co-existence

This section illustrates a case for usage of design patterns is design of system in which safety and control system operate the system under control. The functional safety system patterns introduced in Table I illustrate solutions for safety and control system co-existence. The patterns describe approaches to arrange the responsibilities of the systems and override of the control system. The idea is to divide the responsibilities so that the development of the safety system is as lightweight as possible, but the safety system still is retaining full control over the machinery.

A potential design decision flow to utilize the patterns is illustrated in Fig. 1. The figure illustrates the pattern relations of the patterns in Table I. The Separated safety pattern is applied first to the system under development. This decision results separated safety and control systems and only the safety system has to be developed according to safety standards, which decreases the development burden considerable as the control system (which is typically a larger entity than the safety system) can now be developed

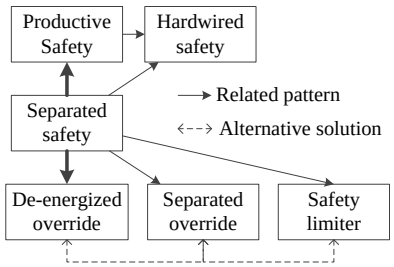

Figure 1. Design flow using functional safety system patterns [5]

without safety standard conformance.

When the system is divided into safety and control system, the Productive safety pattern can be considered. It suggests that the control system implements the basic interlock mechanisms that (try to) keep the system in a normal operational state as far as possible. The interlock mechanisms can be as complex as needed as they do not need to conform to the safety standards. The actual safety functions are implemented on the safety system and they can be rather simple because the control system keeps the machinery in the normal operational state. The safety system can, for instance, only implement an emergency shutdown of the machinery when the control system has failed to retain the normal operational state. This approach simplifies, and thus potentially lowers the cost of, the safety system development and implementation.

As the safety system must be able to drive the system into a safe state regardless of the control system state, the designer needs to implement such functionality. The three override patterns provide three distinct approaches how the safety system can override the control system on the actuator level. The designer can compare the suggested approaches and select the one with most desirable consequences regarding the system under control. For instance, if separation between the safety and control system is the main concern, the Separated or De-energized override pattern is the most appropriate. However, if there is a need to lower the amount of actuators or use advanced safety functionality, the Safety limiter pattern may be a better alternative.

The above workflow illustration also depicts the potential of pattern language utilization. The designer uses a pattern language as a framework and selects the most appropriate patterns to design the system. The pattern language supports the design process by defining relationships between the patterns. The relationships illustrate, e.g., patterns that are applicable after a certain pattern has been applied, conflicting patterns or patterns that solve problems, which may arise when a pattern is applied.

E. Maintainability of common solution models

An important feature of control systems, especially in an industrial domain, is long life-cycles. As safety systems are part of control structures of a system, they also have long life-cycles in similar applications. The maintenance phase of a system may contribute considerably to a large part of system design and development costs when the whole life-cycle costs of the system are considered. Thus the maintainability of a safety system is an important aspect to be ensured during the initial development process of the safety system.

Maintenance of a system is easier if the system is intelligible. Usage of design patterns can improve intelligibility through common vocabulary. If design patterns are used in system development and documentation [1], the maintenance team can more easily understand the system concepts and execute maintenance operations to the system. Naturally this requires that both the developer and the maintenance team know and understand the used patterns. This, in practice, requires either company's internal patterns or widely adopted patterns related to the domain.

VI. CHALLENGES IN DESIGN PATTERN USAGE IN SAFETY SYSTEM ENGINEERING

Patterns have qualities that justify their usage in the development of safety systems. However, some challenging issues can be identified as well. To provide ample insight into patterns the issues of patterns are discussed in this section.

Patterns are not exact. As mentioned, patterns (typically) describe solution on a relatively high abstraction level so that they can be used and implemented in multiple ways. In safety system development exactness and completeness are considered virtues that patterns can, but often do not, provide.

Usage of patterns may lead to *inconsistent understanding* between system developers. A pattern can be implemented in many ways and each person has a unique mindset about a pattern. Thus patterns are not applicable as safety documentation as such. However, when a set of patterns has been used extensively, the patterns may become a part of a communication language that clarifies the ideas shared between individuals [1] and thus may act as a supporting form of documentation.

A developer may *misunderstand pattern solutions* or use them in contexts not suitable for the pattern. A similar issue is naturally related to all situations when documented solutions are applied. One can also misunderstand solutions illustrated in a book, journal article or data of a preceding project.

Patterns are not meant to be detailed illustrations of the solution (though some patterns indeed illustrate details). Instead, they typically provide a *generic framework of the solution*, which the designer can apply in the environment in which the problem is considered. This is one of the strengths of patterns, but it is also a potential issue. A pattern author may have accidentally or intentionally left out some information that would be needed to be fully able to consider all the side-effects of the pattern.

If a pattern reader is *unfamiliar with the domain* the patterns consider, an incorrect overall picture could be adopted. Though patterns consider various aspects of the solution, they cannot take into account all the relevant aspects. In the domain of safety system engineering artefacts relate to each other in complex manners. A single pattern cannot consider all these aspects as it would shift the focus of the pattern. Thus the reader should regard patterns with a healthy sense of criticism when they are applied.

Patterns may encourage designers to *stick with existing solutions*. Often the reuse of solutions is a productive way to go and well-tried solutions are valuable in the field of safety system engineering. However, this should not mean that reuse of solutions is the only way to go. New, more efficient, simpler, and better approaches cannot be developed if old solutions are constantly used. It has to be identified if the design benefits from the reuse of solutions and when one needs to focus on creating a better, novel approach to the problem in hand.

 ISBN: 978-1-61208-268-4

VII. FUTURE WORK

Our future effort is expansion of our safety system development related pattern collection [5] and development of tool support for a semantic search of patterns. The target of the pattern collection expansion is to construct a pattern language that could serve safety system developers. Another aspect is to study the effects of pattern usage in practical development processes. Empirical studies on pattern usage in the development processes of safety systems would provide insight into widening the usage of patterns.

The semantic search for patterns eases pattern discovery. Semantic relations between pattern data are being developed. This enables the search of patterns supported with a semantic deduction engine to identify patterns with similar features and consequences as given in the original search.

VIII. CONCLUSION

In this article, we have illustrated rationale for using design patterns in the development of safety systems. The foundation of usage of patterns lies in the idea of providing a way to document tacit and existing knowledge into an explicit format. When experience is formatted as a design pattern, it can become common knowledge that can serve in documentation, implementation and communication purposes.

Safety systems are parts of critical systems that are able to cause physical damage. The sole purpose of a safety system is to prevent the hazardous situations leading to physical damage. Well-tried solutions and approaches documented in patterns can help in the development of a dependable and cost-effective safety system. Development of safety systems is heavily regulated by standards and legislation, which require methods, techniques and processes to be used, but provide few practical solutions. With design patterns practical solutions can be documented into an intelligible format while providing room for modifiability.

Cooperation between a control and a safety system can prove to be a burdensome task especially if it is made from scratch. This may occur in larger control system projects for large scale unique plants. In such cases patterns provide a valuable engineering resource as they describe solution and approaches on an abstract level. This enables a designer to apply the approach in a suitable way considering the system.

Design patterns also have some drawbacks in context of safety system development. They are not exact and accepted as documentation or proof of compliance. Still patterns can help to improve development process and share knowledge.

REFERENCES

[1] D. Riehle, "Lessons Learned from Using Design Patterns in Industry Projects," in Transactions on Pattern Languages of Programming II, vol. 6510, J. Noble, R. Johnson, P. Avgeriou, N. Harrison, and U. Zdun, Eds. Springer Berlin / Heidelberg, 2011, pp. 1-15.

[2] C. Alexander, S. Ishikawa, and M. Silverstein, A pattern language: Towns, buildings, construction. New York: Oxford University Press, 1977.

[3] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, Design patterns, Elements of reusable object-oriented software. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc, 1995.

[4] J. Koskinen, M. Vuori, and M. Katara, "Safety Process Patterns: Demystifying Safety Standards," Proc. 2012 IEEE International Conference on Software Science, Technology and Engineering (SWSTE), IEEE Computer Society, 2012, pp. 63-71.

[5] J. Rauhamäki, T. Vepsäläinen, and S. Kuikka, "Functional Safety System Patterns," Proc. VikingPLoP 2012 Conference, 2012, pp. 48-68, http://URN.fi/URN:ISBN:978-952-15-2944-3 [retrieved: February, 2013].

[6] E. Freeman, E. Freeman, B. Bates, and K. Sierra, Head first design patterns. O' Reilly & Associates, Inc, 2004.

[7] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, Pattern-oriented software architecture, volume 1: A system of patterns. Chichester, UK: Wiley, 1996.

[8] D. C. Schmidt, M. Stal, H. Rohnert, and F. Buschmann, Pattern-oriented software architecture: Patterns for concurrent and networked objects. 2nd ed.,New York, NY, USA: John Wiley & Sons, Inc, 2000.

[9] M. Fowler, Patterns of enterprise application architecture. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc, 2002.

[10] G. Hohpe and B. Woolf, Enterprise integration patterns: Designing, building, and deploying messaging solutions. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc, 2003.

[11] T. Erl, SOA design patterns. 1st ed.,Upper Saddle River, NJ, USA: Prentice Hall PTR, 2009.

[12] R. S. Hanmer, Patterns for fault tolerant software. Chichester, England ; Hoboken, NJ: John Wiley, 2007.

[13] A. Armoush, Design Patterns for Safety-Critical Embedded Systems. 2010.

[14] B. P. Douglass, Doing hard time: Developing real-time systems with UML, objects, frameworks, and patterns. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc, 1999.

[15] V. Eloranta, J. Koskinen, M. Leppänen, and V. Reijonen, A Pattern Language for Distributed Machine Control Systems. 2010. http://practise.cs.tut.fi/project.php?project=sulake [retrieved: February, 2013].

[16] International Electrotechnical Commission, Functional safety of electrical/electronic/programmable electronic safety-related systems, Part 1: General requirements, IEC, 2010.

[17] European Parliament and of the Council, Directive 2006/42/EC of the european parliament and of the council, vol. L 157/24, 2006.

[18] International Electrotechnical Commission, Functional safety of electrical/electronic/programmable electronic safety-related systems, IEC, 2010.

[19] European Committee for Standardization, Safety of machinery, safety-related parts of control systems, Part 1: General principles for design, 2008.

[20] International Electrotechnical Commission, Functional safety of electrical/electronic/programmable electronic safety-related systems, Part 3: Software requirements, IEC, 2010.

[21] M. Kairaila, Domain-Specific Template-Based Visual Language and Tools for Automation Industry. 2010.

 ISBN: 978-1-61208-268-4

Safety-Related ASIC-Design in Terms of the Standard IEC 61508

Ali Hayek and Josef Börcsök
Chair of Computer Architecture and System Programming
University of Kassel
Kassel, Germany
ali.hayek@uni-kassel.de

***Abstract*—Due to the continuing development of semiconductor structures, it can be allowed nowadays to integrate more robust and high-efficient systems into a very small area of silicon. In such system-on-chip all individual components of a target system can be integrated into a single silicon die at lowest level, which in turn contributes in saving the substantial space and reduces power consumption and production costs. With the consideration of the miniaturization of safety-related systems into system-on-chips, where usually complete redundant architectures along with memory and interfaces are integrated into small silicon structures, many advantages can be taken into account. These advantages extend to all levels of the development cycle. In the present paper, the advantages of the miniaturization of integrated 1oo2D-safety architecture (one out of two with diagnosis) and its safety-aware implementation in terms of the safety standard IEC 61508 are presented.**

Keywords—Safety-related systems;Integrated Circuits; IEC 61508; on-chip redundancy

I. INTRODUCTION

Nowadays embedded System-on-Chip applications are increasingly used in several industrial control processes. Due to the development of silicon structures and thanks to the rapid development of the IP-Core market (Intellectual Property), Application Specific Integrated Circuits (ASICs) are increasingly used in several industrial applications compared to a decade ago. In this regard, one can integrate his own chip functionality concluding whole communication microcontroller units and other digital and analogue components can be today shortly integrated in such circuits. Latter makes from ASICs an interesting platform for realizing safety-related architectures, since those consist of complex redundant components which need to be implemented following stringent procedures and need to offer their functionality under specific conditions. Due to their flexibility of programming and reconfiguring at run-time, Field Programmable Gate Arrays (FPGAs) provide a popular platform for safety-related systems. Thus, the susceptibility of SRAM-based FPGAs to external effects increases with the ongoing miniaturization of silicon structures, such as the susceptibility to single-event upsets (SEUs). For this reason, the usage of SRAM-Based FPGAs in safety critical fields require the adoption of very specific reliability and fault tolerance techniques, in order to protect their functionality against such transient effects. In recent research work [1] a survey of using those FPGAs in safety-related systems was presented. In this paper, the disadvantages of using FPGAs such as the mentioned susceptibility to soft errors and the increasing part costs are solved by targeting ASICs as a platform for integrated safety-related systems. As against SRAM-based FPGAs, the functionality of systems implemented on ASICs is programmed only on time during the design and so permanent and immune to soft errors.

At a glance, this paper deals with the use of ASIC-based system-on-chips in safety-related systems conforming to established safety standards. In our opinion, there are two important points for dealing with this: the safety of the ASIC implementation itself and the safety properties of the hardware description code used to perform the functionality which is translated to the ASIC hardware.

In this paper, we first introduce the safety standards which are relevant for this work. Especially the standard IEC 61508 and its second edition (IEC 61508 Ed. 2) are explained in detail. In addition, the standard DO-254 from the field of aviation is introduced briefly, since it is widely used in the United States of America and has parallels to the standard IEC 61508. Afterwards, the safety-related 1oo2D-architecture is introduced. The heart of this research work is divided in two parts. First, a technical evaluation of using integrated architectures against discrete system solution, which is nowadays conform to the state of the art, is given. Second, an analysis of the use of ASICs according to the standard IEC 61508 second edition is introduced. Latter is divided into two main aspects: modeling and coding methodologies on software and physical measures on hardware. On the one hand, the implementation of standard techniques and measures on ASIC platform is motivated and discussed. This includes techniques for increasing the reliability of such systems like on-chip redundancy and safety-aware placement and routing. On the other hand, the coding methodologies of ASIC programming languages such as VHDL are discussed. In this context, we study the possibility of the use of these languages for realizing safety properties. Coding and verification measures are discussed in this section. Finally, the proposed techniques will be evaluated on ASIC using an example of the 1oo2D-architecture.

In the second section of this paper an overview about the relevant functional safety standards is given. Afterwards the targeted safety-related on-chip 1oo2D-architecture and its advantages are introduced. In section IV the software methodologies for safety-related on-chip systems are discussed. Section V presents hardware-based measures for the physical implementation of safety-related systems-on-chip. Section VI outlines the paper with a short conclusion and future prospects.

 ISBN: 978-1-61208-268-4

II. FUNCTIONAL SAFETY STANDARDS

Norms and standards for safety-related systems are not new; MIL-STD-882 from the US Department of Defense (DoD) developed in 1963, is the first standard in this area. This standard is derived from the military area. The idea was to improve the safety of weapons and to keep the risk of undesired accidental damage to people and the environment in an acceptable range. In 1998, a new paradigm was been developed with the standard IEC 61508 which has been associated with a new definition of the term “functional safety”. The main innovation is that in the context of functional safety only the safety features of a system are considered. The other non-safety-related functions are in accordance with the standard IEC 61508 only a part of quality management. In the following sections the standard IEC 61508 is primarily introduced, as the safety standard applied in Europe. Furthermore, a short insight into the standard DO-254 is given. Latter is irrelevant for the current research work but could be applicable in future considerations.

A. IEC 61508

The standard IEC 61508 [2] is a standard in the area of safety technology, which was developed by the International Electrotechnical Commission (IEC), an international standards organization, and first released in 1998. It is titled "Functional safety of electrical / electronic / programmable electronic systems" (E / E / PES). The standard is also known as basic safety standard, because it is application independent, but it addresses all safety functions of a system. It is regarded as basis for further application-specific standards. The standard IEC 61508 is limited on electrical and electronic programmable electronic safety-related systems. In this context, it defines four safety-integrity-level, so-called SIL. This applies: the higher the SIL, the safer the E / E / PES. The specification of SIL provides developers, producers and customers a clear and unequivocal basis for negotiating basic aspects of safety integrity.

The standard IEC 61508 is seen as a basis for further standards. In this regard, the standard gives sufficient flexibility for technical respectively technological innovations. The standard is also kept consciously abstract and flexible in regard to the methods to cover the requirements on hardware and software, while the requirement is clearly defined, it leaves ample room for researchers and developers to apply own implementation ideas and makes them free of the need to comply with stringent rules. In the context of in this research work considered ASICs, it gives for example a note on the requirements for using ASICs in safety-related applications. Furthermore, innovations find their way into new drafts of the standard. While using on-chip redundancy (OCR) was in the first standard version still unconsidered, it was contained in the following draft standards, and could thereby be taken into consideration by developers and certification bodies in terms of the standard. The main changes in the second edition of the standard are presented briefly in the next section.

B. IEC 61508 Ed. 2

Generally, the standard IEC 61508 is divided into 7 parts and provides a guide for developing safety-related systems. The specific implementation of the requirements is flexible. In the present work, the requirements for the development of safety-related systems based on ASICs conforming to the second edition of the standard IEC 61508 [3] are mainly considered. In the following the main novel features in the second edition are described briefly. In the next sections the applicable features for ASICs are argued in detail:

- New requirements for Application Specific Integrated Circuits (ASICs)
- Clear definition of Systematic Integrity Compliance Route (Route 1_S, Route 2_S and Route 3_S)
- Clear definition of Hardware Integrity Compliance Route (Route 1_H and Route 2_H)
- New definition of Proven-in-Use terms

C. DO-254

The standard DO-254 [4] is performed under the title "Design Assurance Guidance for Airborne Electronic Hardware" and is a standard for the development of complex electronic hardware systems in the aviation field. It was developed in April 2000 by the RTCA (Radio Technical Commission for Aeronautics) and EUROCAE (European Organization for Civil Aviation Equipment) and is today carried as a standard for the development of complex electronic hardware in the aviation field, of both by the American aviation authority FFA, as well as the European Aviation Safety Agency EASA demanded. The DO-254 is, like IEC 61508, a safety standard, which is also application independent, but specifically refers only to the hardware development.

Like in IEC 61508, it includes no binding guidelines for the direct implementation, but it lists conception guidelines for the intended certification of the whole development process. Outside the norm there are further standards, such as DO-178B [5], which deals exclusively with the software development in the aviation field. The standard specifies a complete documentation during development and takes into account the life cycle of the product. A consistent and binding implementation of the product life cycle from concept to decommissioning, as specified in IEC 61508, however is not requested.

III. INTEGRATED 1OO2D-ARCHITECTURE

The standard IEC 61508 gives a basis for realization of qualitative and quantitative analysis in areas of reliability and safety. Particularly, architectural measures were introduced, which are necessary to provide a desired safety or reliability such as the introduction of hardware fault tolerance, system redundancy and implementation of diagnostic and monitoring elements. Considering the use of hardware redundancy and hardware fault tolerance, MooN-system-architectures (M out of N) are usually targeted. The name describes the system architecture and the possible degradation behavior by fault

 ISBN: 978-1-61208-268-4

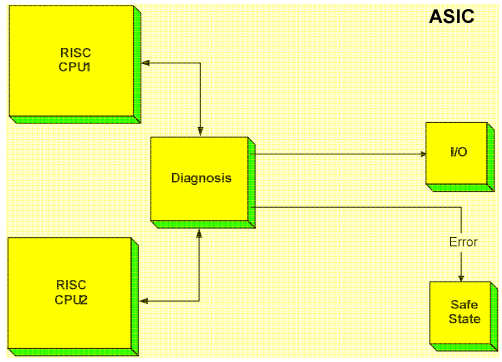

Figure 1. 1oo2D-Architecture

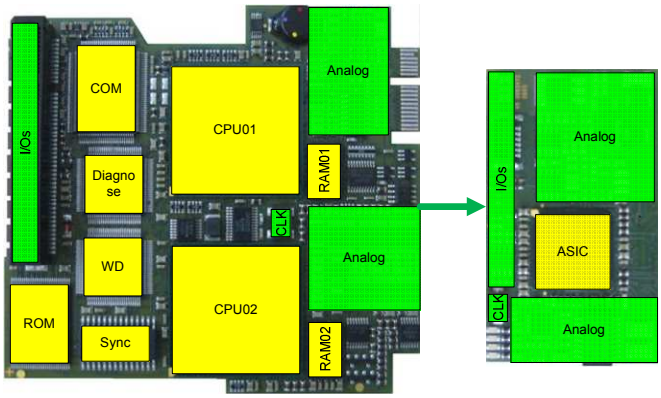

Figure 2. Integrated 1oo2D-Architecture

behavior. By this is meant that M out of N channels of a safety-related system are sufficient to transfer the system into a safe state. The lowest form of this redundancy is the present 1oo2-architecture. This represents a safety-architecture with hardware fault tolerance HFT = 1. In the following, the 1oo2-architecture and its advantages are discussed.

