A pattern Collection to Support the Practical Challenges Involved in Adopting International Standards for Safety-Related Embedded Systems

Farah Lakhani, Nabiha Faisal, and Michael J. Pont

Abstract—**With the introduction of standards such as ISO 26262, developers of many embedded systems are going through a process of "design migration", in order to improve system reliability and – thereby – meet certification requirements. Such a migration process can present many challenges for an organization, not least because long- established (and possibly rather informal) working practices can be seen to be under threat. The research presented in this paper investigates the challenges faced by practitioners during the migration process and explores ways in which design patterns can be used to support the transition between informal "event triggered" system architectures and more predictable time-triggered alternatives. The results obtained suggest that migration to time-triggered architecture using a pattern collection may indeed help to follow the latest industry standards and improve system reliability.**

Index Terms—**Design patterns, event-triggered, migration, reliability, time-triggered.**

I. Introduction

In the literature, many examples can be found in which designers faced the challenge of making changes to the software architecture of real-time embedded applications in order to improve system performance. Two such example systems are discussed in detail in [1] and [2]. Some of the issues that drive the migration of embedded applications are discussed in [3] and [4], though the ultimate goal of achieving reliability in systems is usually paramount. High reliability is the most important criterion for high risk embedded systems operating in unpredictable environments such as military applications (e.g. fly-by-wire aircraft and drive-by-wire passenger cars) [5]. However, even in more mundane domestic applications (e.g. an alarm clock that fails to sound on time or a TV recorder that operates intermittently), poor reliability can have other impacts such as reduced sales [6].

Two key software architectures used for designing modern embedded applications are event-triggered (ET) and time-triggered (TT). In either case, the system is designed as a collection of tasks; however, the method of invocation of tasks is different. If the tasks are invoked as a result of events with unknown arrival times, the system may be described as 'event triggered' [7]. In implementation terms, ET systems are characterized by the use of multiple interrupts. By contrast, TT architectures are based on a "one interrupt per CPU" rule. That is, TT designs have only a single interrupt enabled (per CPU) in the system [8]. This single interrupt is usually linked to a hardware timer which can be configured to overflow periodically (for example every millisecond) or as required. The occurrence of this single periodic interrupt acts as a "time-stamp" for the system and is used to control all the tasks.

ET systems are considered to be flexible and easy to build. However, such designs do not scale easily and – for non-trivial designs – can be very difficult to test fully. By contrast, TT designs can be more difficult to build, but much easier to test. For larger systems and / or safety-related or safety-critical applications, TT designs are generally preferred.

In aerospace and defense sectors (for example), TT designs are commonplace. In other sectors – for example, the automotive sector – TT designs are much less common and ET approaches are the norm. With the introduction of standards such as ISO 26262 in the automotive sector, developers are being forced to re-assess existing designs and begin a process of "design migration", in order to improve system reliability and – thereby – meet certification requirements. Such a migration process can present many challenges for an organization, not least because long-established (and possibly rather informal) working practices can be seen to be under threat.

In this paper, we explore ways in which design patterns may be able to assist in the process of design migration between ET and TT architectures.

The concept of design patterns first emerged from the work of an architect, Christopher Alexander, during the 1960s and 1970s. Alexander and his colleagues published three pioneering texts [9]-[11] between 1975 and 1979 that laid the foundation of use of patterns in the field of architecture. He defines a pattern as "a three part rule which expresses a relation between a certain context, a problem and a solution". As patterns contain time-tested solutions to commonly recurring problems in a domain, the general nature of this concept makes design patterns a useful tool beyond the architecture.

In the field of embedded systems, most previous work with design patterns has focused on the process of system construction (see for example [12]-[17]). In this research we have identified the lack of a standard approach in the use of design patterns in the process of migrating between different software architectures of embedded applications. This research is therefore aimed to provide support to the practitioners in this field by introducing a new pattern collection to assist in the migration process from ET to TT designs. Our particular concern is to explore ways in which we may be able to help developers of embedded systems to improve system reliability by migrating between ET and

Manuscript received February 22, 2015; revised February 8, 2016.

