FARE: A Framework for Benchmarking Reliability of Cyber-Physical Systems

Leon Wu Gail Kaiser
Department of Computer Science
Columbia University
New York, NY 10027, USA
{leon,kaiser}@cs.columbia.edu

***Abstract*—A cyber-physical system (CPS) is a system featuring a tight combination of, and coordination between, the system's computational and physical elements. System reliability is a critical requirement of cyber-physical systems. An unreliable CPS often leads to system malfunctions, service disruptions, financial losses and even human life. Improving CPS reliability requires an objective measurement, estimation and comparison of the CPS system reliability. This paper describes FARE (Failure Analysis and Reliability Estimation), a framework for benchmarking reliability of cyber-physical systems. Some prior researches have proposed reliability benchmark for some specific CPS such as wind power plant and wireless sensor networks. There were also some prior researches on the components of CPS such as software and some specific hardware. But according to the best of our knowledge, there isn't any reliability benchmark framework for CPS in general. FARE framework provides a CPS reliability model, a set of methods and metrics on the evaluation environment selection, failure analysis and reliability estimation for benchmarking CPS reliability. It not only provides a retrospect evaluation and estimation of the CPS system reliability using the past data, but also provides a mechanism for continuous monitoring and evaluation of CPS reliability for runtime enhancement. The framework is extensible for accommodating new reliability measurement techniques and metrics. It is also generic and applicable to a wide range of CPS applications. For empirical study, we applied the FARE framework on a smart building management system for a large commercial building in New York City. Our experiments showed that FARE is easy to implement, accurate for comparison and can be used for building useful industry benchmarks and standards after accumulating enough data.**

***Index Terms*—cyber-physical system, reliability, failure analysis, software reliability, reliability estimation, machine learning, data mining, statistical analysis**

I. Introduction

A cyber-physical system (CPS) is a system featuring a tight combination of, and coordination between, the system's computational and physical elements [1]. Typical applications of CPS include sensor-based systems and intelligent control systems. Sensor-based systems such as smart building management systems and wireless sensor networks utilize many distributed sensors to measure and collect system or environmental data and transmit these information to a centralized system for processing. Intelligent control systems include smart grid operation control systems, autonomous automotive systems, medical monitoring, process control systems, distributed robotics, and automatic pilot avionics.

System reliability is a critical requirement of cyber-physical systems[2]. An unreliable CPS often leads to system malfunctions, service disruptions, financial losses and even human life [3]. Improving CPS reliability requires an objective measurement, estimation and comparison of the CPS system reliability. Some prior researches have proposed reliability benchmark for some specific CPS such as wind power plant and wireless sensor networks. There were also some prior researches on the components of CPS such as software and some specific hardware. But there isn't any reliability benchmark framework for CPS in general, according to the best of our knowledge. This paper describes FARE (Failure Analysis and Reliability Estimation), a framework for benchmarking reliability of cyber-physical systems. The FARE framework provides a set of methods and metrics on failure analysis, data quality measurement and monitoring, operational availability measurement and reliability estimation for benchmarking CPS reliability.

The advantages of FARE framework include a more general and accurate representation of the CPS reliability; additional reliability metrics; CPS-specific holistic system reliability; emphasis of actual use and continual evaluation. The framework is extensible for accommodating new reliability measurement techniques and metrics. It not only provides a retrospect evaluation and estimation of the CPS system reliability using the past data, but also provides a mechanism for continuous monitoring and evaluation of CPS reliability for runtime enhancement.

For empirical study, we implemented FARE framework as a software application and applied it on a smart building management system for a large commercial building in New York City. Our experiments showed that FARE is easy to implement, accurate for comparison and can be used for building useful industry benchmarks and standards after accumulating enough data.

In the following section, we provide definitions for the terms used in this paper. In section III, we describe FARE framework including CPS reliability model, selection of testing environment, failure analysis and reliability estimation. For empirical evaluation, we present our implementation, experiments and results in section IV. We then compare some related work in section V before conclusion in section VI.

II. Definitions

Formal definitions for the reliability related terms used in this paper are described as follows. Some of these definitions are similar to what are defined in the ANSI/ISO/ASQ standards [4], [5].