A. 1oo2D-Architecture

The 1oo2-safety-architecture (one out of two) is one of several system architectures which are described in the standard IEC 61508. This kind of architecture is composed of two parallel channels. If both channels of the system have a failure, the system loses the safety function. Additionally the 1oo2D-architecture is a 1oo2-architecture with integrated diagnostic units. Details of 1oo2D-architecture are explained in the following.

As mentioned above, a 1oo2D-architecture describes a complete system or a subsystem, consisting of two channels (main and redundant channel) with the same function. In case of failure, only one of the two channels is required to transfer the system into a safe state. A diagnostic unit compares continuously the results of both channels. If there is an inequality, this points to a faulty channel. The diagnostic unit signalizes this to the two channels and the faultless channel transfers the complete system to a safe state. The complete system remains therein until the fault is corrected and both channels are functioning again. If both channels fail independently of each other or by a fault with a common cause, the complete 1oo2D (sub-) system is not able to trigger the safe state. For such cases, external diagnostic measures such as watchdog, temperature and voltage monitoring are used to transfer the system into a safe state.

In Fig.1 below the block diagram of general 1oo2D-architecture on ASIC is given.

B. Advantages of the inetgrated 1oo2D-Architecture

Besides the 1oo2-architecture other architectures, such as 1oo3-, 2oo3- or 2oo4-architectures are used. Comparing the parameters for calculating the average probability of failure on demand (PFD) or per hour (PFH) the 1oo2-architecture arise very good values under consideration of minimal redundancy. The 1oo2-architecture has without any doubt their eligibility for systems that can be transformed to a safe state as needed.

The main target of the present work is the integration of the 1oo2D-architecture on a single ASIC. In this case, the redundant processor channels as well as diagnosis units are integrated into a single chip. Fig. 2 shows an example for the difference between an integrated and a discrete 1oo2D-architecture.With reference to the architecture shown in Fig. 2, the following obvious advantages for the integrated 1oo2D-architecture:

- System size and costs: The integration of all digital components of 1oo2D-architecture on a single ASIC reduces the total count of the required resources. Thereby the system size is clearly reduced and also the system costs which are required for the system implementation.
- Power consumption and system performance: By integration the count of individual components and size of the off-chip communication are reduced. This leads to a clear reduction of the power consumption of the complete system. Otherwise, the system performance of the system can be increased by using modern semiconductor process technologies which allow higher system clock rates.
- Reliability and safety: By the integration of 1oo2D-architecture on ASIC the reliability and safety of the complete system increase. This is due to the fact that the count of component and all contiguous factors (such packages, routing lines, solder joints, etc.) decrease. The latter results to a lower failure rate of failure (λ) for the complete system, and thus to better values of the reliability and safety parameters MTTF and PFD.

The integration of diagnostic units on hardware and software level is a further important feature of the 1oo2D-architecture. All important components of the system are monitored by diagnostic units and forwarded to the watchdog, which is responsible for transferring the system to the safe state. On one hand diagnosis units are implemented on hardware level such as system comparator for monitoring the several states of the redundant channels, as well as diagnosis and test blocks for dedicated safety functions such as voltage, temperature and clock monitoring units. On the other hand, the implementation of diagnostic units on software level is performed. In this

 ISBN: 978-1-61208-268-4

context, an essential component of this diagnosis is the implementation of safe operating system. The latter includes in addition to his usual tasks as operating system, a cycle-based monitoring of the system on software level. This includes the integration of different test routines that are responsible for all safety functions. Software test routines like CPU-tests, memory-tests, synchronization-tests are performed at this level.

Summarized, the integrated safety 1oo2D-architecture offers an increased degree of system safety and system reliability on smallest area and allows the realization of SIL3 systems in different areas of embedded systems.

IV. SOFTWARE METHODOLOGIES

In regard to the requirements of hardware and software, the ASIC development cycle is double edged: On the one hand, ASICs are hardware devices. On the other hand the development of ASICs is mainly done by software coding. Concretely, the ASIC description is usually written in so-called hardware description languages (HDL). Such a description is similar in many respects to classic programming languages. Usual representatives are currently VHDL and Verilog. System-C is a target language to generate code for both hardware and software systems. In this work System-C is not considered, but pure hardware description languages. In this section, the methodologies according to the standard IEC 61508 Ed. 2 are presented, which arise on coding and verification level for design safe ASICs.

A. *Safety-related Design Cycle*

To develop safety-related systems on ASIC level, the standard IEC 61508 Ed. 2 recommends an approach based on the V-model shown in Fig. 3. This is due to the fact that ASIC system development is not only a hardware development, but also a software development. In this context, requirements of both Part 2 and Part 3 of the IEC 61508 are considered for the used HDL code. This is especially in view of avoiding systematic faults important and useful. For this, appendix C in Part 3 offers guidance for quantifying the systematic safety integrity. More general requirements for safety-related ASIC-design include:

- Clear, unambiguous, testable requirements;
- Traceable safety requirements specifications;
- Detailed specific hardware and software specifications, among others, Interfaces, Performance and response times;
- Requirements on systematic safety integrity:
  - Avoiding systematic faults according to IEC 61508 Part 2 and Part 3 (Route 1_S),
  - Using of proven-in-use elements (Route 2_S),
  - Only software: requirements of IEC 61508 Part 3 (Route 3_S);
- Requirements on hardware safety integrity: to determine Route 1_H or Route 2_H;
- Systematic safety integrity: systematic ability of the elements of the safety functions, architecture-related restriction of max. SIL.

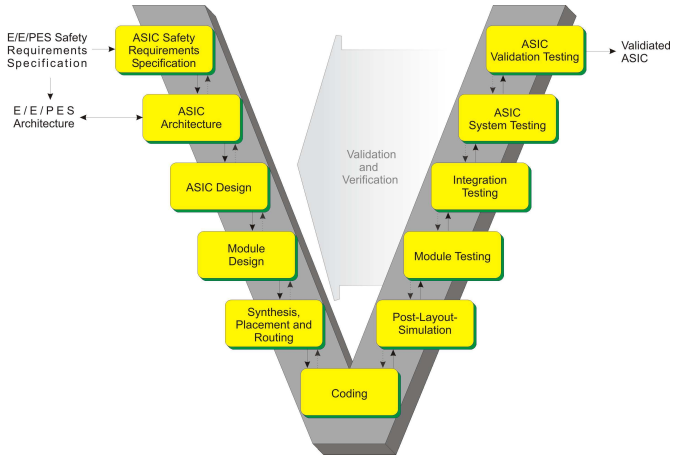

Figure 3. V Model for ASIC-development conforming to the safety standard IEC 61508

The three routes to avoid systematic errors mentioned above can be interpreted as described below.

For proven-in-use elements the residual number of systematic error is assumed to be small enough. Proven-in-use elements are defined as those elements which were used long enough used in similar projects. This primarily means that the field experience with the used elements should be conforming to the targeted SIL. Elements which are even already certified for the intended SIL can surely meet the requirements for systematic safety integrity following the route 2S. Route 1S refers to hardware; in our case to the hardware chip itself and its substrate, layout and manufacturing process, but also the HDL code. Although the latter also has software characteristics, it also applies for HDL code the requirements of Part 2 of the standard IEC 61508. Route 3S is reserved for the software running on the developed ASIC, e. g. safe operating system, driver software and application.

In any case, measures and methodologies for avoiding systematic faults, and thus for increasing systematic safety integrity are treated in Appendix F of Part 2 of the standard. The main part is represented in tabular form, wherein some measures and methodologies for respective SIL are recommended or not recommended. Considering an ASIC design as software, Part 3 of the standard introduces different requirements for software.

B. *Coding Methodologies*

Considering an ASIC design as software, Part 3 of the standard introduces different requirements for software that are applied in HDL code. The most important in our view are listed below:

- Modularity
- Other methods to reduce code complexity;
- Programming conforming to following aspects:
  - functionality,
  - exchange of information between elements,
  - timing behavior,
  - timing constraints,

 ISBN: 978-1-61208-268-4

  - concurrency,
  - data structures,
  - design-related assumptions and dependencies,
  - exception Handling (on HDL-level: Wiring of interrupt control lines),
  - pre-conditions, invariants, results / post-conditions,
  - comments;
- Ability to represent the design at multiple levels (structurally, functionally) - this is generally satisfied with HDL,
- Intelligibility.
- Testability (on verification and validation level).

In this context, the standard also requires the determination of suitable coding rules and naming conventions. But these are not specified, it is left to developers to define in advance useful guidelines.

Finally, for verification issues HDL tests are especially targeted. To illuminate this topic is beyond the scope of this document. For more information this and about requirements on HDL code in general, see our related work in [7].

V. HARDWARE METHODOLOGIES

After the software methodologies have been introduced in the last section, this section deals with the technical implementation of the safety-related systems on ASIC level. In this context, the term “on-chip Redundancy” is introduced in detail. Furthermore, the requirements and implementation methodologies of OCR on ASIC are presented. Furthermore, the handling with common cause failures is argued briefly. Finally an example for an ASIC-based 1oo2D-architecture is represented in the last section.

A. On-chip Redundancy

On-chip redundancy (OCR) is defined as a multiple (redundant) component implementation on a single chip. Hereby is generally not specified whether these components are active or passive redundant components. In Fig. 4 an example of a double on-chip redundancy is shown.

For the purposes of functional safety one usually considers channels; over the entire loop from sensors to control logic and actuators. In this regard, OCR could be used in order to implement redundant control logic or even the whole loop without using multiple chips. In case auf ASICs, the simplest example is a 1oo2-architecture described above; therefore a single ASIC could be used to implement two processors channels and its needed diagnosis components. Architecture-related requirements for ASICs in general with OCR are described in Appendix E in Part 2 of the standard IEC 61508. At a glance, it is noted that the requirements apply to purely digital ASICs with common substrate. Furthermore, there are currently no requirements for ASICs with a mixed design of digital and analog parts, so-called mixed-mode ASICs, or even purely analog ASICs. It is also noted that the standard in terms of OCR and in favor of safety is driving a more conservative

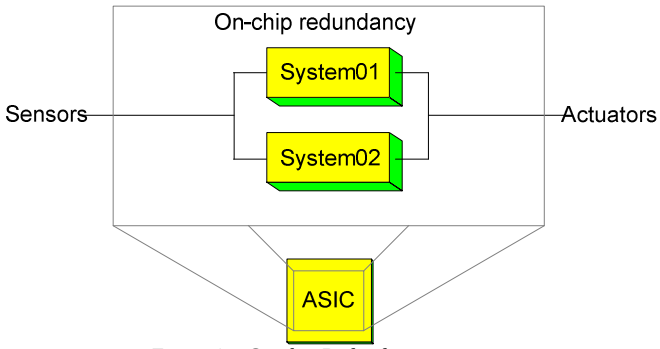

Figure 4. On-chip Redundancy

course. For this reason, the maximum SIL is limited to SIL 3. Nevertheless, the requirements on OCR are the following:

- Restriction to SIL 3,
- No systematic skills upgrading by combination,
- Consideration of random errors by temperature increasing,
- Physical channel separation by formation of blocks with "sufficient" distance to avoid short-circuits, for instance by electron migration and crosstalk
- Short circuits and crosstalk between adjacent lines of different blocks must not lead to failure of a safety function,
- Measures to avoid errors caused by faulty power supply, e.g. noise, crosstalk, high currents caused by short circuits, ...
- Connecting the substrate to ground, independent of the design process, for example n-well or p-well CMOS,

From practical view some of the requirements can be covered by concrete simulation runs for the target process technology design, such as temperature propagation at maximum clock frequency. Other requirements, such as avoiding cross talk, can be covered by applying concrete formal assessments for the routing. For other requirements, such as noise and the migration of soft errors, sufficient probability models or statistical experience results can be applied. For the minimum distance required between physical blocks experience values depending on the targeted process technology can be took into account. In any case, all these measures and methodologies have to be evaluated and fixed in agreements with the suitable certification authority, .e.g. TUV.

Finally, it is important to mention that fulfilling concrete measures to cover the above requirements depends heavily form the target application and the target process technology. In a failure mode and effects analysis (FMEA) and arrangement with the certification authority these aspects should be sufficiently considered.

 ISBN: 978-1-61208-268-4

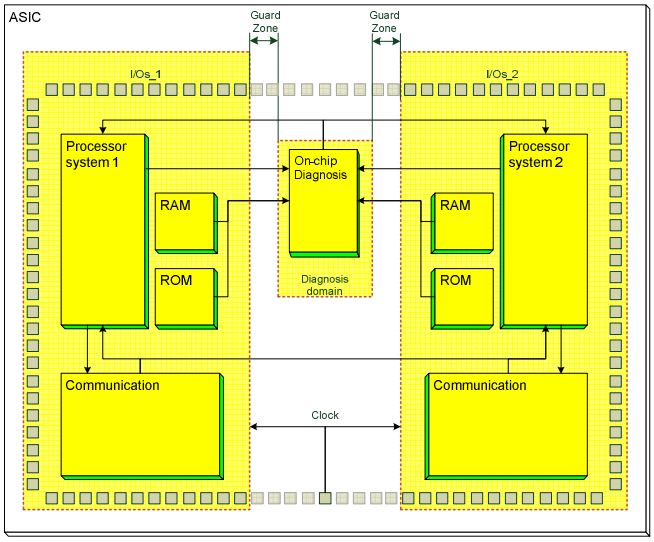

Figure 5. Safety-related ASIC Implementation

B. Commom-Cause Faults

In addition to the previously considered single-point failures, it is important to consider faults which have common cause, so-called common-cause faults. This is described in detail in the standard and be touched upon only briefly here. For ASICs with OCR a base-beta factor β_{IC} of 33% is assumed. By applying additional measures according to the tables given in the standard IEC 61508 this factor may increase or decrease. Thus the resulting beta coefficient is: $\beta_{ASIC} = \beta_{IC} + \Sigma$ modification. This shall not be higher than 25%. More information on this can be found in the standard. In this context, the following aspects are to be considered:

- Recognizing an uncontrollable faults - by diagnostic units, online tests, proof tests - needs to reach or holding the safe condition,
- For each channel and each singular executed monitoring component a diagnostic coverage (DC) of at least 60% should be achieved,
- Only diversely implemented (also differently designed) channels may monitor each other and thus improve as a watchdog the SFF and DC
- Homogeneous channels may only act as watchdogs for other channels if high SFF and DC has been already sufficiently reached,
- Tests regarding electromagnetic compatibility (EMC) with additional safety margin should neither impair the IC functionality neither destroy it
- Unsymmetrical wiring should be avoided as much as possible.

C. ASIC Implementation

This section describes the implementation of the measures presented in the previous sections illustrated by a case example. In the context of a recent research work, the implementation of a redundant 1oo2-architecture with on-chip diagnostic has been presented for FPGA implementation [6]. In Fig. 5 the block diagram of this architecture is shown. In this diagram the implemented measures according to section V A and B are mentioned. The physical separation and the establishment of guard zone are realized by using ASIC design tools. The width of the guard zone is weighted conforming to the guidelines of the standard IEC 61508 and depend on the used semiconductor process technology. Each channel is placed in a separated power domain and has its own power supply pins. The routing between the channels is effected by the use of special pre-routing blocks affected by the used design tools.

VI. CONCLUSION

Safety-related ASIC-design conforming to the safety standard IEC 61508 was introduced in this paper. First the target 1oo2D-architecture was presented. Afterwards, the advantages of the integration of this architecture on ASIC platform were motivated. Furthermore, software and hardware based methodologies for safety-related ASIC-design were presented. The key methodologies are on-chip redundancy and safety-related ASIC implementation. Particular attention was paid to the separation channel by power domains and guard zone. Finally, the determination of the beta factor for on-chip redundant design channels was briefly introduced. Techniques and measures to improve it were also presented and discussed in terms of the standard. In a future work the physical ASIC-implementation will be published.

REFERENCES

[1] A. Hayek and J. Boercsoek, “SRAM-Based FPGA Design Techniques for Safety Related Systems Conforming IEC 61508: a Survey and Analysis,” Second International Conference on Advances in Computional Tools for Engineering Applications (ACTEA), IEEE Conference Publications, Zouk Mosbeh, Lebanon, 2012, pp. 319–324

[2] International Electrotechnical Commission, IEC/EN 61508: International Standard 61508 Functional Safety: Safety-related Systems, Geneva; 2005

[3] International Electrotechnical Commission, IEC/EN 61508: International Standard 61508 Functional Safety: Safety related Systems: Second Edition, Geneva; 2010

[4] RTCA Inc., “DO-254: Design Assurance Guidance for Airborne Electronic Hardware,” Washington D.C., USA , 2000

[5] RTCA Inc., “DO-178: Software Considerations in Airborne Systems and Equipment Certification,” Washington D.C., USA , 1992

[6] J. Boercsoek, A. Hayek, B. Machmur and M. Umar, “Design and Implementation of an IP-based Safety-related Architecture on FPGA”, XXII International Symposium on Information, Communication and Automation Technologies (ICAT), Sarajevo, Bosnia and Herzegovina, IEEE Conference Publications, 2009, pp. 1–6

[7] A. Hayek, M. Schreiber and J. Boercsoek, “Basic VHDL tests conforming to IEC 61508”, Seventh International Conference on Networked Sensing Systems (INSS), Kassel, Germany, 2010, IEEE Conference Publications, 2010, pp. 41–44

 ISBN: 978-1-61208-268-4

Power Grid Safety Assessment Based on Linguistic Casual Network Under Uncertanties

Eugene V. Brezhnev[1], Vyacheslav S. Kharchenko[2], Artem V. Boyarchuk[3]
Centre for Safety Infrastructure Oriented Research and Analysis
National Aerospace University " KhAI"
Kharkiv, Ukraine
E-mail: [1] milesone@list.ru, [2] v.kharchenko@khai.edu, [3] a.boyarchuk@khai.edu

***Abstract* – This paper is devoted to the solution of problems connected with power grid safety. States of power grid are specific conditions of power grid determined by voltage, current and frequency fluctuations. Criticality of grid is a marginal state of grid with nearly to loss of stability. The state of power grid is determined by states of its systems, conditioned by their mutual influence of different nature. These influences cause the change of state of each system during grid life cycle. Problem is closely related to risk to NPP safety due to state of power grid. When grid is close to loss of its stability (call it as grid instabilities) NPP risk increase. It means the power grid conditions might affect NPP safety. Problem is to evaluate risk of NPP-PG interconnections. The proposed approach is based on application of Bayesian Belief Network, where nodes represent different grid systems, and links are stipulated by different types of influences (physical, informational, geographic, etc.). The grid safety is evaluated by its criticality. The criticality and influence are treated as the linguistic values. It is suggested to evaluate criticality of system, considering the change of criticalities of all connected systems. Conditional probabilities are also represented by linguistic values. To demonstrate the approach to the grid safety analysis, Russian Sayano–Shushenskaya hydro power station accident is reviewed. BBN is suggested as a tool for NPP PG risk assessment.**

Keywords-power grid; computing with words; safety; influence; Bayesian Belief Network

I. INTRODUCTION

A. Motivation

The power grid (PG) is an interconnected network composed of power-generation stations, high-voltage transmission lines, lower voltage distribution systems, and other support components. PG is a highly controlled, dynamic and distributed network combined of different systems. This complexity of engineered systems is a consequence of several factors: the sheer size and interconnectivity of the PG, the safety requirements, the need to balance electricity supply and consumption – throughout the grid at all times, and the nature of electricity that it is generated as it is used. This means the PG requires continual surveillance and adjustment to ensure supply always matches demand.

Disturbances in PG operation can originate from natural disasters, failures, human factors, terrorism, and so on. Outages and faults will cause serious problems and failures in the interconnected power systems, propagating into critical infrastructures such as Nuclear Industries, Telecommunication systems, transportation systems, etc.

Therefore, it is of high priority to consider PG safety, mutual influence of its systems and forecast possible accidents and failures, considering their severity and high costs of recovery.

