

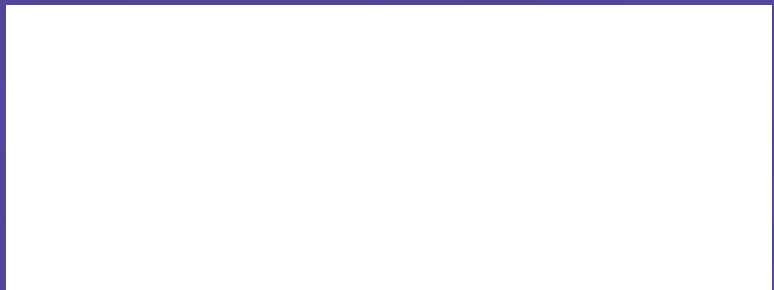
COMPUTING edge

- > Security and Privacy
- > Smart Homes
- > Autonomous Vehicles
- > Robotics



AUGUST 2019

www.computer.org



Keep Your Career Options Open

Upload Your Resume Today!

Whether you enjoy your current position or you are ready for change, the **IEEE Computer Society Jobs Board** is a valuable resource tool.

Take advantage of these special resources for job seekers:



JOB ALERTS



TEMPLATES



CAREER
ADVICE



RESUMES VIEWED
BY TOP EMPLOYERS



WEBINARS

No matter your career level, the IEEE Computer Society Jobs Board keeps you connected to workplace trends and exciting new career prospects.

www.computer.org/jobs



IEEE
COMPUTER
SOCIETY



STAFF

Editor

Cathy Martin

Publications Operations Project Specialist

Christine Anthony

Publications Marketing Project Specialist

Meghan O'Dell

Production & Design

Carmen Flores-Garvey

Publications Portfolio Managers

Carrie Clark, Kimberly Sperka

Publisher

Robin Baldwin

Senior Advertising Coordinator

Debbie Sims

Circulation: ComputingEdge (ISSN 2469-7087) is published monthly by the IEEE Computer Society. IEEE Headquarters, Three Park Avenue, 17th Floor, New York, NY 10016-5997; IEEE Computer Society Publications Office, 10662 Los Vaqueros Circle, Los Alamitos, CA 90720; voice +1 714 821 8380; fax +1 714 821 4010; IEEE Computer Society Headquarters, 2001 L Street NW, Suite 700, Washington, DC 20036.

Postmaster: Send address changes to ComputingEdge-IEEE Membership Processing Dept., 445 Hoes Lane, Piscataway, NJ 08855. Periodicals Postage Paid at New York, New York, and at additional mailing offices. Printed in USA.

Editorial: Unless otherwise stated, bylined articles, as well as product and service descriptions, reflect the author's or firm's opinion. Inclusion in ComputingEdge does not necessarily constitute endorsement by the IEEE or the Computer Society. All submissions are subject to editing for style, clarity, and space.

Reuse Rights and Reprint Permissions: Educational or personal use of this material is permitted without fee, provided such use: 1) is not made for profit; 2) includes this notice and a full citation to the original work on the first page of the copy; and 3) does not imply IEEE endorsement of any third-party products or services. Authors and their companies are permitted to post the accepted version of IEEE-copyrighted material on their own Web servers without permission, provided that the IEEE copyright notice and a full citation to the original work appear on the first screen of the posted copy. An accepted manuscript is a version which has been revised by the author to incorporate review suggestions, but not the published version with copy-editing, proofreading, and formatting added by IEEE. For more information, please go to: http://www.ieee.org/publications_standards/publications/rights/paperversionpolicy.html. Permission to reprint/republish this material for commercial, advertising, or promotional purposes or for creating new collective works for resale or redistribution must be obtained from IEEE by writing to the IEEE Intellectual Property Rights Office, 445 Hoes Lane, Piscataway, NJ 08854-4141 or pubs-permissions@ieee.org. Copyright © 2019 IEEE. All rights reserved.

Abstracting and Library Use: Abstracting is permitted with credit to the source. Libraries are permitted to photocopy for private use of patrons, provided the per-copy fee indicated in the code at the bottom of the first page is paid through the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923.

Unsubscribe: If you no longer wish to receive this ComputingEdge mailing, please email IEEE Computer Society Customer Service at help@computer.org and type "unsubscribe ComputingEdge" in your subject line.

IEEE prohibits discrimination, harassment, and bullying. For more information, visit www.ieee.org/web/aboutus/whatis/policies/p9-26.html.

IEEE Computer Society Magazine Editors in Chief

Computer

David Alan Grier (Interim),
Djaghe LLC

IEEE Software

Ipek Ozkaya, Software
Engineering Institute

IEEE Internet Computing

George Pallis, University of
Cyprus

IT Professional

Irena Bojanova, NIST

IEEE Security & Privacy

David Nicol, University of Illinois
at Urbana-Champaign

IEEE Micro

Lizy Kurian John, University of
Texas, Austin

IEEE Computer Graphics and Applications

Torsten Möller, University of
Vienna

IEEE Pervasive Computing

Marc Langheinrich, University of
Lugano

Computing in Science & Engineering

Jim X. Chen, George Mason
University

IEEE Intelligent Systems

V.S. Subrahmanian, Dartmouth
College

IEEE MultiMedia

Shu-Ching Chen, Florida
International University

IEEE Annals of the History of Computing

Gerardo Con Diaz, University of
California, Davis

AUGUST 2019 • VOLUME 5, NUMBER 8

COMPUTING
edge



14

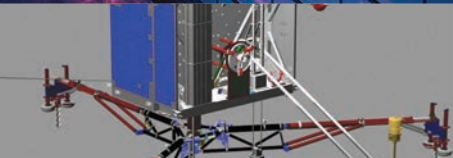
Penetration
Testing in the
IoT Age

18

IoT Safety: State
of the Art

40

Lance Gharavi:
Performance-
Inspired Science
+ Technology



47

A Comet Revisited: Lessons Learned from *Philae's* Landing

Security and Privacy

8 The Fuzzing Hype-Train: How Random Testing Triggers Thousands of Crashes

MATHIAS PAYER

14 Penetration Testing in the IoT Age

CHUNG-KUAN CHEN, ZHI-KAI ZHANG, SHAN-HSIN LEE, AND SHIUHPYNG SHIEH

Smart Homes

18 IoT Safety: State of the Art

JANUSZ ZALEWSKI

23 Physical Computing's Connected and Shape-Changing Future

HEATHER M. PATTERSON

Autonomous Vehicles

28 Next-Generation Smart Environments: From System of Systems to Data Ecosystems

EDWARD CURRY AND AMIT SHETH

37 My Mother the Car (or Why It's a Bad Idea to Give Your Car a Personality)

PHIL LAPLANTE

Robotics

40 Lance Gharavi: Performance-Inspired Science + Technology

BRUCE CAMPBELL AND FRANCESCA SAMSEL

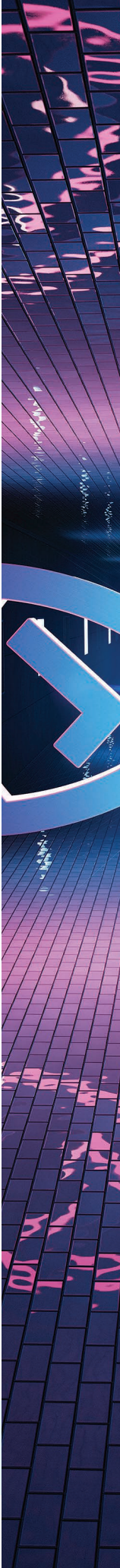
47 A Comet Revisited: Lessons Learned from *Philae's* Landing

ANDRÁS BALÁZS

Departments

- 4 Magazine Roundup
- 7 Editor's Note: Finding Flaws in Security
- 58 Conference Calendar

Subscribe to **ComputingEdge** for free at www.computer.org/computingedge.



Magazine Roundup

The IEEE Computer Society's lineup of 12 peer-reviewed technical magazines covers cutting-edge topics ranging from software design and computer graphics to Internet computing and security, from scientific applications and machine intelligence to visualization and microchip design. Here are highlights from recent issues.

Computer

Basic Cyber Hygiene: Does It Work?

A number of security certifications for small- and medium-size

enterprises have been proposed, but how effective are these schemes? The authors of this article from the April 2019 issue of *Computer* evaluate the effectiveness of Cyber Essentials and find that its security controls work well to mitigate threats that exploit vulnerabilities remotely with commodity-level tools.

Computing in Science & Engineering

Fostering Reuse in Scientific Computing With Embedded Components

Component-based programming is a programming paradigm that

eases software reuse but has yet to be widely adopted in scientific computing. In this article from the March/April 2019 issue of *Computing in Science & Engineering*, the authors propose embedding component frameworks inside high-performance languages directly to improve flexibility. They present this approach through the example of a high-performance Bayesian inference application.

IEEE Annals of the History of Computing

Hacking the Cis-tem

This article from the January–March 2019 issue of *IEEE Annals of the History of Computing* looks at the case of transgender Britons who tried to correct the gender listed on their government-issued ID cards, but ran up against the British government's increasingly

computerized methods for tracking, identifying, and defining citizens. These newly computerizing systems show some of the earliest examples of transphobic algorithmic bias; explicit attempts to program trans people out of the system can be seen in the programming of the early Ministry of Pensions computer system designed to apportion benefits to all tax-paying British citizens. Transgender citizens pushed back against these developments, attempting to hack the bureaucratic avenues and categories available to them, laying the groundwork for a coalescing political movement. This article argues that uncovering the deep prehistory of algorithmic bias and investigating instances of resistance within this history is essential to understanding current debates about algorithmic bias, and how computerized systems have long functioned to create and enforce norms and hierarchies.

IEEE Computer Graphics and Applications

Fast Sketch Segmentation and Labeling With Deep Learning

The authors of this article from the March/April 2019 issue of *IEEE Computer Graphics and Applications* present a simple and efficient method based on deep learning to automatically decompose sketched objects into semantically valid parts. They train a deep neural network to transfer existing segmentations and labelings from 3D models to freehand sketches without requiring numerous well-annotated sketches as training

data. The network takes the binary image of a sketched object as input and produces a corresponding segmentation map with per-pixel labelings as output. A subsequent post-process procedure with multi-label graph cuts further refine the segmentation and labeling result. The authors validate the proposed method on two sketch datasets. Experiments show that the method outperforms the state-of-the-art method in terms of segmentation and labeling accuracy and is significantly faster, enabling further integration in interactive drawing systems. The authors demonstrate the method's efficiency in a sketch-based modeling application that automatically transforms input sketches into 3D models by part assembly.

IEEE Intelligent Systems

Multimodal Foraging by Honey Bees toward Optimizing Profits at Multiple Colonies

Honey bees single out from available foraging sources by evaluating the amount of energy needed to transport an article for livelihood. In this article from the January/February 2019 issue of *IEEE Intelligent Systems*, the authors propose a novel method based on the dynamic and distributed computing behavior of honey bees at distinct colonies in gathering multiple resources for meeting their demands at respective destinations by maximizing the profit. A computational model depicting the multidimensional swarm behavior in bee colonies is developed

as a multi-objective optimization problem for optimizing cost and time. The performance of the algorithm is evaluated by mapping it to a dynamic model of the multi-commodity transportation problem with multiple optimizing parameters. The algorithm is found to terminate successfully in linear time at each destination accounting for uncertainties, which are the natural property of the real world, thus optimizing the objective function at all nodes.

IEEE Internet Computing

meChat: In-Device Personal Assistant for Conversational Photo Sharing

It is still challenging to search in-device photos without the sacrifice of privacy or poor latency, despite the fact that photos have long been an essential part of communication in messaging applications. In this article from the March/April 2019 issue of *IEEE Internet Computing*, the authors propose a novel in-device personal assistant for conversational photo sharing, called meChat. By understanding the user's conversations and in-device photos, meChat intelligently searches in-device photos that are semantically relevant to the conversation context. Notably, meChat works in a stand-alone, privacy-protecting manner without sending out any in-device personal content. The scenario-based user studies show that meChat efficiently searches highly relevant in-device photos with low perceived latency and energy consumption.

IEEE Micro

Samsung M3 Processor

The M3 processor is Samsung's third-generation custom microarchitecture. As performance demands continue to grow, major design improvements are required. In this generation, the core is enhanced with a six-wide microarchitecture, deeper out-of-order resources, and faster instructions. The M3 delivers a new level of performance to the Android ecosystem. Read more in the March/April 2019 issue of *IEEE Micro*.

IEEE MultiMedia

QoE-Oriented Multimedia Assessment: A Facial Expression Recognition Approach

Multimedia services are predominant in current wireless networks and are becoming ubiquitous in the upcoming 5G era in which the video quality of experience (QoE) is a fundamental metric. However, no widely accepted QoE model exists due to its subjective nature. This article from the January–March 2019 issue of *IEEE MultiMedia* proposes a framework for quantifying the QoE of multimedia content based on the facial expression approach, which can directly reflect users' intrinsic attitudes toward the services. To achieve this objective, a face database is established, which contains over 1,000 videos and serves as a dataset for the subsequent experience mining. The semi-supervised clustering

method proposed in this article is applied to calculate the video experience scores and achieves 8% higher average test accuracy than other prevailing methods. Extensive experimental results show that this approach can accurately reveal the user's experience toward video content and is expected to become a valid and useful QoE model.

IEEE Pervasive Computing

Supporting the IoT Business Value Through the Platformization of Pilots

Testbeds are powerful large-scale experimentation tools. Often, however, their limited scope and/or unrealistic functionality result in anecdotal use. Knowledge transfer becomes more effective through pilots, but they involve techno-economic challenges with uncertain outcomes. Platform-based design has proven a valuable strategy to overcome similar barriers. However, the authors of this article from the October–December 2018 issue of *IEEE Pervasive Computing* identify that while this platformization is happening in the cloud elements of the value chain, Internet-of-Things (IoT) devices are still designed in an application-specific manner, thus limiting pilots' initiatives.

IEEE Security & Privacy

Privacy-Preserving Machine Learning: Threats and Solutions

For privacy concerns to be addressed adequately in today's

machine-learning (ML) systems, the knowledge gap between the ML and privacy communities must be bridged. This article from the March/April 2019 issue of *IEEE Security & Privacy* aims to provide an introduction to the intersection of both fields with special emphasis on the techniques used to protect the data.

IEEE Software

Strategies for Competing in the Automotive Industry's Software Ecosystem: Standards and Bottlenecks

The automotive industry includes many actors engaged in software. This article from the May/June 2019 issue of *IEEE Software* focuses on the controlling position of car manufacturers in the automotive software ecosystem and suggests three strategies for software innovators: contesting, cooperating, and circumventing.

IT Professional

Decentralization: The Failed Promise of Cryptocurrencies

Cryptocurrencies promise to revolutionize the financial market due to two main features: security and decentralization. In this article from the March/April 2019 issue of *IT Professional*, the authors analyze whether cryptocurrencies are fully decentralized—in other words, whether the transaction processing is distributed among different entities. In addition, they present the consequences that a possible centralization entails. ●

Finding Flaws in Security

Catching and fixing security and privacy vulnerabilities before they are exploited is a crucial part of computer hardware and software engineering. A growing number of tools are helping engineers assess and test for security weaknesses. This issue of *ComputingEdge* covers two effective testing techniques: fuzzing and penetration testing.

Fuzzing is the process of executing a software program using random inputs with the goal of discovering bugs. *IEEE Security & Privacy's* "The Fuzzing Hype-Train: How Random Testing Triggers Thousands of Crashes" describes fuzzing's many benefits and challenges. Penetration testing, on the other hand, simulates cyberattacks to uncover flaws. *Computer's* "Penetration Testing in the IoT Age" proposes strategies for improving penetration testing of IoT objects.

IoT devices must be not only secure, but also safe—especially in smart homes. "IoT Safety: State of the Art," from *IT Professional*, stresses the importance of preventing Internet-connected appliances and control systems from malfunctioning and hurting people or property. "Physical Computing's Connected and Shape-Changing Future,"

from *IEEE Pervasive Computing*, goes a step further and argues that smart-home devices should not only be secure and safe, but also contribute to social progress and human wellbeing.

Smart devices are also key components in autonomous vehicles. In *IEEE Intelligent Systems's* "Next-Generation Smart Environments: From System of Systems to Data Ecosystems," the authors discuss the enormous amount of data generated by self-driving cars and the opportunities and challenges associated with processing and sharing that data. "My Mother the Car (or Why It's a Bad Idea to Give Your Car a Personality)," from *IT Professional*, explores the concept of anthropomorphic autonomous vehicles through the lens of science-fiction stories.

This *ComputingEdge* issue concludes with two articles on robotics. *IEEE Computer Graphics and Applications's* "Lance Gharavi: Performance-Inspired Science + Technology" features an artist who integrates robots and other cutting-edge technologies into his projects. *IEEE Software's* "A Comet Revisited: Lessons Learned from *Philae's* Landing" provides takeaways from the robotic lander's mission. 🍷

The Fuzzing Hype-Train: How Random Testing Triggers Thousands of Crashes

Mathias Payer | EPFL, Lausanne, Switzerland

Software contains bugs, and some bugs are exploitable. Mitigations protect our systems in the presence of these vulnerabilities, often stopping the program once a security violation has been detected. The alternative is to discover bugs during development and fix them in the code. The task of finding and reproducing bugs is difficult; however, fuzzing is an efficient way to find security-critical bugs by triggering exceptions, such as crashes, memory corruption, or assertion failures automatically (or with a little help). Furthermore, fuzzing comes with a witness (proof of the vulnerability) that enables developers to reproduce the bug and fix it.

Software testing broadly focuses on discovering and patching bugs during development. Unfortunately, a program is only secure if it is free of unwanted exceptions. Security, therefore, requires proof of the absence of security violations. For example, a bug becomes a vulnerability if any attacker-controlled input reaches a program location that allows a security violation, such as memory corruption. Software security testing, therefore, requires reasoning about all possible executions of code at once to produce a witness that violates the security property. As Edsger W. Dijkstra said in 1970: “Program testing can be used to show the presence of bugs, but never to show their absence!”

Digital Object Identifier 10.1109/MSEC.2018.2889892
Date of publication: 20 March 2019

System software, such as a browser, a runtime system, or a kernel, is written in low-level languages (such as C and C++) that are prone to exploitable, low-level defects. Undefined behavior is at the root of low-level vulnerabilities, e.g., invalid pointer

The idea of fuzzing is simple: execute a program in a test environment with random input and see if it crashes.

dereferences resulting in memory corruption, casting to an incompatible type leading to type confusion, integer overflows, or application programming interface (API) confusion. To cope with the complexity of current programs and find bugs, companies such as Google, Microsoft, and Apple integrate dynamic testing into their software development cycle.

Fuzzing, the process of providing random input to a program to intentionally trigger crashes, has been around since the early 1980s. A revival of fuzzing techniques is taking place as evidenced by papers presented at top-tier security conferences showing improvements in the techniques’ effectiveness. The idea of fuzzing is simple: execute a program in a test environment with random input and see if it crashes. The fuzzing process is inherently sound but incomplete. By producing trial cases and observing whether the tested program crashes, fuzzing produces a witness for each

discovered crash. As a dynamic testing technique, fuzzing is incomplete for nontrivial programs as it will neither cover all possible program paths nor all data-flow paths except when run for an infinite amount of time. Fuzzing strategies are inherently optimization problems where the available resources are used to discover as many bugs as possible, covering as much of the program functionality as possible through a probabilistic exploration process. Due to its nature as a dynamic testing technique, fuzzing faces several unique challenges:

- *Input generation:* Fuzzers generate inputs based on a mutation strategy to explore a new state. Because the fuzzer is aware of the program structure, it can tailor input generation to the program. The underlying strategy determines how effectively the fuzzer explores a given state space. A challenge for input generation is finding the balance between exploring new paths (control flow) and executing the same paths with different input (data flow).
- *Execution engine:* The execution engine takes newly generated input and executes the program under test with that input to detect flaws. Fuzzers must distinguish between benign and buggy executions. Not every bug results in an immediate segmentation fault, and detecting a state violation is a challenging task, especially as code generally does

not come with a formal model. Additionally, the fuzzer must disambiguate crashes to identify bugs without missing true positives.

- *Coverage wall:* Fuzzing struggles with some aspects of code. It may, for example, have difficulty handling a complex API, checksums in file formats, or hard comparisons, such as a password check. Preparing the fuzzing environment is a crucial step to increase the efficiency of fuzzing.
- *Evaluating fuzzing effectiveness:* Defining the metrics for evaluating the effectiveness of a fuzzing campaign is challenging. For most programs, the state space is (close to) infinite, and fuzzing is a brute-force search in this state space. Deciding, for example, when to move to another target, path, or input is a crucial aspect of fuzzing. Orthogonally, comparing different fuzzing techniques requires an understanding of the strengths of a fuzzer and the underlying statistics to enable a fair comparison.