- *Reliability* is defined as the probability that a given item will perform its intended function for a given period of time under a given set of conditions.
- *Failure* is the inability of a system or component to perform its required function within the specified performance requirement. It is the manifestation of a fault in the system or a human mistake.
- *Fault* or *Defect* is an incorrect step, process, or data definition in a system or program.
- *Mistake* is a human action that produces an incorrect result.
- *Error* is the difference between a computed, observed, or measured value or condition and the true, specified, or theoretically correct value or condition.

III. Framework

A. CPS Reliability Model

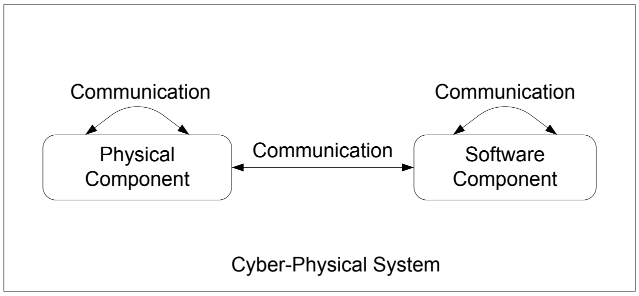


Fig. 1. CPS reliability model.

Composition of CPS [6] is used as the basis of CPS reliability model. As illustrated in Fig. 1, a simple CPS reliability model consists of physical component or hardware, cyber component or software, and communication among them. At the system level, CPS reliability can be measured or estimated and it is an integration of different components' reliability.

B. Selection of Evaluation Environment

Reliability estimation depends on the results or data collected from the tests or the actual use of the system. Fig. 2 illustrates the decision tree approach in selecting different evaluation environments where the failure data will be collected. The top-level category determines whether the results are based on tests in a lab environment or actual use in the operational environment. In a lab environment, some tests are based on life test that simulates the actual running environment. Life test includes Highly Accelerated Life Test (HALT) and Life Test (LT) in a normal pace. The HALT is similar to stress test that creates a situation such that failure is more likely to happen. It is a method based on physics-of-failure.

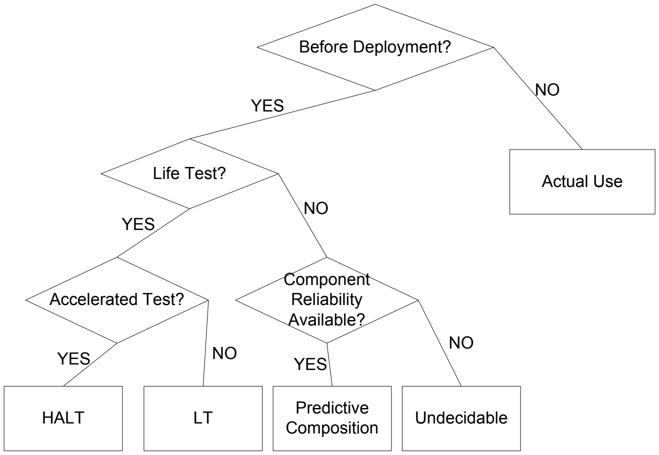


Fig. 2. Decision tree for selecting evaluation environment.

Without a life test, in a lab environment, components' reliability data can be used to construct the whole system's reliability. Although there are many ways to do the compositional reliability, those estimates are often not indicative or accurate for representing the whole system reliability. One reason is the communication failure that is often not easy to be incorporated in these models [7].

Reliability estimation in actual use employs continual failure data processing to enable rolling estimates that are often useful in system performance monitoring, especially for human operators of these systems. There are several advantages for employing reliability estimation in actual use. The first advantage is that it enables real-time feedback to the operators or systems so that corrective actions can be implemented in a manual or autonomic fashion. Second, it enables large long-running systems such as power grids or smart buildings to be continuously monitored for reliability improvement or degradation. Those systems are often not possible to be simulated in a lab environment due to its complexity and the unpredictable running environment. The continual reliability estimates can be further used to construct reliability profile for the system under study. The reliability profiling may show reliability changes in related to different factors such as seasonality and usage pattern.

Not every type of CPS can be evaluated during actual use. For example, medical systems need to be properly tested for reliability prior to its use in the medical operations. Lab testing for these systems is needed.