B. Work Related Analysis

There are a lot of approaches and techniques of PG safety assessment. An approach to PG safety analysis, taking into consideration technical, organizational, and individual aspects, is proposed by Linstone [1]. The PG safety analysis is supplemented by a set of geographic and economic aspects developed by Kaiser [2]. An approach for PG safety assessment, based on processing statistical data related to PG operation, is proposed by Holmgren and Molin [3]. The main task of the safety statistical analysis is to determine the failure probability distribution function and to assess power grid risk. Lack of statistics prevents the use of traditional statistical methods for PG safety assessment.

Beside well known techniques of probabilistic and deterministic PG safety analysis, there are a lot of different approaches used for PG safety assessment. Logic methods (Fault Tree Analysis and Event Tree Analysis), used for safety analysis, are applied in research done by Bedford and Cooke [4] and Hoyland [5]. Typical PG safety analysis techniques are connected with equipment failure analysis, environment and human factor. Nowadays, a new type of grid hazards – intentional attacks occur. This type of hazards is analyzed by the use of probabilistic approach together with conditional probabilities calculation. However, mutual influence of systems, taking into account dynamical aspects of functioning and variation of risks caused by their failures, is not considered. Recently, network modeling has been revived due to computer technology progress and increase of interest in complex systems analysis. Achievements in a graph theory for complex systems analysis are reviewed by Albert and Barabasi [6]. A topology of North American Power System is analyzed. Graph is used as a model by Albert [7]. Evaluations, specifying Power System topology, lack of connectivity, while demounting vertexes that connect transmitting substations, are calculated. Two types of power grid safety hazards are analyzed: random failures and antagonistic (intentional) attacks.

Some methods used for PG safety analysis are qualitative and based on expert evaluations. Analysis results are represented in the form of risk matrix, containing failure effect frequency and severity. Qualitative techniques of safety analysis do not operate

 ISBN: 978-1-61208-268-4

numeric data providing results as descriptions, recommendations. The safety assessment is related to qualitative description of frequency of undesired events, damage and threat scenario. In [8] Glass specified that safety of a PG can be improved by implementing of process automation in disturbance situations.

The PG safety is affected by many factors regarding its design, manufacturing, installation, commissioning, operation and maintenance. Consequently, it may be extremely difficult to construct a complete mathematical model for the system in order to assess the safety because of inadequate knowledge about the basic failure events. This leads inevitably to problems of uncertainty in PG safety assessment.

The power grid is a very complex system. It is characterized by huge number of nodes and links between nodes with increasing structural complexity; links between nodes could change over time, have different weights, directions, etc. There are some PG attributes such as self-adaptation, PG self-healing, etc. which keep us aside from adequate understanding of PG nature, behavior types, accident mechanisms, etc.

There are a lot of risks as the inherited essences of PG life cycle. Due to high PG complexity, its dynamic nature these risks are not static. More over PG life cycle is characterized by complicated risk flow when safety and reliability issues might endanger the cyber security and vise verse. The risk associated with PG weakest link could compromise the safety and reliability of PG as a whole.

As an application of probability theory, Bayesian belief network (BBN) proposed by Heping [9] is a powerful tool both for graphically representing the relationships among a set of variables and for dealing with uncertainties in such variables. Many applications have proven that BBN is a powerful technique for reasoning relationships among a number of variables under uncertainty. BBN was successfully applied to ecological risk assessment and fault diagnosis in complex nuclear power systems.

But, traditional BBN requires too much precise information in the form of prior and conditional probability tables, and such information is often difficult or impossible to obtain. In particular, in dealing with indirect relationships, even domain experts may find that it is usually difficult to make precise judgments with crisp numbers, that is, to assign an exact number to the probability that consequences happen given the occurrence of an event. In certain circumstances, a verbal expression, e.g., “very unlikely” or interval value, e.g., 0.15, 0.20 of probabilistic uncertainty may be more appropriate than numerical values.

Common disadvantages of mentioned approaches are as follows: PG safety is considered a static attribute; no consideration for risk flow inside of PG; no consideration provided for mutual influences between power grid systems; there is a lack of publications for PG safety assessment with BBN using linguistic experts’ judgments.

C. Goal of the Paper

To assure the PG safety, it is necessary to consider and thoroughly analyze the nature of interaction among PG systems and evaluate the risk flow. The goal of the paper is to introduce an approach to power grid safety assessment, considering the different type of influence among its systems and evaluate safety using linguistic BBN. This technique can be useful to evaluate PG safety, taking into consideration mutual influences of its systems when all data available are represented by expert’s knowledge.

II. Principles of Power Grid Safety Analysis With Linguistic Casual Network

A. General Principles of Analysis

The PG safety analysis is carried out taking into consideration principles of dynamism, hierarchy, uncertainty, and influence (interaction) of subsystems.

Principle of dynamical analysis assumes to record changes of system criticality during the operation as a result of changes of its states (transition to state of non-operability). At each stage of life cycle, the criticality assessment specification and adjustment of criticality matrices [10], taking into consideration probable changes, are carried out.

The principle of hierarchy assumes representation of grid structure as a hierarchy.

The principle of influence of subsystem failures of i-level (on subsystem failure criticality of the same level) and influence on subsystems of (i-1)-level (higher) is important.

The safety of all influenced subsystems must be reconsidered.

The principle of uncertainty takes into consideration information incompleteness and uncertainty related to the conditions that cause PG accidents.

The principle of the weakest link risk flow is based on assumption that PG safety might be evaluated on risks associated with the weakest link of the grid.

The PG safety is an integral value composed of grid systems safety values. The grid safety is determined by uncontrolled mutual influence among grid systems. It is worth to note that influence exists on all grid levels and have to be taken into consideration when providing grid systems safety.

B. Types of Influences Between Power Grid Systems

According to the *principle of influence*, all influences (or relationships), existing in PG, can be divided into several hierarchy levels. The influence is an ability of one PG system to determine the state, characteristics or processes in other systems. Any type of influence is a time dependent value. The changes in NPP state and characteristics stipulate the changes in the influence value.

Generally, influences could be classified into different types [11]:

– Physical $I_{phys}(S_1 \to S_2)$ – a physical reliance on electricity flow between PG systems S_1 and S_2;

– Informational $I_{inf\,orm}(S_1 \to S_2)$ – a reliance on information transfer between PG systems S1 and S2 (via through I&C systems);

– Geographic $I_{inf\,orm}(S_1 \to S_2)$ – a local event occurred in PG system S1 affects power grid’s system S2 due to physical proximity;

– Logical $I_{log\,ical}(S_1 \to S_2)$ – an influence that exists between power grid systems S1 and S2 that does not fall into one of the about categories;

 ISBN: 978-1-61208-268-4

– Organizational $I_{organiz}(S_1 \to S_2)$ (influences through policy, regulation, markets). An influence that exists due to policy or procedure that relates a state change in one elements of PG to subsequent effect on other systems;

– The societal influence $I_{organiz}(S_1 \to S_2)$ that one PG system may have one of societal factors as public opinion, fear and confidence, for example, staff of other PG system.

The modern PG has become the informational infrastructure. As a result a new type of influences might be introduced in the scope of analysis. For example, physical state of one system might influence the informational state of other system, etc.

The formalization of influences between PG systems is very helpful for its safety assessment based on criticality matrices. Generally, criticality matrix is represented as FMECA table. The traditional FMECA [12] is the most widely used reliability analysis technique on the initial stages of system development.

For example, if PG system S_1 consists of three subsystems S_{11}, S_{12}, S_{13} then criticality matrix which represents the system S_1 might be presented as shown in the Table 1.

TABLE I. CRITICALITY MATRIX FOR SYSTEM S_1

System S_1		**Severity of Failure Mode**		
Failure rate		H	M	L
	H		S_{12}	
	M			S_{13}
	L	S_{11}		

Traditionally, the criticality assessment is performed by calculating the criticality accident (failure) as a product of its severity and probability:

$$Crt(S_i) = P(S_i) \times Sev(S_i), \quad (1)$$

where S_i is PG system; $P(S_i)$ is probability of S_i accident; $Sev(S_i)$ – severity of accident consequences.

According to the principle of hierarchy, the grid structure might be represented as a hierarchy. In this case, the safety of PG systems of higher level hierarchy might be evaluated as a sum of criticalities of power grid systems of lower level hierarchy. For example, considering the criticalities of S_{11}, S_{12}, S_{13} as subsystems of S_1, its total criticality could be calculated as:

$$Crt(S_i) = P(S_1) \times Sev(S_1) + P(S_2) \times Sev(S_2) + \\ + P(S_3) \times Sev(S_3) = \sum_{i}^{I} P(S_i) \times Sev(S_i). \quad (2)$$

Another approach might introduced considering the weakest link of PG. In this case the system total criticality might be equaled its weakest link criticality.

It is suggested to treat criticality as PG system's safety inverse value. The more system criticality the less its safety and vice versa.

It should be noted that criticality matrix might be used to represent different states of environment and its influence on PG systems. We suggest to use the environmental FMECA where different natural hazards (earthquake, flooding, etc.) are considered as different failures modes characterized by its probability and severity for the nearest PG systems. This probability of system accident (natural disaster) and its severity could be handled as linguistic or numerical variable. Hence, criticality is also treated correspondently either linguistic or numerical variable.

A linguistic variable is characterized by a quintuple (x, T(x), U, G, M) in which x is the name of variable; T(x) is the term set of x, that is, the set of names of linguistic values of x with each value being a fuzzy number defined on U; G is a syntactic rule for generating the names of values of x; and M is a semantic rule for associating with each value its meaning.

The set of state Ω_{Si} of any PG system S_i is determined as:

$$\Omega_{Si} = \{Crt(S_i)=High, Crt(S_i)=Medium, Crt(S_i)=Low\}. \quad (3)$$

Any accident or failure of power grid system leads to the change of criticality of all connected systems due to principle of risk flow. When a failure of one system occurs, the criticalities of all dependent systems are recalculated.

The prognosis and assessment of PG system service life, based on real time measurements, will help to identify grid systems most likely to fail. The potential estimation methods and equipment service life prediction for complicated systems consist of deterministic, statistical, physical-statistical and methods based on expert knowledge. These methods are used to predict the probability of accident of any system S_{ij} of S_i.

This criticality assessment is used to support the subjective expert judgment expressed by linguistic variable on the initial power grid system state. The more system criticality calculated on (2) the more confident expert's opinion on the criticality of each node of PG.

C. Bayesian Belief Network as a Model for Power Grid's safety assessment

BBN is a classical causal network represented as a pair N= {(V, E), P} where V and E are the nodes and the edges of a Directed Acyclic Graph (DAG), respectively, and P is a probability distribution over V. Discrete random variables $V = \{X_1, X_2, ..., X_n\}$ are assigned to the nodes while the edges E represent the causal probabilistic relationship among the nodes. Each node in the network is annotated with a Conditional Probability Table (CPT) that represents the conditional probability of the variable given the values of its parents in the graph. The CPT contains, for each possible value of the variable associated to a node, all the conditional probabilities with respect to all the combinations of values of the variables associated with the parent nodes. For nodes that have no parents, the corresponding table will simply contain the prior probabilities for that variable.

The principles behind BBN are Bayesian statistics and concentrate on how probabilities are affected by both prior and posterior knowledge. In order to extend the classic BBN into fuzzy BBN which is capable of dealing with linguistic variables, fuzzy numbers and their operations must be used.

The state of each PG system is determined by types of influence mentioned above. The Figure 1 represents a

 ISBN: 978-1-61208-268-4

fragment of network, which characterizes the PG, where S_1, S_2 are the parent nodes and system S_3 is a child node.

Generally, the several BBNs might be required to represent only one PG. These networks have the same nodes as PG systems, but different types of influence, which stipulate the different causal links (physical, geographical, organizational, logical, informational and societal) between nodes. The different types of influence are characterized by its own weight. The more weight of the given type of influence (according to the expert judgments) the more PG sensitive to this type of influence.

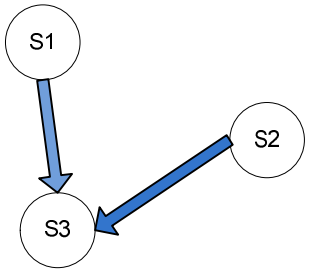

Figure 1. A fragment of network, which characterizes the PG.

Apparently, the physical influence is more important, when PG safety is considered. But all types of influences should be considered to provide more accurate PG safety evaluation. For each type of influence might be introduced its own type of PG system particular criticality. It means that PG could be more vulnerable to the change of one type of influence and, at the same time, be insensitive to other type influence change.

Considering the types of influence mentioned it is assumed that the total PG system criticality is a function of power grid system's particular criticalities, stipulated by the particular types of influence, i.e.,

$$Crt(S_i) = f(Crt^{org}(S_i), Crt^{fhys}(S_i), Crt^{geo}(S_i), Crt^{\log}(S_i), Crt^{soc}(S_i), Crt^{\inf}(S_i)) \quad (4)$$

where $Crt(S_i)$ - the total power grid system criticality; $Crt^{org}(S_i)$ – particular criticality of power grid system, conditioned by organizational influence in PG; $Crt^{fhys}(S_i)$ – particular criticality of power grid system, conditioned by physical influence in PG; $Crt^{\log}(S_i)$ – particular criticality of power grid system, conditioned by logical influence in PG; $Crt^{\inf}(S_i)$ – particular criticality of power grid system, conditioned by informational influence in PG; $Crt^{soc}(S_i)$ – particular criticality of power grid system, conditioned by societal influence in PG.

Depending on the scale used to evaluate criticality, each PG system could be characterized by the tuple of its criticalities values, considering the types of influence, which determine these criticalities.

Example of power grid system criticality tuple is shown in the Table 2.

TABLE II. EXAMPLE OF POWER GRID SYSTEM CRITICALITY TUPLE

	Type of influence					
	Physical	***Informational***	***Geographic***	***Logical***	***Organizational***	***Societal***
PG Si	***Criticalities caused by the given type of influence***					
PG S_1	H	H	M	L	L	L
PG S2	H	M	M	L	L	H
…..						
PG Sn	L	H	M	M	M	L

This tuple might be interpreted as combination of risks for particular PG systems due to different type of influences caused by other systems inside of PG.

The following task is to calculate the particular criticality, stipulated by the given type of influence. We suggest using BBN to evaluate the criticalities of the PG systems.

According to approach, it is suggested to construct BBN for each type of influence. Each node of BBN is represented by criticality matrix. Nodes are connected by links, which represent the different types of influence.

Hence, BBNs, which describe the PG system safety, consist of set of nodes. For each node the set of state is introduced. As mentioned, the state of node is characterized by a value of its criticality, calculated according to (2).

Every node also has a conditional probability table (CPT), associated with it. Conditional probabilities represent likelihoods based on prior information or past experience. A conditional probability is stated mathematically as, i.e., the probabilities of power grid system (child node) being at state characterized by expressions "Criticality is High (Medium, Low)", considering all possible combinations of other PG systems (parents' nodes) criticalities (High, Medium, Low).

As mentioned these conditional probabilities might be represented by linguistic values (for example High, Medium, Low).

Fragment of linguistic CPT is shown in the Table 3.

TABLE III. FRAGMENT OF CPT

S_1			**S_2**			**S_3**	
Criticality			***Criticality***			***Criticality***	
H	***M***	***L***	***H***	***M***	***L***	***H***	**…**
+			+			P(Crt(S_3)=H/Crt(S_1)=H, Crt(S_2)=H)=*High*	…
+				+		P(Crt(S_3)=H/Crt(S_1)=H, Crt(S_2)=M)=*Low*	…
						…….	..
	+		+			P(Crt(S_3)=H/Crt(S_1)=M, Crt(S_2)=H)=*Low*	…

Let us consider the fragment of BBN of S_1, S_2, S_3 represented on Figure 1., where criticality of S_3 (child node) is conditioned by criticalities both of S_2, S_3 (parents' nodes).

According to [13], probability of S_3, being at one of the established state Ω_{S3} depending on the states of parents nodes, could be determined as:

$$P(S_3^{(k)}) = \sum_i \sum_j P(S_3^{(k)} / S_1^{(i)}, S_2^{(j)}) * P(S_1^{(i)}) * P(S_2^{(j)}), \quad (5)$$

where $P(S_3^{(k)})$ – a probability for S_3 being at *k*-th state; $P(S_3^{(k)} / S_1^{(i)}, S_2^{(j)})$ – a conditional probability for PG system S_3 to be at *k*-th state, provided system S_1 being at *i* th state

and system S_2 being at j – th state; $P(S_1^{(i)})$ – probability for S_1 being at i-th state determined by expert, taking into account value (2); $P(S_2^{(j)})$ – probability for S_2 being at j-th state determined by expert, taking into account value (2).

Whereas linguistic BBN is used for PG safety analysis all probabilities in formula (5) are represented as linguistic variables.

The probability for system S_1 of being at the state described by expression Criticality - High" is calculated as

$$
\begin{aligned}
&P(Crt(S_3)=High))=\\
&P(Crt(S_3)=H/Crt(S_1)=H,Crt(S_2)=H)*P(Crt(S_1)=H)*P(Crt(S_2)=H)+\\
&+P(Crt(S_3)=H/Crt(S_1)=H,Crt(S_2)=M)*P(Crt(S_1)=H)*P(Crt(S_2)=M)+\\
&+P(Crt(S_3)=H/Crt(S_1)=H,Crt(S_2)=L)*P(Crt(S_1)=H)*P(Crt(S_2)=L)+\\
&+P(Crt(S_3)=H/Crt(S_1)=M,Crt(S_2)=H)*P(Crt(S_1)=M)*P(Crt(S_2)=H)+\\
&+P(Crt(S_3)=H/Crt(S_1)=M,Crt(S_2)=M)*P(Crt(S_1)=M)*P(Crt(S_2)=M)+\\
&+P(Crt(S_3)=H/Crt(S_1)=M,Crt(S_2)=L)*P(Crt(S_1)=M)*P(Crt(S_2)=L)+\\
&+P(Crt(S_3)=H/Crt(S_1)=L,Crt(S_2)=H)*P(Crt(S_1)=L)*P(Crt(S_2)=H)+\\
&+P(Crt(S_3)=H/Crt(S_1)=L,Crt(S_2)=M)*P(Crt(S_1)=L)*P(Crt(S_2)=M)+\\
&+P(Crt(S_3)=H/Crt(S_1)=L,Crt(S_2)=L)*P(Crt(S_1)=L)*P(Crt(S_2)=L).
\end{aligned}
$$

Similarly all system S_1 of being at the state described by expression Criticality – Meduim and Criticality-High.

Semantics of linguistic variables are supported by fuzzy sets. Fuzzy sets are obtained by the means of fuzzy arithmetic for triangular fuzzy numbers.

A triangular fuzzy number denoted by M = <m, α, β>, has the membership function:

$$
\mu_M(x)=\begin{cases}0, \ for\ x\le m-\alpha\\ 1-\dfrac{m-x}{\alpha}, \ for\ m-\alpha<x<m\\ 1-\dfrac{x-m}{\beta}, \ for\ m<x<m+\beta\\ 0, \ for\ x\ge m+\beta\end{cases}. \tag{6}
$$

The point m, with membership grade of 1, is called the mean value and α, β are the left hand and right hand spread of M respectively.

If M = <m, α, β> and N = <n, γ, δ> are two TFNs then their addition is expressed as:

$$M \oplus N = <m + n, \alpha + \gamma, \beta + \delta>. \tag{7}$$

Multiplication M ⊗ N of two TFNs is not necessarily a triangular.

A good approximation is as follows:

$$M \otimes N = <m,\alpha,\beta> \otimes <n,\gamma,\delta> \cong <mn,\ m\gamma + n\alpha,\ m\delta + n\beta>. \tag{8}$$

Division of two TFNs is

$$\frac{M}{N} \cong \left\langle \frac{m}{n}, \frac{m\delta + n\alpha}{n^2}, \frac{m\gamma + n\beta}{n^2} \right\rangle. \tag{9}$$

These fuzzy arithmetic operations are used to calculate new linguistic probabilities represented by TFNs. These fuzzy probabilities usually do not match any linguistic term in the initial term set (High, Medium, Low), so a computing with words (CWW) procedure is needed to express the result in the original expression domain.

The CWW is used to express the result in the original expression domain. CWW procedure uses the linguistic assessments and makes computations with them. Foundations and applications, providing the current status of theoretical and empirical developments in CWW, can be found in [14].