Input Generation

Input generation is essential to the fuzzing process as every fuzzer must automatically generate test cases to be run on the execution engine. The cost of generating a single input must be low, following the underlying philosophy of fuzzing where iterations are cheap. Through input generation, the fuzzer implicitly selects which parts of the tested program are executed. Input generation must balance data-flow and control-flow exploration (discovering new code areas compared to revisiting previously executed code areas with alternate data) while considering what areas to focus on. There are two fundamental forms of input generation: model- and mutation-based input generation. The first is aware of the input format while the latter is not.

Knowledge of the input structure given through a formal description enables model-based input generation to produce (mostly) valid test cases. The model specifies the input format and implicitly indicates the explorable state space. Based on the model, the fuzzer can produce valid test cases that satisfy many checks in the program, such as valid state checks, dependencies between fields, or checksums such as a CRC32. For example, without an input model, most randomly generated test cases will fail the equality check for a cor-

Through input generation, the fuzzer implicitly selects which parts of the tested program are executed.

rect checksum and quickly error out without triggering any complex behavior. The model allows input generation to balance the created test inputs according to the underlying input protocol. The disadvantage of model-based input generation is that it needs an actual model. Most input formats are not formally described and will require an analyst to define the intricate dependencies.

Mutation-based input generation requires a set of seed inputs that trigger valid functionality in the program and then leverages random mutation to modify these seeds. Providing a set of valid inputs is significantly easier than formally specifying an input format. The input-mutation process then constantly modifies these input seeds to trigger behavior that researchers want to study.

Regardless of the input-mutation strategy, fuzzers need a fitness function to assess the quality of the new input and guide the generation of new input. A fuzzer may leverage the program structure and code coverage as fitness functions. There are three approaches to observing the program

during fuzzing to provide input to the fitness function. White-box fuzzing infers the program specification through program analysis but often results in untenable cost. For example, the scalable automated guided execution white-box fuzzer leverages symbolic execution to explore different program paths. Black-box fuzzing blindly generates new input without reflection. The lack of a fitness function limits black-box fuzzing to functionality close to the provided test cases. Grey-box fuzzing leverages lightweight program instrumentation instead of heavier program analysis to infer coverage during the fuzzing campaign itself, merging analysis and testing.

Coverage-guided grey-box fuzzing combines mutation-based input generation with program instrumentation to detect whenever a mutated input reaches new coverage. Program instrumentation tracks which areas of the code are executed, and the coverage profile is tied to specific inputs. Whenever an input mutation generates new coverage, it is added to the set of inputs for mutation. This approach is highly efficient due to the low-cost instrumentation but still results in broad program coverage. Coverage-guided fuzzing is the current de facto standard, with American fuzzy lop¹ and honggfuzz² as the most prominent implementations. These fuzzers leverage execution feedback to tailor input generation without requiring the analyst to have deep insight into the program structure.

A difficulty for input generation is finding the perfect balance between the need to discover new paths and the need to evaluate existing paths with different data. While the first increases coverage and explores new program areas, the latter explores already covered code through the use of different data. Existing metrics have a heavy control-flow focus as coverage measures how much of the

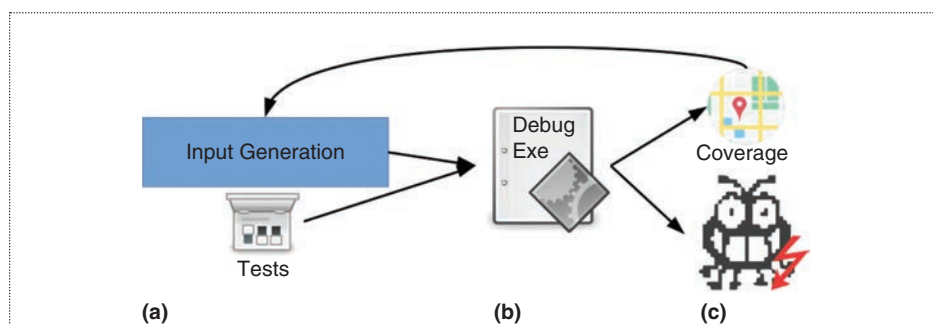


Figure 1. Fuzzing consists of an execution engine and an input-generation process that runs executables, which are often instrumented with explicit memory safety checks. (a) The input-generation mechanism (the blue box marked “Input Generation”) may leverage existing test cases (“Tests”) and execution coverage to generate new test inputs. For each discovered crash, the fuzzer provides a witness (the input that triggers the crash). (b) The execution engine. (c) A “bug” triggers the crash. The icon marked “Coverage” indicates input that has passed through the execution engine. Some of that input may pass through the input-generation process again. Arrows indicate the direction of process. Exe: executable.

program has already been explored. Data-flow coverage is only measured implicitly with inputs that execute the same paths but with different data values. A good input-generation mechanism balances the explicit goal of extending coverage with the implicit goal of rerunning the same input paths with different data.

Execution Engine

After the fuzzer generates test cases, it must execute them in a controlled environment and detect when a bug is triggered. The execution engine takes the fuzz input, executes the program under test, extracts runtime information, such as coverage, and detects crashes

(Figure 1). Ideally, a program would terminate whenever a flaw is triggered. For example, an illegal pointer dereference on an unmapped memory page results in a segmentation fault, which terminates the program, allowing the executing engine to detect the flaw. Unfortunately, only a small subset of security violations will result in program crashes. Buffer overflows into adjacent memory locations, for instance, may never be detected at all or may only be detected later if the overwritten

data are used. The challenge for this component of the fuzzing process is to efficiently enable the detection of security violations. For example, without instrumentation, only illegal pointer dereferences to unmapped memory, control-flow transfers to nonexecutable memory, division by zero, or similar violations will trigger an exception.

To detect security violations early, the tested program may be instrumented with additional software guards. It is especially tricky

The main goal of the execution engine is to conduct inputs as fast as possible.

to find security violations through undefined behavior for code written in system languages. Sanitization analyzes and instruments the program during the compilation process to detect security violations. Address Sanitizer,³ the most commonly used sanitizer, employs probability to detect spatial and temporal memory safety violations by placing red zones around allocated memory objects, keeping track of allocated memory, and checking memory accesses. Other LLVM-based

sanitizers cover undefined behavior, uninitialized memory, or type safety violations through illegal casts.⁴ Each sanitizer requires a certain type of instrumentation, which increases the performance cost. The use of sanitizers for fuzzing, therefore, has to be carefully evaluated as, on one hand, it makes error detection more likely but, on the other hand, it reduces fuzzing throughput.

The main goal of the execution engine is to conduct inputs as fast as possible. Several fuzzing optimizations, such as fork servers, persistent fuzzing, or special operating system (OS) primitives, reduce the time for each execution by adjusting system parameters. Fuzzing with a fork server executes the program up to a certain point and then forks new processes at that location for each new input. This allows the execution engine to skip over initialization code that would be the same for each execution. Persistent fuzzing allows the execution engine to reuse processes in a pool with new fuzzing input, resetting the state between executions. Different OS primitives for fuzzing reduce the cost of process creation by, for example, simplifying the creation of page tables and optimizing scheduling for short-lived processes.

Modern fuzzing is heavily optimized and focuses on efficiency, measured by the number of bugs found per unit of time. Sometimes fuzzing efficiency is implicitly measured by the number of crashes found per unit of time. However, crashes are not necessarily unique, and many crashes could point to the same bug. Disambiguating crashes to locate unique bugs is an important but challenging task. Multiple bugs may cause a program crash at the same location, whereas one input may trigger multiple bugs. A fuzzer must triage crashes conservatively

so that no true bugs are removed. Yet the triaging must not overload the analyst with redundant crashes.

Coverage Wall

In addition to massive parallelism, a key advantage of fuzzing compared to more heavyweight analysis techniques is its simplicity. However, due to this simplicity, fuzzing can get stuck in local minima in front of a coverage wall. When this happens, continuous input generation will not result in either additional crashes or new coverage. A common approach to circumvent the coverage wall is to extract seed values used for comparisons. These seed values are then used during the input-generation process. Orthogonally, a developer can comment out hard checks, such as CRC32 comparisons, or checks for magic values. Removing these noncritical checks from the program requires a knowledgeable developer to tailor fuzzing for each program.

Several recent extensions⁵⁻⁸ try to bypass the coverage wall by automatically detecting when the fuzzer gets stuck and, then, if the problem is detected, leveraging an auxiliary analysis to either produce new inputs or modify the program. It is essential that this (sometimes heavyweight) analysis is executed only rarely, as alternating between analysis and fuzzing is costly and reduces fuzzing throughput.

Fuzzing libraries also face the challenge of experiencing low coverage during unguided fuzzing campaigns. Programs often call exported library functions in sequence, building up a complex state in the process. The library functions execute sanity checks and quickly detect an illegal or missing state. These checks make library fuzzing challenging, as the fuzzer is not aware of the dependencies between library functions.

Existing approaches, such as LibFuzzer, require an analyst to prepare a test program that calls the library functions in a valid sequence to build up the necessary state to fuzz complex functions.

Evaluating Fuzzing

In theory, evaluating fuzzing is straightforward: in a given domain, if technique A finds more unique bugs than technique B, then technique A is superior to technique B. In practice,

Rerunning the same experiment with a different random seed may result in vastly different numbers of crashes, discovered bugs, and iterations.

evaluating fuzzing is very difficult due to the randomness of the process and domain specialization (e.g., a fuzzer may only work for a certain type of bug or in a certain environment). Rerunning the same experiment with a different random seed may result in vastly different numbers of crashes, discovered bugs, and iterations. A recent overview of the state of the art⁹ evaluated the common practices of recently published fuzzing techniques. The study's authors, after identifying common benchmarking mistakes when comparing different fuzzers, drew four observations from their findings:

- *Multiple executions:* A single execution is not enough due to the randomness in the fuzzing process. Input mutation relies on randomness to decide, according to the mutation strategy, where to mutate input and what to mutate. In a single run, one mechanism could discover more bugs simply by chance. To evaluate different mechanisms and measure noise, we require multiple trials and statistical tests.

- *Crash triaging:* Heuristics cannot be the only way to measure performance. For example, collecting crashing inputs or even stack bucketing is insufficient to identify unique bugs. Ground truth is needed to disambiguate crashing inputs and correctly count the number of discovered bugs. A benchmark suite with ground truth will help.
- *Seed justification:* The choice of seed must be documented, as different starting seeds provide vastly different starting configurations, and not all techniques cope equally well with different seed characteristics. Some mechanisms require a head start with seeds to execute reasonable functionality, while

others are perfectly fine to start with empty inputs.

- *Reasonable execution time:* Fuzzing campaigns are generally executed over days or weeks. Comparing different mechanisms based on a few hours of execution time is not enough. A realistic evaluation, therefore, must run fuzzing campaigns for at least 24 h.

These recommendations make fuzzing evaluation more complex. Evaluating each mechanism now takes considerable time with experiments running multiple days to get enough statistical data for a fair and valid comparison. Unfortunately, such a thorough evaluation is required for a true comparison and analysis of factors leading to better fuzzing results.

A Call for Future Work

With the advent of coverage-guided grey-box fuzzing,^{1,2} dynamic testing has seen a renaissance. Many new techniques that improve security testing have appeared. An important advantage of fuzzing is that each reported bug comes with

a witness that enables the deterministic reproduction of the bug. Sanitization, the process of instrumenting code with additional software guards, helps in discovering bugs closer to their source. Overall, security testing remains challenging, especially for libraries or complex code, such as kernels or large software systems. As fuzzers become more domain specific, an interesting challenge will be to make comparisons across different domains (e.g., comparing a grey-box kernel fuzzer for use-after-free vulnerabilities with a black-box protocol fuzzer). Given the significant recent improvements in fuzzing, exciting new results can be expected. Fuzzing will help make our systems more secure by finding bugs during the development of code before they can cause harm during deployment.

Fuzzing is a hot research area with researchers striving to improve input generation, reduce the impact of each execution on performance, better detect security violations, and push fuzzing to new domains, such as kernel fuzzing or hardware fuzzing. These efforts bring excitement to the field. ■

References

1. M. Zalewski, "American fuzzy lop (AFL)," 2013. [Online]. Available: http://lcamtuf.coredump.cx/afl/technical_details.txt
2. R. Swiecki, "Honggfuzz," 2010. [Online]. Available: <https://github.com/google/honggfuzz>
3. K. Serebryany, D. Bruening, A. Potapenko, and D. Vyukov, "AddressSanitizer: A fast address sanity checker," presented at the 2012 USENIX Annual Technical Conference, Boston, MA. [Online]. Available: <https://www.usenix.org/conference/atc12/technical-sessions/presentation/serebryany>
4. Y. Jeon, P. Biswas, S. A. Carr, B. Lee, and M. Payer, "HexType: Efficient detection of type confusion errors for C++," in *Proc. 2017 ACM SIGSAC Conf. Computer and Communications Security*, pp. 2373–2387. doi: 10.1145/3133956.3134062.
5. N. Stephens et al., "Driller: Augmenting fuzzing through selective symbolic execution," in *Proc. ISOC Network and Security System Symp.*, 2016. doi: 10.14722/ndss.2016.23368.
6. S. Raway, V. Jain, A. Kumar, L. Cojocar, C. Giuffrida, and H. Bos, "VUzzer: Application-aware evolutionary fuzzing," in *Proc. ISOC Network and Security System Symp.*, 2017. doi: 10.14722/ndss.2017.23404.
7. H. Peng, Y. Shoshitaishvili, and M. Payer, "T-Fuzz: Fuzzing by program transformation," in *Proc. 2018 IEEE Symp. Security and Privacy*. doi: 10.1109/SP.2018.00056.
8. I. Yun, S. Lee, M. Xu, Y. Jang, and T. Kim, "QSYM: A practical concolic execution engine tailored for hybrid fuzzing," presented at the 27th USENIX Security Symp., Baltimore, MD, 2018.
9. G. Klees, A. Ruef, B. Cooper, S. Wei, and M. Hicks, "Evaluating fuzz testing," in *Proc. ACM Conf. Computer and Communications Security (CCS)*, 2018. doi: 10.1145/3243734.3243804.

Mathias Payer is a security researcher and an assistant professor at the EPFL School of Computer and Communication Sciences, leading the HexHive group. His research focuses on protecting applications in the presence of vulnerabilities, with a focus on memory corruption and type violations. Contact him at mathias.payer@nebelwelt.net.

SHARE AND MANAGE YOUR RESEARCH DATA

IEEE DataPort is an accessible online platform that enables researchers to easily share, access, and manage datasets in one trusted location. The platform accepts all types of datasets, up to 2TB, and dataset uploads are currently free of charge.

 Open Access Options	 Generates Citations	 2 TB of Cloud Storage	 Links to Manuscripts
 Reproducible Research	 ORCID Integration	 Hosts Data Competitions	 DOI Provided

IEEE DataPort

UPLOAD DATASETS AT IEEE-DATAPORT.ORG

This article originally appeared in IEEE Security & Privacy, vol. 17, no. 1, 2019.



PURPOSE: The IEEE Computer Society is the world's largest association of computing professionals and is the leading provider of technical information in the field.

MEMBERSHIP: Members receive the monthly magazine *Computer*, discounts, and opportunities to serve (all activities are led by volunteer members). Membership is open to all IEEE members, affiliate society members, and others interested in the computer field.

COMPUTER SOCIETY WEBSITE: www.computer.org

OMBUDSMAN: Direct unresolved complaints to ombudsman@computer.org.

CHAPTERS: Regular and student chapters worldwide provide the opportunity to interact with colleagues, hear technical experts, and serve the local professional community.

AVAILABLE INFORMATION: To check membership status, report an address change, or obtain more information on any of the following, email Customer Service at help@computer.org or call +1 714 821 8380 (international) or our toll-free number, +1 800 272 6657 (US):

- Membership applications
- Publications catalog
- Draft standards and order forms
- Technical committee list
- Technical committee application
- Chapter start-up procedures
- Student scholarship information
- Volunteer leaders/staff directory
- IEEE senior member grade application (requires 10 years practice and significant performance in five of those 10)

PUBLICATIONS AND ACTIVITIES

Computer: The flagship publication of the IEEE Computer Society, *Computer* publishes peer-reviewed technical content that covers all aspects of computer science, computer engineering, technology, and applications.

Periodicals: The society publishes 12 magazines, 15 transactions, and two letters. Refer to membership application or request information as noted above.

Conference Proceedings & Books: Conference Publishing Services publishes more than 275 titles every year.

Standards Working Groups: More than 150 groups produce IEEE standards used throughout the world.

Technical Committees: TCs provide professional interaction in more than 30 technical areas and directly influence computer engineering conferences and publications.

Conferences/Education: The society holds about 200 conferences each year and sponsors many educational activities, including computing science accreditation.

Certifications: The society offers three software developer credentials. For more information, visit www.computer.org/certification.

2019 BOARD OF GOVERNORS MEETINGS

(TBD) October: Teleconference

EXECUTIVE COMMITTEE

President: Cecilia Metra

President-Elect: Leila De Floriani

Past President: Hironori Kasahara

First VP: Forrest Shull; **Second VP:** Avi Mendelson;

Secretary: David Lomet; **Treasurer:** Dimitrios Serpanos;

VP, Member & Geographic Activities: Yervant Zorian;

VP, Professional & Educational Activities: Kunio Uchiyama;

VP, Publications: Fabrizio Lombardi; **VP, Standards Activities:**

Riccardo Mariani; **VP, Technical & Conference Activities:** William D. Gropp

2018–2019 IEEE Division V Director: John W. Walz

2019 IEEE Division V Director Elect: Thomas M. Conte

2019–2020 IEEE Division VIII Director: Elizabeth L. Burd

BOARD OF GOVERNORS

Term Expiring 2019: Saurabh Bagchi, Gregory T. Byrd, David S. Ebert, Jill I. Gostin, William Gropp, Sumi Helal

Term Expiring 2020: Andy T. Chen, John D. Johnson, Sy-Yen Kuo, David Lomet, Dimitrios Serpanos, Hayato Yamana

Term Expiring 2021: M. Brian Blake, Fred Douglass, Carlos E. Jimenez-Gomez, Ramalatha Marimuthu, Erik Jan Marinissen, Kunio Uchiyama

EXECUTIVE STAFF

Executive Director: Melissa A. Russell

Director, Governance & Associate Executive Director: Anne Marie Kelly

Director, Finance & Accounting: Sunny Hwang

Director, Information Technology & Services: Sumit Kacker

Director, Marketing & Sales: Michelle Tubb

Director, Membership Development: Eric Berkowitz

COMPUTER SOCIETY OFFICES

Washington, D.C.: 2001 L St., Ste. 700, Washington, D.C.

20036-4928; **Phone:** +1 202 371 0101; **Fax:** +1 202 728 9614;

Email: hq.ofc@computer.org

Los Alamitos: 10662 Los Vaqueros Cir., Los Alamitos, CA 90720;

Phone: +1 714 821 8380; **Email:** help@computer.org

Asia/Pacific: Watanabe Building, 1-4-2 Minami-Aoyama,

Minato-ku, Tokyo 107-0062, Japan; **Phone:** +81 3 3408 3118;

Fax: +81 3 3408 3553; **Email:** tokyo.ofc@computer.org

MEMBERSHIP & PUBLICATION ORDERS

Phone: +1 800 678 4333; Fax: +1 714 821 4641;

Email: help@computer.org

IEEE BOARD OF DIRECTORS

President & CEO: Jose M.D. Moura

President-Elect: Toshio Fukuda

Past President: James A. Jefferies

Secretary: Kathleen Kramer

Treasurer: Joseph V. Lillie

Director & President, IEEE-USA: Thomas M. Coughlin

Director & President, Standards Association: Robert S. Fish

Director & VP, Educational Activities: Witold M. Kinsner

Director & VP, Membership and Geographic Activities:

Francis B. Grosz, Jr.

Director & VP, Publication Services & Products: Hulya Kirkici

Director & VP, Technical Activities: K.J. Ray Liu



Penetration Testing in the IoT Age

Chung-Kuan Chen, Zhi-Kai Zhang, Shan-Hsin Lee, and Shiuhyng Shieh,
National Chiao Tung University

Internet of Things (IoT) objects offer new services but also pose new security threats. Due to the heterogeneity, large number, and resource constraints of these objects, new penetration testing tools and techniques are needed to complement defensive mechanisms.