Nabiha Faisal is with Bahria University Karachi Campus, Pakistan (e-mail: nabiha_faisal@bimcs.edu.pk).

equivalent (and more effective) TT architectures. The research introduces a new pattern language for system migration, and to illustrate the efficacy of this approach the paper presents case studies based on two non-trivial embedded applications.

Following the introduction, this paper is organized as follows: Section II describes the current trends and standards which the industry is moving towards to improve the reliability of applications and the challenges faced by the practitioners to adopt these standards. Section III will give a quick introduction to the contributions made by this research and a brief introduction of the patterns in the language. Section IV presents a discussion on the usability of pattern language how it can help the developers of embedded applications to face the challenges imposed by current trends and standards. Finally, Section V concludes the paper.

II. Industry and the Challenge of Migration

A. Current Trends and Standards

To maintain a benchmark for the functional safety of devices the International Electro-technical Commission (IEC) has introduced a set of standards for the general market called IEC 61508. Because IEC 61508 is very general and as the state of the industry cannot be reflected in a general standard, the specific standards are described for different types of applications such as IEC 60730 for white goods, ISO 26262 for automotive industry and DO-178 for aerospace. By the use of these standards both systematic and random failures can be reduced and managed to ensure system reliability and functional safety. For example for ensuring functional safety of road vehicles, ISO 26262 has reserved a separate part (Part 6) which is further divided into various sections on guidelines for software architectural design. The standard does not favor or oppose any specific software architecture but has defined guidelines to follow to get safe and reliable systems. Some of the principles defined for the software architectural design are: hierarchical structure of software components, restricted size of software components, restricted coupling between components, appropriate scheduling properties and restricted use of interrupts. Some mechanisms are defined for error detection and error handling at the software architectural level. For example the recommended error detection mechanisms are: detection of data errors, external monitoring facility and control flow monitoring. Mechanisms defined for error handling are: static recovery mechanism and graceful degradation. There is strong emphasis on testing and verification of the software architectural design and methods recommended are informal verification by walkthrough or inspection of the design, semi-formal verification by simulating dynamic parts of the design or by the proto type generation or animation, formal verification which involves rigorous mathematical models, control flow and data flow analysis for the system.

In order to achieve systems which are safety complaint and certified by these standards the competitors in the automotive industry are inclined towards adhering to ISO 26262 guidelines. As there is no single architecture which is recommended or rejected by ISO 26262 it is still to be decided by the software designers and developers which architecture has the ability to follow the guidelines. The underlying goal of the research presented in this paper is to investigate what are the real challenges which are faced by the practitioners and to provide support accordingly.

B. Challenges Faced by the Real Stakeholders

One of the major objectives of the research presented in this paper is to investigate the challenges faced by the practitioners working on safety-critical applications. There are various ways to conduct such an investigation for example live interviews, online discussion forums and blogs or web-based surveys. In the context of this research a real challenge was to collect maximum responses from highly skilled and sophisticated personnel in the field of embedded application development around the world. Mindful of the practical constraints of this aspect of the research it was decided to go for a web-based survey methodology to obtain the required information. Web-based surveys have the potential for a global audience and are more inclusive allowing a broader reach than phone or postal survey or direct interviews [18]. Once setup, web-based surveys are easy to carry out, making it easier to recruit large number of participants or to collect data repeatedly. Since the data is captured directly in electronic format, it also makes data analysis faster and cheaper than other investigation methods.

Finally a short questionnaire was designed and an open-ended question was included in the questionnaire to obtain the feedback of respondents on how the industry is currently coping with the problem of migration between different software architectures and what the challenges are around it. In order to approach the targeted audience different social networks were used. To achieve more responses a method of personalized emails was adopted to reach those people who are not very active on public forums. The e-mail addresses were obtained from contacts in the research group. An initial target was set to achieve at least 25 respondents in total to be able to draw some useful results.