A linguistic aggregation operator based on the extension principle acts according to

$$S^n \xrightarrow{\tilde{F}} F(R) \xrightarrow{app_1(\cdot)} S, \tag{10}$$

where S^n symbolizes the n Cartesian product of S, $\tilde{F}$ is an aggregation operator based on extension principle, F(R) the set of fuzzy sets over the set of real number R, app_1: $F(R) \rightarrow S$ is a linguistic approximation function that returns a label from the linguistic term S, whose meaning is the closest to the obtained unlabeled fuzzy number, and S is the initial term set.

According to (5), the probabilities for system S_1, being at the state described by expression "Criticality - High", "Criticality – Medium" and "Criticality-High" might be calculated.

The power grid system S_i state, conditioned by the given type of influence, is determined on the criterion:

$$
\begin{aligned}
&Crt(S_i) = arg\ max(P(Crt(S_i) = High),\\
&P(Crt(S_i) = Medium), P(Crt(S_i) = Low),
\end{aligned} \tag{11}
$$

where $P(Crt(S_i) = High)$ – a probability of power grid system of being at the state described by linguistic value High; $P(Crt(S_i) = Medium$ – probability of power grid system of being at the state described by linguistic value Medium; $P(Crt(S_i) = Low$ – probability of power grid system of being at the state described by linguistic value *Low*.

III. HPP Accident Case Study Based on Linguistic Casual Network

To demonstrate the approach to the power grid safety analysis, using the linguistic BBN, Russian Sayano–Shushenskaya HPP failure (August, 2009) is reviewed [15]. This HPP is one of the largest (together with Bratskaya HPP) one, used for power control of the whole power system with installed capacity - 6,4 mm kW, annual output - 22,8 bln kW p.h. Ten hydraulic units, each of 640 kW, are installed in the plant.

The BBN is built for fragment of Siberian power systems. BBN's nodes are criticalities matrixes of Sayano–Shushenskaya HPP – S_1, Mayansk HPP – S_2, Bratskaya HPP– S_3, Thermal Power Plant (TPP) of Bratsk – S_4.

Only physical influence is considered to evaluate state of S_1 when conditional probabilities are expressed in linguistic values. Each node of Siberian power systems is completed by linguistic conditional probabilities table (see Table 4).

The increasing of load from Bratskaya HPP and Mayansk HPP increased the criticality of S_1 and, finally, led to destruction of HPU – 2 (S_{32}). Increasing of criticality of S_1 led to increasing criticality of S_4.

 ISBN: 978-1-61208-268-4

TABLE IV. FRAGMENT OF SIBERIAN POWER SYSTEMS CONDITIONAL PROBABILITIES TABLE (BEFORE ACCIDENT)

Mayansk HPP, S_1			**Bratskaya, HPP S_2**			**Sayano–Shushenskaya HPP, S_3**	
Criticality			***Criticality***			***Criticality***	
H	***M***	***L***	***H***	***M***	***L***	***H***	**…**
+			+			P(Crt(S_3)=H/Crt(S_1)=H, Crt(S_2)=H)=*High*	…
+				+		P(Crt(S_3)=H/Crt(S_1)=H, Crt(S_2)=M)=*High*	…
						…….	..
	+		+			P(Crt(S_3)=H/Crt(S_1)=M, Crt(S_2)=H)=*Low*	…

According to (5) the linguistic probabilities of S_3 being at the different states are calculated as:

$$P(Crt(S_3) = High)) = High;$$
$$P(Crt(S_3) = Medium)) = Low;$$
$$P(Crt(S_3) = Low)) = Low.$$

Considering (7), it is suggested that before Sayano–Shushenskaya HPP accident its criticality value might had been *High.*

IV. CONCLUSION AND FUTURE WORK

The proposed technique may be applied to PG safety value prediction, taking into account its systems influence. The technique is based on the use of dynamical criticality matrices hierarchy. The power grid's capacity used to predict the possible safety change could be improved by implementing of the decision making system.

The technique suggested in the paper is considered as a part of this system. The PG safety assessment is carried out taking into consideration principles of dynamism, hierarchy, uncertainty and mutual influence of systems. BBN is used to predict the particular criticality of PG system, conditioned by the given type of influence. CWW is suggested to determine the probabilities of PG states expressed by linguistic values.

The results of analysis may be used to determine effective safety management strategies.

Consideration of the difference types of influence allows improving the accuracy of PG safety value.

Next step of technique enhancement will be related to consideration of Ukrainian NPP safety analysis, taking into consideration the types of influences of power grid and development of decision making tool-based system.

Main math tool would be based on dynamic BBN which allow considering the changes of systems states and perform safety assessment in time-related manner.

REFERENCES

[1] H. Linstone, "Multiple Perspectives for Decision-Making: Bridging the Gap Between Analysis and Action", North-Holland, Amsterdam, 1984.

[2] A. Kaijser, The Swedish Infrastructure – Historical Development and Future Challenges, Carlsson, Stockholm, 1984.

[3] A. Holmgren and S. Molin, Using Disturbance Data to Assess Vulnerability of Electric Power Delivery Systems, Journal Infrastructure systems, American Society of Civil Engineers (ASCE), Volume 12, Issue 4, pp. 243-251, 2006.

[4] T. Bedford and R. M. Cooke, Probabilistic Risk Analysis: Foundation and Methods, Cambridge University Press, Cambridge, England, 2001.

[5] A. Hoyland and M. Rausald, System Reliability Theory: Models and Statistical Methods, Wiley, New York, USA, 1996.

[6] R. Albert and A.-L. Barabasi, Statistical Mechanics of Complex Networks, Review of Modern Physics, Volume 74, pp. 23-26, 2002.

[7] R. Albert, I. Albert, and G. L. Nakarado, Structural Vulnerability of North American Power Grid. Physical review E., 69, 025193 (R) (Rapid Communication), pp. 95-104, 2004.

[8] R. Glass, Simulation and Analysis of Cascading Failure in Critical Infrastructure, Proceeding of Working Together: R&D Partnerships in Homeland Security, Boston, April 2005, pp. 45-56, 2005.

[9] P. Heping and L. Lin, "Fuzzy Bayesian networks a general formalism for representation, inference and learning with hybrid Bayesian networks", IJPRAI, 14(7), 941–962, 2000.

[10] I. Elyasi Komari, V. Kharchenko, and E. Babeshko, Extended Dependability Analysis of Information and Control Systems by FME(C)A-Technique: Models, Procedures, Application, in Proceedings of the Conference on Dependability of Computer systems, Brunow, Poland, 30 June-02 July 2009, IEEE Computer Society, Los Alamitos, California, pp. 25-32, 2009.

[11] E. Brezhnev, Risk-analysis in critical information control system based on computing with words' model, in Proceeding of 7th International Workshop on Digital Technologies, Circuit Systems and Signal Processing, Zilina, Slovakia, 11-21 November, pp. 67-72, 2010.

[12] J. B. Bowles, "An assessment of PRN prioritization in a failure modes effects and criticality analysis", Journal of the IEST, Volume 47, N: 51-6. 2004.

[13] On-line Tutorial on Bayesian nets and probability:http://www.dcs.qmw.ac.uk/%7Enorman/BBNs/BBNs.htm [retrieved: February, 2013].

[14] L. Zadeh and J. Kacprzyk, Computing with words in Information / Intelligent Systems – Part : Foundation; Part 2, USA, 2001.

[15] "Causes of destruction of Sayano–Shushenskaya HPP hydraulic unit 2", http://zhurnal.lib.ru/b/boris_i_k/prichinyrazruschenijagidroagregata2sschges.shtml [retrieved: December, 2012].

 ISBN: 978-1-61208-268-4

Robust Monitoring of a Conditional Distribution in Economic Data Streams Using Statistical Depth Functions

Daniel Kosiorowski
Department of Statistics
Cracow University of Economics
Cracow, Poland
e-mail: daniel.kosiorowski@uek.krakow.pl

***Abstract*—Estimation of a conditional distribution is a building block of a variety of statistical procedures used in the modern economics. This estimation is especially difficult in case of an economic data stream, i.e., when data are generated by the multidimensional non-stationary process of unknown form which may contain outliers. In this paper we propose a novel approach for robust monitoring conditional and unconditional distributions in the data streams. Our proposals are based on the idea of adjusted Nadaraya-Watson estimator proposed in [6] and they appeal to the so called data depth concept. We show very promising statistical properties of our proposals in cases of selected linear and nonlinear data streams models.**

Keywords-data stream; robust procedure; depth function

I. INTRODUCTION

An economic data stream could be informally defined as *a random sequence of observations of an undetermined length* – see [21], [16]. We should notice that in case of stochastic process analysis, say $\{X_t\}$, we assume a fixed interval of time, say $[0,T]$. All our calculations concern this interval, so we infer on base of information consisted in this interval – see [4], [10], [15]. In case of the data stream analysis we do not fix any interval. Each consecutive while denotes a new stochastic process analysis. The terminology originates from the Informatics, where the data streams were considered at first. In the Economics, we use by default stochastic methodological framework appealing to a nonlinear time series theory and generally consider different research tasks than in the Informatics – see [3], [4], [16]. We can indicate several specific features of the economic data stream analysis: 1. Data are generated by a process exhibiting a nonlinear structure of dependence between the observations. 2. Data streams usually exhibit several regimes. 3. Data stream analysis is performed on base of a constantly updated sample – on base of a sliding window or windows (the windows may differ with respect to their length or probing frequency for purposes related to a different time scales). 4. The streams usually consist of a huge amount of multivariate observations containing outliers, which is not stored in computer memory. 5. A signal carried by the stream is observed at irregularly spaced time points and has to be processed on-line. By the *signal* we mean a relation between numerical characteristics of the stream rather than a result of removal a noise from the stream. Let $\mathbf{x}_1, \mathbf{x}_2, ...$ be an observed economic data stream.

A window $\mathbf{W}_{i,n}$ denotes the sequence of points ending at $\mathbf{x}_i$ of size n, i.e., $\mathbf{W}_{i,n} = (\mathbf{x}_{i-n+1}, ..., \mathbf{x}_i)$. Many approaches to data stream analysis is based on a monitoring various distance measures between distributions estimated from two or more windows – see [1], [12], [13]. In this paper, we study certain aspects of robust monitoring of a *one-dimensional economic data stream* using a moving window of a fixed length. We consider the following problems:

PROBLEM 1: We monitor a one-dimensional stream $X_1, X_2, ...$, and our aim is to detect changes in unconditional distribution of the X_i, on base of the moving window $\mathbf{W}_{i,n}$, $i = 1, 2, ...$, i.e., changes of $P(X_i \in A)$, $A \subset \mathbb{R}$, $i = 1, 2, ...$.

PROBLEM 2: We monitor a one-dimensional stream $X_1, X_2, ...$, and our aim is to detect changes in a conditional distribution of the X_{i+1}, conditioned on the observed window $\mathbf{W}_{i,n}$, $i = 1, 2, ...$, i.e., changes of $P(X_{i+1} \in A \mid \mathbf{W}_{i,n} = \mathbf{x})$, $A \subset \mathbb{R}$, $i = 1, 2, ...$.

In order to solve the above problems we focus our attention on the adjusted Nadaraya–Watson estimator of the conditional distribution proposed in [6]. The authors assumed that data are available in the form of strictly stationary stochastic process $\{(Y_i, \mathbf{X}_i)\}$, where Y_i is a scalar and $\mathbf{X}_i$ is a d-dimensional vector. They proposed two estimators for estimating the conditional distribution function $F(y \mid \mathbf{x}) \equiv P(Y_i \le y \mid \mathbf{X}_i = \mathbf{x})$, local logistic method and adjusted Nadaraya-Watson estimator, which have better statistical properties then known local and/or nonparametric approaches. Their proposals however are not robust. In the economic time series context, $\mathbf{X}_i$ typically denotes a vector of lagged values of a phenomenon Y_i, in which case $F(\cdot \mid \mathbf{x})$ is the *predictive distribution* of Y_i, given $\mathbf{X}_i = \mathbf{x}$ representing the past. Let $p_i = p_i(\mathbf{x})$, for $1 \le i \le n$, denote weights (functions of the data $\mathbf{x}_1, ..., \mathbf{x}_n$ as well as of $\mathbf{x}$) with the property that each $p_i \ge 0$, $\sum_{i=1}^{n} p_i = 1$ and

$$\sum_{i=1}^{n} p_i(\mathbf{x})(\mathbf{X}_i - \mathbf{x}) K_h(\mathbf{X}_i - \mathbf{x}) = 0 . \qquad (1)$$

In a spirit of ideas presented in [6] we can define following estimators of the unconditional and conditional densities

 ISBN: 978-1-61208-268-4

$$\tilde{f}(x)=\frac{1}{n}\sum_{i=1}^{n}p_i(x)K_h(x_i-x) \quad , \tag{2}$$

$$\tilde{g}(y\mid \mathbf{x})=\frac{h_1^{-1}\sum_{i=1}^{n}K_{h_1}^{1}(y_i-y)\,p_i(y,\mathbf{x})\mathbf{K}_h(\mathbf{x}_i-\mathbf{x})}{\sum_{i=1}^{n}p_i(\mathbf{x})\mathbf{K}_h(\mathbf{x}_i-\mathbf{x})} \quad , \tag{3}$$

where K is a kernel function (e.g., Gaussian), $K_h(\cdot)=h^{-1}K(\cdot/h)$, K_h^1 is one-dimensional Kernel, $\mathbf{K}_h$ denotes d-dimensional kernel, $d\geq 2$ (e.g., a product kernel or a kernel based on a norm of $\mathbf{x}$), h denotes a bandwidth.

It is well known that a crucial issue concerning the kernel density estimation involves an appropriate choice of the bandwidth h (i.e., providing a balance between unbiasedness, dispersion and computational complexity). In this paper, we "*robustify*" the above approach by the choice of weights $p_i(\mathbf{x})$ using adjusted sample depth function.

II. ROBUST DATA STREAM ANALYSIS

We understand the robustness of our proposals in a spirit of an approach presented in [5]. According to the authors a crucial property of an estimator is that it takes different values for different sample realizations. If a continuum of sample realizations is possible and the estimator is continuous in the sample, we expect a continuum of possible values for the estimator. We can look for the fraction of contamination for which this property is lost. In particular, we look for the fraction of outliers such that the estimator, or more specifically the measure of badness, can take only a finite number of different values despite a continuum of possible uncontaminated sample realizations. The statistical procedures, statistical decision rules, considered in this paper are functions of the estimators calculated on base of a moving window from the stream. In our proposals we use a very promising methodological approach of the multivariate analysis called *data depth concept* – see [14], [19], [23].

A *data depth* is a way to measure the "depth" or "outlyingness" of a given point with respect to a multivariate data cloud or its underlying distribution. A data *depth function* provides an order of the multivariate observations on base of their departure from the center. This ordering enables us for quantifying many complex multivariate features of the underlying distribution, including location, quantiles, scale, skewness and kurtosis. There are a variety of statistical depth functions known in the literature and implemented in the statistical software – see [2]. For the technical convenience purposes (vanishing value of the depth outside the convex hull of a sample) we further use so called *simplicial or Liu depth* – see [14], [19].

Let $\mathbf{X}^n=\{\mathbf{X}_1,...,\mathbf{X}_n\}$ be a random sample from the distribution $G(\cdot)$ in $\mathbb{R}^d$, $d\geq 1$. Let $I(\cdot)$ be the indicator function, that is, $I(A)=1$ if A occurs and $I(A)=0$ otherwise. Given the sample $\mathbf{X}^n$, the *sample simplicial depth* of $\mathbf{x}\in\mathbb{R}^d$ is defined as

$$D(\mathbf{x},\mathbf{X}^n)=\binom{n}{d+1}^{-1}\sum_{(*)}I(\mathbf{x}\in s[\mathbf{X}_{i_1},...,\mathbf{X}_{i_{d+1}}]), \tag{4}$$

where $(*)$ runs over all possible subsets of $\mathbf{X}^n$ of size $d+1$, $s[\mathbf{X}_{i_1},...,\mathbf{X}_{i_{d+1}}]$ is closed simplex with vertices $\mathbf{X}_{i_1},...,\mathbf{X}_{i_{d+1}}$.

When the distribution G is known, then the simplicial depth of $\mathbf{x}$ with respect to G is defined as $D(\mathbf{x},G)=P_G\{\mathbf{x}\in s[\mathbf{X}_1,...,\mathbf{X}_{d+1}]\}$, where $\mathbf{X}_1,...,\mathbf{X}_{d+1}$ are $d+1$ random observations from G. This depth is affine invariant and $D(\mathbf{x},\mathbf{X}^n)$ converges uniformly and strongly to $D(\mathbf{x},G)$. The affine invariance ensures that our proposed inference methods are coordinate–free, and the convergence of $D(\mathbf{x},\mathbf{X}^n)$ to $D(\mathbf{x},G)$ allows us to approximate $D(\mathbf{x},G)$ by $D(\mathbf{x},\mathbf{X}^n)$ when G is unknown. For our purposes it is useful to consider a rescaled version of the sample depth

$$\tilde{D}(\mathbf{x},\mathbf{X}^n)=D(\mathbf{x},\mathbf{X}^n)\Big/\sum_{i=1}^{n}D(\mathbf{x}_i,\mathbf{X}^n). \tag{5}$$

III. PROPOSALS

There is a long tradition in applying nonparametric methods in time series analysis involving nonparametric regression, runs tests for randomness, permutation tests or certain rank tests. However *nonparametric and robust analysis of a non-stationary time series* still seems to be a great challenge for the statistical and econometrical community – see [4], [9]. Our proposals concern decision making process basing on the stream and they aimed at detecting changes in certain very important for the decision makers properties of the stream – its unconditional and conditional distributions.

PROPOSAL 1: Let $W_{j,n}=\{x_{j-n},...,x_j\}$, denotes a window from the stream of length n in a time point $j=l,...$, and let g denotes a certain *fixed reference density*. In order to monitor the unconditional distribution of the stream, determined by density f, monitor *Hellinger* distance

$$d_j(\tilde{f}_j,g),\ j=l,..., \tag{6}$$

where

$$\tilde{f}_j(x)=\frac{1}{n}\sum_{i=1}^{n}K_h(x_{ij}-x)\tilde{D}(x,W_{j,n}), \tag{7}$$

is the adjusted kernel density estimate and K is a kernel function, $K_h(\cdot)=h^{-1}K(\cdot/h)$, $\tilde{D}(x,W_{j,n})$ denote the adjusted sample depth (5) of x, $x_{ij}\in W_{j,n}$, $i=1,...,n$, $j=l,...$.

PROPOSAL 2: Let $W_{j-N,n}=\{x_{j-N-n},...,x_{j-N}\}$,..., $W_{j-1,n}=\{x_{j-n-1},...,x_{j-1}\}$, $W_{j,n}=\{x_{j-n},...,x_j\}$ denote N windows from the stream each of length n, $j=l,...$,

$k, N \in \mathbb{N}$, $N \gg k$ and let g denotes a fixed reference density. Let $Y_j^N = \{x_{j-N},...,x_{j-1},x_j\} \equiv \{y_1^j,...,y_N^j\}$, $\mathbf{X}_j^N = \{(x_{j-k-N},...,x_{j-1-N}),...,(x_{j-k-1},...,x_{j-1})\} \equiv \{\mathbf{x}_1^j,...,\mathbf{x}_N^j\}$.

In order to monitor the conditional distribution of X_j determined by density f_j, given the small section of the past such as $(X_{j-1},...,X_{j-k})$, $k = 2,3$, we propose to monitor the *Hellinger* distance between the densities

$$d_j(\tilde{f}_j, g), \quad j = l,..., \tag{8}$$

where

$$\tilde{f}_j(y \mid (X_{j-1},...,X_{j-k}) = \mathbf{x}) = \frac{h_1^{-1}\sum_{i=1}^{N} K_{h_1}^1(y_i^j - y)\tilde{D}\left((y,\mathbf{x}),(Y_j^N,\mathbf{X}_j^N)\right)K_h(\mathbf{x}_i^j - \mathbf{x})}{\sum_{i=1}^{N}\tilde{D}\left(\mathbf{x},\mathbf{X}_j^N\right)K_h(\mathbf{x}_i^j - \mathbf{x})}, \tag{9}$$

is the adjusted kernel density estimate of $\tilde{f}_j$ and $K_{(\cdot)}$ is univariate or multivariate kernel, $\mathbf{K}_h(\cdot) = h^{-1}\mathbf{K}(\cdot/h)$, $\tilde{D}(\cdot,\cdot)$ is the adjusted sample depth (5).