Internet of Things (IoT) devices and services are now integral to most daily activities. However, the IoT brings not only added convenience but, by connecting more and more objects to the Internet, new security threats.¹ Many applications in IoT ecosystems, from smart homes to customized healthcare, contain sensitive personal information that can become the targets of network attacks.

Unfortunately, ensuring the security of IoT objects is not straightforward for three major reasons. First, the IoT's heterogeneous nature makes it vulnerable to many kinds of attacks. Second, heavyweight protection mechanisms are infeasible for resource-constrained IoT devices. Third, many IoT objects are deployed only once and thereafter are rarely maintained or updated.

PENETRATION TESTING

Due to these challenges, penetration testing (PT), which

employs offensive attack techniques to discover vulnerabilities, is often used to complement defensive security methods before IoT objects are deployed. Because malicious attacks need only a single exploit to be successful, improving PT coverage is crucial. To enhance manual PT, security researchers use automated tools to carry out three types of specialized PT: interface testing, trans-

portation testing, and system testing.

Interface testing targets interfaces that interact with external users or devices. Major vulnerabilities can exist in an application if its input validation mechanisms are not in effect. In the Open Web Application Security Project (OWASP) tester guidelines for IoT applications (www.owasp.org/index.php/IoT_Testing_Guides), the categories “insecure web interface” and “insecure network services,” among others, would be addressed by interface testing.

Transportation testing focuses on misuse issues and design flaws in communication protocols and weak cryptographic schemes. In the OWASP guidelines, “insufficient authentication/authorization,” “lack of transport encryption/integrity verification,” and “privacy concerns” fall into this type of testing.

System testing examines firmware, OSs, and system services for implementation flaws, insecure system settings,



and other known vulnerabilities. In the OWASP guidelines, “insufficient security configurability” and “insecure software/firmware” are relevant for system testing.

To cope with the heterogeneity and large quantity of IoT objects, we propose modularization of test modules to scale up all three types of testing. At the same time, due to IoT devices’ resource limitations, intelligent approaches are desirable for generating test plans based on available test modules to reduce wasted resources and redundant effort while extending test coverage.

INTERFACE TESTING

Many user-facing IoT objects have web-based interfaces, and these can have various vulnerabilities. Among the most common is input validation failure. Unlike traditional web interfaces, which are linked to operations closely coupled with data manipulation, IoT object interfaces can also be linked to code-oriented operations such as controlling system programs. Code-oriented attacks such as command injection and code injection could be even more severe than data-oriented attacks. Improving input validation testing is thus critical for the IoT. Although testing web-based interfaces is our focus, the same modularization and intelligence mechanisms described below can be applied to other types of IoT applications.

Modularized design

Testers employ various techniques for different input validation vulnerabilities. However, these methods are conceptually similar in that they all crawl to the entry points and submit the test payload. Modularizing interface testing would make it easier to create testing tools for specific vulnerabilities and install them on demand. Moreover, algorithms could be developed in

a more systematic way—for example, to implement adaptive, prioritized, or automutation test strategies.

Intelligent payload mutation

Because IoT objects can lack comprehensive input validation mechanisms, extending the coverage of test payloads is desirable. A widely used method, fuzz testing, employs randomly generated payloads, but this is inefficient due to resources wasted on meaningless inputs. An alternative is to exhaustively or randomly generate syntax-correct inputs. This method provides better test coverage but is still inefficient, as the space of syntax-correct inputs is usually large.

environment with numerous network services is difficult and time-consuming. Because service entry points can be dynamically generated, the links between them can be complex, and loops might be produced across IoT objects. In addition, a dispatcher might be built into an IoT application to manage entry points. As the dispatcher can be in either a centralized or distributed structure, a crawler should be able to discover as many entry points as possible in both types of structures to locate more test targets. A proof-of-concept vulnerability scanner that does this, VulScan,² has been developed to complement manual PT.

Modularizing interface testing would make it easier to create testing tools for specific vulnerabilities and install them on demand.

Intelligently mutating known payloads is a compromise between manual testing and exhaustive/random testing. Combining existing evasion techniques provides greater ability to circumvent validation mechanisms. In this case, conflicting or overlapping techniques should be manipulated carefully to prune unnecessary test cases.² On the other hand, converting payloads to syntactically or semantically equivalent payloads is worthy of further investigation. Syntactic mutation generates payloads with slight changes at the syntax level. For example, SQL code “`or 1 = 1`” can be mutated to “`' || 1 = 1`”. Semantic mutation converts the whole payload to functional equivalent ones. For instance, “`id = 1 or 1`” is semantically equivalent to “`id = id xor 0`”.

Intelligent entry-point crawling

Entry-point discovery in an IoT

TRANSPORTATION TESTING

Transportation testing is performed both on the network infrastructure interconnecting IoT objects as well as the associated cryptographic schemes and communication protocols used to protect messages.

New network infrastructures

Messages between IoT objects traverse heterogeneous networks such as TCP/IP, Zigbee, and 6LoWPAN. To allow more efficient object communication, new infrastructures such as FIA (www.nets-fia.net), HUB4NGI (www.hub4ngi.eu), and PNS³ have been proposed. New PT tools are needed to test these infrastructures, the protocols, and the gateways or converters between the infrastructures and protocols. Because network heterogeneity is a key issue in IoT communication, transportation testing should be modularized to provide better flexibility.

Cryptographic issues

In general, the cryptographic algorithms that protect network communication are believed to be secure due to theoretical proofs. When vulnerabilities are discovered, they're generally attributable to misuse, implementation failures, and bad protocol design. However, resource-constrained IoT objects can't afford heavyweight cryptographic mechanisms. Moreover, messages between devices usually are well formatted and lack entropy. The combination of these factors could make differential cryptanalysis or statistical attacks possible.

Trusted platform modules (TPMs) enable new applications but also raise new threats. For example, the ROCA vulnerability⁴ is caused by a weak

In conventional computing environments the x86/x64 instruction-set architecture (ISA) dominates, but other ISAs such as ARM, MIPS, and PPC are also used in the IoT. OSs vary among IoT objects as well, with general-purpose OSs such as Linux, Windows, and Android often customized. The diversity of IoT objects makes automated reverse-engineering challenging.

Encapsulation

To mitigate the impacts of system diversity, encapsulation can enable cross-platform analysis. Encapsulation involves using an abstract language such as LLVM (<http://llvm.org>) or VEX (<http://valgrind.org>) to create an intermediate representation (IR) of different machine languages to emulate

infeasible. An alternative approach is virtual machine introspection (VMI), which monitors VM execution in the hypervisor outside the VM.^{6,7} Because VMI doesn't modify the guest OS, IoT objects are easier to deploy. Through VMI, the emulator's out-of-box monitoring, memory forensics, and debugging features can be developed more easily to enable both manual and automatic PT.

Intelligent grey-box testing

As the boundary of grey-box testing is more obscure than white- and black-box testing, a systematic division of testing phases enables the development of future testing techniques. Intelligent grey-box PT can be divided into four phases: vulnerability model construction, execution path exploration, vulnerability path searching, and vulnerability path verification. To discover vulnerabilities, the model of abnormal behaviors is first constructed. Next, control flows are analyzed to find each execution path. The vulnerability risk for each path is then estimated using information from the IR and VMI to prioritize testing order. Once the path with highest risk is identified, the symbolic execution resolves inputs to the path. During the final phase, if the resolved input is available, a verifier can monitor the program with the input to check whether the vulnerability model can be satisfied. Using this systematic approach, intelligent grey-box PT can discover system-level vulnerabilities.

To mitigate the impacts of system diversity, encapsulation can enable cross-platform analysis.

prime-number generator in the RSA library within TPMs. This vulnerability affects many vendors including Microsoft, Google, and HP. Another example is KRACK attacks,⁵ which exploit a flaw in Wi-Fi's WPA2 encryption and affects all major software platforms. As cryptographic operations are rarely computed in cleartext, developing PT methods to discover such vulnerabilities in the IoT is challenging.

SYSTEM TESTING

In contrast to interface testing, which focuses on commonly used technologies such as web interfaces, proprietary programs are the main targets of system testing. Without having knowledge of such systems, testers often resort to black-box methods, such as fuzz testing. Given the large number of IoT objects to be tested, exhausting all test cases is infeasible. It's therefore helpful generating test cases through automatic reverse-engineering, what is termed grey-box PT.

ISAs. Hardware-assisted emulators are used to test programs running on specific ISAs, but software-based emulators such as QEMU (www.qemu.org) and Bochs (<http://bochs.sourceforge.net>) can leverage multiple ISAs and are more suitable for IoT objects. Another method for building an IR is symbolic execution, which translates a program to mathematical constraints and evaluates whether certain properties can be satisfied. With these constraints, developing an intelligent PT method with a more formalized foundation is possible.

Virtual machine introspection

While symbolic execution mostly deals with per-process information, system-wide runtime information is also important for PT. However, runtime analysis tools might not be available for IoT objects. Due to resource constraints, object diversity, and proprietary architectures, developing debugging and analysis tools for different objects is usually

To cope with the heterogeneity, large number, and resource constraints of IoT objects, PT tools and techniques should apply the principles of modularization and intelligence. Modularization provides the flexibility to test various targets, and intelligence enlarges test coverage and improves accuracy. In interface testing, input validation mechanisms should be tested using an intelligent mutation engine and entry-point

discovery automated. Transportation testing must address the problem of messages between IoT objects traversing heterogeneous networks. To deal with emerging IoT network infrastructures, PT tools should be compatible with the overlay networks. Cryptographic misuse issues and implementation flaws must also be considered. In system testing, the challenge is IoT objects with various ISAs and OSs. If encapsulation and related translation modules are available, cross-platform analysis becomes feasible. VMI and symbolic execution can be applied on top of encapsulation. In this way, intelligent analysis methods can be used to discover vulnerabilities in variant platforms.

REFERENCES

1. Z.-K. Zhang et al., "IoT Security: Ongoing Challenges and Research Opportunities," *Proc. IEEE 7th Int'l Conf. Service-Oriented Computing and Applications (SOCA 14)*, 2014, pp. 230-234.
2. H.-C. Huang et al., "Web Application

Security: Threats, Countermeasures, and Pitfalls," *Computer*, vol. 50, no. 6, 2017, pp. 81-85.

3. Z.-K. Zhang et al., "Identifying and Authenticating IoT Objects in a Natural Context," *Computer*, vol. 48, no. 8, 2015, pp. 81-83.
4. M. Nemeč et al., "The Return of Coppersmith's Attack: Practical Factorization of Widely Used RSA Moduli," *Proc. 2017 ACM SIGSAC Conf. Computer and Communications Security (CCS 17)*, 2017, pp. 1631-1648.
5. M. Vanhoef and F. Piessens, "Key Reinstallation Attacks: Forcing Nonce Reuse in WPA2," *Proc. 2017 ACM SIGSAC Conf. Computer and Communications Security (CCS 17)*, 2017, pp. 1313-1328.
6. K. Nance, M. Bishop, and B. Hay, "Virtual Machine Introspection: Observation or Interference?," *IEEE Security & Privacy*, vol. 6, no. 5, 2008, pp. 32-37.
7. C.-W. Wang et al., "Cloudebug: A Programmable Online Malware Testbed," *Computer*, vol. 47, no. 7, 2014, pp. 90-92.

CHUNG-KUAN CHEN is a PhD candidate in the Department of Computer Science at National Chiao Tung University (NCTU). Contact him at ckchen@cs.nctu.edu.tw.

ZHI-KAI ZHANG is a PhD candidate in the Department of Computer Science at NCTU. Contact him at skyzhang.cs99g@g2.nctu.edu.tw.

SHAN-HSIN LEE is a PhD student in the Department of Computer Science at NCTU. Contact him at shlee.cs06g@nctu.edu.tw.

SHIUHPYNG WINSTON SHIEH is a university chair professor and past chair of the Department of Computer Science at NCTU. Contact him at ssp@cs.nctu.edu.tw.

This article originally appeared in Computer, vol. 51, no. 4, 2018.

IEEE
SECURITY & PRIVACY

IEEE *Security & Privacy* magazine provides articles with both a practical and research bent by the top thinkers in the field.

- ✓ Stay current on the latest security tools and theories and gain invaluable practical and research knowledge,
- ✓ Learn more about the latest techniques and cutting-edge technology, and
- ✓ Discover case studies, tutorials, columns, and in-depth interviews and podcasts for the information security industry.

IoT Safety: State of the Art

Janusz Zalewski

Florida Gulf Coast University

Editors: **Phillip A. Laplante**, Penn State (plaplante@psu.edu), **Ben Amaba**, IBM (baamaba@is.ibm.com)

■ **AMONG MULTIPLE ISSUES** or concerns with respect to the Internet of Things (IoT), it is definitely security that is given the most attention. Users, developers, managers, and other stakeholders are concerned that the heterogeneity and complexity of this technology, essentially composed of systems of systems, may open the door to security breaches on an unprecedented scale. There is, however, another important system property, which is not very often brought up as an imminent concern, but is equally important. This is device safety, the violation of which may cause severe harm to the environment in which the device operates.

While normally, in the use of technology, concerns are placed on actions the device takes to accomplish its desired functions accurately, as specified, safety is a concern related to ensuring that the device causes no damage or harm. Just like with a regular TV set, which must meet specific regulations that prevent it from catching fire, there are multiple instances of IoT applications, where the environment can suffer due to unintended misbehavior of the device. Examples of Internet-enabled devices that may lead to such consequences include wearable medical device causing harm to a patient, a vehicle causing an accident due to the malfunction of software, a thermostat in a smart home causing overheating, etc.

Digital Object Identifier 10.1109/MITP.2018.2883858

Date of current version 26 February 2019.

BASIC NOTIONS

Technically, in computing systems, safety as a system property is strictly related to security. They are complementary. While safety relates to ensuring that the computing device does not cause harm to the environment, security relates to ensuring that the computing device is not hurt from the environment. This is illustrated in Figure 1, where a smart device may impact the environment negatively due to malfunctioning, which is a safety concern, but may also be affected negatively and experience harm due to the environment acting maliciously, which is a security concern. The environment is understood here very broadly and means everything with which the computing system or smart device may interface but is normally limited to one or more of the following: network, a human operator, a database, or a plant (technical object with which it interacts).

In practice, IoT security violations occur as breaches due to threats (attacks) exploiting vulnerabilities in hardware or software. IoT safety violations, on the other hand, usually occur as a result of computer failures due to hardware or software faults activated by hazards. The terminology for safety and security has a lot of parallel concepts and mutually matching terms,¹ which are illustrated in Table 1. In addition, the two properties can hardly be viewed in isolation, because poor security can have negative impact on safety, and *vice versa*, safety violations may negatively affect security.

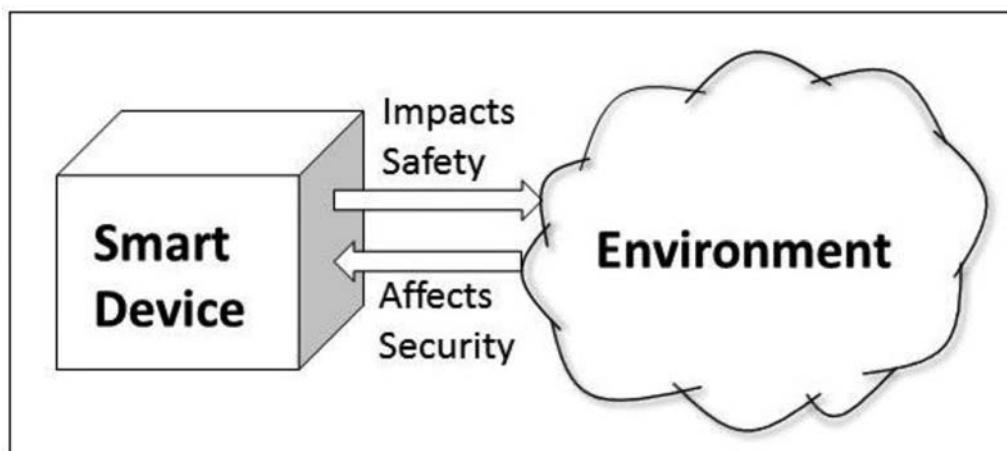


Figure 1. Analogy between safety and security properties.

Referring to IoT safety, one can list a number of industries, where IoT device safety is crucial, but rarely is a different perspective brought up: how a cross section of our society is influenced by safety concerns, which involves the following (see Figure 2):

- individual level, with wearables, such as a smart watch, or medical devices, such as pacemakers or insulin pumps;
- households, with smart homes, where properly operating home appliances, air conditioners, door locks, animal feeders, and others make up crucial components of us feeling safe;
- smart cities, where street vehicles, traffic navigation systems, parking spaces, building automation, power and energy distribution may all adversely impact safety, if improperly used or controlled as a part of IoT;

Table 1. Illustration of a dualism between safety and security concepts.

Security			Safety		
Concept	Definition	Consequences	Concept	Definition	Consequences
Threat	Any circumstance or event with the potential to adversely impact an information system through unauthorized access, destruction, disclosure, modification of data, and/or denial of service. [CNSS]	Exploits vulnerabilities	Hazard	Intrinsic property or condition that has the potential to cause harm or damage. [SSEV]	Activates a fault
Vulnerability	Weakness in an information system, system security procedures, internal controls, or implementation that could be exploited. [CNSS]	Results in a breach	Fault	Manifestation of an error in software. [SSEV]	Results in a failure
Breach	An event in which a system or system component is compromised, so its required functions within specified limits are impaired. [Author]	Leads to losses	Failure	Termination of the ability of a system to perform a required function or its inability to perform within previously specified limits. [SSEV]	Leads to harm or damage

SSEV – Software and Systems Engineering Vocabulary – <http://computer.org/sevocab>

CNSS – Committee on National Security Systems Glossary

– <https://rmf.org/wp-content/uploads/2017/10/CNSSI-4009.pdf>

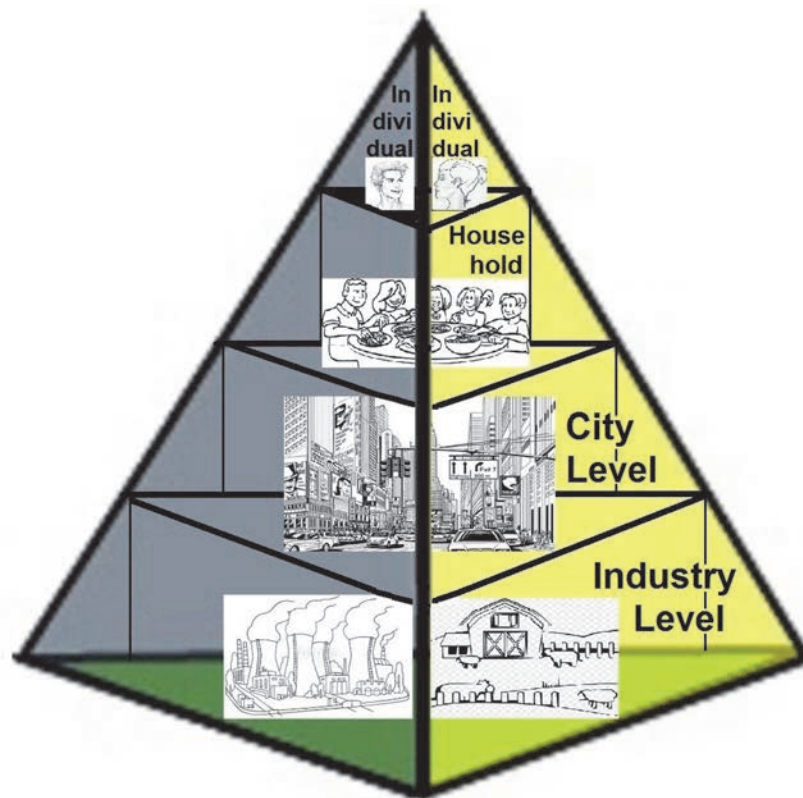


Figure 2. Layered perspective on IoT safety concerns.

- industrial world, in which factories, agricultural fields, critical infrastructure (water and waste processing plants, gas and oil pipelines) make us all vulnerable if unsafe.

Each upper layer is immersed into the lower layers, forming a consistent view of IoT safety concerns.

As viewed from Figure 2, computer and software safety is of paramount importance at all layers and has been addressed by various industries before, resulting in multiple guidelines, both industry specific,² and general international standards.³ With the emergence of IoT, however, the situation changes drastically, because these standards may no longer be 100% applicable. They have been developed mostly for a single layer, say, industrial operation, rarely taking into account cross passing to other layers. Nowadays, because IoT has emerged, there are other “players” in the picture, which have to be accounted for in each standard.