Analyzing results from open-ended questions is more challenging and requires techniques such as ‘text analysis’ to interpret results. Text analysis is a research technique for making replicable and valid inferences from texts to the context of their use [19]. This involves establishing categories and then counting the number of instances when those categories are used in a particular item of text [20]. This is usually achieved through creating nodes and coding to find some sort of order and coherence within the dataset and to see how data relates to the research question. Coding is a way of classifying data so that it can be reviewed by category as well as source. A node within a coding scheme is a representation of an idea, theme or category in data. Segments of data from across the dataset are coded to these nodes. This enables to retrieve all the data related to a node. Over a period of two months twenty-eight people responded to the questionnaire and provided useful feedback on the challenges of software architecture migration in the industry. The option of leaving the contact details for respondents at the end of the questionnaire helped to identify some of the organizations from which the participants were associated. Useful feedback responses were received from professionals from organizations based in Australia, India, Malaysia, Miami, Pakistan, Singapore, Switzerland, Tunisia, UK and USA.

The comments received by most of the respondents are useful and self-explanatory however it was important to apply some text analysis techniques to conclude results from the textual data. Text analysis is a research technique for making replicable and valid inferences from texts to the context of their use [19]. This involves establishing categories and then counting the number of instances when those categories are used in a particular item of text [20]. This is usually achieved through creating nodes and coding to find some sort of order and coherence within the dataset and to see how data relates to the research question. Coding is a way of classifying data so that it can be reviewed by category as well as source. A node within a coding scheme is a representation of an idea, theme or category in data. Segments of data from across the dataset are coded to these nodes. This enables to retrieve all the data related to a node.

In this regard the QSR NVIVO software tool is used for text analysis and original comments are distributed into different nodes. Related to the research question and analyzing the respondents' comments the main nodes are identified as:

Unawareness
Time to market
Legacy systems
Lack of support
Lack of experience
Industry standards
Cost
Agile methods
Adapting architectures

Responses from the practitioners indicated that currently people in the industry are aware of the fact that migration between different software architectures is a huge challenge but at present there is no sophisticated way adopted to cope with this challenge.

With the help of NVIVO tool a graphical analysis of the text data is achieved and the percentage covered by each node is given in Fig. 1.

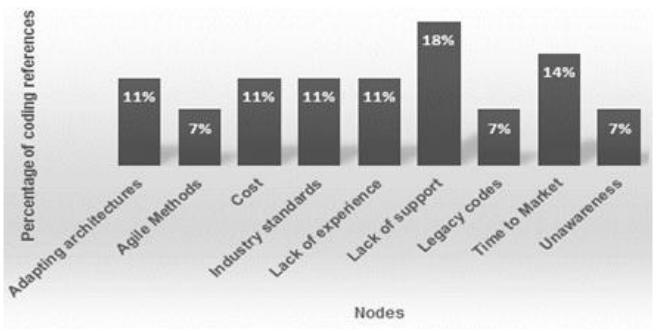

Fig. 1. Illustrating the number of coding references covered for each node.

This coding analysis of text data obtained through respondent's comments has revealed some interesting results. As emphasized by the respondents shown in Fig. 1 'Lack of support' appeared as the most important issue during the migration process. Further this lack of support is more towards 'Lack of documentation' and the 'tool support'. The other areas where practitioners feel that industry lack support are towards 'Methodology' which should be adopted while migrating architectures, 'Automation' that must be required during the various stages and support for 'Multi-core and parallel hardware' designs.

The other main challenge indicated is the 'Time to Market'. 'Time to Market' is one of the most important design metrics and is considered as the time required to develop a system to the point that it can be released and sold to customers [21]. In this sense it is directly related to business profit and reputation of the organization. As one of the respondent has mentioned that migration across architectures may increase 'time to market' and managers at higher level always think in terms of gaining more profit for the organization. Overall it is concluded that people in industry are reluctant to any change in existing architectures unless required by the customer as migration might not be a favorable decision in terms of organization's repute and business profit.

As 'Time to Market' is directly related to 'Cost' so the next important concern shown by the practitioners is the involvement of 'Cost' for industrial projects which are required to pass through the process of migration. The cost might be in form of outsourcing manpower or tools required during the process and therefore avoided unless a reasonable profit return is envisioned by a team of experts in the organization. Few respondents mentioned about following agile approaches to software development. This truly makes sense as agile practices bring a number of business benefits as better project adaptability and reaction to changes, reduced production cost and better performance [22].