For the both proposals, in order to choose the bandwidths h we presently use a variant of cross-validation from [7] applied to the most central points in the window with respect to the reference sample, e.g., $\{y \in Y_j^N : D(y, Y^g) \geq \alpha\}$, where Y^g denote the reference sample. In (7) and (9) we propose to use *adjusted simplicial depth*, however it is possible to make use of other "more smoothly trimming" depth function, e.g., *projection depth*. For the computational convenience purposes we propose to choose the well known *Hellinger* or *Kolmogorov* distance as the distance between the density estimate and the reference density. For "regular distributions", it seems to be sufficient to approximate this distance using usual pointwise distances between the densities in say 100 – 1000 points. In case of a complete lack of the knowledge about the stream model (we need the reference densities) we propose to estimate the densities first by means of the statistics (7) or (9), eventually decompose the output density by means of a non hierarchical clustering algorithm (e.g., k-trimmed means) and then simulate the reference samples by means of the well known *inverse distribution function* method.

IV. PROPERTIES OF THE PROPOSALS

Similarly as in [6] we compared the proposed statistics with various estimators of the unconditional $f(\cdot)$ and conditional density function $f(\cdot|\cdot)$ through several simulated models of the data streams, involving independent observations, nonlinear time series and time series models exhibiting several regimes including TAR with trend (*threshold autoregressive model*), and CHARME (*conditional autoregressive mixture of models*) – for details of the models see [4] and [18]. As benchmark estimators we used kernel density estimator with normal kernel, k- nearest neighbors' density estimator, and the adjusted Nadaraya-Watson estimator originally proposed in [6]. For each simulated sample, the performance of the estimators used in the proposals was evaluated in terms of the *mean absolute deviation error*, *integrated mean square error* and visually by means of *functional boxplot* (i.e., the estimated densities were the observations) – see [17]. For example, in order to investigate finite window properties of the proposals we 500 times generated samples, each of length 10000 observations, from the time series model CHARME consisted of two AR-GARCH sub-models or consisted of three AR or SV sub-models). We used moving window of a fixed length of 100 obs. We considered streams with and without up to 5% of the additive outliers – for details see [15]. The unconditional and conditional densities of the sub-models, which comprised on the used CHARME model, were closer or more distant from each other according to the *Hellinger* distance. One of the densities was treated as a *null hypothesis*; subsequent densities represented the *alternative hypotheses*. Next we calculated kernel density estimates of the proposed statistics (8) and (9) under null and alternative hypotheses. Significant differences of the distributions (e.g., location shifts) of the proposals under null and alternative hypotheses indicated their good discriminative properties – their usefulness in the monitoring of the economic data stream. The estimated distributions of the statistics were similar for different density families of submodels – what give us a hope for their universal consistency (i.e., distributions of the statistics are independent from the underlying distributions). Results of the simulations were quite promising especially in cases of the data streams containing outliers. However we had to cope with the crucial issues of appropriate and computationally feasible bandwidth choice and weights calculation. In the simulations we used the cross-validation approach from [7] applied to the most central points (for which sample simplicial depth function takes value higher than a certain prefixed threshold) and an approximate depth calculation algorithm implemented in [2]. We presently study a possibility of an application new promising approaches to approximate calculation of the sample depth proposed recently in [21] and [22]. We implement our ideas in [2].

Fig. 1 presents a part of the results of the simulation studies of small samples properties of our proposals. Fig. 1 presents the functional boxplot for 100 density estimates obtained on base of 100-obs. samples drawn from Student t distribution with 5 degree of freedom (left), and the functional boxplot for 100 conditional density estimates obtained on base of 100-obs. samples drawn from bivariate normal distribution with mean vector (0,0) and covariance matrix consisted of rows (10,3) and (3,2) – conditional density of the first coordinate under the condition that second coordinate equals 1 (right). Each of the samples consisted of up to 5% additive outliers. The estimates obtained by means our proposals were not affected by the

 ISBN: 978-1-61208-268-4

outliers – we therefore conclude that they are quite promising in the context of robust analysis of the economic stream. The proposals need further studies of the issues concerning the bandwidth choice and tuning the depth based weights adjusting the kernel estimates (7) and (9).

V. CONCLUSIONS

We proposed two depth based statistics for the robust monitoring of unconditional and conditional distributions of the data stream. Results obtained so far are quite promising in the context of robust analysis of the data stream. They are robust to outliers being sensitive to the major changes of the stream at the same time. Most of the robust and nonparametric multivariate statistical procedures are computationally very intensive and has to cope with so called "*curse of dimensionality*" (i.e., sparsity of the data in many dimensions). We actually intensively study the possibility to overcome these substantial difficulties.

ACKNOWLEDGMENT

The author thanks for financial support from Polish National Science Center grant UMO-2011/03/B/HS4/01138.

REFERENCES

[1] Aggerwal Ch. C. (ed.), Data Streams – Models and Algorithms, Springer, New York, 2007.
[2] Bocian, M. Kosiorowski, D., Węgrzynkiewicz, A., Zawadzki, Z. Depth Procerures R package {depthproc}, 2012, https://r-forge.r-project.org/projects/depthproc/ [retrieved: Feb. 2013]
[3] Donoho, D., High-dimensional Data Analysis: The Curses and Blessings of Dimensionality, Manuscript, 2000, http://www-stat.stanford.edu/~donoho/Lectures/AMS2000/Curses.pdf
[4] Fan, J. Yao, Q., Nonlinear Time Series: Nonparametric and Parametric Methods, Springer, New York, 2005.
[5] Genton M. G., Lucas A., Comprehensive Definitions of Breakdown Points for Independent and Dependent Observations, Journal of the Royal Statistical Society Series B, 2003, 65, 81 – 84.
[6] Hall, P., Rodney, C. L. and Yao, Q., Methods for Estimating a Conditional Distribution Function. Journal of the American Statistical Association, vol. 94, 1999, pp. 154-163.
[7] Hall, P., Racine, J., Li, Q, Cross-Validation and the Estimation of Conditional Probability Densities, Journal of the American Statistical Association, vol. 99, pp. 1015-1026.
[8] Hahsler, M., Dunhamr, H. M., EMM: Extensible Markov Model for Data Stream Clustering in R, Journal of Statistical Software, vol. 35, 2010, pp. 2 – 31.
[9] Härdle, W., Hautsch, N. and Overbeck, L. Applied Quantitative Finance, 2nd edition, Springer, Heidelberg, 2009.
[10] Jacod, J., Shiryaev, A.N., Limit Theorems for Stochastic Processes, Second ed., Springer-Verlag, New York, 2003.
[12] Kosiorowski, D., Student Depth in Robust Economic Data Stream Analysis, Colubi A. (Ed.) Proceedings COMPSTAT'2012, ISI/IASC, 2012, pp. 437 – 449.
[13] Kosiorowski, D., Snarska, M., Robust Monitoring of a Multivariate Data Stream, 2013, unpublished, https://r-forge.r-project.org/projects/depthproc/ [retrieved: Feb. 2013]
[14] Li, J., Liu, R. Y. New Nonparametric Tests of Multivariate Locations and Scales Using Data Depth. Statistical Science, vol. 19, 2004, pp. 686 – 696.
[15] Maronna, R. A., Martin, R. D., Yohai, V. J., Robust Statistics - Theory and Methods. Chichester: John Wiley & Sons Ltd., 2006.
[16] Muthukrishan, S., Data Streams: Algorithms and Applications, Now Publishers, 2006.
[17] Ramsay, J. O., Hooker, G., Graves, S., Functional Data Analysis with R and Matlab, New York, Springer, 2009.
[18] Shalizi C. R., Kontorovich, A., Almost None of the Theory of Stochastic Processes A Course on Random Processes, 2007, http://www.stat.cmu.edu/~cshalizi/almost-none/ [Feb. 2013]
[19] Serfling, R., Depth Functions in Nonparametric Multivariate Inference, In: Liu R.Y., Serfling R., Souvaine D. L. (Eds.): Series in Discrete Mathematics and Theoretical Computer Science, AMS, vol. 72, 2006, pp. 1 - 15.
[20] Stockis, J-P.,Franke, J., Kamgaing, J. T., On Geometric Ergodicity of CHARME Models, Journal of the Time Series Analysis, vol. 31, 2010, pp. 141 – 152.
[21] Szewczyk, W., Streaming Data, Wiley Interdisciplinary Rev.: Computational Statistics, vol. 3, 2010, [retrieved: Feb. 2013]
[22] Torti, F., Perrotta, D., Atkinson, A. C, Riani, M., Benchmark Testing of Algorithms for Very Robust Regression, Computational Statistics and Data Analysis, vol. 56, 2012, pp. 2501–2512.
[23] Shao, W., Zuo,Y. (2012). Simulated Annealing for Higher Dimensional Projection Depth. Computational Statistics and Data Analysis, vol. 56, 2012, pp. 4026–4036.

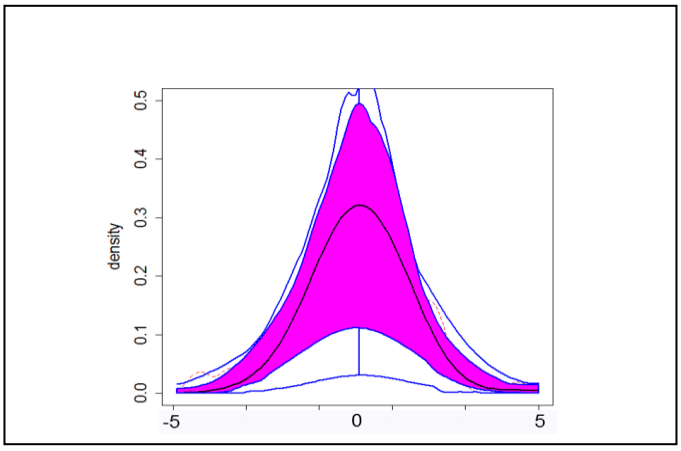

Figure 1. Functional boxplots for 100 density estimates obtained on base of 100-obs. samples drawn from Student t with 5 degree of freedom consisted of 5% outliers (left), and for 100 conditional density estimates obtained on base of 100-obs. samples drawn from bivariate normal distribution with mean vector (0,0) and covariance matrix consisted of rows (10,3) and (3,2) (right). Each sample consisted of 5% outliers. Our own calculations using {fda} R package.

 ISBN: 978-1-61208-268-4

Static versus Dynamic Group-Screening Models

Dieter Claeys, Joris Walraevens, Bart Steyaert, Herwig Bruneel
SMACS Research Group, Department TELIN
Ghent University
Ghent, Belgium
Email: Dieter.Claeys@telin.ugent.be

***Abstract*—Group screening can lead to a tremendous reduction in number of tests and costs, and therefore has attracted considerable attention in literature. For several decades, a model from Robert Dorfman was the de facto standard to determine the optimal group size. However, more recently, it has been pointed out that the model from Dorfman is rather static, i.e., a predetermined large number of items has to be screened, whereas in reality the context is rather dynamic: items arrive at random moments in time, in groups of different and random size.**

In this paper, we investigate to what extent the optimal group size in the static model from Dorfman remains efficient in a dynamic context. This is of vital importance for practitioners, as the model from Dorfman was the de facto standard for several decades and thus has to be validated. On top of that, even though dynamic models exist nowadays, these are much harder to implement and require a time-consuming processing time, due to the numerical work that is involved, such as repeatedly calculating zeroes of functions and solving sets of equations.

***Keywords*-group screening; optimal group size; static model; dynamic model; model validation**

I. INTRODUCTION

Classification of items as good or bad can often be achieved more economically by screening the items in groups rather than individually. The underlying reason is that when a test on a group returns good, it can be concluded (after one test only) that all items within the group are good. Dorfman [1] was the first to introduce the paradigm of group screening and he found an immediate application in the detection of syphilitic men drafted into military service during WWII. He suggested to apply this procedure also to manufacturing processes where the defective goods have to be eliminated from the collection of all produced goods. Later on, many researchers applied this paradigm to screen blood for the presence of HIV [2][3][4][5][6][7], Influenza [8] and West Nile Virus [9] (group screening is in this context generally referred to as blood pooling). The range of application even streches further. Macula [10] and Manoli et al. [11] applied group screening to DNA screening and Dean and Lewis [12] (chapter 3), Xie et al. [13] and Zhu et al. [14] utilized it for drug discovery. Finally, group screening has also found its entrance in the field of computer science, for instance in the study of web services [15][16], image compression [17], multiple access protocols [18], optical networks [19], encryption [20], etc.

When group screening is feasible, the *selection of the group size is crucial*: the larger a group size, the more items can be screened by only one test, but the more likely it becomes that one or more items of the group are bad, inferring that retesting becomes necessary. This can, for instance, be achieved by retesting all items of the group individually, which is often referred to as *group-individual screening policy* (see, e.g., [2]). However, in many occasions, a *group-subgroup screening policy* is adopted, whereby the group is divided into subgroups which are each subjected to a new group test. Traditionally, a bad group is divided in two subgroups of equal size whereby the items of a bad subgroup are retested individually [21].

For several decades, a mathematical model from Robert Dorfman [1] was the de facto standard to determine the optimal group size. This model is essentially **static**: it postulates that a population consisting of a predetermined large number of items has to be screened whereby all items are present from the beginning. However, Abolnikov and Dukhovny [2] correctly pointed out that the practical context is usually **dynamic**: items are not all present from the start, and arrive at random moments in time, possibly in groups of different and random size. For instance, trucks from various regions of a country arrive at the blood screening laboratory at random moments of the day, with a variable number of blood samples to be screened. Abolnikov and Dukhovny [2] dealt with the dynamic nature by relying on queueing theory. Since then, Bar-Lev et al. [22] and Claeys et al. [21] further developed and analyzed queueing models to better include the dynamic nature of item arrivals.

An important disadvantage of dynamic models and analyses, is that they are *much harder to implement* and the *processing time is slow* due to the numerical work that is involved, such as repeatedly calculating zeroes of functions and solving sets of equations. A natural question crucial to practitioners is thus the following: **under which circumstances does the static model yield accurate results in a dynamic context**? This is the question we wish to

 ISBN: 978-1-61208-268-4

answer in this paper. More specifically, we examine to what extent the optimal group size in the static model from Dorfman [1] remains optimal or efficient in the dynamic (queueing) model from Claeys et al. [21]. We compare with [21] and not with the dynamic model from Bar-Lev et al. [22] as they study an incomplete-identification scenario, which means that when a group test returns bad, the whole group is discarded. In Claeys et al. [21] and Dorfman [1] on the other hand, the bad items need to be separated from the good, i.e., complete identification is necessary, incurring that retesting is essential. We also prefer to not adopt the model from Abolnikov and Dukhovny [2], as only the system content at service completion times is established in their paper.

The remainder of this paper is structured as follows: in Section II, we describe the static and the dynamic model in detail. Thereafter, we evaluate in Sections III and IV to what extent results obtained by the static model are valid in a dynamic context. Finally, conclusions are drawn in Section V.

II. Description of the models

In this section, the static model of Dorfman [1] and the dynamic model of Claeys et al. [21] are reviewed in detail.

A. Static Model

In the static model, the bad items in a (huge) population have to be identified. Dorfman [1] has deduced a formula for the mean number of tests ($\mathrm{E}[T]$) required to screen a population consisting of N items when the group-individual policy is adopted with group size c. Let $\overline{p}$ characterise the probability that a random item is good. Then $(1-\overline{p}^c)$ is the probability that a group of c items is bad (at least one item is bad), in which case c additional individual tests are necessary after this group test to actually identify the bad item(s). Hence, the mean number of tests to screen a group of c customers equals $1+(1-\overline{p}^c)c$. As the population is divided in N/c subgroups, the average number of tests to screen the entire population reads

$$\mathrm{E}[T] = \frac{N}{c} + N(1-\overline{p}^c) \ . \tag{1}$$

Along the same lines, it is possible to deduce an expression for the average number of required tests in case of other screening policies. For instance, for the group-subgroup policy whereby a bad group is divided in two subgroups of equal size which are retested individually when the (sub)group test returns bad, it can be found that

$$\begin{aligned}\mathrm{E}[T] = \frac{N}{c}\Big[1 &+ \left(1-\overline{p}^{\lceil c/2\rceil}\right)\overline{p}^{\lfloor c/2\rfloor}\left(2+\lceil c/2\rceil\right) \\ &+ \left(1-\overline{p}^{\lfloor c/2\rfloor}\right)\overline{p}^{\lceil c/2\rceil}\left(2+\lfloor c/2\rfloor\right) \\ &+ \left(1-\overline{p}^{\lceil c/2\rceil}\right)\left(1-\overline{p}^{\lfloor c/2\rfloor}\right)(2+c)\Big] \ . \end{aligned} \tag{2}$$

The first term between brackets expresses that at least one test is required per group, the second represents the situation whereby only the first subgroup is bad, the third term corresponds with only the second subgroup being bad, whereas in the final term both subgroups are bad.

B. Dynamic Model

In [2], it was stated that the dynamic nature of the item arrivals can be captured by a queueing model. We here briefly recapitulate the queueing model that is presented in [21]. It is a discrete-time queueing model whereby the numbers of item arrivals during consecutive time slots are modelled by a sequence of independent and identically distributed random variables, with common mass function

$$a(n) \triangleq \Pr[n \text{ arrivals in a random slot}] \ ,$$

and probability generating function (PGF) $A(z)$, i.e.,

$$A(z) \triangleq \sum_{n=0}^{\infty} a(n) z^n .$$

The mean value, often referred to as mean arrival rate, is denoted by λ and is by definition equal to $A^{'}(1)$ (we use primes to indicate derivatives).

The items join the queue in awaitance of being screened by the testing facility ("the server"). The items are screened ("served") in groups, which is in queueing theory called batch service (e.g., [23][24][25][26]) or bulk service ([27][28][29][30]). It is assumed that a single test takes exactly one slot and that tests are initiated and completed at slot boundaries. In order to avoid confusion, we adopt the term group screening for the complete process, i.e., for the first test on the entire (original) group and the other tests (if any) on subgroups or individual items of the group. The "service time" of a group of items corresponds with the number of tests required to screen the group and can thus take several slots. As the number of tests required to screen a group depends on the number of items in that group, the service time of a group is dependent on the number of items within the group (a larger group has a larger probability of being bad). The service time of a group consisting of j items is represented by S_j and its corresponding PGF by $S_j(z)$. The following expression for $S_j(z)$ is established in [21] for the group-individual screening policy

$$S_1(z) = z \ ,$$

$$S_j(z) = \overline{p}^j z + (1-\overline{p}^j) z^{j+1} \ , \qquad j \geq 2 \ .$$

This can be comprehended as follows: a group consisting of one item requires only one test. In the other case, a group of size j is good with probability $\overline{p}^j$ and thus requires one (group) test. When the group is bad (with probability

 ISBN: 978-1-61208-268-4

$(1-\overline{p}^j)$), j additional individual tests are necessary, leading to a service time of $j+1$ slots.
For the group-subgroup screening policy presented in Section II-A it was found in [21] that

$$S_1(z) = z \ ,$$

$$S_2(z) = \overline{p}^2 z + (1-\overline{p}^2)z^3 \ ,$$

$$S_3(z) = \overline{p}^3 z + \overline{p}^2 p z^3 + (1-\overline{p}^2)z^5 \ ,$$

$$\begin{aligned} S_j(z) &= \overline{p}^j z + \overline{p}^{\lceil j/2 \rceil}\left(1-\overline{p}^{\lfloor j/2 \rfloor}\right) z^{\lfloor j/2 \rfloor + 3} \\ &\quad + \overline{p}^{\lfloor j/2 \rfloor}\left(1-\overline{p}^{\lceil j/2 \rceil}\right) z^{\lceil j/2 \rceil + 3} \\ &\quad + \left(1-\overline{p}^{\lfloor j/2 \rfloor}\right)\left(1-\overline{p}^{\lceil j/2 \rceil}\right) z^{j+3} \ , \qquad j \geq 4 \end{aligned}$$

The cases $j < 4$ are considered separately to avoid that a bad group or subgroup consisting of one item gets retested. In the expression for $j \geq 4$, the first term corresponds with a good group, the second with only the first subgroup being bad, the third with only the second subgroup being bad and the final term with both subgroups being bad.