Overall, due to the massive amount of applications that are Internet-enabled, unintended connections arise, new uses emerge, and vendors are losing control over how their products are utilized and may not even anticipate some of the purposes the devices serve. This situation, if not properly addressed, may have a dramatic impact on the entire ecosystem and endanger life safety and consumer safety, leading to sustainability concerns. Therefore, the issue of developing safety standards for various industries to include IoT has to be revisited.

STATE OF THE ART

There is an advocacy for IoT safety from professional communities. Cerf *et al.*, discussing⁴ the shared governance of the Internet and the IoT, state that “user’s safety must be the first priority for all hardware and software providers” and focus the discussion on “how to address safety issues that become much more prominent with the spread of Internet-enabled physical

Table 2. Sample structure of the proposed IoT Safety Case (based on IPA).⁹

IoT safety case #001		
Name [optional]		Automotive vehicles (cars) and homes
Description	Scenario	The driver of a car uses the on-board voice-operated system [...] to control the lighting, thermostat, and security systems in the home, such as opening/closing the garage door, turning on/off the lighting of the home's entrance, and turning on/off the home security system. The person relaxing on a couch at home uses the home's cloud-based voice recognition service to start/stop the car's engine, lock/unlock the car's door, and check its fuel gauge.
	Negative consequences	See Reference
	Potential countermeasures	See Reference
References	See ⁹	

environments.” During the IEEE Experts Technology and Policy Forum,⁵ Robert Martin of Mitre said “We need to make sure we don’t fall prey to calling this end-to-end security, when really we want to talk about end-to-end security and safety.” Further: “For the IoT, safety needs to be considered along with privacy, the performance issues, reliability, resilience, and, of course, the security of these systems.”

Baba *et al.* present briefly a similar view in their report,⁶ published by the Internet Engineering Task Force: “Recognition of the importance of information security has grown in step with the rising use of the Internet. Closer examination reveals that the IoT era may see a new direct physical threat to users. [...]. These kinds of scenarios may occur without identity fraud, hacking, and other means of compromising information security. Therefore, [...] this issue shall be referred to as “IoT Safety” to distinguish it from Information Security.”

Partially in response to such concerns, the Object Management Group (OMG) defined⁷ Safety-Sensitive Consumer Device (SSCD) as a category of industrial products used by consumer users, including automobiles, service robots, medical devices and clinical systems, and smart houses. They argue the following: “Taking the future of electronics systems into consideration, each electronics system is going to be one of the terminals of IoT and will be expected to play a significant role as a part of smart city. This consideration indicates that the safety of electronics systems cannot be achieved alone, but have to be achieved together with other electrical and

electronic systems as a whole.” Consequently, the OMG document defines the Dependability Assurance Framework (DAF) as a standard specification to assess and assure the dependability of SSCD’s. The DAF approach and their document look very promising on its face, but the life will show how practical is this to apply.

At the government level, the Consumer Product Safety Commission (CPSC) issued in March 2018 a Notice of Public Hearing and Request for Written Comments on The IoT on Consumer Product Hazards, to which a wide response has been received. The Center for Democracy and Technology (CDT) provided an extensive comment,⁸ in which it outlined five Case Studies on IoT Safety, including smart: thermostats, TV’s, lights, locks, and speakers, arguing the following: “As these examples demonstrate, consumer IoT devices have the potential to introduce physical harms that are not present in other products.” Further, the CDT letter recommended five specific actions to be taken to improve IoT product safety.

On the international arena, similar activities are observed. The Japanese Information-technology Promotion Agency (IPA) issued guidelines on IoT safety/security development. The most recent one⁹ acknowledges that “an increasing number of corporations are placing the creation of value through interconnectivity of IoT devices and relevant systems [...] as one of their major business strategies.” Although the approach advocated in the IPA report is a little confusing, since it relies on providing high reliability rather than safety or security, a detailed discussion of risk analysis for five IoT use cases

is very valuable, as it involves, among other things, structural analysis of IoT components, expected threats/damage and main factors or issues, and possible countermeasures. A more extensive list of case studies like this could form a database similar to Common Vulnerabilities and Exposures (CVE) maintained by Mitre,¹⁰ which is a directory of entries—each containing an identification number, a description, and at least one public reference—for publicly known cyber security vulnerabilities. Each entry in this new safety list, tentatively named IoT safety cases, could have the data fields for safety concerns as shown in Table 2.

CONCLUSION

There is no doubt that safety of IoT devices is a critical issue for the society and must be taken into account in the design processes to develop the safe products. It appears that the industry and the government understand the challenges coming with the advent of IoT and are taking steps to improve the design and operational principles in respective processes. This note presented a professional perspective on IoT safety concerns and proposed establishing the IoT safety cases database. Two important issues have not been discussed here, which is risk assessment, including the legal aspects, and required certification.

ACKNOWLEDGMENTS

Work presented in this article has been done in part during the author's summer fellowships at the Air Force Research Lab in Rome, New York.

REFERENCES

1. A. Kornecki and J. Zalewski, "Aviation software: Safety and security," in *Wiley Encyclopedia of Electrical and Electronics Engineering*, J. G. Webster, Ed. New York, NY, USA: Wiley, 2015.

2. DO-178C, Software Considerations in Airborne Systems and Equipment Certification. RTCA SC-205, Jan. 2012.
3. *Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems*. IEC 61508:2010, Int. Electrotechn. Comm., Geneva, Switzerland, 2010.
4. V. G. Cerf *et al.*, "IoT safety and security as shared responsibility," *J. Bus. Inform.*, vol. 1, no. 36, pp. 7–19, 2016.
5. IEEE Experts Technology and Policy (ETAP) Forum on Internet Governance, Cybersecurity and Privacy, Washington, DC, USA, Feb. 5, 2016.
6. H. Baba *et al.*, Problems in and among Industries for the Prompt Realization of IoT and Safety Considerations. Internet Draft, IETF, Nov. 15, 2018. [Online]. Available: <https://tools.ietf.org/id/draft-baba-iot-problems-06.html>
7. Dependability Assurance Framework for Safety-Sensitive Consumer Devices (DAF). Ver. 1.0. Object Management Group, Feb. 2016.
8. J. L. Hall *et al.*, Response to Docket No. CPSC-2018-0007—The Internet of Things and Consumer Product Hazards. A Letter to the CPSC. Jun. 15, 2018.
9. Guidance for Practice Regarding IoT Safety/Security Development Guidelines, Software Reliability Enhancement Center, Information-Technology Promotion Agency, Tokyo, Japan, Dec. 2017.
10. Common Vulnerabilities and Exposures Database. [Online]. Available: <https://cve.mitre.org/>

Janusz Zalewski is a professor of software engineering and computer science at Florida Gulf Coast University. He has extensive experience in the design and development of real-time safety critical systems. His recent research interests include security of cyberphysical systems and IoT. He is currently the secretary of the IEEE P1876 WG Networked Smart Learning Objects for Online Laboratories. Contact him at zalewski@fgcu.edu.

*This article originally appeared in
IT Professional, vol. 21, no. 1, 2019*



Physical Computing's Connected and Shape-Changing Future

Heather M. Patterson, Intel Labs

Technologies have a way of unexpectedly upending established social practices, often at a pace far outstripping the ability of a given society to absorb or process disruptions with awareness. From stirrups to gunpowder to magnets, history teaches that even seemingly mundane objects have the capacity to change everything.¹

Physical computing has become an umbrella term whose practitioners draw theoretical and practical inspiration from research in a cluster of associated research domains, including tangible interactions, human-material interactions, shape-changing interfaces, organic user interfaces, interactive materiality, and material ecology. For many, a common goal is to bridge analog and physical worlds by infusing objects with compute and sense-making capabilities, in some cases enabling changes in a material's physical characteristics. The most intriguing implementations also strip away layers of abstraction and allow people to interact directly with objects with fewer intermediate interfaces, potentially reducing interaction friction, improving legibility, and fulfilling unmet human needs (see the sidebar for a list of potential applications).

The transformation of material objects into intelligent, information-rich ones holds great promise for fruitful partnerships between humans and

technology. However, new opportunities also bring new challenges. It behooves us to examine how these new objects impact humans so that we can thoughtfully engineer systems that are mindful of our social practices—sidestepping potentially troubling outcomes and creating technologies worth having.

NOTEWORTHY TRANSITIONS

At this point in time, several transitions associated with physical computing stand out as being particularly worthy of further scrutiny. I will elaborate on two.

How will we discover what knowledge is embedded in a particular object, as well as the limits of that knowledge?

From Material to Informational

The first concerns the shift from the *material* to the *informational* and presents questions about how interactions between humans, objects, and physical spaces will change over time. From shaping the way that we configure space to guiding the daily flows of activities, our built environments (and the objects within them) organize our lives.² However, in a world

in which complex interactivities will be designed into even the most ordinary “stuff”—from faucets to furniture to walls themselves³—how will we go about determining an information object's utility, intended use, or capacity?⁴ How will we discover what knowledge is embedded in a particular object, as well as the limits of that knowledge? More importantly, how can technological designs be adapted to anticipate and meet the needs of their human users, and not the other way around?

From Contextual to Universal

The second transition concerns an acceleration of merging social contexts brought about by novel data flows that connect and enliven these new informational objects. By selectively embedding objects with computing technologies, we enable new forms of data collection, analysis, and distribution, eroding natural data dams that have, up until now, shaped social conceptions of privacy. What might be the consequences of unexpected data tides, eddies, and floods? If the future is one in which our devices are watching us, listening to us, and even physically reconfiguring themselves to enhance our experiences, are we still able to truly do, feel, be, share, and withhold portions of ourselves at will?

PHYSICAL COMPUTING APPLICATION AREAS

What does the future hold? Virtual, in-air touchpad interfaces that enable real-time sensing of arm, hand, and finger positions could support surgical training by tracking fine hand movements.¹ Deformable interfaces that mimic mechanical properties of anatomical materials, water, and clay could help medical practitioners distinguish between tumor types, or assist geologists in modeling tsunami or earthquake outcomes.² Screens that curve inward, remain flat, or skitter away in response to the presence of authorized (or unauthorized) persons could provide better privacy, security, and personalization.³ Water faucets that narrow their aperture or bend away from users could provide unobtrusive nudges to conserve in periods of higher-than-normal water usage.³ Shape-changing tablets with co-located 3D graphics could simulate wave frequency and wind strength.⁴ Tactile representations of navigable spaces (and more),⁵ such as insoles that buzz to instruct a wearer to change direction, could improve accessibility and correct some setbacks unwittingly imposed upon communities for whom flat screens might be as functionally meaningless as sheets of glass.⁶

REFERENCES

1. T. Okonski, "ZeroTouch: A New Multifinger Sensing Technology," *Texas A&M Computer Science and Engineering Magazine*, 2011, pp. 3–6; <http://ecologylab.net/research/publications/Kerne-Moeller-ZeroTouch-tamuEngNews.pdf>.
2. K. Nakagaki et al., "Materiale: Rendering Dynamic Material Properties in Response to Direct Physical Touch with Shape Changing Interfaces," *Proc. 2016 CHI Conf. Human Factors in Computing Systems (CHI)*, 2016, pp. 2764–2772.
3. A. Roudaut, A. Karnik, and S. Subramanian, "Morpheus: Toward High 'Shape Resolution' in Self-Actuated Flexible Mobile Devices," *Proc. SIGCHI Conf. Human Factors in Computing Systems (CHI)*, 2013, pp. 593–602.
4. D. Lindlbauer et al., "Combining Shape-Changing Interfaces and Spatial Augmented Reality Enables Extended Object Appearance," *Proc. 2016 CHI Conf. Human Factors in Computing Systems (CHI)*, 2016, pp. 791–802.
5. E. Geissler, K. Harnack, and A. Mühlenberend, "Senseole: An Insole-Based Tickle Tactile Interface," *Proc. 10th Int'l Conf. Tangible, Embedded, and Embodied Interaction (TEI)*, 2016, pp. 717–722.
6. L. Gannes, "The Disappearing Interface," *All Things D*, 4 Mar. 2013; <http://allthingsd.com/20130304/the-disappearing-interface>.

INTERACTING WITH INFORMATIONAL OBJECTS

A central goal of design is to make objects and interfaces disappear—to get out of the way and let users achieve their goals. Designer Don Norman, for example, explicitly challenges developers to create tools that “fit the task so well that [they] become part of the task, feeling like a natural extension of the work, a natural extension of the person.”⁵

Over the past decades, an influx of flat-screen technologies have shifted the daily human experience from one in which the vast majority of interaction affordances were readily apparent through objects’ physical properties,

such as cup handles that invite grasping and doorknobs that invite turning, to one in which the execution of even simple tasks requires explicit *signifiers*, or indicators of use.⁵ Flexible and shape-changing interfaces appear to be introducing yet another phase of interactivity, in which direct, intuitive relations between objects and human users will reemerge as a priority, but without explicitly signaling the existence or management of the full set of interaction possibilities.

This presents an interesting challenge. As our environments become rich with “physical interfaces [that] can physically change to accommodate

different users, uses, and contexts,”⁶ are we moving toward a reality that has the appearance of being concrete and unchanging, but that is actually far less perceptible or discoverable?⁷ When presented with new materialities, will we be more able to superficially engage with objects but less able to understand what they really do, what they really know, and who they really talk to? And if so, how should we, as developers, designers, and users, set about managing this complexity?

Although answers to these questions are far from settled, one early intriguing approach was taken by the MIT Media Lab’s theoretical explorations of “radical atoms”—materials that can dynamically change their form and appearance to make information directly manipulable. In this line of research, objects change shape to reflect changes in their underlying computational states, such that affordances change concomitantly and dynamically in order to inform users of these alterations.⁸ [See this issue’s Interview department for a related discussion with Hiroshi Ishii.]

But dynamic affordances raise additional intriguing questions. As each generation has discovered, the introduction of new technologies transforms the things we think about, the things we think with, and the arenas in which we think.¹ How might early learners in future generations come to conceptualize object constancy, time, and even theory of mind when their daily interactions are dominated by objects with readily changeable areas, curvatures, and densities, and in which each form factor comes with an attendant set of computational capabilities customized to particular users,⁹ perhaps appearing to merge with the ambient environment altogether?¹⁰

Intelligence and connectivity also reconfigure connections between objects, our environments, and ourselves, leading to an erosion of boundaries and an accompanying expansion of connected systems. Some predict that we will continue to become immersed in the

so-called “infosphere” as technologies move from being mere enhancements of our bodies (like hammers) to augmenting interfaces between different environments (like washing machines) to re-engineered realities, in which large, distributed, and connected systems envelope us, altering how we use space and how we configure our bodies in relation to one another.¹¹

Philosopher Luciano Floridi argues, for example, that conflating the material/physical with the informational reshapes our relationship with our physical environments and even our own informational identities, leading to new notions of *ourselves* as informational objects that collect experiences, keep memories, and transmit curated, reshaped historical narratives of our own lives. In Floridi’s view, future generations will be doubly cursed: Forced to acquire unwanted characteristics and disallowed the possibility of forgetting or reinvention, they will nonetheless feel “deprived, excluded, handicapped, or poor to the point of paralysis and psychological trauma...like fish out of water”¹¹ if ever disconnected from the infosphere.

UNDERSTANDING ERODING BOUNDARIES

A second important consequence of fusing physical and computational environments is the new prospect of collecting, analyzing, and distributing vast amounts of personal data that support inferences about people’s habits, preferences, lifestyles, and social affiliations. As the engineering accomplishments of physical computing advance, so does the importance of understanding how products, data flows, and policies can be architected to respect user privacy and maximize fairness. Although individuals might be eager to sample new technologies, appearing on the surface to forsake privacy for convenience or utility, research suggests that a host of doubts lurk beneath: “Why is this object here with me, and whose interests is it serving—mine, the company who made it, or the providers

of the delivered services? What are the implications of these new information flows for me, my family, and society? Should I participate in this brave new world?”

Analytic philosopher scholar Helen Nissenbaum has argued that people do not care about having complete control over information about themselves. Rather, what they care about is that information is shared *appropriately*.¹² Nissenbaum’s privacy framework of contextual integrity provides a process for determining the appropriateness of new information flows by reference to the ends and values embedded within a particular social context, such as home life, employment, or medical care. An indiscretion confessed to a trusted family member might be considered “too much information” for one’s work colleagues; health data willingly shared with a medical professional bound by the Hippocratic Oath might be considered “off limits” to a commercial wellness app; location data given to a navigation service in exchange for route planning might take on a new meaning when it is later shared with law enforcement. As Nissenbaum explains, “when actions or practices violate entrenched informational norms, they provoke protest, indignation, or resistance. When actions or practices are in compliance, they respect contextual integrity.”¹² From this perspective, determining whether and how new information flows violate privacy requires an assessment of whether and how they violate or enhance current contextual social norms.¹³

However, a central challenge to privacy today is that physical computing is hastening the blurring of previously well-established contextual boundaries. To take but one example, consider so-called “aging-in-place” devices and services. Embedding sensors in common household objects such as refrigerators, door locks, and even mattresses has the laudable aim of helping the elderly maintain independence and forestall moves to assisted living facilities. But recent analyses make clear that this

implementation of physical computing has created a new ecosystem most accurately conceptualized as, among other things, a hybrid of home life and clinical medical care, each context of which has radically different norms of appropriate information sharing.¹³ In one, information might facilitate a sense of camaraderie with family and friends. In another, it might present thorny issues regarding compliance, adherence, or physical safety, threatening the cost or availability of insurance. Similarly, smart home thermostats, lighting, and water-monitoring services are caught in a liminal space between norms of data sharing in the service of energy-saving, and norms that keep home life free from potentially meddlesome outside parties.¹⁴

This transition toward making private activities more easily seen, tracked, and potentially controlled by others matters for several reasons, not least because in technology, as with many domains, “the benefits and deficits are not distributed equally,”¹ and it is not always clear who will “win” and who will lose.¹ Clear and visible accountability, as well as signals of loyalty and discretion, will become particularly important as computation becomes further embedded in structural elements in our environments, including furniture, walls, and other infrastructural components that are able to observe, react, and always remember.

RECOMMENDATIONS

We are still in the earliest days of determining how new systems should behave when facing uncertain circumstances. In practice, which positive steps must be taken to maximize human value and prevent or minimize harm? There are no easy answers, but for now, I suggest that we think aspirationally. What would a “good” future world look like? How can physical computing systems be designed to bring that future closer?

To start, we would be wise to optimize for coordination, discoverability, and understanding.³ *Coordinating* with

humans means complementing, rather than duplicating, our strengths and creating objects that understand and work with human mental models, rather than imposing their own. *Discoverability* and *understanding* challenge us to create the conditions under which humans can easily grasp core system functionalities and comprehend its basic operations and limitations. This entails also thinking carefully about what types of feedback are informative and actionable without being too irritating or intrusive, too often.

From a privacy and information flow perspective, best practices suggest designing privacy features into systems at the outset, rather than attempting to tack them on at the end of the development process.¹⁵ In practice, this means carefully exploring the aims, values, and ends of the social context in which a particular technology or set of technologies will be implemented and creating an integrated set of information flow settings that enhance these values. If contextual boundaries are blurry, the challenge becomes determining how to design for the optimum degree of transparency and granular control: when, how, and how frequently should users be offered information about what information is being collected about them, how it is being processed, stored, and distributed? What types of simple tools would allow them to selectively dismiss or reject unwanted information flows without losing access to core services?

In this vein, we might look beyond task objectives and toward larger social goals. Luciano Floridi introduces the term “infraethics,” which he defines as “the design of environments that can facilitate ethical choices, actions, or processes.”¹¹ Different from Ethics by Design, which privileges a set of behaviors pre-determined by a designer to meet particular ethical standards, a “pro-ethical” design privileges user reflection.

For example, rather than designing a system with defaults and allowing a person to opt in or opt out (which

carries different social implications in the cases of say, information collection vs. organ donation vs donating grocery bag fees to charity), a pro-ethical system would leave choices open and ask a user to make a decision before he or she can proceed with a transaction. In this approach, the challenge is to first create the reflection infrastructure (infraethics) and then approach the contents (ethics) itself, considering how to present the implications of various alternative choices while being mindful of the nudges that design imposes upon human users.