C. Failure Analysis

Failure analysis includes failure detection and diagnostics. As illustrated in Fig. 3, failure detection provides information for further diagnostics, along with domain knowledge and heuristics.

1) Failure Detection: The failure manifestation can be used as a proxy to system failure. For example, an out of range measurement indicates a system failure. Failure might be

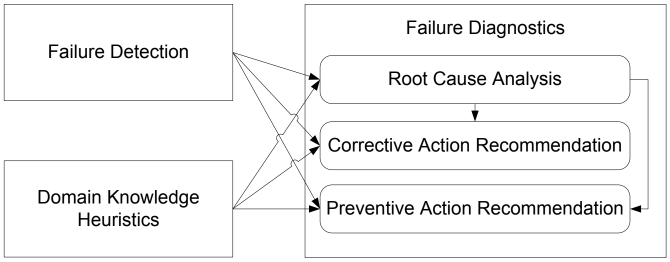

Failure Detection
Domain Knowledge Heuristics
Failure Diagnostics
Root Cause Analysis
Corrective Action Recommendation
Preventive Action Recommendation

Fig. 3. Failure analysis.

Rating	Description of Detection
1	Almost certain to detect
2	Very high chance of detection
3	High chance of detection
4	Moderately high chance of detection
5	Medium chance of detection
6	Low chance of detection
7	Slight chance of detection
8	Remote chance of detection
9	Very remote chance of detection
10	No chance of detection; no inspection

TABLE I
RATING OF DETECTION OF FAILURE.

induced by the external environment, a human mistake or an internal system fault. Automated anomaly detection techniques such as those using machine learning and data mining can be used for more intelligent detection of failures. Table I lists the ratings for detection of failure [8].

2) Failure Diagnostics:

- Root cause analysis (RCA) is used to classify failure type, analyze its nature and mechanism.
- Corrective action recommendation is used to correct the current failure and avoid future recurrence of the same type of failure.
- Preventive action recommendation is used to prevent occurrence of a certain potential failure before it happens. Cost versus benefits can be a factor in determining preventive action such as replacement or inspection of the components.

3) Failure Severity and Impact: To evaluate failure severity and impact, the U.S. military developed Failure Mode Effects Analysis (FMEA) [8] in the 1940s. FMEA and its standards were further developed by the aerospace, automotive and other industries. Table II lists the ratings for failure severity.

Rating	Severity Description
1	The effect is not noticed by customer
2	Very slight effect noticed by customer, does not annoy or inconvenience customer
3	Slight effect that causes customer annoyance, but they do not seek service
4	Slight effect, customer may return product for service
5	Moderate effect, customer requires immediate service
6	Significant effect, causes customer dissatisfaction; may violate regulation or design code
7	Major effect, system may not be operable; elicits customer complaint; may cause injury
8	Extreme effect, system is inoperable and a safety problem. May cause severe injury.
9	Critical effect, complete system shutdown; safety risk
10	Hazardous; failure occurs without warning; life threatening

TABLE II
RATING FOR SEVERITY OF FAILURE.

D. Reliability Estimation

Reliability may be measured in different ways depending on the particular situation [9]. Reliability can be estimated using a qualitative or a quantitative method. Some systems' reliability cannot be estimated quantitatively due to various reasons such as lacking of failure data. For these systems, qualitative method using heuristics may be applicable.

FARE framework primarily employs quantitative methods for reliability estimation. The following are some commonly used reliability metrics that are also applicable to the FARE framework:

- *Failure rate* is defined as the total number of failures within an item population, divided by the total time expended by that population, during a particular measurement interval under stated conditions [10].
- *Mean Time Between Failures (MTBF)* is the mean (expected) time between system failures.
- *Mean Time To Failure (MTTF)* is sometimes used instead of MTBF in cases where a system is replaced after a failure, since MTBF denotes time between failures in a system, which is repaired.
- *Mean Time To Repair (MTTR)* is the mean time required to repair a failed component or device.
- *Availability or Mission Capable Rate* is the proportion of time a system is in a functioning condition. This is also called system uptime ($x\%$). Using a simple representation, it can be calculated as a ratio of the expected value of the uptime of a system to the aggregate of the expected values of up and down time,
- *Availability at time t* is the probability that the item is able to function at time t [9].
- *Survival Probability* is the probability that the item does not fail in a time interval $(0, t]$ [9].