At this point, it is important to realize that the dynamic nature of the item arrivals entails **two additional differences** as compared to the static model. First, it is necessary to select a **minimum group size** next to the maximum c from the static model (in the sequel we denote this minimum by l), as it might occur that less items are present than the maximum group size when the testing facility is available. When less than l items are present, screening is postponed, whereas otherwise screening is initiated even if less than c items are present.
Second, whereas in the static model only the number of required tests to screen the entire population matters, **various performance measures** can be of importance in case of a dynamic model. In this paper, we restrict ourselves to the, in our opinion, most important, performance measures for dynamic models. The first is the *testing probability* f, defined as the fraction of slots during which the testing facility is busy. It is equal to the probability that the testing facility is testing (a group, a subgroup or an individual item) during a random time slot. This performance measure is especially of importance from an operational point of view: the smaller the testing probability, the cheaper the testing strategy. The second is the *mean delay* of items ($\overline{D}$), i.e., the average time that an item remains in the test center (the "system"). More specifically, the delay of an item is the time, starting at the end of the time slot wherein the item arrives, until the item has been screened. As screening starts and ends at slot boundaries, the delay of an item is expressed as an integral number of time slots. As opposed to the testing probability, the mean delay is especially of importance from the point of view of the items to be screened. For instance, when items represent blood samples, it is necessary to inform the patients as soon as possible whether or not they are infected by some disease and the mean delay is a measure for this. The following formulas have been deduced in [21] for these performance measures:

$$f = 1 - \sum_{n=0}^{l-1} d_n \ , \tag{3}$$

$$\begin{aligned} \overline{D} = \Bigg[& 2c\mathrm{E}\left[S_c\right]\lambda \sum_{n=0}^{l-1} d_n + c(c-1)\sum_{n=0}^{l-1} d_n + 2c\sum_{n=0}^{l-1} d_n n \\ & + \sum_{n=l}^{c-1} d_n g_n - c(c-1) + S_c^{''}(1)\lambda^2 + \mathrm{E}\left[S_c\right] A^{''}(1) \Bigg] \\ & / \left[2\lambda(c - \mathrm{E}\left[S_c\right]\lambda)\right] \ , \end{aligned} \tag{4}$$

with

$$\begin{aligned} g_n \triangleq \ & \mathrm{E}\left[S_n\right] c\left[c-1+2n\right] - \mathrm{E}\left[S_c\right] n\left[n-1+2c\right] \\ & + 2\lambda(c-n)\mathrm{E}\left[S_n\right]\mathrm{E}\left[S_c\right] + \lambda\left[cS_n^{''}(1) - nS_c^{''}(1)\right] \ , \end{aligned}$$

and $\mathrm{E}\left[S_n\right] = S_n^{'}(1)$ by definition. The boundary probabilities d_n ($n = 0, \ldots, c-1$) are the solutions of the following set of c linear equations:

$$\sum_{n=0}^{l-1} d_n z_i^n + \sum_{n=l}^{c-1} d_n \frac{z_i^n - S_n(A(z_i))}{1 - A(z_i)} = 0 \ , \ 1 \leq i \leq c-1 \ ,$$

$$-c + \mathrm{E}\left[S_c\right]\lambda = -c\sum_{n=0}^{l-1} d_n + \sum_{n=l}^{c-1} d_n[n\mathrm{E}\left[S_c\right] - c\mathrm{E}\left[S_n\right]] \ .$$

The z_i's ($1 \leq i \leq c-1$) are the $c-1$ zeroes of $z^c - S_c(A(z))$ inside the closed complex unit disk $\{z \in \mathbb{C} : |z| \leq 1\}$ that are different from 1.

Before we compare results of the static and the dynamic model, it is important to stress that in order to determine optimal group sizes l and c, it is necessary to calculate (3) or (4) (whichever is the intended criterium) for various values of l and c and then select those values that minimize f or $\overline{D}$. On top of that, (3) and (4) rely on the d_n's, which in turn are dependent on l and c. Therefore, **those boundary probabilities have to be calculated for every l and c, by each time calculating zeroes z_i and solving a set of equations**. Hence, this procedure is complicated for practitioners due to the numerical work involved and requires much processing time.

 ISBN: 978-1-61208-268-4

Remark 1: It should be noted that formulas (3) and (4) are valid under the assumption that the system is in steady state. The system can reach steady state if and only if

$$\lambda < \frac{c}{\mathrm{E}\left[S_c\right]} \ .$$

This inequality guarantees that items that enter the test center will eventually (i.e., after a finite time) be screened. It expresses that the average number of items that enter the test center in a random slot must be smaller than the average number of items that can leave the test center at the end of a slot if many items are present. This is a natural assumption in practice.

In the remainder of this paper, we compare the optimal group size in the static model (in the sequel referred to as optimal static group size) with the minimum and maximum group sizes that produce the smallest testing probability f (Section III) and those that generate the smallest average delay $\overline{D}$ (Section IV) in the dynamic model.

III. TESTING PROBABILITY

In this section, we compare the optimal static group size with the group sizes that minimize the testing probability (f) in the dynamic model. Note first that formulas (1)-(2) and (3)-(4) for the static and dynamic model do not consist of the same parameters. Before we can fairly compare the optimal group sizes, we have to study the influence of these parameters. We therefore represent in Tables I and II the optimal static group size versus the population size N for respectively the group-individual and the group-subgroup screening procedure as presented in Section II-A. To find the optima, we have calculated $\mathrm{E}\left[T\right]$ for a wide range of values of c and selected the value that produces smallest $\mathrm{E}\left[T\right]$ as optimum (see [21] for more information). We observe that the population size has no impact on the optimal static group size.

Next, we turn to the dynamic model and evaluate the influence of $A(z)$ on the group sizes that minimize f. The optimal maximum group size is illustrated in Tables III and IV for several values of λ and $\overline{p}$ and for two distributions for the number of item arrivals during a random slot: the Poisson distribution, i.e., with PGF

$$A(z) = e^{\lambda(z-1)} \ ,$$

and the geometric distribution:

$$A(z) = \frac{1}{1+\lambda-\lambda z} \ .$$

We perceive from Tables III and IV that the mean arrival rate λ and even the whole distribution $A(z)$ have no impact on the optimal maximum group size. Note that the **optimal minimum group size** is not mentioned as it is **equal to the optimal maximum group size** when the testing probability has to be minimized.

Table I
GROUP SIZE THAT MINIMIZES $\mathrm{E}\left[T\right]$ IN CASE OF GROUP-INDIVIDUAL SCREENING (STATIC MODEL)

	$\overline{p}=0.95$	$\overline{p}=0.975$	$\overline{p}=0.99$
$N=1000$	5	7	11
$N=2000$	5	7	11
$N=5000$	5	7	11
$N=10000$	5	7	11

Table II
GROUP SIZE THAT MINIMIZES $\mathrm{E}\left[T\right]$ IN CASE OF GROUP-SUBGROUP SCREENING (STATIC MODEL)

	$\overline{p}=0.95$	$\overline{p}=0.975$	$\overline{p}=0.99$
$N=1000$	8	10	14
$N=2000$	8	10	14
$N=5000$	8	10	14
$N=10000$	8	10	14

These findings are of crucial importance: in order to compare the static and dynamic models, it is not necessary to "map" the parameter N of the (static) population size on the (dynamic) mean arrival rate λ and the PGF of the arrival process $A(z)$: the **exact values of N and λ and the exact expression for $A(z)$ do not have an influence** on the optimal group sizes. In addition, when comparing Tables I-II with Tables III-IV, it is clear that **the static and the dynamic model produce equal optimal group sizes**.

We now prove these findings. Let us start by inspecting the dynamic model. As already mentioned, the optimal minimum group size always equals the optimal maximum group size. The reasoning behind this is that when less items are present than the maximum group size, it is, from the point of view of minimizing the testing probability, better to wait until enough items are present, in order to fully exploit the benefit of group screening (less tests required). As a result, if the testing facility is screening (with probability f), it always screens a group consisting of c items. As a consequence, the average screening time equals $\mathrm{E}\left[S_c\right]$ slots, so that in a random slot wherein the testing facility is screening, the ongoing screening is finished at the end of that slot with probability $1/\mathrm{E}\left[S_c\right]$. Hence, the average number of items leaving the system in a slot because screening is completed equals $fc/\mathrm{E}\left[S_c\right]$. We now rely on this result to translate the well-known "rate-in-rate-out" principle (see e.g., [31]) in terms of the system parameters. The "rate-in-rate-out" principle expresses that in a queueing system in steady state, the mean number of items entering the system per slot equals the mean number of items leaving the system per slot.

 ISBN: 978-1-61208-268-4

Table III
GROUP SIZE THAT MINIMIZES f IN CASE OF GROUP-INDIVIDUAL SCREENING (DYNAMIC MODEL); * MEANS THAT THE TEST CENTER IS NOT ABLE TO COPE WITH ALL SAMPLES, BECAUSE $\lambda \geq c/\mathrm{E}[S_c]$ FOR EVERY VALUE OF c

	$\overline{p} = 0.95$ Poisson	$\overline{p} = 0.95$ geometric	$\overline{p} = 0.99$ Poisson	$\overline{p} = 0.99$ geometric
$\lambda = 1$	5	5	11	11
$\lambda = 2$	5	5	11	11
$\lambda = 3$	*	*	11	11
$\lambda = 4$	*	*	11	11
$\lambda = 5$	*	*	11	11

Table IV
GROUP SIZE THAT MINIMIZES f IN CASE OF GROUP-SUBGROUP SCREENING (DYNAMIC MODEL); * MEANS THAT THE TEST CENTER IS NOT ABLE TO COPE WITH ALL SAMPLES, BECAUSE $\lambda \geq c/\mathrm{E}[S_c]$ FOR EVERY VALUE OF c

	$\overline{p} = 0.95$ Poisson	$\overline{p} = 0.95$ geometric	$\overline{p} = 0.99$ Poisson	$\overline{p} = 0.99$ geometric
$\lambda = 1$	8	8	14	14
$\lambda = 2$	8	8	14	14
$\lambda = 3$	*	*	14	14
$\lambda = 4$	*	*	14	14
$\lambda = 5$	*	*	14	14

Putting these elements together yields

$$\lambda = f \frac{c}{\mathrm{E}[S_c]} \quad . \tag{5}$$

Next, define

$$g(c) \triangleq \frac{\mathrm{E}[S_c]}{c} \quad . \tag{6}$$

As a result, (5) can be transformed into

$$f = \lambda g(c) \quad .$$

At this point, it is crucial to realize that $g(c)$ is independent of λ and $A(z)$, simply because $\mathrm{E}[S_c]$ only depends on c and $\overline{p}$. As a result, the group size c that produces the smallest value f, minimizes $g(c)$ and is independent of λ and $A(z)$. In the static scenario on the other hand, the mean number of tests required to screen a population of size N is equal to

$$\mathrm{E}[T] = \frac{N}{c} \mathrm{E}[S_c] \quad ,$$

because the population is divided in N/c groups with average testing time $\mathrm{E}[S_c]$. Owing to (6), we obtain

$$\mathrm{E}[T] = N g(c) \quad .$$

Analogously as for the dynamic model, we can state that the optimal group size minimizes $g(c)$ and that N has no impact. We can thus conclude that the optimal group size is the same in both models as it minimizes in essence the same function. Finally, it is worth noting that the proof is independent of the screening policy, which thus infers that the conclusions are valid for all screening policies.

In order to illustrate that the optimal group size minimizes $g(c)$ ($\triangleq \mathrm{E}[S_c]/c$), $\mathrm{E}[S_c]$ and $g(c)$ are shown in Table V, for various values of c and in case of the group-subgroup screening policy. We indeed notice that for $\overline{p} = 0.95$, 0.975, 0.99, $g(c)$ is minimized when $c = 8$, 10, 14 respectively (compare with Tables II and IV).

Table V
$\mathrm{E}[S_c]$ AND $g(c)$ FOR VARIOUS VALUES OF c IN CASE OF GROUP-SUBGROUP SCREENING

	$\mathrm{E}[S_c]$ $\overline{p} = 0.95$	$g(c)$ $\overline{p} = 0.95$	$\mathrm{E}[S_c]$ $\overline{p} = 0.99$	$g(c)$ $\overline{p} = 0.99$
$c = 1$	1	1	1	1
$c = 2$	1.1950	0.5975	1.0398	0.5199
$c = 3$	1.4803	0.4934	1.0992	0.3664
$c = 4$	1.7610	0.4402	1.1584	0.2896
$c = 5$	2.0753	0.4151	1.2269	0.2454
$c = 6$	2.3856	0.3976	1.2952	0.2159
$c = 7$	2.7732	0.3962	1.3826	0.1975
$c = 8$	3.1571	**0.3946**	1.4697	0.1837
$c = 9$	3.6126	0.4014	1.5756	0.1751
$c = 10$	4.0647	0.4065	1.6813	0.1681
$c = 11$	4.5829	0.4166	1.8055	0.1641
$c = 12$	5.0982	0.4248	1.9295	0.1608
$c = 13$	5.6744	0.4365	2.0716	0.1594
$c = 14$	6.2479	0.4463	2.2136	**0.1581**
$c = 15$	6.8777	0.4585	2.3735	0.1582

IV. MEAN DELAY

In the previous section, we have shown that the minimum and maximum group sizes that minimize f in the dynamic model are both equal to the optimal static group size. In this section, we investigate whether this also holds when $\overline{D}$ is minimized instead of f in the dynamic model. First, note that the **optimal minimum group size now equals 1**. Indeed, if very few items are present when the testing facility is available, these items would probably suffer a considerable delay if the testing facility would postpone screening until more items are present. In addition, the probability of a group of very few items to be infected is very small, so that most likely the screening of such a group only lasts one slot, in which case the testing facility will be available again at the beginning of the next slot for possible newly arrived items. As a result, we fix the minimum group size to 1.

Next, we illustrate the maximum group size that minimizes $\overline{D}$ for various values of λ and $\overline{p}$, both for the group-individual (Table VI) and the group-subgroup screening policy (Table VII). We observe that the **optimal maximum group size increases as a function of** λ and that **for large enough** λ **it equals the static optimum**. The latter is a consequence of the fact that when the system is heavily loaded, nearly always many items are present, meaning that the system becomes almost equivalent with a static system. In the previous section, we have proved that the optimal

 ISBN: 978-1-61208-268-4

static group size minimizes $g(c)$, the average number of tests per item. It is thus natural to ask why for small and medium values of λ it is better to select a smaller maximum group size than the group size that minimizes the average number of tests per item. In order to understand this, we consider the example with the group-subgroup policy and with $\overline{p} = 0.975$ (Table VII). Assume that 10 items are present, the testing facility is available and $\lambda = 1$. It seems evident to execute 1 group test on the 10 items, as this would lead to the smallest average number of tests per item (Table IV). However, Table VII indicates that a maximum group size equal to 6 is the best option. Let us observe what happens if the maximum group size equals 6 instead of 10, by relying on Table V. When the 10 items are screened together, this takes on average 2.6364 slots. When, on the other hand, only the first 6 items are screened together, these 6 items are screened in, on average, 1.72 slots. Thereafter, the remaining 4 items can be screened, in other words, these items have already been delayed on average 1.72 slots before their screening is initiated. As $\lambda = 1$, on average 5.72 items are present when the screening of the first 6 items is completed. Assume that 6 items are present at that time. These items are screened together and it takes on average 1.72 slots. Hence, the first 6 items benefit and suffer on average a delay of 1.72 instead of 2.6364 slots, whereas the screening of the other 4 items is completed after on average 3.34 instead of 2.6364 slots. As a consequence, the average screening time of these 10 items is $(6 * 1.72 + 4 * 3.34)/10 = 2.368$, which is better than 2.6364 slots. This example thus illustrates why the maximum group size that minimizes $\overline{D}$ can be smaller than the optimal static group size for smaller values of λ.

Table VI
GROUP SIZE THAT MINIMIZES $\overline{D}$ IN CASE OF GROUP-INDIVIDUAL SCREENING; * MEANS THAT THE TEST CENTER IS NOT ABLE TO COPE WITH ALL SAMPLES, BECAUSE $\lambda \geq c/\mathrm{E}[S_c]$ FOR EVERY VALUE OF c

	$\overline{p} = 0.95$	$\overline{p} = 0.975$	$\overline{p} = 0.99$
$\lambda = 1$	3	4	6
$\lambda = 2$	4	5	6
$\lambda = 2.25$	5	5	6
$\lambda = 2.5$	*	5	6
$\lambda = 3$	*	6	7
$\lambda = 3.25$	*	7	7
$\lambda = 3.5$	*	*	7
$\lambda = 4$	*	*	8
$\lambda = 5.11$	*	*	11

Hence, for low and medium λ, the static model overestimates the optimal group size if the mean delay of items has to be optimized. A question that arises in this context is: when the static optimal group size (say c_s) is selected instead of the group size that actually minimizes $\overline{D}$ (we call this c_d),

Table VII
GROUP SIZE THAT MINIMIZES $\overline{D}$ IN CASE OF GROUP-SUBGROUP SCREENING; * MEANS THAT THE TEST CENTER IS NOT ABLE TO COPE WITH ALL SAMPLES, BECAUSE $\lambda \geq c/\mathrm{E}[S_c]$ FOR EVERY VALUE OF c

	$\overline{p} = 0.95$	$\overline{p} = 0.975$	$\overline{p} = 0.99$
$\lambda = 1$	4	6	8
$\lambda = 2$	6	6	8
$\lambda = 2.5$	8	8	8
$\lambda = 3$	*	8	10
$\lambda = 3.75$	*	10	10
$\lambda = 4$	*	*	10
$\lambda = 6$	*	*	14

what is the relative error

$$100(\overline{D}_s - \overline{D}_d)/\overline{D}_s \ ,$$

(in %), with $\overline{D}_s$ the mean delay when $c = c_s$ and $\overline{D}_d$ the mean delay when $c = c_d$. We have therefore depicted this relative error versus λ in Figures 1 and 2, for the group-individual and the group-subgroup screening policy respectively. We observe that the **relative error is extremely small for small values of** λ, that it **first increases as a function of** λ, and **then decreases again when** λ **becomes large**. When for instance $c = 14$ is selected instead of $c = 8$ for small values of λ, it does not matter much because it seldom occurs that more than 8 items are present when the testing facility is available. If λ increases, this occurs more, which then leads to a larger relative error. Finally, when λ becomes large, c_d tends to c_s, which leads to a decaying relative error.

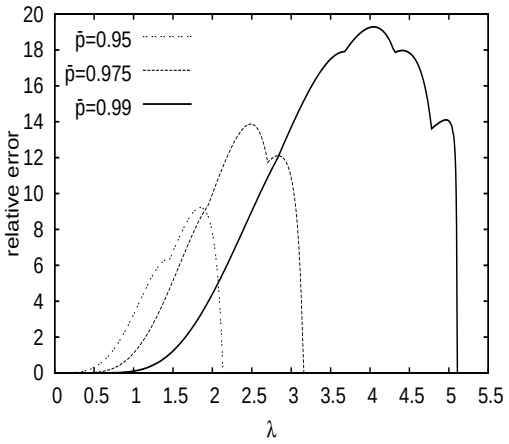

Figure 1. Relative error (in %) versus λ; Group-individual screening policy

Remark 2: Even in those cases whereby it is necessary to adopt the dynamic model because the relative error is significant, this paper provides precious information. Indeed, we have pointed out that the optimal minimum group size equals one and that the optimal maximum group size is upper bounded by the optimal static group size. As a result, $\overline{D}$ has to be calculated only for values of c not larger than

 ISBN: 978-1-61208-268-4

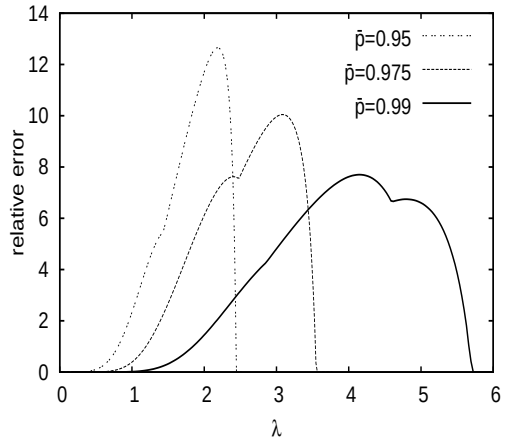

Figure 2. Relative error (in %) versus λ; Group-subgroup screening policy

the optimal static group size, instead of an extensive range of combinations of both l and c, which thus reduces the calculation time considerably.

V. Conclusion and future work

In this paper, we have investigated to what extent the optimal static group size of Dorfman remains efficient in a dynamic context. We have explained that instead of one group size, a minimum and a maximum group size have to be determined in case of dynamic item arrivals. In addition, various performance measures exist, whereas only the number of tests required to screen the population matters in the static model. We have considered the testing probability and the mean delay of items. The former is especially of importance from an operational point of view, whereas the latter is crucial for the items to be tested.