III. Research Contribution

This research presents a new pattern language named 'PMES' (Patterns for Migration of Embedded Systems). The map of the language (in the current state) is shown in Fig. 2.

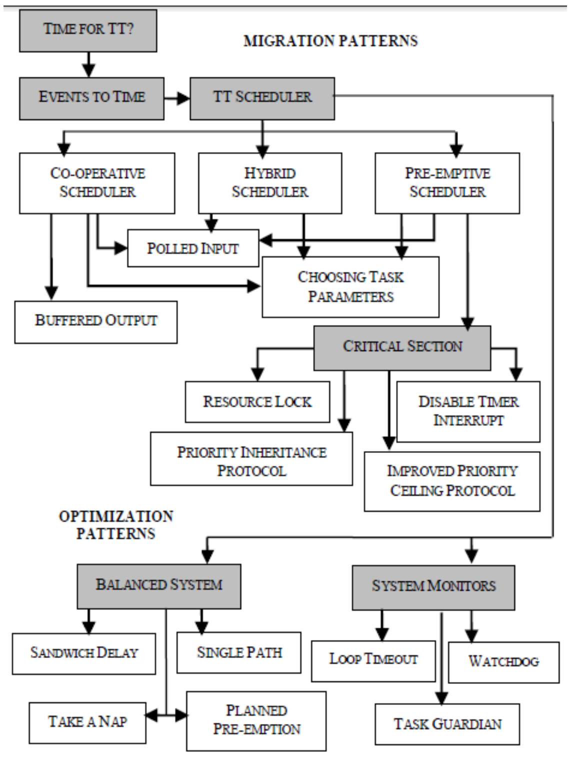

Fig. 2. Map for the PMES collection.

In the present state the pattern language contains 23 patterns and represents the extension of the work we have previously introduced in [23]. At the current stage, the PMES collection is mainly divided into two major partitions ‘Patterns for migration’ and ‘Patterns for optimization’.

Collectively the underlying goals of the collection are:

- To provide sufficient understanding of the challenges involved in migrating from an ET to a TT design and in which situations either of the two designs is appropriate.
- To help the practitioners in choosing a suitable TT architecture for migration.
- To assist in handling with the problems after migrating to a TT design such as long task problem in co-operative design and shared resource access in pre-emptive designs.
- To provide techniques for further optimizing the TT design.

Both migration and optimization patterns are structured in a hierarchy of abstract patterns and patterns. This division of patterns in a pattern language has been introduced previously in [24]. An abstract pattern is meant to identify a class of design problems for which (more than one) design pattern is available in the collection. For example in Fig. 1, the TT SCHEDULER is an abstract pattern connected to three patterns that implies there are three possible designs, CO-OPERATIVE, HYBRID and PRE-EMPTIVE to have a TT scheduler in the system. Abstract patterns do not directly tell the user how to construct a piece of hardware or software; instead, they assist a developer in deciding whether or not the use of a particular design solution would be an appropriate way of solving a particular design challenge. The information contained in the design pattern outlines the solution that is required to be implemented once the major architectural issues (discussed in the abstract pattern) surrounding the design problem are solved. For example the pattern TIME FOR TT discusses the architectural issues with both ET and TT designs and help developers to decide whether migration to the TT architecture is an appropriate choice for their application. Pattern EVENTS TO TIME is documented with the aim to help in determining what architectural changes in the ET design will be required and the pattern TT SCHEDULER discusses possible TT solutions. Patterns CO-OPERATIVE SCHEDULER, HYBRID SCHEDULER [14] and PRE-EMPTIVE SCHEDULER [25] are aimed to provide concrete solutions to each of the possible TT design.

The collection is also designed to guide the developers through the most important issues that need to be handled with TT designs. The example patternsare BUFFERED OUTPUT (to handle with long task problem in the co-operative designs), POLLED INPUT (handling external inputs in all TT designs) and CHOOSING TASK PARAMETERS (to decide about task order and task offset values). Patterns which help to avoid conflicts over shared resources that employ a pre-emptive scheduler include the pattern CRITICAL SECTION, RESOURCE LOCK, DISABLE TIMER INTERRUPT, PRIORITY INHERITANCE PROTOCOL and IMPROVED PRIORITY CEILING PROTOCOL. The resultant time-triggered designs can be further optimized by reducing jitter values using techniques such as those discussed in the patterns BALANCED SYSTEM, SANDWICH DELAY, SINGLE PATH DELAY, TAKE A NAP and PLANNED PRE-EMPTION, and finally by applying system monitoring techniques described in the pattern SYSTEM MONITORS, LOOP TIMEOUT, WATCHDOG and TASK GUARDIAN.