Additionally, we introduce three new reliability measurement metrics in the FARE framework in order to provide coverage for some specific evaluation scenarios:

- *Theta* is the rate of reliability change over time. If MTBF is used as the reliability measurement, then Theta can be calculated as

$$\Theta(t) = \frac{MTBF(t) - MTBF(t + \Delta t)}{\Delta t}.$$

  On a MTBF versus time scatter plot chart, Theta indicates the slop of the linear regression. In a long running continual evaluation environment, Theta provides a useful indicator of the reliability improvement or degradation over time.

- *Vega* is the rate of reliability change over a selected variable, which can be any factor of interest. Similar to Theta, it is a derivative measurement of the reliability for better indication of the reliability improvement or degradation with respect to a specific variable.
- *Cross-Sectional Failure Percentage (CSFP)* is the percentage of total failed items within an item population in use at a specific time t. This may look similar to failure rate or availability at time t. But they are not the same. In a simple form, failure rate is the total number of failures divided by the total time. Availability at time t is the probability of a single item does not fail at time t. Cross-sectional failure percentage is the total number of failed items divided by the total number of items in use at a given time t. It is a ratio based on actual measurement. This metric is especially useful for a large system that has a large number of subsystems or components running in parallel.

To give some further explanation, we use $\lambda(t)$ to denote the failure rate at time t, and $R(t)$ to denote the reliability function (or survival function), which is the probability of no failure before time t. Then the failure rate is:

$$\lambda(t) = \frac{R(t) - R(t + \Delta t)}{\Delta t \cdot R(t)}.$$

As Δt tends to zero, the above λ becomes the instantaneous failure rate, which is also called hazard function (or hazard rate) $h(t)$:

$$h(t) = \lim_{\Delta t \to 0} \frac{R(t) - R(t + \Delta t)}{\Delta t \cdot R(t)}.$$

A failure distribution $F(t)$ is a cumulative failure distribution function that describes the probability of failure up to and including time t:

$$F(t) = 1 - R(t), t \geq 0.$$

For system with a continuous failure rate, $F(t)$ is the integral of the failure density function $f(t)$:

$$F(t) = \int_0^t f(x)\,\mathrm{d}x.$$

Then the hazard function becomes

$$h(t) = \frac{f(t)}{R(t)}.$$

For the Weibull [11], [12] failure distribution, the failure density function $f(t)$ and cumulative failure distribution function $F(t)$ are

$$f(t; \lambda, k) = \begin{cases} \frac{k}{\lambda}(\frac{t}{\lambda})^{k-1} e^{-(t/\lambda)^k}, & t \geq 0 \\ 0, & t < 0 \end{cases}$$

$$F(t; \lambda, k) = \begin{cases} 1 - e^{-(t/\lambda)^k}, & t \geq 0 \\ 0, & t < 0 \end{cases}$$

where $k > 0$ is the shape parameter and $\lambda > 0$ is the scale parameter of the distribution. The hazard function when $t \geq 0$ can be derived as

$$h(t; \lambda, k) = \frac{f(t; \lambda, k)}{R(t; \lambda, k)} = \frac{f(t; \lambda, k)}{1 - F(t; \lambda, k)} = \frac{k}{\lambda}\left(\frac{t}{\lambda}\right)^{k-1}.$$

A value of $k < 1$ indicates that the failure rate decreases over time. A value of $k = 1$ indicates that the failure rate is constant (*i.e.*, k/λ) over time. In this case, the Weibull distribution becomes an exponential distribution. A value of $k > 1$ indicates that the failure rate increases with time.

The mean time to failure is given by

$$MTTF = \int_0^\infty t \cdot f(t)\,\mathrm{d}t = \int_0^\infty R(t)\,\mathrm{d}t.$$

If $MTTR$ is known, then the availability is

$$Availability = \frac{MTTF}{(MTTF + MTTR)}.$$

IV. Empirical Study

A. Implementation

We developed a prototype software application named FARE for this study using Java programming language and MATLAB. As illustrated in the software architecture diagram in Fig. 4, the software includes a data preprocessor, a failure detector, a reliability estimator with metrics and profiler, and a user interface along with data output component. It was designed with modular components so that it can be used along with or embedded in another larger system.