FUTURE APPLICATIONS: BUILDING WITH SELF-AWARENESS

Looking ahead, much interesting work will be happening in a variety of production environments, where a shift from instruction-based to behavior-based fabrication offers intriguing possibilities for industry. Sometimes referred to as Industry 4.0 or the Fourth Industrial Revolution, a merger of material synthesis and connected computing is ushering in an ecosystem in which machines embedded with sensors and actuators are predicted to substitute real-time physical sensing for predictive modeling. No longer limited to executing predetermined tasks, robotic entities will be able to sense, learn, and create adaptively, reconfiguring themselves to new environments.¹⁶

In the architectural and design domain, scholars such as Achim Menges have asked, “what happens if the production machine no longer remains just the obedient executor of predetermined instructions, but begins to have the capacity to sense, react and act; in other words, to become self-aware?”¹⁷ A large-scale adoption of machines that “self-predict, self-configure, and self-organize”¹⁷ could have enormous economic upsides for manufacturers that benefit from dynamically reconfigured and streamlined workflows. Data from supply chains and production lines will enable factories to keep track of their own (and each other’s) production and

maintenance needs, adjust performance in real time to meet fluctuating targets, and even reconfigure themselves, taking into consideration the state of the system as a whole. Digitization of manufacturing technologies will enable greater individualization, bringing new opportunities in fields such as automotive design.¹⁸

This transition will also implicate system security, reliability, and, no less pressing, employment prospects for the humans whose professional expertise in operating complex machinery may be subjected to new demands made for, and perhaps by, replacements of the objects they once controlled. As machines begin to break free from deterministic instructions and train themselves, we would be wise to ensure that they receive human guidance along the way. Whether for purposes of planning, operations, or maintenance and repair, human input is critical for developing procedures for assigning control, accountability, and liability when systems fail.

Let us remember that even the most promising new technologies may reinforce social inequalities and create new forms of disadvantage for which we are ill-equipped to deal. Developing a stronger sense of what really matters to people—politically, economically, and culturally—will set us on a path toward creating new technologies that are not merely transformative but also responsive to human needs.¹⁹ ■

REFERENCES

1. N. Postman, *Technopoly: The Surrender of Culture to Technology*, Knopf, 1992.
2. M. McCullough, *Digital Ground*, MIT Press, 2004.
3. K. Nakagaki et al., “Materiable: Rendering Dynamic Material Properties in Response to Direct Physical Touch with Shape Changing Interfaces,” *Proc. 2016 CHI Conf. Human Factors in Computing Systems (CHI)*, 2016, pp. 2764–2772.
4. D.A. Norman, *The Design of Everyday Things*, Basic Books, 2013.

5. D.A. Norman, *The Invisible Computer: Why Good Products Can Fail, the Personal Computer Is So Complex, and Information Appliances Are the Solution*, MIT Press, 1998.
6. M. Coelho and J. Zigelbaum, "Shape-Changing Interfaces," *J. Personal and Ubiquitous Computing*, vol. 15, no. 2, 2011, pp. 161–173.
7. J.J. Gibson, "The Theory of Affordances," *Perceiving, Acting, and Knowing: Toward an Ecological Psychology*, Wiley, 1977.
8. H. Ishii et al., "Radical Atoms: Beyond Tangible Bits, Towards Transformable Materials," *Interactions*, Jan./Feb. 2012, pp. 38–51.
9. A. Roudaut, A. Karnik, and S. Subramanian, "Morpheus: Toward High 'Shape Resolution' in Self-Actuated Flexible Mobile Devices," *Proc. SIG-CHI Conf. Human Factors in Computing Systems (CHI)*, 2013, pp. 593–602.
10. D. Lindlbauer et al., "Combining Shape-Changing Interfaces and Spatial Augmented Reality Enables Extended Object Appearance," *Proc. 2016 CHI Conf. Human Factors in Computing Systems (CHI)*, 2016, pp. 791–802.
11. L. Floridi, *The Fourth Revolution: How the Infosphere Is Reshaping Human Reality*, Oxford University Press, 2014.
12. H. Nissenbaum, *Privacy in Context: Technology, Policy, and the Integrity of Social Life*, Stanford Univ. Press, 2009.
13. P. Bruening and H. Patterson, "A Context-Driven Rethink of the Fair Information Practice Principles," *J. Business & Technology Law*, vol. 13, 2017.
14. H. Nissenbaum and H. Patterson, "Bio-sensing in Context: Health Privacy in a Connected World," *Quantified: Bio-sensing Technologies in Everyday Life*, 2016, pp. 79–100.
15. A. Cavoukian, "Privacy by Design," *IEEE Technology and Society Magazine*, Winter 2012, pp. 18–19.
16. T. Schwinn and A. Menges, "Fabrication Agency," *Material Synthesis: Fusing the Physical and the Computational*, vol. 85, no. 5, 2015, pp. 93–99.
17. A. Menges, "The New Cyber-Physical Making in Architecture: Computational Construction," *Material Synthesis: Fusing the Physical and the Computational*, vol. 85, no. 5, 2015, pp. 28–33.
18. B. Baudy et al., "Computational Design and Automotive Material Gestalt," *Material Synthesis: Fusing the Physical and the Computational*, vol. 85, no. 5, 2015, pp. 114–121.
19. C. Madsbjerg, *Sensemaking: What Makes Human Intelligence Essential in the Age of the Algorithm*, Brown, 2017.

Heather M. Patterson is a senior research scientist at Intel Labs. Contact her at heather.m.patterson@intel.com.



ADVERTISER INFORMATION

Advertising Personnel

Debbie Sims: Advertising Coordinator
Email: dsims@computer.org
Phone: +1 714 816 2138 | Fax: +1 714 821 4010

Advertising Sales Representatives (display)

Central, Northwest, Southeast, Far East:
Eric Kincaid
Email: e.kincaid@computer.org
Phone: +1 214 673 3742
Fax: +1 888 886 8599

Northeast, Midwest, Europe, Middle East:
David Schissler
Email: d.schissler@computer.org
Phone: +1 508 394 4026
Fax: +1 508 394 1707

Southwest, California:

Mike Hughes
Email: mikehughes@computer.org
Phone: +1 805 529 6790

Advertising Sales Representative (Classifieds & Jobs Board)

Heather Buonadies
Email: h.buonadies@computer.org
Phone: +1 201 887 1703

Advertising Sales Representative (Jobs Board)

Marie Thompson
Email: marie@4caradio.org
Phone: 714-813-5094

Next-Generation Smart Environments: From System of Systems to Data Ecosystems

Edward Curry
Lero, NUI Galway

Amit Sheth
Kno.e.sis, Wright State
University

Digital transformation is driving a new wave of large-scale data-rich smart environments with data on every aspect of our world. The resulting data ecosystems present new challenges and opportunities in the design of intelligent systems and

system of systems.

Smart Environments are generating significant quantities of data due to a convergence of digital infrastructure from the Internet of Things (IoT), Edge, Fog, and Cloud Computing that is driving a new wave of data-driven intelligent systems. Through the generation and analysis of data from the smart environment, data-driven systems are transforming our everyday world, from the digitization of traditional infrastructure (smart grid, water, and mobility), the revolution of industrial sectors (smart autonomous cyber-physical systems, autonomous vehicles, and industry 4.0), to changes in how our society operates (smart government and cities). At the other end of the scale, we see more human-centric thinking in our systems¹ where users have growing expectations for highly personalized digital services for the “market-of-one.” The digital transformation is creating a data ecosystem with data on every aspect of our world spread across a range of intelligent systems. Data ecosystems present new challenges to the design of intelligent systems and system of systems requiring a rethink in how we deal with the needs of large-scale data-rich smart environments. How can intelligent systems leverage their data ecosystem to be “smarter?” How can we support data sharing data between smart systems in an ecosystem? How can systems adapt to take advantage of the data within the ecosystem? What are practical approaches to the governance of data within an ecosystem? How can we make trusted decisions using data and humans within the ecosystem? Solving these problems is critical if we are to progress towards next-generation, data-intensive intelligent systems.

FROM DETERMINISTIC TO PROBABILISTIC DECISIONS IN SMART ENVIRONMENTS

Within a smart environment a range of reliability is required. Consider the example of the autonomous connected car/vehicle. We have the strict requirements of safety-critical autonomous driving system, and a failure may lead to loss of life or serious personal injury. Compare that to the “good enough” infotainment systems, where a failure is acceptable and merely an inconvenience to the user. When it comes to making decisions in smart systems, there are two general approaches: deterministic (model-driven) and probabilistic (data-driven). A critical difference between the approaches can be explored by considering the costs and level of reliability and adaptability each provides. There is a tension between reliability, predictability, and cost:¹ usually the more dependable and reliable the system needs to be, the more cost is associated with its development. Typically, we can see deterministic systems as reliable but with high costs to develop and adapt (i.e., autopilot), and probabilistic as low-cost to build and adapt, but less reliable (i.e., infotainment).

Where high-levels of reliability are needed, deterministic approaches are an obvious choice for the design of smart systems. This is because the environment is optimized based on a formal deterministic model, and a set of rules and/or equations details the decision logic for the system that is used to control the activity in the environment in an efficient and predictable manner. Adapting the system to meet changes in the environment is a costly process, as the model and its rules need to be updated by expert system engineers.

In the probabilistic approach, the core of the decision process is a statistical model that has been learned from an analysis of training data to learn the structure of a decision model automatically from the observed data (i.e., driver behavior). Thus, a fundamental requirement of data-driven approaches is the need for data to train the algorithms. A lack of data, and training data, within a smart environment limits the use of data-driven approaches.

As the IoT is enabling the deployment of lower-cost sensors, we are seeing broader adoption of intelligent systems and gaining more visibility (and data) into smart environments. Not only are smart environments generating more data, but they are also producing different types of data with an increase in the number of multimedia devices deployed such as vehicle and traffic cameras. The emergence of the Internet of Multimedia Things (IoMT) is resulting in large quantities of high-volume and high-velocity multimedia event streams that need to be processed. The result is a data-rich ecosystem of structured and unstructured data (i.e., images, video, audio, and even text) detailing the smart environment that can be exploited by data-driven techniques.

The increased availability of data has opened the door to the use of the data-driven probabilistic models, and their use within smart environments is becoming increasingly commonplace for “good enough” scenarios. It is estimated that a single connected car will upload about twenty-five gigabytes of data per hour (http://www.cisco.com/web/about/ac79/docs/mfg/Connected-Vehicles_Exec_Summary.pdf), while a vehicle fitted with an autonomous vehicle imaging and scanning system generates and processes about 4 TB of data for every hour of autonomous driving (<https://www.datamakespossible.com/evolution-autonomous-vehicle-ecosystem/>).

As a result, the conventional rule-based approach is now being augmented with data-driven approaches that support optimizations driven by techniques including machine learning, cognitive, and AI techniques that are opening up new possibilities in the design of smart systems. For example, pedestrian detection is difficult to implement in a rule-based approach. However, deep learning models for object detection and semantic segmentation using a dash-mounted camera are very effective at detecting pedestrians. Systems can now adapt to changes in the environment by leveraging the data generated in the environment within their learning process to improve performance. If systems share data on their operational experiences, then the pooled data can be used to improve the overall learning processes of all the systems, giving us a form of collective artificial intelligence through the “wisdom of the systems.” Because the process is data-driven, it can be run and re-run at low cost. This critical role of data in enabling adaptability and collective machine intelligence makes it a precious resource.

SYSTEM OF SYSTEMS

The need to bring together multiple systems within a smart environment to work together is becoming a standard requirement. Initiatives such as Smart Cities are showing how different systems within the city (i.e., energy and transport) can collaborate to maximize the potential to optimize overall city operations. Autonomous connected vehicles can support smart city mobility by providing a vital feedback loop for cities on the state of traffic volumes, flows, roadway design and maintenance, and the mobility requirements (trip information) of its occupants. This requires a System of Systems (SoS) approach to connect systems that cross organizational boundaries (i.e. city, automotive, personal data), come from different domains (i.e., entertainment, manufacturing, logistics, etc.), and operate at different levels (i.e., city, district, neighborhood, fleet, vehicle, or individual passenger). The joint ISO/IEC/IEEE definition of a SoS brings together a set of systems for a task that none of the systems can accomplish on its own. Each constituent system keeps its management, goals, and resources while coordinating within the SoS and adapting to meet SoS goals.² Maier³ identified a set of characteristics to describe a SoS:

- Operational independence: constituent systems can operate independently from the SoS and other systems.
- Managerial independence: constituent systems are managed by different entities.
- Geographic distribution is the degree to which a system is widely spread or localized.
- Evolutionary development: the evolution of a SoS and its behavior, which requires changes to system interfaces to be maintained and kept consistent.
- Emergent behavior: new emergent behavior can be observed when the SoS changes.

There are many challenges in bringing together the constituent systems into a SoS at the data, service, process, and organizational levels that require advanced systems engineering. At the data-level, data-driven approaches can benefit from leveraging data from multiple systems within the smart environment. This requires support for the sharing of data at new scales between multiple complex interconnected system of systems within a smart environment.

DATA ECOSYSTEMS

Within a data ecosystem, participants (individuals or organizations) can create new value that no single participant could achieve by itself.⁴ A data ecosystem can form in different ways—around an organization, an activity (mobility), a community of interest (music), a geographical location (city), or within or across industrial sectors (automotive, manufacturing, pharmaceutical). In the context of a smart environment, the data ecosystem metaphor is useful to understand the challenges in maximizing the value of data within the environment. The cross-fertilization and sharing of vital resources and datasets from different participants is a key benefit of data ecosystems, leading to new business opportunities and easier access to knowledge and data.

Connected and Autonomous Vehicle Data Ecosystem

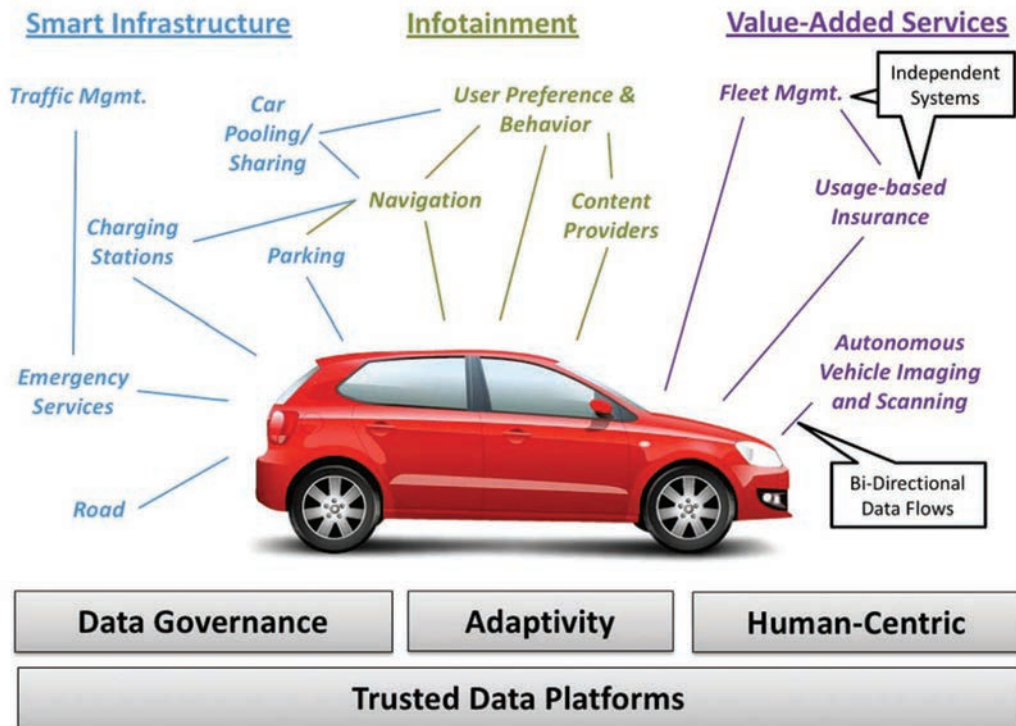


Figure 1. Connected and Autonomous Vehicle Data Ecosystem

Figure 1 details the data ecosystem for connected and autonomous vehicles where a community of interacting data-intensive systems share and combine their data to provide a holistic functional view of the car, passenger, city mobility, and service & infrastructure providers. Systems within the ecosystem can also come together to form a SoS. The ecosystem supports the flow of data between systems, enabling the creation of data value chains to understand, optimize, and reinvent processes that deliver insight to optimize the overall ecosystem. Data may be shared about the current operating conditions of the vehicle, traffic flows, or context of the passengers; a family on holiday, or a business executive moving between meetings. The pooled data can be used to support personalized digital services (i.e. delivering the latest episode of the family's favorite sitcom) and real-time decision-making (i.e. delivering relevant information for the business executive's next meeting). Data on past operating conditions can be shared to improve the learning processes of all systems in the ecosystem.

The nature of the ecosystem, the systems themselves, and the system dynamics will affect the design and operation of the ecosystem. Enabling data ecosystems for smart environments will require a rethink in the design of intelligent systems to consider ecosystem concerns including governance, economics, and technical challenges. Data infrastructure is needed to support data sharing within the ecosystem—from data provided by a single dominant actor on their proprietary infrastructure, to a community pooling their data in a managed open source data platform.

To understand the dynamics of a smart environment data ecosystem we can look to the literature on SoS³ and business ecosystem⁵ to help us understand the different types of data ecosystem that can exist. In Figure 2 we bring together these two areas in the design of a data ecosystem for a smart environment: Koenig⁵ identified two key criteria regarding the design of a business ecosystem that will also influence a data ecosystem, namely, resource control, and interdependence:

- Control of key data resources: Who controls the essential data resources in the ecosystem? Does a single “keystone”⁶ actor control the key data resources that all others depend on, or is control of the key data resources spread across multiple actors in the ecosystem?
- Participant interdependence: Interdependence is based on the degree to which different participants in the ecosystem must interact and exchange data for performing their activities. Reciprocal interdependence requires high levels of coordination between the participants, while pooled interdependence enables loose coupling between participants.

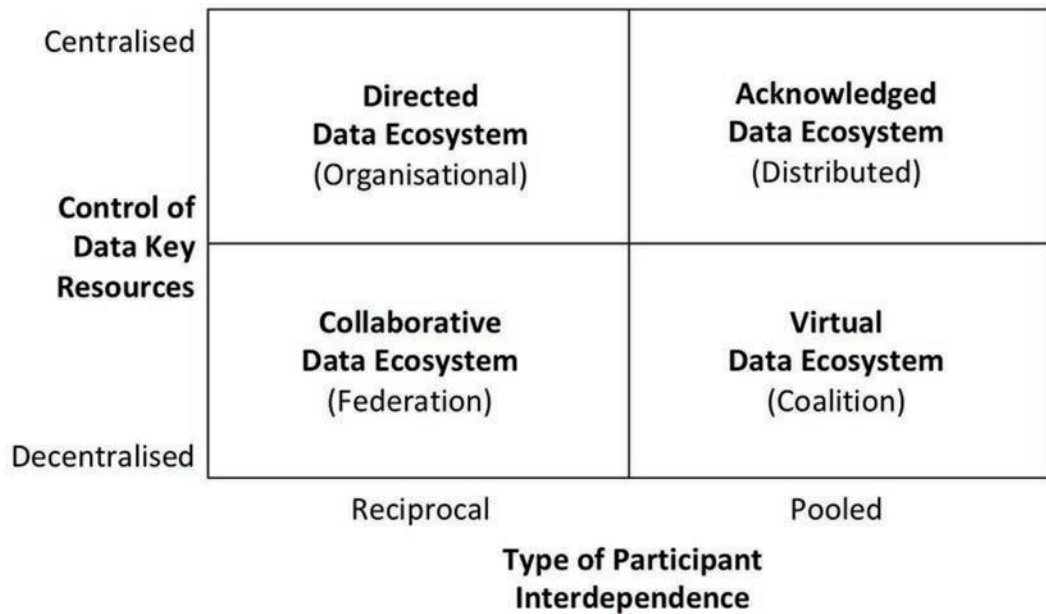


Figure 2. Topology of Data Ecosystems (adapted from Koenig⁵ and Maier³)

Drawing inspiration from the SoS classification by Maier³ (which defines Virtual, Collaborative, Acknowledged, and Directed categories) and the ecosystem topology by Koenig, we can consider the different types of data ecosystems that may exist within a smart environment (Figure 2).

- Directed data ecosystems are centrally controlled to fulfill a specific purpose. Typically found within an organization setting or following a keystone model, participants within a directed ecosystem maintain an ability to operate independently, but their standard operational mode is subordinated to the centrally managed purpose of the ecosystem.
- Acknowledged data ecosystems have defined objectives and pooled dedicated resources. The constituent systems retain their independent ownership and objectives. Changes in the ecosystem are based on collaboration between the distributed participants.
- Collaborative data ecosystems have participants interact voluntarily to fulfill an agreed-upon central purpose. The primary players collectively decide the means of enforcing and maintaining standards between the federations of participants.
- Virtual data ecosystems have no central management authority and no centrally agreed upon purpose. Bottom-up coalitions of participants emerge from a virtual data ecosystem to pool decentralized resources to achieve specific goals.

FUTURE DIRECTIONS

Enabling a smart environment data ecosystem will require many challenges to be overcome regarding infrastructure, governance, systems engineering, and human-centricity.