Please note that this collection has evolved in stages and most of the individual patterns have been published previously in [14] and [26]–[28]. A complete collection altogether is formed by linking the individual patterns together and is described in detail in [29].

IV. PMES Usability in the Industrial Context

Section II mentioned that for the guaranteed safety of high integrity systems the industry is backed up by international safety standards incorporated in IEC 61508 which is a benchmark standard for developing and validating safety-related electronic systems. The standard is entitled: “Functional safety of Electrical/Electronic/Programmable Electronic Safety related Systems (E/E/PE)” (IEC, 2012) [30]. The rules defined in IEC 61508 are quite general and are intended to define a basic functional safety standard applicable to all kinds of industry. It is also mentioned that the automotive industry has adapted IEC 61508 to produce ISO 26262, a new benchmark standard for developing and validating safety-related systems that are installed in passenger cars. The vehicle production industry is now looking forward to design applications for modern passenger cars which are ISO 26262 compliant. The aerospace industry has adapted the IEC 61508 and introduced the DO-178 standard for the development of software for aircrafts.

The research aimed to explore ways of assisting developers and organizations in situations where systems need to meet certification requirements. The particular goal was to avoid “reinventing the wheel” by helping people adapt their existing designs to make them suitable for use in systems with what IEC 61508 referred to as “Safety Integrity Level” or SIL. The following sections will describe how the PMES collection can help to achieve this aim.

A. *Predictability in Systems*

The main reason to introduce predictability in systems is to make them safe and reliable. According to the documentation for IEC 61508 Edition 2, the definition of Safety is, ‘Freedom from unacceptable risk’. One possible way of avoiding risks is to design/configure a software system such that it exhibits predictable behaviour and allows one to determine in advance – before the system begins executing – exactly what it will do at every moment of time during which it is running. In other words, the architecture must allow testing of the system in a convenient way to guarantee the safety and reliability of the system. In the proposed PMES collection patterns TIME FOR TT and TT SCHEDULER has discussed how one can make their system safe and reliable and when and what TT architectures can perfectly match an application requirements.

For example the safety process for automotive vehicles is described in various parts in the documentation for ISO 26262 structure. Part 6 of the documentation specifically refers to the development of the software aspect of the product and section 6-7 of the standard is “Restricted use of interrupts" to guarantee the fault free and safer systems. In the PMES collection the pattern EVENTS TO TIME [31] has described ways of transforming systems with restricted use of interrupts.

B. System Requirements for Safety-Integrity

During a system design once the system architecture is specified and the design begins the functional safety requirements begin to be refined into specific design safety requirements. In order to achieve functional safety, one of the main emphases of IEC 61508 and ISO 26262 is on risk assessment to determine the steps necessary to reduce the risk of each hazard to an acceptable level, usually through 'safety integrity' for an electronic system. IEC 61508 has defined Safety Integrity Levels 'SIL' to relate to the probability of a dangerous failure. ISO 26262 has adapted these levels for the automotive industry as 'ASIL' or Automotive Safety Integrity Levels; however these are not defined as a probabilistic requirement.

In real practice the software safety requirements are particularly related to the determination of Worst Case Execution Time (WCET) of tasks designed for the system. The accurate WCET prediction matters as the variation in task timings could potentially lead to system failure. In the PMES collection pattern BALANCED SYSTEM and its associated patterns this issue has been discussed in detail and solutions provided to possible hazards that can introduce variations in task timings.

C. System Monitoring

It is almost impossible to design a systems which is 100% fault-free even if designed with due care. At runtime systems are susceptible to fault occurrences which in the worst case scenario could lead to a loss of precious human lives and property. This places a responsibility on designers to avoid any such fault occurrences both at the initial design stage and while systems are in a running state. In the documentation for ISO 26262, Part 6 Annex D is specifically focused on software elements' freedom from interference. These sources of interference could be due to hardware problems, or software mismanagement such as task overruns. In the optimization patterns of the PMES collection pattern SYSTEM MONITOR and the associated patterns discusses the possible run time error issues and how to cater to them in advance.