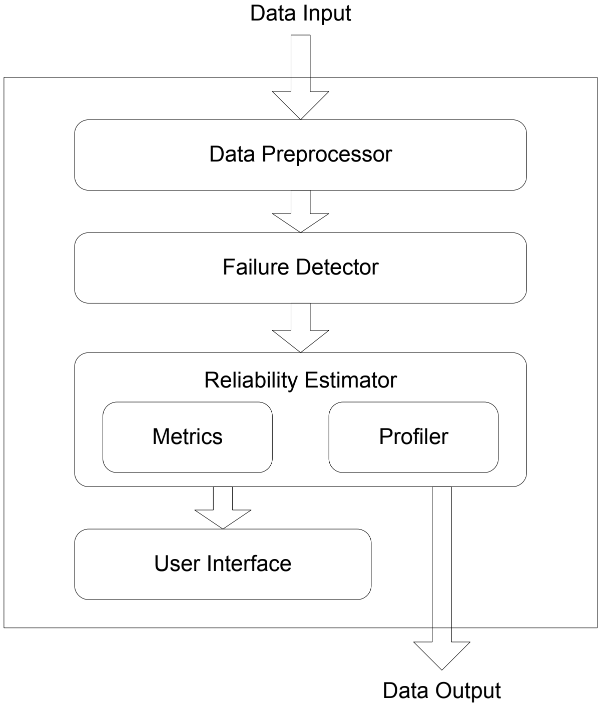


Fig. 4. FARE software architecture.

B. Smart Building CPS

Smart Building Management System (BMS) at a large office building in New York City was used for this study. A *Building Management System (BMS)* is a type of CPS consisting of both software and hardware components that controls and monitors a building's mechanical and electrical equipment, such as ventilation, lighting, power systems, fire systems and security systems. The building energy control system is an important component of the BMS that reads data feeds representing internal and exogenous conditions (*e.g.*, temperature, humidity, electrical load, peak load, fluctuating electricity pricing and building work schedule) and takes control actions (*e.g.*, adjust lighting, turn on/off the air-conditioning and shut off partial elevators) accordingly. Building operators usually have the ability to change or override control actions taken by the BMS to accommodate special situations such as severe weather or changes in the building's work schedule.

C. Experimental Setup

Fig. 5 illustrates our experimental setup. The building's BMS's software collects various data sources and stores them in the local BMS database. We established a data transmission link between the BMS server and the remote server where FARE software is installed and running.

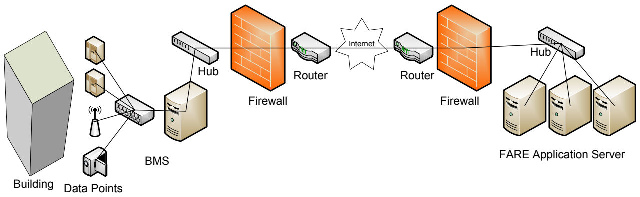


Fig. 5. Experimental setup.

Determination of the failure trigger condition depends on the data source we use. We first use FARE software to process the data through the decided failure criteria to obtain a dataset of failure time series. These data are then processed by the FARE software to obtain reliability estimates on the fly.

D. Experimental Results

1) Failure Detection: Fig. 6 shows an example BMS data time series for six months starting from July 1, 2012 to January 1, 2013. As a simple threshold failure detection method, the data points with value above 80 or below 65 were determined to be nonconformity or failure.

2) Reliability Estimation: After the failure incidence time series data is collected, FARE then estimates reliability metrics as described in Section III-D. To follow the example described above, the weekly failure rates for the six months are listed and charted in Fig. 7. Also, the linear regression $y = -0.0229x + 1.1169, R^2 = 0.03829$ shows the improved results over the time. In this case, the Theta equals -0.0229 using failure rate as the reliability measurement.