Trusted Data Platforms

To support the ecosystem and the interconnection of systems, there is a need to enable the sharing of data between systems. Platform approaches have proved successful in many areas of technology, and the idea of large-scale "data" platforms have been touted as a possible next step. A data platform focuses on the secure and trusted data sharing among a group of participants (i.e., industrial consortiums sharing private or commercially sensitive data) within a clear legal framework. An ecosystem data platform would have to be infrastructure agnostic and have to support continuous, coordinated data flows, seamlessly moving data between systems. Data exchange could be based on models for monetization or reciprocity. Data platforms can create possibilities for smaller organizations and even individual developers to get access to large volumes of data, enabling them to explore their potential. Data platforms open up many research areas including data discovery, curation, linking, synchronization, standardization, and decentralization. However, the challenges go beyond the technical to issues of data ownership, privacy, business models, and licensing and authorized reuse by third parties.

Ecosystem Data Governance

For mass collaboration to take place within data ecosystems, we need to overcome the challenges of dealing with large-scale agreements between potentially decoupled interacting parties. Research is needed on decentralized data governance models for data ecosystems that support collaboration and fully consider ethical, legal, and privacy concerns. Data governance within an ecosystem must recognize data ownership, sovereignty, and regulation while supporting economic models for the sustainability of the data ecosystem. A range of decentralized governance approaches may guide a data ecosystem from authoritarian to democratic alternatives, including majority voting, reputation models (i.e., eBay), proxy-voting, and dynamic governance (i.e., sociocracy: circles and double linking).⁷ Finally, economic concerns may be considered as an incentivization factor within governance models with "data-vote exchange" models where participants pay for votes with data.

Incrementally Evolving Systems Engineering: Cognitive Adaptability

The design of adaptive systems will need to consider the implication of operating within an ecosystem. The boundaries of systems will be fluid and will change and evolve at runtime to adapt to the context of the current situation. However, we must also consider the cost of system participation, and support "pay-as-you-go" approaches at both the system and data-levels. For data management, dataspace⁸ represent one avenue where a pay-as-you-go approach has been applied to integrate data on an "as-needed" basis with the labor-intensive aspects of data integration postponed until they are required.⁹ How can the pay-as-you-go approach be extended to the design of incremental and evolving systems?

Work on evolving systems engineering¹⁰ will need to consider the inclusion of data-driven probabilistic techniques that can provide "cognitive adaptability" that will help systems adapt to changes in the environment that were unknown at design-time. Adaptive systems require new iterative development processes that require training and deploying machine learning models over massive volumes of training data with close collaboration between data scientists, software developers, data engineers, and governance professionals. System design will need to consider the varying levels of accuracy offered by data-driven approaches, providing best-effort or approximate results using the data accessible at the time.⁸ How can we mix deterministic and statistical approaches? How can we test and verify these systems? What are the challenges in making decisions using multiple sources from the ecosystem?

Towards Human-Centric Systems

Currently, intelligent systems make critical decisions in highly-engineered systems (i.e., autopilots) where users receive specialized training to interact with them (i.e., pilots). As we move forward, intelligent systems will be making both critical and lifestyle decisions—from the course of treatment for a critical illness and safely driving a car, to choosing what takeout to order and the temperature of our shower. Data-driven decision approaches (including cognitive and AI-based techniques) will need to provide explanations and evidence to support their decisions and guarantees for the decisions they recommend. The role of users in data ecosystems will not be a passive one. Users are a critical part of socio-technical systems, and we need to consider more ways of including the “human in the loop” within future systems. Active participation of users can improve their engagement and sense of ownership of the system. Indeed, active involvement of the user could be a condition for them granting access to their private data. Research is needed to build trust in algorithms and data—in the trusted co-evolution between humans and AI-based systems, and in the legal, ethical, and privacy issues associated with making data-driven critical decisions.

ACKNOWLEDGEMENTS

This work was supported, in part, by Science Foundation Ireland grant 13/RC/2094 and co-funded under the European Regional Development Fund through the Southern & Eastern Regional Operational Programme to Lero - the Irish Software Research Centre (www.lero.ie).

REFERENCES

1. A. Sheth, “Computing for Human Experience: Semantics-Empowered Sensors, Services, and Social Computing on the Ubiquitous Web,” *IEEE Internet Computing*, vol. 14, no. 1, 2010, pp. 88–91.
2. *ISO/IEC/IEEE 15288: 2015 Systems and Software Engineering - System Life Cycle Processes*, standard ISO/IEC/IEEE 15288, ISO/IEC/IEEE, 2015.
3. M. W. Maier, “Architecting Principles for Systems-of-Systems,” *Systems Engineering*, Wiley, 1998.
4. *New Horizons for a Data-Driven Economy: A Roadmap for Usage and Exploitation of Big Data in Europe*, J. M. Cavanillas, E. Curry, and W. Wahlster, Springer International Publishing, 2016.
5. G. Koenig, “Business Ecosystems Revisited,” *Management*, vol. 15, 2012, pp. 208–224.
6. H. Kim, J.-N. Lee, and J. Han, “The Role of IT in Business Ecosystems,” *Communications of the ACM*, vol. 53, 2010, p. 151.
7. J. A. Buck and S. Villines, *We the People : Consenting to a Deeper Democracy : a Guide to Sociocratic Principles and Methods*, Sociocracy.info, 2007.
8. M. Franklin, A. Halevy, and D. Maier, “From Databases to Dataspaces: A New Abstraction for Information Management,” *ACM SIGMOD Record*, vol. 34, no. 4, 2005, pp. 27–33.
9. E. Curry et al., “Internet of Things Enhanced User Experience for Smart Water and Energy Management,” *IEEE Internet Computing*, vol. 22, no. 1, 2018.
10. M. Hinchey and L. Coyle, “Evolving Critical Systems: A Research Agenda for Computer-Based Systems,” *217th IEEE International Conference and Workshops on Engineering of Computer Based Systems*, 2010, pp. 430–435.

ABOUT THE AUTHORS

Edward Curry is a funded investigator at Lero: The Irish Software Research Centre and a lecturer in informatics at the National University of Ireland Galway. <http://edwardcurry.org>

Amit Sheth is the LexisNexis Ohio Eminent Scholar and the executive director of Kno.e.sis - Ohio Center of Excellence in Knowledge-enabled Computing and BioHealth Innovations. He is a fellow of the IEEE and the AAAI. <http://knoesis.org/amit>

*This article originally appeared in
IEEE Intelligent Systems, vol. 33, no. 3, 2018.*



IEEE Security & Privacy magazine provides articles with both a practical and research bent by the top thinkers in the field.

- stay current on the latest security tools and theories and gain invaluable practical and research knowledge,
- learn more about the latest techniques and cutting-edge technology, and
- discover case studies, tutorials, columns, and in-depth interviews and podcasts for the information security industry.



computer.org/security

IEEE

COMPUTER ARCHITECTURE

LETTERS

IEEE Computer Architecture Letters is a forum for fast publication of new, high-quality ideas in the form of short, critically refereed technical papers. Submissions are accepted on a continuing basis and letters will be published shortly after acceptance in IEEE Xplore and in the Computer Society Digital Library.

Submissions are welcomed on any topic in computer architecture, especially:

- Microprocessor and multiprocessor systems
- Microarchitecture and ILP processors
- Workload characterization
- Performance evaluation and simulation techniques
- Interactions with compilers and operating systems
- Interconnection network architectures
- Memory and cache systems
- Power and thermal issues at the architectural level
- I/O architectures and techniques
- Independent validation of previously published results
- Analysis of unsuccessful techniques
- Domain-specific processor architecture (embedded, graphics, network)
- High-availability architectures
- Reconfigurable computer architectures

www.computer.org/cal

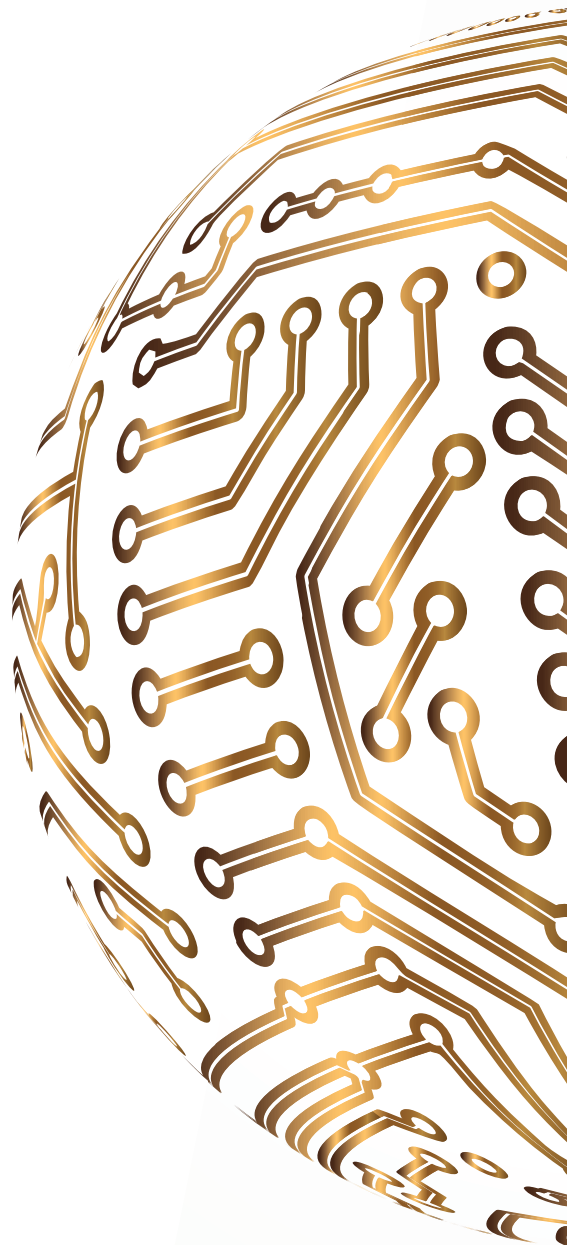


Join the IEEE Computer Society
for subscription discounts today!

www.computer.org/product/journals/cal



IEEE
COMPUTER
SOCIETY



My Mother the Car (or Why It's a Bad Idea to Give Your Car a Personality)

Phil Laplante

Penn State

Abstract—What happens when we endow an autonomous vehicle with a personality? It sounds like a good idea—witty banter, friendly advice, and encouragement from your car. But science fiction writers predicted this technology and both the good and bad scenarios that could arise. A brief sample of some of these situations from American television and film is enlightening but suggests that granting a personality to an auto is ill-advised. This theme issue explores the state and potential benefits and ethical dilemmas of connected and autonomous vehicles. While fully autonomous vehicles are already experimentally deployed, they will become ubiquitous within a few years. Soon after we should see an experimental deployment of vehicles that have a level of artificial intelligence such that they could be considered “sentient” (self-aware) or even “anthropomorphic” (human-like). These vehicles could do much more than self-drive or give us driving advice, travel information, and various forms of infotainment. They could also remind us of important stops to be made, listen to our complaints about traffic and other drivers, and even provide us with soothing advice. And this advice could be given with the benefit of an artificial personality—one that could be formal, friendly, or even replicates that of a famous person, celebrity, or a loved one.

■ **BUT ADDING A** personality to the car could make things worse, as predicted by writers of science fiction novels, short stories, television plays, and movie scripts. Let us review some examples from American film and television.

ANTHROPOMORPHIC VEHICLES IN AMERICAN FILM AND TELEVISION

In the early silent short, *Nothing Matters* (1926), the hero dreams of “anthropomorphic cars cowering and fleeing in fear of the new car in town”¹ an interesting twist on the idea of communicating (connected) cars. The vignette raises several interesting questions.

Digital Object Identifier 10.1109/MITP.2019.2896775

Date of current version 27 March 2019.

For example, will anthropomorphic vehicles form cliques, show prejudice or discriminate? And will connected vehicles organize to give preferential treatment to certain types of vehicles, for example at merge points?

In a 1953 public service announcement (PSA) film, *The Talking Car*, a fatherly vehicle advises children about safe street crossing. In a later version of the PSA (1969) a trio of vehicles (each, with a very distinct personality) sat in judgment of a young boy who disregards basic street crossing protocol. These PSAs raise questions about the extent to which smart vehicles will have authority over humans, how they may cooperate, and even if their “testimony” will be admitted in courts.

In the short-lived and forgettable TV series, *My Mother the Car* (1965–66), a man discovers that his car hosts the reincarnated spirit of his deceased mother. Talking through the radio of a 1928 Porter Touring car (this was an anachronism since the dashboard radio wasn’t invented until 1930), the protagonist, Dave (played by the actor, Jerry Van Dyke, who turned down the role of Gilligan in the much more successful, *Gilligan’s Island*, for this clunker of a show), is pestered, advised and loved by the car. But mother also exhibits human emotions such as anger, jealousy, and greed.

In addition to these emotions, cars could also be heroic. For example, the namesake car in the film *Chitty Chitty Bang Bang* (1968) (this is a British film, but it enthralled me as a child, so I am including it in this brief survey) is self-aware. This 1900s era modified roadster was autonomous and could anticipate the need to sprout pontoons and a propeller for water travel, or wings for the flight to rescue its human family from an evil baron.

An internationally successful film featuring another heroic car, *Herbie the Love Bug* (1968), spawned four sequels over 40 years. Herbie was self-driving, sentient Volkswagen Beetle with a mischievous, loyal, and loving personality. Time and again, even while causing trouble with his antics, the precocious Herbie managed to rescue his owners from various predicaments. Similarly, the vehicle in the animated “Speed Buggy” television program (1973) featured a goofy talking car who heroically

participated in the human protagonists’ mystery-solving adventures.

The popular TV series *Knight Rider* (1982–1986) starred K.I.T.T., a sentient Pontiac Firebird powered by a “Knight 2000 microprocessor.” K.I.T.T. was self-aware and capable of learning. In addition to providing navigational aid and entertainment, the car helped his crusading driver Michael solve crimes and resolve many dangerous situations. The vehicle also exhibited a very human-like personality that was egotistical, stodgy and funny.

Cars such as K.I.T.T. could not only save the day when called, they could also act autonomously and proactively. For example, the “muggle” car owned by the Weasley family in *Harry Potter and Chamber of Secrets* (1998 novel, 2002 movie) anticipates Harry and Ron’s need for rescue from the giant spider den, arriving in the nick of time. More recently, the *Cars* (2006 *et al.*) and *Transformers* (2007 *et al.*) movies and sequels have featured vehicles with human-like personalities, ambitions, and even expressing love. Of course, the main premise of the *Transformers* movies and television program featured the epic battle between the heroic autobots and villainous decepticon vehicles.

DANGERS OF SENTIENT AND PSEUDO SENTIENT VEHICLES

What happens when a “self-aware” vehicle errs, gets confused, or turns bad? Movies and Television have covered this aspect quite well. In an episode from *My Mother the Car*, “What Makes Auntie Freeze,” Mother gets “drunk” on antifreeze leading to all kinds of erratic behavior. In the episode, “I Remember Mama, Why Can’t You Remember Me?” Dave’s mother gets amnesia following a fender bender. Even K.I.T.T. could be obstinate with Michael and Herbie could be prideful and stubborn.

The movie, *Total Recall* (1990), which was adapted from a 1966 book, predicted autonomous taxis and some of the problems they can present in the case of misunderstanding. In one case, the helpful “Johnny Cab” confuses the destination when the protagonist shouts expletives. But Johnny Cab exhibits even worse behavior when the fare is not paid—the heretofore obliging cab turns homicidal and tries to run over the hero.

Self-awareness can also lead to paranoia. The evil car from in Steven King's *Christine* (novel and movie 1983) involves a possessed 1958 Plymouth Fury that exacts murderous revenge on the enemies of its owner. Both K.I.T.T. and Mother exhibited some signs of paranoia at times—what if they had completely turned evil?

And what happens when an anthropomorphic vehicle impersonates a human in some sort of Turing inspired nightmare? *My Mother the Car* foreshadowed this possibility too. In the episode, “TV or Not TV” Dave puts a TV in the garage for Mother, who creates a dilemma when she calls into a game show and wins a chance to appear on live television.

While the premises of these situations seem ridiculous, it is easy to imagine a computer virus or malware negatively altering the “personality” of an anthropomorphic to manifest analogously.

ROAD AHEAD

Artificially intelligent vehicles should provide great benefit, but new risks could emerge as the capabilities increase. Adding a personality to a vehicle will increase these risks. Some of the risks are simply annoying—do we need our cars hectoring us about our weight or driving habits

and acting like an overbearing mother? Or do we need the car interrupting the radio to tell us we are going too fast (or reporting us to the police). But, more significantly, what happens if the car becomes capricious, supercilious, paranoid or goes berserk?

Science fiction writers have shown us the potential benefits and real dangers of anthropomorphic vehicles. Most of these dangers are related to the car violating our trust and untrustworthiness is a personality flaw. While much more research work is needed in this area (e.g., Future of Life Institute <https://futureoflife.org/>) I think we just should not give cars personalities. Most people trust their mothers, if we cannot trust our mother the car, which car can we trust?

■ REFERENCE

1. J. Roots, *100 Greatest Silent Film Comedians*. Lanham, MD, USA: Rowman & Littlefield, 2014.

Phillip A. Laplante (M'86–SM'90–F'08) is a Professor of Software and Systems Engineering with Pennsylvania State University, Malvern, PA. Lately, his research interests include the Internet of Things, blockchain, and artificial intelligence. Contact him at plaplante@psu.edu.

*This article originally appeared in
IT Professional, vol. 21, no. 2, 2019.*

Department

Lance Gharavi: Performance Inspired Science + Technology

Bruce Campbell
Rhode Island School of Design

Francesca Samsel
University of Texas

Editors: Bruce Campbell, bcampbel01@risd.edu, and Francesca Samsel, figs@cat.utexas.edu

Abstract—We caught up with Lance Gharavi after we heard of and investigated *Beneath: A journey within*, a live performance motivated by the intent to get a wider audience interested in the lithosphere and mantle beneath our feet. *Beneath* is an archetype of art-science-tech projects and Lance came to coordinating that show from a long history of art-science projects. We asked him about insights and lessons learned.

■ **DR. GHARAVI HAS** been working at the intersection of art, science, and engineering through collaborating with large teams typically of artists, designers, scientists, engineers, and others to make media rich works of performance.

Like many others who had a stick-to-it-ness during the first heyday of virtual reality (VR), Lance brought the skills he acquired and fine-tuned to various projects when hired by Arizona State University (ASU) to be an artist working with digital technologies. VR was not officially on the list of his job responsibilities but as Lance suggested, “I realized it’s a matter of what you call VR. Many of those themes I explored transferred well into other digital technologies.”

A Brief Anniversary of Time, a prior performance project, represents well the role he was hired at ASU to perform. “We worked with a system I call a universe in a box,” he said. “The team designed the performance for ASU’s flat screen planetarium—the Marston Exploration Theater that uses planetarium software called Sky-Skan. The Marston Exploration Theater software runs on a stack of ten servers and every known object in the universe is plotted in four dimensions. You can fly around in this universe anywhere you want to go—even through time. It is a way to experience the mind-bogglingly immense scope of our universe. I like to think of it as an existential-crisis-machine.”

The show was a celebration of the 25th anniversary of Stephen Hawking’s *A Brief History of Time* (see figure 1) with a mix of pre-recorded video and live performance played

Digital Object Identifier [10.1109/MCG.2019.2892863](https://doi.org/10.1109/MCG.2019.2892863)

Date of current version 22 March 2019.



Figure 1. *A Brief Anniversary of Time*. Media design by Daniel Fine. Photo credit: Matthew Ragan. (Used with permission.)

out in that planetarium software. As Lance observed, “Much of what I do involves media design for live performance, with some interactive data visualization.”

Lance discussed the motivation of *Beneath*:

“Scientists know a remarkable amount about what exists far above us. We know the weight of the moon. We know the composition of stars in galaxies millions of light years away. But we know comparatively little about what lies just a few dozen miles below our feet. That which is beneath is our mystery and science is working to cast light on the subject. *Beneath* takes audiences on a multisensory journey to the Earth’s deep interior.”

“*Beneath*’s fusion of science and live performance features Christy Till, a geologist ballerina dancing catastrophic planetary cycles (see figure 2); Ed Garner, a bass-playing geophysicist interacting with his data through trip-hop bass-lines; and Patrick Young, a belly-dancing theoretical astrophysicist embodying seismic waves. Audiences virtually visit the lab of Dan Shim, a mineral physicist who uses diamonds in startling experiments, and talk with Lindy Elkins-Tanton, the first woman to lead a NASA mission beyond the Earth’s orbit.”