We have shown that when the testing probability has to be minimized, both the optimal minimum and the maximum group size equal the optimal static group size. When, on the other hand, the average delay has to be minimized, the optimal minimum group size equals one, whereas the optimal maximum group size increases as a function of the mean arrival rate (λ) and eventually becomes equal to the optimal static group size for large mean arrival rates. We have demonstrated that selecting the optimal static group size instead of the optimal maximum group size, leads to a small relative error in the mean delay for sufficiently small or large mean arrival rates and to a larger relative error for medium mean arrival rates.

We can thus conclude that this paper clearly indicates under which circumstances the static model from Dorfman produces satisfying results in a dynamic context. This is of vital importance for practitioners, as the model from Dorfman was the de facto standard for several decades and thus has to be validated. On top of that, even though dynamic models exist nowadays, these are much harder to implement and require a time-consuming processing time, due to the numerical work that is involved, such as repeatedly calculating zeroes of functions and solving sets of equations.

Finally, we would like to stress that even when it is necessary to rely on a dynamic model, this paper provides valuable insights which aid in reducing the calculation time considerably.

Although this paper provides precious insights, there are several directions for future research. First, the conclusions in this paper are based on numerical examples and intuitive reasoning. Therefore, we will continue our research in order to prove our findings on a rigorous manner. Next, we will investigate whether the conclusions from this paper also hold when the number of item arrivals during consecutive time slots is not independent and identically distributed, but exhibits some kind of correlation. Finally, we will also study other performance measures, such as the probability that the delay of an item exceeds some large threshold, the mean number of items waiting in the queue to be screened, etc.

Acknowledgment

This research has been funded by the Interuniversity Attraction Poles Programme initiated by the Belgian Science Policy Office.

References

[1] R. Dorfman, "The detection of defective members of large populations," *Annals of Mathematical Statistics*, vol. 14, pp. 436–440, 1943.

[2] L. Abolnikov and A. Dukhovny, "Optimization in HIV screening problems," *Journal of Applied Mathematics and Stochastic Analysis*, vol. 16(4), pp. 361–374, 2003.

[3] S. Fiscus, C. Pilcher, W. Miller, K. Powers, I. Hoffman, M. Price, D. Chilongozi, C. Mapanje, R. Krysiak, S. Gama, F. Martinson, and M. Cohen, "Rapid, real-time detection of acute HIV infection in patients in Africa," *Journal of Infectious Diseases*, vol. 195(3), pp. 416–424, 2007.

[4] T. Quinn, R. Brookmeyer, R. Kline, M. Shepherd, R. Paranjape, S. Mehendale, D. Gadkari, and R. Bollinger, "Feasibility of pooling sera for HIV-1 viral RNA to diagnose acute primary HIV-1 infection and estimate HIV incidence," *AIDS*, vol. 14(17), pp. 2751–2757, 2000.

[5] M. Sherlock, N. Zetola, and J. Klausner, "Routine detection of acute HIV infection through RNA pooling: survey of current practice in the United States," *Sexually Transmitted Diseases*, vol. 34(5), pp. 314–316, 2007.

[6] J. Stekler, P. Swenson, R. Wood, H. Handsfield, and M. Golden, "Targeted screening for primary HIV infection through pooled HIV-RNA testing in men who have sex with men," *AIDS*, vol. 19(12), pp. 1323–1325, 2005.

[7] L. Wein and S. Zenios, "Pooled testing for HIV screening: Capturing the dilution effect," *Operations Research*, vol. 44, pp. 543–569, 1996.

 ISBN: 978-1-61208-268-4

[8] M. Hourfar, A. Themann, M. Eickmann, P. Puthavathana, T. Laue, E. Seifried, and M. Schmidt, "Blood screening for influenza," *Emerging Infectious Diseases*, vol. 13(7), pp. 1081–1083, 2007.

[9] B. Lee and B. Biggerstaff, "Screening the United States blood supply for West Nile Virus: a question of blood, dollars, and sense," *PLoS Medicine*, vol. 3(2), pp. 168–169, 2006.

[10] A. Macula, "Probabilistic nonadaptive and two-stage group testing with relatively small pools and DNA library screening," *Journal of Combinatorial Optimization*, vol. 2, pp. 385–397, 1999.

[11] T. Manoli, N. Gretz, H. Grne, M. Kenzelmann, R. Eils, and B. Brors, "Group testing for pathway analysis improves comparability of different microarray datasets," *Bioinformatics*, vol. 22(20), pp. 2500–2506, 2006.

[12] A. Dean and S. Lewis, *Screening: Methods for Experimentation in Industry, Drug Discovery, and Genetics*. Springer, 2006.

[13] M. Xie, K. Tatshuoka, J. Sacks, and S. Young, "Group testing with blockers and synergism," *Journal of the American Statistical Association*, vol. 96, pp. 92–102, 2001.

[14] L. Zhu, J. Hughes-Oliver, and S. Young, "Statistical decoding of potent pools based on chemical structure," *Biometrics*, vol. 57, pp. 922–930, 2001.

[15] W. Tsai, Y. Chen, R. Paul, N. Liao, and H. Huang, "Cooperative and group testing in verification of dynamic composite web services," *Proceedings of the 28th Annual International Computer Software and Applications Conference (COMPSAC '04)*, vol. 2, pp. 170–173, 2004.

[16] W. Tsai, X. Zhou, Y. Chen, and X. Bai, "On testing and evaluating service-oriented software," *Computer*, vol. 41(8), pp. 40–46, 2008.

[17] E. Hong and R. Ladner, "Group testing for image compression," *IEEE Transactions on Image Processing*, vol. 11(8), pp. 901–911, 2002.

[18] A. D. Bonis and U. Vaccaro, "Constructions of generalized superimposed codes with applications to group testing and conflict resolution in multiple access channels," *Theoretical Computer Science*, vol. 306(1-3), pp. 223–243, 2003.

[19] N. Harvey, "Non-adaptive fault diagnosis for all-optical networks via combinatorial group testing on graphs," *Proceedings of the 26th IEEE International Conference on Computer Communications (INFOCOM2007)*, pp. 697–705, 2007.

[20] J. Fang, Z. Jiang, S. Yiu, and L. Hui, "Checking key integrity efficiently for high-speed quantum key distribution using combinatorial group testing," *Optics Communications*, vol. 284(1), pp. 531–535, 2011.

[21] D. Claeys, J. Walraevens, K. Laevens, and H. Bruneel, "A queueing model for general group screening policies and dynamic item arrivals," *European Journal of Operational Research*, vol. 207(2), pp. 827–835, 2010.

[22] S. Bar-Lev, M. Parlar, D. Perry, W. Stadje, and F. V. der Duyn Schouten, "Applications of bulk queues to group testing models with incomplete identification," *European Journal of Operational Research*, vol. 183, pp. 226–237, 2007.

[23] D. Claeys, J. Walraevens, K. Laevens, and H. Bruneel, "Analysis of threshold-based batch-service queueing systems with batch arrivals and general service times," *Performance Evaluation*, vol. 68(6), pp. 528–549, 2011.

[24] M. Neuts, "The busy period of a queue with batch service," *Operations Research*, vol. 13(5), pp. 815–819, 1965.

[25] K. Sikdar and U. Gupta, "Analytic and numerical aspects of batch service queues with single vacation," *Computers and Operations Research*, vol. 32, pp. 943–966, 2005.

[26] X. Yi, N. Kim, B. Yoon, and K. Chae, "Analysis of the queue-length distribution for the discrete-time batch-service $Geo/G^{a,Y}/1/K$ queue," *European Journal of Operational Research*, vol. 181, pp. 787–792, 2007.

[27] A. Banerjee and U. Gupta, "Reducing congestion in bulk-service finite-buffer queueing system using batch-size-dependent service," *Performance Evaluation*, vol. 69(1), pp. 53–70, 2012.

[28] S. Chang and D. Choi, "Performance analysis of a finite-buffer discrete-time queue with bulk arrival, bulk service and vacations," *Computers and Operations Research*, vol. 32, pp. 2213–2234, 2005.

[29] S. Chang and T. Takine, "Factorization and stochastic decomposition properties in bulk queues with generalized vacations," *Queueing Systems*, vol. 50, pp. 165–183, 2005.

[30] M. Neuts, "A general class of bulk queues with Poisson input," *Annals of Mathematical Statistics*, vol. 38, pp. 759–770, 1967.

[31] H. Bruneel and B. Kim, *Discrete-Time Models for Communication Systems Including ATM*. Kluwer Academic Publishers, 1993.

 ISBN: 978-1-61208-268-4

Algorithm-Based Master-Worker Model of Fault Tolerance in Time-Evolving Applications

Md. Mohsin Ali and Peter E. Strazdins
Research School of Computer Science
The Australian National University
Canberra, ACT 0200, Australia
{md.ali, peter.strazdins}@anu.edu.au

***Abstract*—In order to make the High Performance Computing (HPC) applications fault-tolerant, many application developers are investigating Algorithm-Based Fault Tolerance (ABFT) techniques to improve the efficiency of these applications recovery beyond what existing checkpoint/restart techniques alone can provide. Unfortunately, the standard library Message Passing Interface (MPI) used for implementing this type of application do not have standardized fault tolerance semantics. This paper presents how the fault tolerance semantics of Fault-Tolerant MPI (FT-MPI) can be used as a part of ABFT to design and implement a fault-tolerant algorithm applicable for time-evolving applications which could survive process failures. The model of the presented technique is a master-worker scheme which can tolerate the failures of all worker processes. As an example of time-evolving application, we consider the upwind scheme of one dimensional advection equation solution. We focus on communication-level issues, data prevention techniques, as well as time-evolving control issues. This paper also highlights a common set of issues including failure detection, failed process recovery, duplicate message handling, etc. This contribution will help application developers to resolve different issues of design and implementation of fault-tolerant algorithms for more complex time-evolving applications.**

Keywords-fault tolerance; MPI; FT-MPI; process failure;

I. INTRODUCTION

Today's High Performance Computing (HPC) systems use hundreds of thousands of processing elements to concurrently execute millions of threads and this number is increasing day-by-day. The computational clusters composed of such multiple processing elements called cores are connected with high-speed networks designed to minimize the communication costs and maximize reliability, see for example [1]. A concerted effort is required in order to exploit the full performance of these new computational clusters. This performance is critically needed in areas like climate and environmental research and in physics and energy research characterized by complex scientific models. The most common such models use the solution of systems of partial differential equations in an iterative way. The time-evolving solution of simple one dimensional advection equation is a very basic one among them and is discussed in [2].

Besides exploiting the full performance of such large clusters, a critical issue is how to deal with hardware and software faults that lead to process failures. The failure rate of a system is roughly proportional to the number of processor elements in that system [3]. For instance, a recent study shows that in a particular model of the Blue Gene system located at the Oak Ridge National Laboratory, a 100,000-processor machine experiences a processor failure every few minutes [4]. Since the size of the HPC systems are becoming larger, as we mentioned before, the failure rates of these large systems are increasing day-by-day [5].

The Message Passing Interface (MPI) [6] specification, which is widely used as a parallel programming paradigm for HPC, could not deal with one or more process failures at run-time. Generally, MPI provides two options for handling failures.

- The first option with error handler `MPI_ERRORS_ARE_FATAL`, which is also the default mode of MPI, is to immediately abort the application.
- The second option, which uses error handler `MPI_ERRORS_RETURN`, is just slightly more flexible; handing the control back to the user application without guaranteeing that any further communication can occur. Its purpose is to mainly give an application developer the option to perform some local operations before exiting.

Another important challenge in HPC for dealing with the issues of fault tolerance is the deficiency of availability of both theoretical and practical literature to get an idea about the range of issues during the development of the fault-tolerant program. Besides this, there is a discrepancy between the capabilities of current HPC systems and the most widely used parallel programming paradigm (MPI). Although the MPI specification proves itself for fully exploiting the capabilities of the current architectures, it can not handle the failure of processes similarly. As a result, one of the main reasons why many researchers prefer Fault-Tolerant MPI (FT-MPI) [7] as an interface to implement their applications is because of its capability to handle process

 ISBN: 978-1-61208-268-4

failures in run-time. It is actually an MPI-1 implementation that extended the MPI communicator states and modified the MPI communicator construction functions. Details of this are discussed in Section III.

The contributions of this paper are as follows:

- Design and implement a fault-tolerant algorithm applicable for time-evolving applications which could survive process failures.
- The presented model is a master-worker model which could survive the failure of all workers in the system.
- The failed processes are rebuilt including the recovery of their data.
- Presenting how the fault tolerance semantics of FT-MPI can be used for failure recovery as a part of an ABFT technique.
- This is a very basic model which is currently not scalable, but there is scope of modifying this model to make it scalable.

The rest of the paper is organized as follows. Related research work is discussed in Section II. Section III describes the semantics and interfaces of FT-MPI that are used for implementing fault-tolerant MPI applications surviving process failures. Section IV describes a fault-tolerant version of time-evolving solution of one-dimensional advection equation demonstrating the techniques of detection and recovery of process failure, recovery of lost data, handling of duplicate messages, and controlling of iteration after the failure. Performance comparison of Open MPI with FT-MPI and experimental results demonstrating failure recovery performance of FT-MPI is provided in Section V. Finally, concluding remarks for this paper are given in Section VI.

II. Related Work

A master-worker model of MPI programs that could recover from process failure by using multiple intercommunicators is proposed in [8]. In this model, the management of multiple sets of intercommunicators for a single group of processes is cumbersome in comparison to directly using a single set of intracommunicators. Moreover, this model is not used for time-evolving applications.

A fault-tolerant time-evolving program applicable for ring type communication is proposed in [9]. The focus of this work is on the run-through stabilization component of the developing proposal which is being extended to include flexible recovery strategies of MPI Forum's Fault Tolerance Working Group [10]. The run-through stabilization component of the proposal provides an application with the ability to continue running and using MPI even when one or more processes in the MPI universe fail, but failed processes become permanently unresponsive to communications. Moreover, data recovery issues are not considered in this component. So, this approach will not be applicable for the systems which require the recovery of the lost data due to process failure.

An algorithm-based fault tolerance technique using checksum for detecting and recovering one error in HPC is proposed in [11]. This is applicable for specific problems like parallel matrix-matrix multiplication. Other fault-tolerant algorithms related to matrix operations are available in [12] and [13].

A floating-point arithmetic coding approach into diskless checkpointing is proposed in [14] to address the associated round-off errors. This approach could survive only a small number of process failures and could not survive all process failures.

A natural fault-tolerant algorithm for iterative problems is proposed in [15] where the algorithm computes a new approximate solution from the data of the non-failed processes after the failure. The main drawback of this approach is that the convergence after failure of the processes is no longer the same as the original method. Moreover, this algorithm is also not applicable for the case where it needs actual solution.

An algorithm-based recovery approach for iterative methods is proposed in [16] where neither a checkpoint nor a roll-back is necessary for recovering the data of the failed processes. It demonstrates that, for many iterative methods, if the parallel data partition scheme satisfies certain conditions, the iterative methods themselves can maintain enough inherent redundant information to tolerate failures in the computation. Under this condition, the computation can be restarted from where the failure occurs without any checkpointing. Although this approach is scalable, it cannot be used for generalized iterative problems.

III. FT-MPI Semantics and Interfaces

Since current semantics of MPI could not guarantee further communication to occur after a failure, as we mentioned before, we need to modify its semantics so that it can take some corrective actions to rebuild the communicator with an aim to continue the communication after detecting a failure. FT-MPI is such an MPI-1 implementation, as we mentioned before, that extended the MPI communicator states and modified the MPI communicator construction functions. The modified semantics of FT-MPI include state of the communicator, state of the process, mode of communicator, mode of the message, etc. As for example, FT-MPI extends the MPI communicator states from {`valid, invalid`} to a range {`FT_OK, FT_DETECTED, FT_RECOVER, FT_RECOVERED, FT_FAILED`}. It can be usually described as {`OK, PROBLEM, FAILED`}, whereas the remaining are used for the internal fault recovery algorithm of FT-MPI. Similarly, typical states of MPI processes called {`OK, FAILED`} are replaced by {`OK, Unavailable, Joining, Failed`} in FT-MPI.

A communicator in FT-MPI changes its state after detecting a probable error when either

- an MPI process changes its state due to its failure or anything else, or
- a communication within that communicator fails for some reason.

For the first case, all communicators that include this process are changed. Whereas for the second case, not all communicators are forced to be updated. Changing the communicator state includes rebuilding that communicator in order to recover from the probable error. A modified version of one of the communicator functions e.g. `MPI_Comm_{create, split or dup}` is used for this rebuild. Depending on the mode of failure, a newly built communicator can hold several modes such as `SHRINK`, `BLANK`, `REBUILD`, and `ABORT`. In order to keep the data structure contiguous, the communicator is reduced by `SHRINK` mode to fill the rank(s) of failed process(s) by the rank of the following process(es). So, the ranks of the processes are changed and forcing the application to recall `MPI_COMM_RANK`. `BLANK` is similar to `SHRINK` except that the communicator can contain gaps where there were problems in the processes during communication. These gaps can be filled later when necessary, but communicating with a gap causes an invalid rank error. Moreover, `MPI_COMM_RANK` returns the total number of processes including failed one which is no more valid. The more complex mode is `REBUILD`, which forces the creation of new processes to fill all the gaps containing empty ranks of the communicator. The new processes can be placed either in the empty ranks, or the communicator can be shrunk at first and then the remaining processes filled at the end. The last mode `ABORT` forces the application to abort immediately after detecting an error and there is nothing to do for a user. Example 3 of [17] shows how a communicator is simply rebuilt and reused when the communicator detects an error.

The MPI standard does not return additional error codes and classes except standard ones. But FT-MPI notifies the process failure once the application attempts to communicate directly (e.g., point-to-point operations) or indirectly (e.g., collective operations) with the failed process through the return code called `MPI_ERROR_OTHER` of the function, and error handler set on the associated communicator. This return code also makes additional information available via the *attribute caching mechanism* including a human readable form [17]. The first form returns the error information for a complete communicator in terms of the number of failures per rank (example 2 of [17]) since last recovery. The second form returns the failed ranks in the same order as they happened locally (example 1 of [17]). Using these information, an application developer can write down a fault-tolerant program to handle the error from the user level. Other than this, communications within a communicator is controlled by a message mode and can be either `NOP` or `CONT`. For `NOP`, there is nothing to do from the user level and allows the application to return from any point in the code to a state where it can take appropriate action as soon as possible based on the error. On the other hand, with `CONT` mode, all communication consisting of unaffected processes can continue as normal and attempts to communicate with a failed process reports an error until the communication state is reset. A sample FT-MPI master-worker code is available in example 4 of [17], where the communicator mode is `BLANK` and the communication message mode is `CONT`. The master keeps track of the work allocated and on an error it checks whether there is any surviving workers remaining or not. If any of these are available, then it just reallocates the work to them to continue the computation.

IV. Time-Evolving Applications: One-Dimensional Advection Equation Solution

The solution of advection equation is an important subject in scientific HPC. There are two reasons behind this. Firstly, advection is a part of important applications of HPC: meteorology, climatology, and air pollution. Secondly, many of the computations done on HPC systems involve the solution of partial differential equations. Since advection equation is actually a relatively simple partial differential equation, it provides a good starting point to study a broad class of computations performed on supercomputers.

There are broad classes of advection equations ranging from simple to complex. The one-dimensional advection equation is the most basic one among them. Algorithms for solving such basic equations are available in [2], along with the definition and uses of ghost values. The principle of these algorithms are like that the original advection values are divided into parts and then distributed them into a number of processors, say n processors. Then in each iteration, the following activities are performed.

- Each processor updates their ghost values by exchanging messages with their *left* and *right* neighbors (*left* of processor 1 is processor n and *right* of processor n is processor 1), see Fig. 1.
- Each processor computes their flux values and update their advection values according to the type of advection equation.

Finally, at the end of the iteration, the computed advection values from each of the processors are combined to generate the actual advection values.

Let us discuss the fault-tolerant version of this algorithm. Literally, there are many meanings of "fault-tolerant". See [8] for details. But in this paper, by fault-tolerant, we mean tolerance of process failure and this failure may happen for any reason. We consider a process failure as a *fail-rebuild-working* failure, that is, failed processes will be rebuilt and become available to communications. Since the data of a failed process is lost, this algorithm also recovers this lost data.