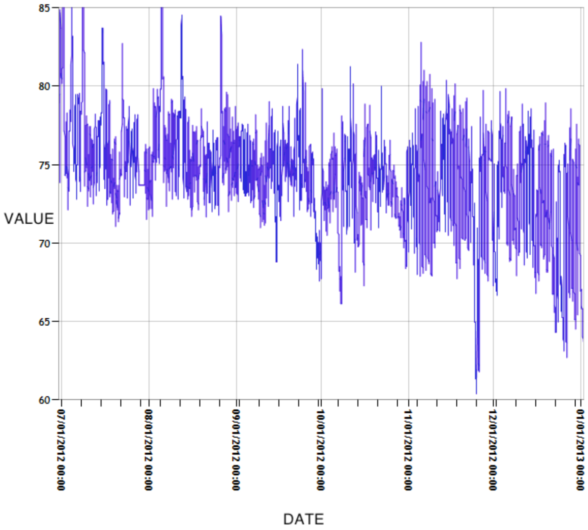
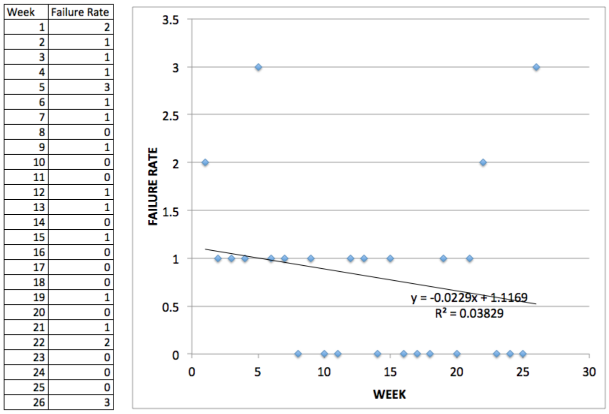


Fig. 6. BMS time series data.



Week	Failure Rate
1	2
2	1
3	1
4	1
5	3
6	1
7	1
8	0
9	1
10	0
11	0
12	1
13	1
14	0
15	1
16	0
17	0
18	0
19	1
20	0
21	1
22	2
23	0
24	0
25	0
26	3

Fig. 7. Weekly failure rate.

V. Related Work

As stated in section III, CPS reliability consists of overall system reliability and component reliability including physical component, software and communication. Lee stated in his paper [1], CPS cannot be deployed for certain mission-critical applications such as traffic control, automotive safety or healthcare without improved reliability and predictability.

Some prior researches have been done on component reliability. Pechet and Nash gave a comprehensive review of the predictive methods for predicting the reliability of electronic equipment [13]. He *et al.* described a theoretical framework for analyzing communication reliability using frequency domain analysis and reliability calculus [7]. In our BUGMINER paper [14], we described an approach of software reliability analysis using Weibull distribution [11], [12] and data mining of bug reports. These prior work are complementary to the FARE framework for benchmarking CPS reliability.

Failure analysis has been a popular research area for many years. Stamatis described theory and execution of Failure

Mode Effect Analysis (FMEA) [15]. Robitaille categorized and described some common corrective actions in his handbook [16]. Their work is complementary to our approach and has different applicable domain.

VI. Conclusion

This paper described FARE (Failure Analysis and Reliability Estimation), a framework for benchmarking reliability of cyber-physical systems. FARE employs a generic CPS reliability model, a set of methods and metrics on the evaluation environment selection, failure analysis and reliability estimation for benchmarking CPS reliability. The framework is extensible for accommodating new reliability measurement techniques and metrics. Our empirical evaluation demonstrated that FARE is easy to implement, accurate for comparison and can be used for building useful industry benchmarks.

Acknowledgment

Wu and Kaiser are members of the Programming Systems Laboratory, funded in part by NSF CCF-1161079, NSF CNS-0905246, and NIH 2 U54 CA121852-06. Wu is also a member of the Center for Computational Learning Systems, supported in part by U.S. Department of Energy, Consolidated Edison, General Electric, FedEx, Finmeccanica and Rudin Management Company.