As Lance suggests:

“*Beneath* is the product of a multiyear collaboration among a team of planetary scientists, theatre makers, performance artists, and media designers based at ASU in collaboration with

animators, media designers, and artists outside of ASU, including Cloud Eye Control, Obscura Digital, and Ohio State University. These internal and external partnerships across the ASU campus and beyond have served to redefine the ways in which performance can function as both an arts-led research practice and forum for engaged learning. In both its goals and methods, *Beneath* provides a model for transdisciplinary collaboration and public outreach in science.”

“The project has three central goals: to make current scientific research artful, accessible, and compelling for the public; to create new visualization tools that aid scientists in research, communication, and education; and to engage and explore new models of collaboration between artists and scientists.”

“The creative team behind *Beneath* fuses theatre and science to tell a compelling story and communicate scientific research in an engaging and accessible way. It illustrates the dynamic systems of the Earth while showing the ways in which humans are connected to the immense and ancient processes of our planet.”

“The best part of the collaboration was definitely working with the people (see figure 3). My main scientist collaborator, Ed Garner, is a geophysicist and a seismologist. He tries to understand vibrations and the interior dynamics and structures of the Earth. There’s a huge scientific



Figure 2. Petrologist Christy Till dances in *Beneath* at the Marston Exploration Theater. Audiences view the stereoscopic media through 3-D glasses. Choreography by Liz Lerman. Stage direction by Erika Hughes, University of Portsmouth. Systems design by Matthew Reagan, Obscura Digital; and Ian Shelanskey, BRDG Studios. Media design by Jake Pinholster, Arizona State University; Dallas Nichols; Daniel Fine, University of Iowa; Alex Oliszewski, The Ohio State University; Miwa Matreyek; Boyd Branch, University of Kent; and Elora Mastison, Arizona State University. Sound design by Stephen Christensen, Arizona State University. Photo credit: Tim Trumble. (Used with permission.)

project called *EarthScope*, one of the largest project ever funded by NSF, and they are rolling these seismometers all across North America along with permanent ones mounted all over the place—to try and get a 3-D picture of the Earth’s interior. We took that data and created a 3-D model of the Earth’s interior based on an enormous spreadsheet of several terabytes of numbers. Ed is also a semiprofessional bass player. Through our software, he uses his bass to ‘play’ his data, lighting up the different parts of the Earth’s interior depending on pitch (see figure 4).”

“Christy Till studies rocks and magma; she is also a former professional ballerina and so dances expressively in the performance. Patrick Young is a semiprofessional belly dancer, who used his body in the show to demonstrate how seismic waves work. For the scientists who were live in the performance, the process was deeply moving. It was a way of bringing together different aspects of themselves.”

When we asked about his opinions on sonification, Lance said:

“I love sonification of data but you often don’t know what you are hearing. We took the sounds from an earthquake and increased the pitch so you could hear it. We turned out all the lights in the theater so people could experience this earthquake wave in the dark for a minute. It’s intense and intimidating, rather like someone banging on a trashcan. You feel the vibrations in your body.”

“When we can ask questions of scientists that they weren’t expecting, or hadn’t thought of it that way, we can help. Science is not just a bunch of data or a lot of numbers. Science is stories, and thus, it depends on metaphor. That’s what we do as artists, many of us, as storytellers—we deal in metaphor. Finding new metaphors with stories to tell is a contribution to science.”

“Some artists just use the outputs of science as fodder for art work. Some translate science for the public (as in *A Brief Anniversary of Time*). Some get involved explicitly trying to show data in new ways. On very rare occasions, working with scientists, artist collaborations can lead to advances in the science. That is always a useful goal. I have various projects where that is really a big focus, having gained the trust of scientists through a longer term collaboration.”

Lance’s current work with Ars Robotica speaks well to the potential of shifting toward a partnership of artists and scientists pursuing science together. As Lance explains:

“The focus of the Ars Robotica initiative is advancing robotics through art and design. I initially worked with a robot called Baxter created by Rethink Robotics, a company that makes anthropomorphic robots. I partnered with ASU’s Autonomous Systems Technologies Research and Integration Laboratory (ASTRIL), and it was

very much about trying to get a robot to fill the needs of the production we were creating (see figure 5). We built some software and an interface to control the robot and it was like an anthropomorphic remote controlled car. At the end of it, Srikanth Saripalli, the lab's Director, asked, 'Can we keep working together? Because working with you has helped advance our research.' With that *Ars Robotica* was born."

"I saw that many artists will take the science and bend it to the purposes of the art they are making. I thought, 'What if we put the interests of the research first and then make art to support the research goals?' That's what *Ars Robotica* does. For instance, now we are working to design a testbed for swarms of driverless cars and remotely operated vehicles."

"Initially I said to them, 'you can get your undergraduates and graduate students to build models for this work. You can go to the nearest train store and pull out buildings and stuff. Why come to an artist?' They were like, 'we don't know.' I said, 'that's a totally acceptable answer. We can figure that out together.' And so the project will be simultaneously a test bed, a laboratory, but also a performance and art installation. It took a period of trust and working together before we found a common process for advancing explicit goals to advance the technology and science."

Lance's work with robotics now focuses on creating a context or a site for science. "But also a performance event, in the case with robots, that wants art to do more. It has to serve a technical and utilitarian function."



Figure 3. *Beneath* team at work. From left: Lance Gharavi, Ian Shelanskey, and Ed Garnero discuss ways of visualizing and sonifying vast amounts of seismic data to reveal the formations of the Earth's deep interior in 3-D. Such visualizations will have a life beyond the performance as tools for scientists. Photo credit: Tim Trumble. (Used with permission.)



Figure 4. In *Beneath*, Heather Lee Harper, Lance Gharavi, and Ed Garnero perform a trip-hop spoken word number about the vibrations in the Earth's interior. Photo credit: Tim Trumble. (Used with permission.)

The hard part as a liaison is figuring out what is the best way to inject the practice and training of art into serving the science. Lance adds:

"With *Robotopolis*, an upcoming project, one of the practical challenges was figuring out how to fulfill the needs of the robotics lab while doing



Figure 5. Baxter performing in *The Mirror*. The ASTRIL lab that used Baxter has moved with its Director to another institution. Ars Robotica has now teamed with the Center for Human, Artificial Intelligence, and Robot Teaming, specifically to work with swarm robotics. Instead of a single robot, they coordinate swarms of robots, like bees, in the ground, air, or water. The robots can be large or tiny, but they work together to perform tasks. Photo credit: Tim Trumble. (Used with permission.)



Figure 6. Brian Foley in *Immerge*, a large-scale transdisciplinary work of digital performance, a dramatic myth on the free movement of data and creativity. Featuring architectural projection, virtual life forms, and other media. Lead artist: Lance Gharavi; media design: Jake Pinholster and David Tinapple; production design: Anastasia Schneider, Brunella Providente, and Adam Vachon. At the Emerge Festival 2012. Photo credit: Tim Trumble. (Used with permission.)

the things that art needs to do. How am I going to create something that I feel pulls all my triggers as an artist on the way to constructing something useful for the lab?”

“*Robotopolis* is a city for robots. We plan on taping out the ground plan the first of the year and once we confirm the ground plan is going to work for our needs, we’ll move forward with finalizing architecture and other aspects and start the build and complete the process of media design. There is a whole aspect of media design in this because we are going to hit the city from several directions with projections and map those onto all the buildings—quite a complex projection mapping task.”

Another project in the works worth tracking as it advances is *Port of Mars*. Lance suggests:

“Some of the biggest challenges of human space exploration aren’t technological. They are social. How can we best sustain healthy human communities in space? What social systems and structures do we need? We have created a project to find solutions. It is a game-based social science experiment called *Port of Mars*. In the game, all players are members of the first Martian community and they have to support the goals of the community in surviving while also pursuing their individual goals. All the behaviors of the players are monitored and tracked. We will analyze their behavior to see what kinds of things lead to success and what leads to failure. What systems and processes will people invent to get along?”

“We go into experiments in February and March. The experiments were designed by a couple of applied mathematicians, Marco Janssen and Marty Anderies, doing work in the social sciences, especially commons issues. Also onboard are game designers like Michael Yichao of Riot Games, planetary scientists like Mars expert Tanya Harrison, visual artists like Titus Lunter, and many other specialists. These relationships will lead to useful contributions to the science, and also something fun and artful.”

“The term ‘commons’ refers to any resource shared by a group. Earth’s ecosystem is a giant commons, for instance. The great thing about Earth is we have such abundant resources. But now we are bumping up against some serious ceilings—climate change, mass extinctions, etc. If we send people to Mars, they will be much more dependent on shared resources and those resources will be much scarcer; there’s a very narrow margin for error. Before we spend the billions of dollars and put people’s lives at risk, we had better learn how to navigate commons dilemmas in such a hostile environment. *Port of Mars* is about finding solutions to such dilemmas.”

Collaborative projects involving the artistic community and the sciences, pursuing common ground and bridging professional norms, have required a lot of patience historically, as well as the ability to bridge languages (see figure 6). In Lance’s case:

“People always ask me, ‘How do you manage collaborations like these across such diverse

disciplines?’ It’s true there are different disciplinary cultures, different languages, goals, and tools. Part of what’s exciting about such work is that difference. It’s a source of allure as opposed to a problem. When you do interdisciplinary work, gathering people with such different skills and knowledge, everyone is a wizard to everyone else. But really, it’s not so difficult. We are all passionately curious. We are all skilled in asking questions, seeking answers, and solving problems. We are all trained to work with a team toward a goal based on limited input, with limited time and resources. I find the infectious sense of wonder and excitement about the questions and challenges bonds us and makes our differences feel negligible.”

“Mostly, I am interested in big stories and big ideas. Science has them. Art gives them breath.”

Bruce Campbell is a faculty member of Web Design + Interactivity at the Rhode Island School of Design. His research interests include ocean data visualization and procedural design. He received the Ph.D. degree in systems engineering from the University of Washington. Contact him at bcampbel01@risd.edu.

Francesca Samsel is a research associate with the Center for Agile Technology, University of Texas–Austin and an artist-in-residence at the Los Alamos National Laboratory. Contact her at figs@cat.utexas.edu.

Contact department editor Bruce Campbell at bcampbel01@risd.edu or department editor Francesca Samsel at figs@cat.utexas.edu.

*This article originally appeared in
IEEE Computer Graphics and Applications,
vol. 39, no. 2, 2019.*

IEEE

SECURITY & PRIVACY

IEEE Security & Privacy is a bimonthly magazine communicating advances in security, privacy, and dependability in a way that is useful to a broad section of the professional community.

The magazine provides articles with both a practical and research bent by the top thinkers in the field of security and privacy, along with case studies, surveys, tutorials, columns, and in-depth interviews. Topics include:

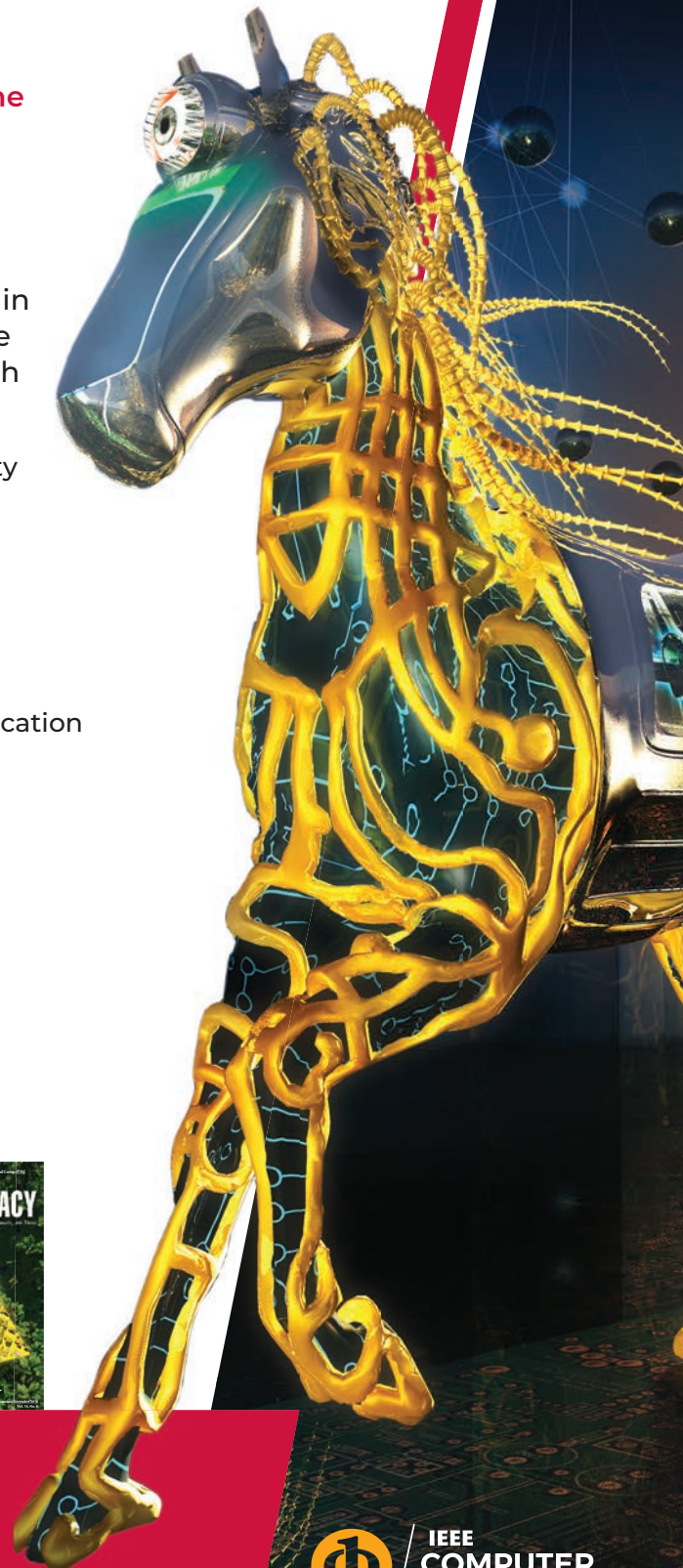
- Internet, software, hardware, and systems security
- Legal and ethical issues and privacy concerns
- Privacy-enhancing technologies
- Data analytics for security and privacy
- Usable security
- Integrated security design methods
- Security of critical infrastructures
- Pedagogical and curricular issues in security education
- Security issues in wireless and mobile networks
- Real-world cryptography
- Emerging technologies, operational resilience, and edge computing
- Cybercrime and forensics, and much more

www.computer.org/security



Join the IEEE Computer Society
for subscription discounts today!

www.computer.org/product/magazines/security-and-privacy





A Comet Revisited

Lessons Learned from *Philae's* Landing

András Balázs

From the Editors

Two years ago, we published a column on the software involved in landing on a comet. It's clear by now that although the landing itself was an impressive accomplishment, not everything went as planned. We thank András Balázs for his thorough, honest analysis of what went wrong, what was done, and, importantly, what more could have been done. The software community could benefit from more such evaluations of the problems that so frequently occur in projects. —*Michiel van Genuchten and Les Hatton*

AFTER A 10-YEAR journey across the Solar System and many maneuvers, the *Rosetta* spacecraft—carrying the *Philae* lander, a scientific mini-laboratory (see Figure 1)—smoothly approached comet 67P/Churyumov-Gerasimenko. *Rosetta* then flew a multitude of low- and high-altitude orbits around the comet, performing scientific experiments and mapping the comet's shape and surface in detail never seen before.

The *Philae* mission comprised these phases:

1. cruise (onboard *Rosetta*);
2. separation, descent, and landing;
3. first comet science;
4. hibernation; and
5. long-term science.

On 12 November 2014, *Rosetta* initiated the ballistic delivery of *Philae*

to the comet's nucleus (see Figure 2a). The launch occurred approximately 500 million km from Earth, approximately 3 astronomical units (AUs) from the sun, and 22.5 km from the comet.

Upon first touching down after a descent phase of 7 hours, the lander couldn't attach itself to the comet, owing to unexpected, probably systematic failures in both parts of the dual-redundant anchoring subsystem and a malfunction of the nonredundant hold-down thruster.¹ However, thanks mostly to the comet's gravitational attraction, *Philae* still completed its touchdown. After several hours of bouncing and uncontrolled tumbling, *Philae* reached its final parking position, tilted to one side (see Figure 2b) roughly 1.2 km from the planned landing site. *Philae* remained functionally intact, despite

its anomalous landing. It also managed to maintain radio contact with *Rosetta*, which served as a relay station between *Philae* and the flight operations control center on Earth.

Philae's batteries supplied energy for doing science on the comet's surface for roughly 60 hours on the first run. Thereafter, the lander went into hibernation owing to the disadvantageous thermal and solar-illumination conditions at its parking site. After about 6 months of hibernation, *Philae* woke up at 1.8 AU from the sun. Its central onboard computer (CDMS) then autonomously entered the long-term-science phase.

Philae's Operation Control

A previous Impact department article² and a more detailed reference paper³ elaborate on the technical, hardware, software, and operational

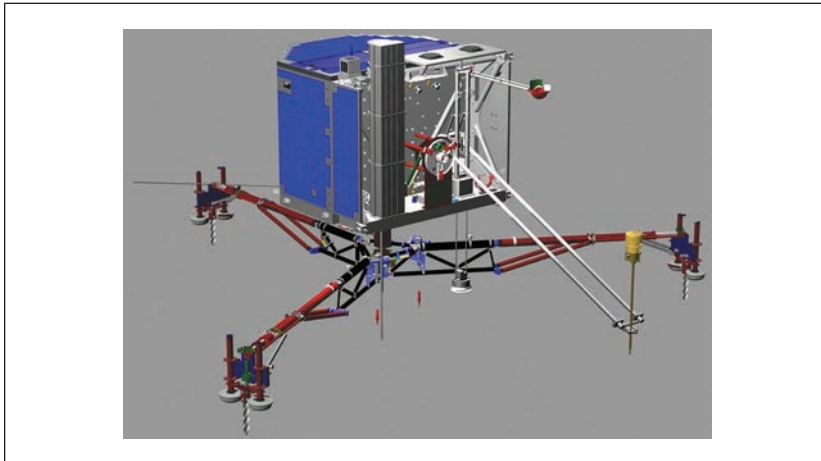


FIGURE 1. The *Philae* lander explored comet 67P/Churyumov-Gerasimenko. (Source: DLR/Cologne; used with permission.)

requirements and their implementation in the CDMS.

Once *Philae* landed, the starting times and durations of the radio visibility windows between *Philae* and *Rosetta* depended on such factors as *Rosetta*'s flight track, the comet's 12.6-hour rotation period, and *Philae*'s location and orientation on the comet. Although these time windows were nominally calculable, *Philae*—in particular, its CDMS—had to be prepared for deviations from the predictions. We, as the CDMS hardware and software developer group, anticipated that the flight operators might have sporadic, time-restricted opportunities for intervention. The primary requirements were flexibility and onboard autonomy for serial and parallel sequencing, and control and harmonization of the operation of the five subsystems and nine scientific instruments.

The energy available for the science programs from the primary and secondary batteries was limited. The CDMS also had to cope with extreme environmental and operational conditions throughout the long-term-science phase. That phase,

driven solely by solar power and the rechargeable battery, required comprehensive thermal control and power flow management.

When Good Intentions Face Reality

As the International Academy of Astronautics noted, the *Philae* mission was “the first-ever trajectory development for a ballistic comet landing, the first on-comet operations, and the first cometary in-situ science collection.”⁴ However, not everything went as planned. Here, I explore some lessons learned from our and the *Philae* team's experiences with problems that occurred in the hardware and software and in mission operations control.

Lesson 1: No One Can Conquer Fate

Originally, this statement from Douglas Adams inspired us to “conquer fate”:

The major difference between a thing that might go wrong and a thing that cannot possibly go wrong is that when a thing that cannot possibly go wrong goes wrong, it

*usually turns out to be impossible to get at and repair.*⁵

Fault tolerance and the graceful degradation of vital subsystems in *Philae*, and particularly in the CDMS, were primary design goals. To help achieve these goals, we implemented a multilevel hardware and software scheme in the CDMS. The first level involved constructing a “thing” (system) that “cannot possibly go wrong.” Such a system—for example, the CDMS, has these elements:

- redundant hardware and the proper architecture of redundant elements;
- self-repairing capability, supported by the hardware architecture and specific software;
- robust fallback software with limited functionality (that is, communication capability); and
- reprogrammability of the acting software equipped with full functionality.

Such a scheme is in principle fault tolerant. However, in practice, it isn't. What if, as an extreme example, a buggy software version is accidentally uplinked to such a system and falls into a deadlock without the system being able to communicate and thus receive the corrected software? Such a system, thought to be fault tolerant, would then fail. To enable intervention (“get at and repair”) from the Earth, we extended the CDMS with triple-redundant emergency telecommand decoders, along with other basic functions to get out of deadlocks.