We believe that appropriate use of the proposed patterns may help in facing the challenges in adapting architectures in order to follow the latest industry standards, reduce the cost and time to market and would provide support in taking the right decisions during the process of architecture modifications.

V. Conclusion

In this paper, we have described investigations conducted in order to find the practical challenges faced by the developers of embedded applications in order to follow the latest industry standards. The investigations helped to conclude where the gap exist and what are the real challenges needed to be mitigated. With the intention to fill this gap we have introduced a new and substantial pattern language aimed to help developers in migrating embedded applications from event-triggered design to time-triggered design. The usability of the patterns is then described in the industrial context and how they can support the practitioners in following the latest standards. We are aiming to expand this collection to address more issues involved in the migration process. In future, we are also aiming to provide a pattern- oriented tool support for the migration between different software architectures.

References

[1] T. Shepard and M. Gagne, "A model of the F18 mission computer software for pre-run-time scheduling," in *Proc. ICDCS '90,* May 1990.

[2] J. Turley, "Gaming the system—high end networking on the cell processor," 2009 .

[3] D. Mosley, "When to migrate legacy embedded applications," in *Proc. SIGAda'06* , Albuquerque, New Mexico, USA, 2006.

[4] O. N. Oest, "Migrating complex embedded systems," *Military embedded Systems*, June 2008.

[5] D. Ayavoo and M. J. Pont, "A hardware in the loop testbed representing the operation of a cruise control system in a passenger car," in *Proc. the 2nd UK Embedded Forum*, Birmingham, UK, 2005.

[6] J. Xu and D. Parnas, "Priority scheduling versus pre run-time scheduling," *The International Journal of Time-Critical Computing Systems*, 2000.

[7] H. Kopetz, *Real-Time Systems Design Principles for Distributed Embedded Applications*, Kluwer Academic Publishers, 1997.

[8] H. Kopetz, "Event-triggered versus time triggered real-time systems," in *Proc. the Workshop on Operating Systems of the 90's and Beyond*, 1991.

[9] C. Alexander, M. Silverstein, and S. Angel, *The Oregon Experiment*, Oxford University, 1975.

[10] C. Alexander, S. Ishikawa, and M. Silverstain, *A Pattern Language*, Oxford University Press, 1977.

[11] C. Alexander, *The Timeless Way of Building*, Oxford University Press, 1979.

[12] M. Adams and J. Coplien, "Fault-tolerant telecommunication system patterns: pattern languages design 2," Boston, USA, 1996.

[13] M. Bottomley, "A pattern language for simple embedded systems," in *Proc. Pattern Languages of Programming(PLoP99)*, Chicago, USA, 1999.

[14] M. J. Pont, *Patterns for Time-Triggered Embedded Systems*, ACM Press, 2001.

[15] W. Herzner, W. Kubinger, and M. Gruber, "Triple-T (time-triggered transmission) a system of patterns for reliable communication in hard real-time systems," in *Proc. 5th European Conference on Pattern Languages of Programming*, Irsee, Germany, 2005.

[16] R. J. Cloutier and D. Verma, "Applying the concepts of patterns to systems architecture," *Journal of Systems Engineering*, vol. 10, no. 2, pp. 138-154, 2007.

[17] V. P. Eloranta and J. Koski, "Software architecture patterns for distributed embedded control systems," in *Proc. 14th European Conference on pattern languages of Programming*, Irsee, Germany, 2009.

[18] T. M. Archer, "Web based survey," *The Journal of Extension*, vol. 41, no. 4, August 2003.

[19] K. Krippendorff, *Content Analysis an Introduction to Its Methodology,* Sage Publications, 2004.

[20] D. Silverman, *Interpreting Qualitative Data*, *Sage Publications*, 2011.

[21] F. Vahid and T. Givargis, *Embedded System Design: A Unified Hardware/Software Introduction*, John Wiley & Sons, 2002.