References

[1] E. A. Lee, "Cyber physical systems: Design challenges," in *International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing (ISORC)*, May 2008, invited Paper. [Online]. Available: http://chess.eecs.berkeley.edu/pubs/427.html

[2] CPS Steering Group, "Cyber-physical systems executive summary," in *CPS Summit 2008*, March 2008, http://varma.ece.cmu.edu/Summit/.

[3] S. M. Amin, "U.S. electrical grid gets less reliable," *IEEE Spectrum*, p. 80, January 2011.

[4] ISO, *ISO 8402:1994 Quality management and quality assurance – Vocabulary*. ISO, 1994.

[5] ANSI/ISO/ASQ, *Q9001-2000 Quality Management Systems–Requirements*. ANSI/ASQ, 2001.

[6] J. Sztipanovits, "Composition of cyber-physical systems," in *14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS '07)*, March 2007, pp. 3–6.

[7] W. He, X. Liu, L. Zheng, and H. Yang, "Reliability calculus: A theoretical framework to analyze communication reliability," in *2010 IEEE 30th International Conference on Distributed Computing Systems (ICDCS)*, June 2010, pp. 159–168.

[8] American Society for Quality, "http://asq.org/learn-about-quality/process-analysis-tools/overview/ fmea.html," 2013.

[9] M. Rausand and A. Høyland, *System Reliability Theory: Models, Statistical Methods, and Applications*, 2nd ed. Wiley, December 2003.

[10] B. Littlewood and J. L. Verrall, "A bayesian reliability model with a stochastically monotone failure rate," *IEEE Transactions on Reliability*, vol. R-23, no. 2, pp. 108–114, 1974.

[11] W. Weibull, "A statistical distribution function of wide applicability," *ASME Journal of Applied Mechanics*, pp. 293–297, September 1951.

[12] S. E. Rigdon and A. P. Basu, "Estimating the intensity function of a Weibull process at the current time: Failure truncated case," in *Journal of Statistical Computation and Simulation (JSCS)*, vol. 30, 1988, pp. 17–38.

[13] M. G. Pecht and F. R. Nash, "Predicting the reliability of electronic equipment," *Proceedings of the IEEE*, vol. 82, no. 7, July 1994.

[14] L. Wu, B. Xie, G. Kaiser, and R. Passonneau, "BugMiner: Software reliability analysis via data mining of bug reports," in *Proceedings of the 23th International Conference on Software Engineering and Knowledge Engineering (SEKE)*, July 2011.

[15] D. H. Stamatis, *Failure Mode Effect Analysis: FMEA From Theory to Execution*, 2nd ed. ASQ Quality Press, 2003.

[16] D. Robitaille, *The Corrective Action Handbook*. Paton Press, 2002.

Leon Wu (M'07) is a PhD candidate at the Department of Computer Science and a Senior Research Associate at the Center for Computational Learning Systems of Columbia University. He received his MS and MPhil in Computer Science from Columbia University and BSc in Physics from Sun Yat-sen University.

Gail Kaiser (M'85-SM'90) is a Professor of Computer Science and the Director of the Programming Systems Laboratory in the Computer Science Department at Columbia University. She was named an NSF Presidential Young Investigator in Software Engineering and Software Systems in 1988, and she has published over 150 refereed papers in a range of software areas. Her research interests include software testing, collaborative work, computer and network security, parallel and distributed systems, self-managing systems, Web technologies, information management, and software development environments and tools. She has consulted or worked summers for courseware authoring, software process and networking startups, several defense contractors, the Software Engineering Institute, Bell Labs, IBM, Siemens, Sun and Telcordia. Her lab has been funded by NSF, NIH, DARPA, ONR, NASA, NYS Science & Technology Foundation, and numerous companies. Prof. Kaiser served on the editorial board of IEEE Internet Computing for many years, was a founding associate editor of ACM Transactions on Software Engineering and Methodology, chaired an ACM SIGSOFT Symposium on Foundations of Software Engineering, vice chaired three of the IEEE International Conference on Distributed Computing Systems, and serves frequently on conference program committees. She also served on the Committee of Examiners for the Educational Testing Service's Computer Science Advanced Test (the GRE CS test) for three years, and has chaired her department's doctoral program since 1997. Prof. Kaiser received her PhD and MS from CMU and her ScB from MIT.