 ISBN: 978-1-61208-268-4

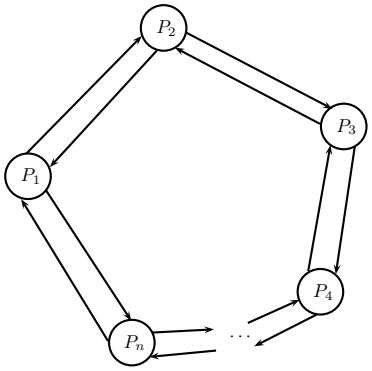

Figure 1. Communication in non-fault-tolerant advection equation solution.

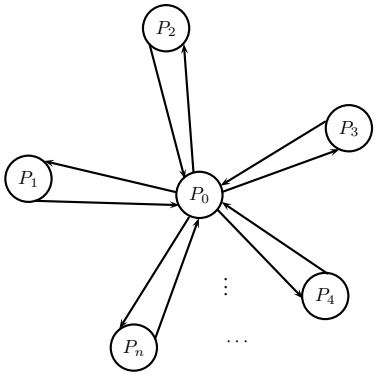

Figure 2. Communication in fault-tolerant advection equation solution.

The model of our fault-tolerant algorithm is master-worker where communications between a worker to its neighbors are done through master similar to Fig. 2 when process P_0 serves as master. In order to update the ghost values for each worker by exchanging messages with *left* and *right* neighbors (see Fig. 1), the following activities are performed.

- The master receives messages containing advection values from all workers and stores them in memory.
- The master calculates *left* and *right* ghost values for each worker from these stored values and send these calculated values to corresponding workers to update their ghost values.

Storing these values have additional benefits which are discussed in Section IV-C. However, it is easily observed that the total number of communication for updating ghost values of the workers through master is the same as that of without master except the increased size of the exchanged messages. See Fig. 1 and 2 for comparison.

The fault-tolerant algorithm for solving one-dimensional advection equation is shown in Fig. 3. The main points of this algorithm are as follows.

- A user-defined error handler is registered on Lines 8 and 9 (details in the algorithm shown in Fig. 8) for handling error including process failure and rebuilding the failed processes.
- Original advection values (input) generation and distributing them to workers are done on Line 11.
- Saving the values of workers to master is done on Line 13 such that the master can provide these values to workers when they rebuild after the failure.
- The main iteration is going on between Lines 16–22.
- The master is receiving advection values from all the workers on Line 18 (details in the algorithm shown in

Function `int main(`*int argc, char *argv[]*`)`

```
1:  /* Initialize MPI */
2:  MPI_Comm MCW = MPI_COMM_WORLD;
3:  MPI_Errhandler errh;
4:  int rc_init = MPI_Init(&argc, &argv);
5:  MPI_Comm_rank(MCW, &process_id);
6:  MPI_Comm_size(MCW, &procs);
7:
8:  MPI_Errhandler_create(recover, &errh);
9:  MPI_Errhandler_set(MCW, errh);
10:
11: input_generate_and_distribute();
12:
13: save_values_in_master();
14:
15: /* Main Iteration */
16: for (i = 0; i < MAX_TIME_STEPS; i++) do
17:     if (process_id == MASTER) then
18:         master_rcv_activity(procs);
19:         ghost_iter_activity(procs);
20:         master_send_activity(procs);
21:     else // process_id == WORKER
22:         worker_activity(process_id);
23:
24: master_collects_distributed_values();
25:
26: MPI_Finalize();
27: return 0;
```

Figure 3. Main function of the algorithm.

Function `void master_rcv_activity(`*int procs*`)`

```
1:  for (j = 1; j < procs; j++) do
2:      do
            master_is_receiving_from_worker(j);
3:          if (rc_init == MPI_INIT_RESTARTED_NODE)
            then
4:              any_worker_re-spawned = 1;
        while (receiving is not SUCCESSFUL);
```

Figure 4. Master is receiving from workers.

```
Function void ghost_iter_activity(int procs)

1:  /* updating ghost values */
2:  for (j = 1; j < procs; j++) do
3:      if (any_worker_re-spawned == 1) then
4:          /* Load previous advection values
            */
5:          copy prev_advection_values(j) to advection_values(j);
6:      calculate_ghost_values_for_j(advection_values);
7:
8:      /* Saving advection values */
9:      save advection_values(j) to prev_advection_values(j);
10: if (any_worker_re-spawned == 1) then
11:     /* reset any_worker_re-spawned */
12:     any_worker_re-spawned = 0;
13:
14:     /* load previous time step */
15:     i - -;
```

Figure 5. Updating ghost values and saving/restoring advection values.

```
Function void master_send_activity(int procs)

1:  for (j = 1; j < procs; j++) do
2:      do
3:          master_is_sending_to_worker(j);
        while (sending is not SUCCESSFUL);
```

Figure 6. Master is sending to workers.

Fig. 4). Received values in the master may come from processes which are just rebuilt after failure. These values are invalid, because, upon process failure, FT-MPI destroys all MPI objects with non-local information (e.g., communicators and groups) including its current address space, except MPI_COMM_WORLD, requiring the application to manually recreate these objects after every failure in the same order [9]. So, a flag called *any_worker_re-spawned* is set in the algorithm shown in Fig. 4 on Line 4 to mark that the value in received buffer is invalid. Whether the value is received from restarted processes or not is checked on Line 3 in the algorithm shown in Fig. 4.

- Calculation of *left* and *right* ghost values for workers is done on Line 19 (details in the algorithm shown in Fig. 5). This calculation depends on the value of the flag *any_worker_re-spawned* stated in the algorithm shown in Fig. 4. If that flag is set, then we have to load the saved buffer values in previous iteration (done on Line 9 in the algorithm shown in Fig. 5) into the current buffer before calculating ghost values. Saving the values in current buffer and calculating the ghost values are done on Lines 5 and 6, respectively, in the algorithm shown in Fig. 5. The purpose of Lines 10–15 of the algorithm shown in Fig. 5 is to reset the flag *any_worker_re-spawned* and decrease the main iteration by one, if the flag was set before, so that the algorithm could continue for the correct number of iterations.
- The master is sending advection values including *left* and *right* ghost values as a message to each of the corresponding worker on Line 20 (details in the algorithm shown in Fig. 6) to update their ghost values and replace the buffer with the advection values if it is just rebuilt after failure.
- Sending advection values to master from each worker, receiving advection values including *left* and *right* ghost values from master to each worker, and calculating flux

```
Function void worker_activity(int process_id)

1:  do
2:      worker_is_sending_to_master(process_id);
    while (sending is not SUCCESSFUL);
3:  do
4:      worker_is_receiving_from_master(process_id);
    while (receiving is not SUCCESSFUL);
5:
6:  calculate_flux_and_update_advection_values(process_id);
```

Figure 7. Workers are sending to and receiving from master.

```
Function void recover(MPI_Comm *com, int *er)

1:  MPI_Comm oldcomm, newcomm;
2:  int rc;
3:  int size, rank;
4:  if (*er == MPI_ERR_OTHER) then
5:      oldcomm = MPI_COMM_WORLD;
6:      newcomm = FT_MPI_CHECK_RECOVER;
7:      /* collective recovery occurs here!
        */
8:      rc = MPI_Comm_dup (oldcomm, &newcomm);
9:      rc = MPI_Comm_rank (MPI_COMM_WORLD, &rank);
10:     rc = MPI_Comm_size (MPI_COMM_WORLD, &size);
11: else
12:     printf("ERR: Error occured with error code %d\n", *er);
13:     sleep(30);
```

Figure 8. Failure recovery function.

 ISBN: 978-1-61208-268-4

as well as updating advection values in each worker is done on Line 22 (details in the algorithm shown in Fig. 7).

- Finally, upon completion of main iteration, each worker sends their computer advection values to master so that master can combine these values to generate complete advection values.

A. Failure Detection

Any failure of processes or other errors in communication in FT-MPI is detected by error code `MPI_ERR_OTHER`. This error code is invoked inside a user-defined error handler function (Fig. 8) which is registered as an error handler in main function (Fig. 3) for detecting errors.

B. Failed Process Recovery

The next task after detecting process failure is to recover these failed processes to reconstruct communicator. So, it actually means recovering MPI environment including its communicator. This is done by substituting failed processes with the new ones by passing the FT-MPI attribute `FT_MPI_CHECK_RECOVER` to the collective function `MPI_Comm_dup` shown in the algorithm shown in Fig. 8.

C. Data Recovery Techniques

As we mentioned before, processes replacing failed processes require the application to manually recreate MPI objects other than MPI_COMM_WORLD and needs to initialize the variables in the new address space. There are two following scenarios of process failures on which data recovery technique depends.

- Sending from master is failed.
- Sending from worker is failed.

For the first case, master resends its current buffer to the worker waiting for re-receiving that after recovering from failure. The received data from master after the recovery is valid, because master's address space is not changed due to the worker's failure. The `Do···While` loops of Figs. 6 and 7 are used for resending and re-receiving the buffer. Data recovery techniques under this scenario is demonstrated in the algorithm shown in Fig. 9.

For the second case, on the other hand, worker resends its current buffer after recovering from failure to the master waiting for re-receiving the data of that buffer. The `Do···While` loops of Figs. 4 and 7 are used for re-receiving and resending the buffer. However, the received data from worker after the recovery is invalid to be used as advection values as well as ghost values, because the worker's address space is changed due to its failure as we mentioned before. As a result, a technique should be applied to replace this invalid data into valid one. The technique that we applied is saving the data (advection values) into another buffer in each time-step so that it can be used to replace that invalid

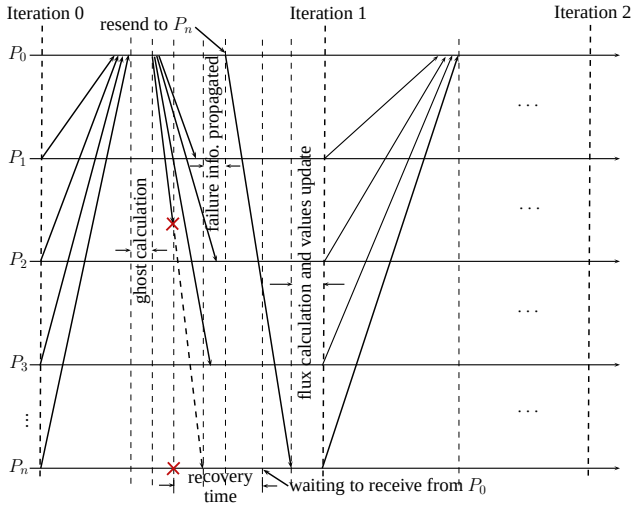

Figure 9. When sending from master is failed.

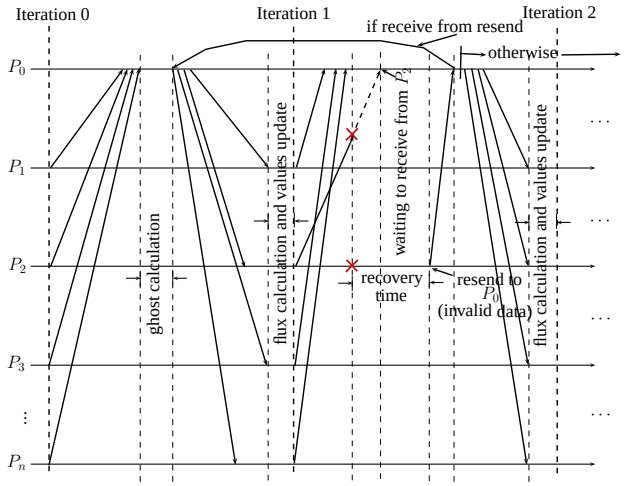

Figure 10. When sending from worker is failed.

buffer. This saving is done on Line 9 in the algorithm shown in Fig. 5. Later, under this scenario, the invalid current buffer is replaced with the saved buffer using Line 5 of the algorithm shown in Fig. 5. Data recovery techniques under this scenario are demonstrated in the algorithm shown in Fig. 10.

D. Time-Evolving Control

The next task after recovering the data for the restarted processes is to control the total number of updates on advection values of the workers. For the scenario of Fig. 10 described in previous section (Section IV-C), the number of such update is decreased by one for each such failure scenario due to invalid data. In order to keep the total number of updates in the presence of failure as the same as that of without failure, the iteration counter should be decreased by one for each such scenario. Lines 10–15 of the algorithm

 ISBN: 978-1-61208-268-4

shown in Fig. 5 is used for this purpose.

E. Duplicate Message Handling

Duplicate message handling in application development using FT-MPI is also an important issue. For the scenario of Figs. 9 and 10, a master or worker does not receive duplicate messages from each other. A re-receive is done only when the previous receive did not succeeded. So, no control is needed for duplicate message handling except `Do···While` loop for resending and re-receiving. However, we should control the receive and send operations on Lines 11 and 13, respectively, of the algorithm shown in Fig. 3 in case of re-spawned process initialization. These two operations are not needed for re-spawned processes. Although there is no problem for send operation in this case as multiple sending operations without corresponding receive operations do not complain, but we should strictly control receive operations. Otherwise, the application waits forever for receiving from master where master sends nothing. An attribute of FT-MPI called `MPI_INIT_RESTARTED_NODE` is used for the purpose of this control.

V. Experimental Result

Although FT-MPI is build with MPI-1 implementation, a significant amount of effort goes into making it competitive with other open source implementations by considering their execution time. In order to prove this issue, we perform an experiment for the non-fault-tolerant (Open MPI) and the Fault-Tolerant (FT-MPI) version of the algorithm applying for the problem discussed in Section IV. This experiment is done on a cluster with a standard GigE Switch with four nodes, each with AMD Phenom(tm) II X4 945 Quad-Core Processor with 3.0GHz of speed and 4.0GB of memory, having a total of 16 cores. The way of measuring the execution time is the `time` and `difftime` functions of `C++`. The result of this experiment is shown in Table I, which shows that FT-MPI is almost similar to Open MPI (version 1.4.5) in case of considering execution time.

Experiment on process failure recovery and recovery time is also conducted for the same problem and on the same cluster, where process failure is simulated by killing the process(es) by issuing the `kill` command and time is measured by the `time` and `difftime` functions of `C++` as before. The experimental result which is performed on a grid with 120 points and 300 time steps is shown in Table II. This result shows that this algorithm could recover from any number of worker process failures. Moreover, the recovery time of process failure is minimum and acceptable.

TABLE I. Execution time of non-fault-tolerant version of algorithm in Open MPI and FT-MPI.

# Grid Points	# Time Steps	Open MPI (Sec)	FT-MPI (Sec)
15360	38400	41	61
30720	76800	173	204
46080	115200	382	441

TABLE II. Experiment on process failure recovery and recovery time.

Total Process Failed	List of Killed Processes	Failure Recovered?	Recovery Time (Sec)
1	Any 1 of the 15 worker processes	YES	1
2	Any 2 of the 15 worker processes	YES	1
3	Any 3 of the 15 worker processes	YES	2
4	Any 4 of the 15 worker processes	YES	2
5	Any 5 of the 15 worker processes	YES	2
6	Any 6 of the 15 worker processes	YES	3
7	Any 7 of the 15 worker processes	YES	3
8	Any 8 of the 15 worker processes	YES	3
9	Any 9 of the 15 worker processes	YES	3
10	Any 10 of the 15 worker processes	YES	4
11	Any 11 of the 15 worker processes	YES	4
12	Any 12 of the 15 worker processes	YES	4
13	Any 13 of the 15 worker processes	YES	5
14	Any 14 of the 15 worker processes	YES	5
15	All worker processes	YES	5

VI. Conclusion

This paper proposes a master-worker model for designing and implementing a fault-tolerant algorithm applicable for time-evolving problems. One of the emerging problems in such category is the solution of advection equations which are modeled by partial differential equations. We have applied this model on the iterative solution of one dimensional advection equations so that it can survive the failure of all the worker processes in that system. We have used the semantics of FT-MPI to implement this algorithm focusing on different issues related to fault tolerance like failure detection, failed process recovery, data recovery techniques, time-evolving control, duplicate message handling, etc. This model is not scalable, but it can recover the failure of all workers in the system and there are scopes to modify this model to make it scalable. This contribution will also help application developers to resolve different issues of design and implementation of fault-tolerant algorithms for more complex time-evolving applications.

We are currently working on modifying this model so that it turns into a scalable solution. One of the scopes include using an extra process for each working process replacing single master process so that each working process can

 ISBN: 978-1-61208-268-4

communicate with another working process directly and save their data to the corresponding extra processes. The purpose of this saving is that the extra processes can send their saved data to the corresponding processes when they re-spawned after the failure. Another approach avoids requiring a master process and saves the data on their *left* and *right* neighbors during communication. This saved data can be send to its neighbors when they are re-spawned after the failure. There are also many approaches like this which can be proposed to make this fault-tolerant application scalable.

ACKNOWLEDGMENT

This work was supported by the Australian Research Council (ARC) and Fujitsu Laboratories of Europe (FLE) through the ARC National Competitive Grants Program (NCGP) Linkage Project LP110200410.

REFERENCES

[1] Y. Ajima, S. Sumimoto, and T. Shimizu, "Tofu: A 6d mesh/torus interconnect for exascale computers," Computer, vol. 42, no. 11, November 2009, pp. 36–40.

[2] L. D. Fosdick, E. R. Jessup, C. J. C. Schauble, and G. Domik, An Introduction to High-Performance Scientific Computing, ser. Scientific and Engineering Computation. MIT Press, 1996.

[3] B. Schroeder and G. A. Gibson, "A large-scale study of failures in high-performance computing systems," in Proc. International Conference on Dependable Systems and Networks, ser. DSN '06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 249–258.

[4] A. Geist and C. Engelmann, "Development of naturally fault tolerant algorithms for computing on 100,000 processors," 2002. [Retrieved: December 10, 2012], URL: http://www.csm.ornl.gov/~geist/Lyon2002-geist.pdf

[5] G. Gibson, B. Schroeder, and J. Digney, "Failure tolerance in petascale computers," Software Enabling Technologies for Petascale Science, vol. 3, no. 4, November 2007, pp. 4–10.

[6] Message Passing Interface Forum, "MPI: A message passing interface," in Proc. Supercomputing. IEEE Computer Society Press, November 1993, pp. 878–883.

[7] G. E. Fagg and J. J. Dongarra, "FT-MPI: Fault tolerant mpi, supporting dynamic applications in a dynamic world," 2000.

[8] W. Gropp and E. Lusk, "Fault tolerance in mpi programs," Special issue of the International Journal High Performance Computing Applications (IJHPCA), vol. 18, 2002, pp. 363–372.

[9] J. Hursey and R. Graham, "Building a fault tolerant mpi application: A ring communication example," in IEEE International Symposium on Parallel and Distributed Processing Workshops and PhD Forum (IPDPSW), May 2011, pp. 1549–1556.

[10] Fault Tolerance Working Group, "Run-through stabilization interfaces and semantics." [Retrieved: December 10, 2012], URL: svn.mpi-forum.org/trac/mpi-forum-web/wiki/ft/run_through_stabilization

[11] G. Bosilca, R. Delmas, J. Dongarra, and J. Langou, "Algorithm-based fault tolerance applied to high performance computing," Journal of Parallel and Distributed Computing, vol. 69, no. 4, 2009, pp. 410–416.

[12] K.-H. Huang and J. A. Abraham, "Algorithm-based fault tolerance for matrix operations," IEEE Trans. Comput., vol. 33, no. 6, June 1984, pp. 518–528.

[13] P. Du, A. Bouteiller, G. Bosilca, T. Herault, and J. Dongarra, "Algorithm-based fault tolerance for dense matrix factorizations," SIGPLAN Not., vol. 47, no. 8, February 2012, pp. 225–234.

[14] Z. Chen, G. E. Fagg, E. Gabriel, J. Langou, T. Angskun, G. Bosilca, and J. Dongarra, "Fault tolerant high performance computing by a coding approach," in Proc. tenth ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming, ser. PPoPP '05. New York, NY, USA: ACM, 2005, pp. 213–223.

[15] J. Langou, Z. Chen, G. Bosilca, and J. Dongarra, "Recovery patterns for iterative methods in a parallel unstable environment," SIAM J. Sci. Comput., vol. 30, no. 1, November 2007, pp. 102–116.

[16] Z. Chen, "Algorithm-based recovery for iterative methods without checkpointing," in Proc. 20th International Symposium on High Performance Distributed Computing, ser. HPDC '11. New York, NY, USA: ACM, 2011, pp. 73–84.

[17] G. E. Fagg and J. Dongarra, "Building and using a fault-tolerant mpi implementation," vol. 18, no. 3, 2004, pp. 353–361.

 ISBN: 978-1-61208-268-4