[22] T. Bozheva and M. E. Gallo, "Framework of Agile patterns," in *Proc. the 5th Workshop from Code Centric to Model Centric*, Paris, France, 2010.

[23] S. Kurian and M. J. Pont, "Building reliable embedded systems using abstract patterns, patterns, and pattern implementation examples," in *Proc. the 2nd UK Embedded Forum*, Birmingham, UK, 2005.

[24] S. Kurian and M. J. Pont, "Re-structuring a pattern language which supports time-triggered co-operative software architectures in resource-contrained embedded systems," in *Proc. 11th European Conference on Pattern Languages of Program,* Irsee, Germany, 2006.

[25] A. Maaita, "Techniques for enhancing the temporal predictability of real-time embedded systems employing a time-triggered software architecture," PhD Thesis, University of Leicester, 2008.

[26] F. Lakhani, A. Das, and M. J. Pont, "Towards a pattern language which supports migration of systems from event-triggered pre- emptive to time-triggered co-operative software architectures," in *Proc. 14th European Conference on Pattern Languages of Program*, Irsee, Germany, 2009.

[27] F. Lakhani, A. Das, and M. J. Pont, "Creating systems with predictable patterns of behaviour: migrating systems from event- triggered to time-triggered software architectures," in *Proc. 15th European Conference on Pattern Languages of Program*, Irsee, Germany, 2010.

[28] H. Wang, M. J. Pont, and S. Kurian, "Patterns which help to avoid conflicts over shared resources in time-triggered embedded systems which employ a pre-emptive scheduler," in *Proc. 12th European Conference on Pattern Languages of Program*, Irsee, Germany, 2007.

[29] F. Lakhani, H. Wang, and M. J. Pont, "Support the migration betrween event-triggered and time-triggered software architectures: A small pattern collection intended for use by the developers of reliable embedded systems," Technical Report ESRG 2011-09-01.

[30] International Electrotechnical Commision IEC, *Functional Safety and IEC 61508,* August 2012.

[31] F. Lakhani and M. J. Pont, "Using Design patterns to support between different system architectures," in *Proc. 5th IEEE International Conference on Systems of Systems Engineering*, Loughborough, UK, 2010.

Farah Lakhani received her bachelor's and master 's degrees in computer engineering from NED University Karachi, Pakistan in 1999 and 2004 respectively. Her PhD in embedded systems engineering from University of Leicester UK in 2013.

She is currently working as an Assistant Professor in the Computer & Software Engineering Department at Bahria University Karachi Pakistan. Priorly worked as a support worker at Accessibility Center and Lab Demonstrator at University of Leicester for 2 years, software trainer at TTE Systems for 3 years. She has published around 10 publications in various international conferences and journals.

Dr. Lakhani's area of research interests are software architectures for modern embedded applications, reliability issues for safety-critical applications and software engineering paradigms.

Nabiha Faisal received her B.E degree in computer & information systems engineering from NED University of Engineering & Technology Karachi, Pakistan in 2004 and her MEngg degree in computer architecture & system design from NED University of Engineering & Technology Karachi, Pakistan in 2007.

She is currently working as an Assistant Professor in the Computer & Software Engineering department at Bahria University Karachi, Pakistan. Priorly worked as a lecturer at Muhammad Ali Jinnah University Karachi for 2 years. She has published a book Computer Graphics-A Practical Approach for Beginner's Karachi, Pakistan, Lambert Academic Publishing, 2012.

Ms. Faisal's research interests include embedded systems, artificial intelligence and computer graphics.

Michael J. Pont did his BSc in engineering from University of Glasgow UK and his PhD in computer science from University of Southampton UK.

He is currently the CEO at SafeTTy Systems Ltd for the past 1 year. Previously worked as a lecturer at University of Sheffield for 2 years and at University of Leicester for 8 years. He was the head of Embedded Systems Laboratory at University of Leicester for 13 years. He was a CEO at TTE Systems Ltd for 6 years and Professor at University of Leicester for 4 years. He has around 21 publications to his credit in various international Conferences and journals.

Dr. Pont has his significant interest in the fields of embedded systems, software engineering, embedded C and systems engineering.