The irony of fate was that exactly at the final step of design, when we thought we had achieved perfect fault tolerance, we introduced

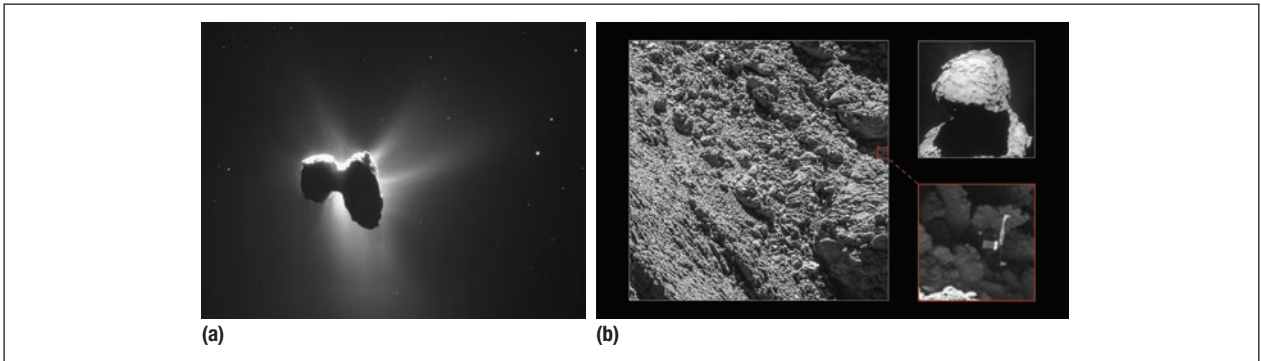


FIGURE 2. Images from the *Philae* mission. (a) Comet 67P/Churyumov-Gerasimenko. (b) *Philae* on the comet. (Images by the *Rosetta* spacecraft's navigation camera and Osiris camera.)

a source of error that remained unnoticed for a long time. A cyclic redundancy check code was supposed to protect the received telecommands from misinterpretation. Nevertheless, the telecommand decoders sometimes (but not often) misinterpreted bit-serially transmitted cross-coupled telemetry packets (which were feedback noise caused by improper harnessing) as true emergency telecommands. The software workaround turned out to be surprisingly simple (but energy consuming): both receiver units were kept powered simultaneously, thus eliminating the receiver buffers' vulnerability to cross-coupled noise.

Lesson 2: Being Prepared for the “Unbelievable”

During the cruise phase, we faced an astonishing incident. In a very narrow temperature range around the CDMS's thermal equilibrium (−27 degrees C), one of the dual-redundant processor units remained unpowered after being turned on.

As I mentioned before, the CDMS was required to be fully operable under extreme environmental conditions. Careful design and screening turned this requirement into practice. But in spite of the CDMS

passing all the environmental qualification tests, the problem remained hidden and only came to light accidentally during flight. Fortunately, the other processor unit automatically took over the missing one's functionality, demonstrating redundancy's vital role in critical systems. Moreover, the unpowered processor unit could be brought to life—”thanks to Douglas Adams” (see Lesson 1)—by a hardware-decoded emergency telecommand. Even more fortunately, the CDMS always worked outside the critical temperature range during comet operations.

Lesson 3: When Even Redundancy Is Useless

Because the success of anchoring was a mission-critical requirement, the anchoring-subsystem developer team and we had devoted much effort to its design, quality assurance, and implementation. In spite of all this preparation, a string of surprising engineering issues occurred during the cruise phase:

- One of the two touchdown event detectors—an accelerometer—was unusable because its sensitivity range coincided with the vibration of *Philae*'s flywheel.

- The touchdown sensor status was cleared after each evaluation attempt, eliminating the chances for repeated evaluation to exclude any transient errors.
- Doubts arose as to whether the sequence and timing of actions in the anchoring algorithm would be efficient enough to shoot the lander's harpoons.

These issues gave the two teams strong incentives to revise the anchoring strategy and control. We didn't have access to *Philae*'s flight hardware; all we had was the changeable software of the CDMS. With the telescope sensors in *Philae*'s dumping mechanism as the basis, an alternative touchdown detection algorithm was designed as a replacement for the unusable detector. By implementing an “Or-Majority” (Touchdown = A or MajorityOf(B, C, D)) voting scheme for four (A, B, C, D) touchdown event sources or paths, we made the detection of the touchdown event more robust against transmission errors and false alerts. We together with the anchor team also completely reworked the anchoring-control software algorithm, tested it on the ground, and uplinked it to the CDMS.

Even so, and even though *Philae* properly detected the touchdown event, the anchoring failed, which severely affected the rest of the mission.

Lesson 4: Conflicts Can Exist between Safety and Science

For a long time, the *Philae* team felt that the delivery strategy should meet the requirements of both maximal scientific throughput and battery redundancy (parallel use of the primary and secondary batteries). The latter requirement would have necessitated a brief descent.

To not violate *Rosetta*'s safety margins, the spacecraft ejected *Philae* relatively far from the comet, which resulted in a prolonged descent. The candidate scientific experiments for the descent required more energy than the secondary battery could provide. So, science won out over safety in terms of a redundant battery supply.

For landing-site-targeting reasons, regardless of whether *Philae*'s main or backup ejection mechanism was deployed, the team decided to set the changeable push-off velocity of the main ejection mechanism to be equal to that of the nonchangeable backup mechanism (0.19 m/s). The dominant factors in *Philae*'s delivery were *Rosetta*'s attitude and orbital velocity vector. Compared to these, *Philae*'s push-off velocity was nearly negligible. This suggests that both ejection mechanisms could have had a simpler but still robust design that produced an even lower nonchangeable push-off velocity.

Lesson 5: Being Unprepared for the Conceivable

The unexpected situation of an unanchored lander at an unknown location and attitude necessitated reshuffling—and to a large extent

discarding—the originally planned timelines of the prestored science sequences. By assessing risks and re-prioritizing scientific objectives, the *Philae* team was obliged to make a series of ad hoc decisions under time pressure before each radio link session, in a continuous day–night work regime of three to four days.

Before the on-comet phases of *Philae*, the team extensively analyzed potential failure sources and prepared lengthy documents with recovery procedures. After the mission's active phases, several papers were published on the impact of the failed anchoring, modeling *Philae* as a mechanical system. You might ask, why didn't the team do that beforehand? During the cruise phase, so many problems had emerged in conjunction with *Philae*'s anchoring (see Lesson 3) that it was doubtful whether it would succeed. More important, the possibility that *Philae* would come to rest tilted to one side, instead of being fixed firmly on its feet, appeared nonnegligible.

And yet no one appreciated that such a scenario was realistic but didn't necessarily have immediate fatal consequences. The entire *Philae* team suffered from a sort of group-think and didn't take measures in advance to prepare both the operational ground segment and (in particular) *Philae*'s onboard system.

On the whole, the team could have exploited the available battery energy more efficiently by shortening the standby periods by relying more on onboard autonomy. That would have allowed additional science experiments, even involving risk-taking actions if necessary. For example, we could have adjusted *Philae*'s attitude for better solar illumination and to take images from additional surface elements of the comet.

Lesson 6: The Revenge of Missed Opportunities

Throughout the first-comet-science phase, the entire *Philae* team was fully occupied with mastering *Philae*'s unexpected situation. So, it missed the opportunity to uplink science sequences and telecommands for subsystems control and autonomous scientific research tailored for the long-term-science phase. The first telemetry data after six months of hibernation reported that *Philae* was in good health, and no one reckoned on a stepwise degradation of its vital hardware subsystems, particularly the redundant radio communication units.

In addition, as the risk grew of completely losing *Philae*, the team arguably set the wrong priorities and adopted in part inadequate methods to achieve reliable contact with it. For example, because *Philae* was close to the sun at that time, the thermal and solar-power conditions were excellent, and the control software for quick battery recharging and even for an extended day–night work regime was obviously operable. Yet, the team had concerns that any premature battery discharging might lead to an irreparable deficit.

After not uplinking in advance the set of telecommands, the team showed again that it hadn't realized the urgent need to establish favorable conditions—including flown orbits of *Rosetta*—for commanding *Philae* to collect and downlink scientific data, instead of focusing on investigating how and why the telecommunication units were degrading. It was of the utmost importance to get a second set of science data for drawing conclusions on comet evolution.


The awareness that the lander did nothing useful in standby mode for

at least 2.5 months from wake-up until its last contact with Earth was distressing. In the end, sadly, all the telecommunication units must have broken down before the team could deploy successful countermeasures.

The *Philae* mission was a jump into the unknown. Besides the standard systematic procedures and workflow, innovative, heuristic ideas had a specific role during software development. In the beginning, the requirements to achieve all the scientific objectives and the technical constraints in such a complex system weren't fully clear. This led to our inability to complete the flight software in the less than three years of hardware and software design and implementation before the spacecraft launch. Afterward, during the cruise phase, we had to prepare the system for many nominal operational scenarios and emergency situations. All in all, we devoted more than 50 percent of development time to making the system—not only the CDMS but also *Philae*—as fault tolerant as possible.

Some of the lessons I described illustrate how design and implementation errors remained unrevealed before the mission launch, despite

ABOUT THE AUTHOR



ANDRÁS BALÁZS is an embedded-hardware-and-software system engineer at the Wigner Research Centre for Physics. Contact him at balazs.andras@wigner.mta.hu.

comprehensive on-ground testing and validation. This shows why software reprogrammability is so important.

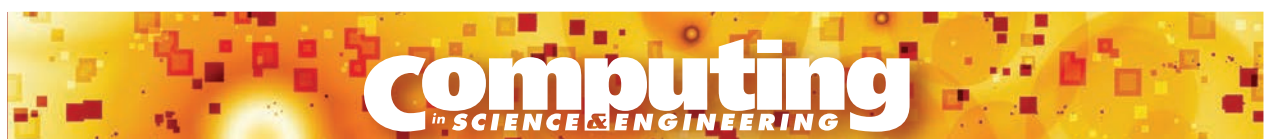
The *Philae* team exploited the onboard autonomy and flexibility in many respects, but not to the extent that would have been—in retrospect—purposeful. In the end, we might have been able to compensate for the relatively slow hardware degradation, at least to partly save the long-term-science phase. In hindsight, this proved a bridge too far for all of us. ☹

References

1. J.-P. Bibring et al., “The Rosetta Lander (‘Philae’) Investigations,” *Space Science Reviews*, vol. 128, nos. 1–4, 2007, pp. 205–220; doi:10.1007/s11214-006-9138-2.
2. A. Baksa et al., “Software on a Comet: The Philae Lander’s Central

- Onboard Computer,” *IEEE Software*, vol. 33, no. 2, pp. 13–16.
3. A. Balázs et al., “Command and Data Management System (CDMS) of the Philae Lander,” *Acta Astronautica*, Aug.–Sept. 2016, pp. 105–117; doi:10.1007/s11214-006-9138-2.
4. “The Laurels for Team Achievement Award 2015 to Philae Lander Mission,” Int’l Academy of Astronautics; <https://iaaweb.org/content/view/143/243>.
5. D. Adams, *Mostly Harmless*, Harmony Books, 1992.

This article originally appeared in IEEE Software, vol. 35, no. 4, 2018.



Subscribe today for the latest in computational science and engineering research, news and analysis, CSE in education, and emerging technologies in the hard sciences.

www.computer.org/cise

IDX, LLC seeks a Computer Programmer to work in Eugene, Oregon to address architectural issues of applications; analyze application designs; modify software; design, develop, modify proprietary application software systems. Mail resume to: IDX, LLC, 100 East Broadway, Eugene, OR 97401, ATTN: Celeste Marshall.



The University of Alabama in Huntsville

The Department of Computer Science at The University of Alabama in Huntsville (UAH) invites applicants for a tenure-track faculty position at the Assistant Professor level beginning January 2020. All applicants with a background in traditional areas of computer science will be considered; however, special emphasis will be given to applicants with expertise in cybersecurity, gaming, software engineering, cloud computing, and systems related areas.

A Ph.D. in computer science or a closely related area is required. The successful candidate will have a strong academic background and be able to secure and perform funded research in areas typical for publication in well-regarded academic conference and journal venues. In addition, the candidate should embrace the opportunity to provide undergraduate education.

The department has a strong commitment to excellence in teaching, research, and service; the candidate should have good communication skills, strong teaching potential, and research accomplishments.

UAH is located in an expanding, high technology area, in close proximity to Cummings Research Park, the second largest research park in the nation and the fourth largest in the world. Nearby are the NASA Marshall Space Flight Center, the Army's Redstone Arsenal, numerous Fortune 500 and high tech companies. UAH also has an array of research centers, including information technology and cybersecurity. In short, collaborative research opportunities are abundant, and many well-educated and highly technically skilled people are in the area. There is also access to excellent public schools and inexpensive housing.

UAH has an enrollment of approximately 9,500 students. The Computer Science department offers BS, MS, and PhD degrees in Computer Science and contributes to interdisciplinary degrees. Faculty research interests are varied and include cybersecurity, mobile computing, data science, software engineering, visualization, graphics and game computing, multimedia, AI, image processing, pattern recognition, and distributed systems. Recent NSF figures indicate the university ranks 30th in the nation in overall federal research funding in computer science.

Interested parties must submit a detailed resume with references to info@cs.uah.edu or Chair, Search Committee, Dept. of Computer Science The University of Alabama in Huntsville, Huntsville, AL 35899. Qualified female and minority candidates are encouraged to apply. Initial review of applicants will begin as they are received and continue until a suitable candidate is found.

The University of Alabama in Huntsville is an affirmative action/equal opportunity employer/ minorities/ females/ veterans/ disabled.

Please refer to log number: 19/20-545



IEEE TRANSACTIONS ON BIG DATA

► **SUBSCRIBE AND SUBMIT**

For more information on paper submission, featured articles, calls for papers, and subscription links visit: www.computer.org/tbd

TBD is financially cosponsored by IEEE Computer Society, IEEE Communications Society, IEEE Computational Intelligence Society, IEEE Sensors Council, IEEE Consumer Electronics Society, IEEE Signal Processing Society, IEEE Systems, Man & Cybernetics Society, IEEE Systems Council, and IEEE Vehicular Technology Society

TBD is technically cosponsored by IEEE Control Systems Society, IEEE Photonics Society, IEEE Engineering in Medicine & Biology Society, IEEE Power & Energy Society, and IEEE Biometrics Council





Call for 2019 Major Awards Nominations

Deadline: 1 October 2019

Help Recognize Computing's Most Prestigious Individuals

IEEE Computer Society awards recognize outstanding achievements and highlight significant contributors in the teaching and R&D computing communities. All members of the profession are invited to nominate individuals they consider most eligible to receive international recognition through an appropriate society award.

Charles Babbage Award

Certificate/\$1,000

In recognition of significant contributions in the field of parallel computation.

Computer Entrepreneur Award

Sterling Silver Goblet

Vision and leadership resulting in the growth of some segment of the computer industry.

Edward J. McCluskey Technical Achievement Award

Certificate/\$2,000

Contributions to computer science or computer technology.

Harry H. Goode Memorial Award

Bronze Medal/\$2,000

Information sciences, including seminal ideas, algorithms, computing directions, and concepts.

Hans Karlsson Award

Plaque/\$2,000

Team leadership and achievement through collaboration in computing standards.

Richard E. Merwin Award for Distinguished Service

Bronze Medal/\$5,000

Outstanding volunteer service to the profession at large, including service to the IEEE Computer Society.

Computer Pioneer Award

Silver Medal

Pioneering concepts and development of the computing field.

W. Wallace McDowell Award

Certificate/\$2,000

Recent theoretical, design, educational, practical, or other tangible innovative contributions.

Taylor L. Booth Award

Bronze Medal/\$5,000

Contributions to computer science and engineering education.

Computer Science & Engineering Undergraduate Teaching Award

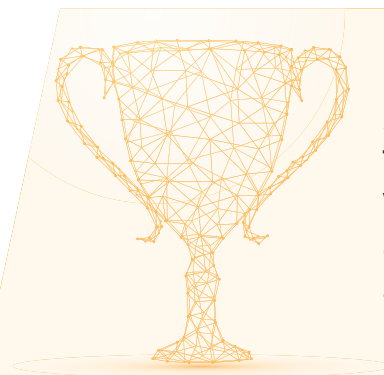
Plaque/\$2,000

Recognizes outstanding contributions to undergraduate education.

Harlan D. Mills Award

Plaque/\$3,000

Contributions to the practice of software engineering through the application of sound theory.



Nomination Deadline

Submit your nomination by
1 October 2019 to
www.computer.org/awards

Contact us at
awards@computer.org



IEEE
COMPUTER
SOCIETY

Education Awards Nominations

*Call for Award Nominations
Deadline: 1 October 2019*

Taylor L. Booth Education Award

A bronze medal and \$5,000 honorarium are awarded for an outstanding record in computer science and engineering education. The individual must meet two or more of the following criteria in the computer science and engineering field:

- Achieving recognition as a teacher of renown.
- Writing an influential text.
- Leading, inspiring, or providing significant education content during the creation of a curriculum in the field.
- Inspiring others to a career in computer science and engineering education.

Two endorsements are required for an award nomination.

Read more information at bit.ly/taylor-booth



Susan H. Rodger
Duke University
2019 Award Recipient



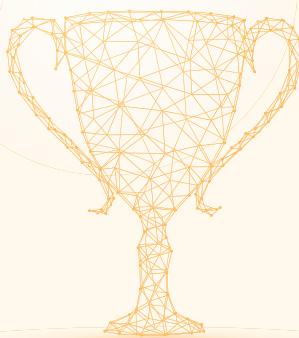
Robert R. Kessler
The University of Utah
2019 Award Recipient

Computer Science and Engineering Undergraduate Teaching Award

A plaque, certificate, and a honorarium of \$2,000 is awarded to recognize outstanding contributions to undergraduate education through both teaching and service and for helping to maintain interest, increasing the visibility of the society, and making a statement about the importance of undergraduate education.

The award nomination requires a minimum of three endorsements.

Read more award details at bit.ly/cs-eu



Nomination Deadline

Submit your nomination by
1 October 2019

Contact us at
awards@computer.org





Harry H. Goode Memorial Award

Call for Award Nominations

Deadline: 1 October 2019

2019 Harry H. Goode Memorial Award Recipient



Marilyn C. Wolf

Georgia Institute of Technology

For contributions to embedded, hardware-software codesign, and real-time computer vision systems.

Named for a pioneer and leader in the field of systems engineering, the **Harry H. Goode Memorial Award** was established to encourage further developments in and honor outstanding contributions to the field of information processing sciences. A bronze medal and \$2,000 are awarded by the IEEE Computer Society on the basis of achievements in the information processing field.

About Harry H. Goode

One of the first scientists to fully comprehend the powers and abilities of computers, Harry H. Goode formulated many principles of systems engineering and developed techniques for the design, analysis, and evaluation of large-scale systems. He was instrumental in initiating early systems projects, including the Typhoon computer and Whirlwind computer at MIT. He participated in the study that led to the creation of the Bomarc missile and conceived and developed the Air Defense Integrated System Project.

Nomination Requirements

This award requires a minimum of three endorsements. Nominations are being accepted electronically by **1 October 2019** to bit.ly/harry-goode.

Questions?

Visit bit.ly/harry-goode or contact awards@computer.org





Richard E. Merwin Award for Distinguished Service

*Call for Award Nominations
Deadline: 1 October 2019*

The Richard E. Merwin Award is the highest-level volunteer service award of the IEEE Computer Society for outstanding service to the profession at large, including significant service to the IEEE Computer Society or its predecessor organizations.

2018 Richard E. Merwin Award Recipient for Distinguished Service



Sorel Reisman

California State University,
Fullerton

*For sustained
contributions,
leadership, and
service to the
IEEE Computer
Society, IEEE, and
the computing
profession at large.*

About Richard Merwin

Richard Merwin was a pioneer in digital computer engineering who participated in the development of the ENIAC, MANIAC, and STRETCH computers. Despite a busy and productive technical career, Merwin found time to be active in professional societies, including the IEEE Computer Society, ACM, and AFIPS. His generosity of spirit and genuine helpfulness was an important element in the progress of the computer profession.

Award

A bronze medal and \$5,000 honorarium are awarded.

Presentation

The Richard E. Merwin Award is presented at the IEEE Computer Society's Annual Awards Ceremony.

Nomination Requirements

This award requires 3 endorsements.

Nominations are being accepted electronically by
1 October 2019 to bit.ly/richard-merwin

Questions?

Email awards@computer.org



IEEE Internet Computing

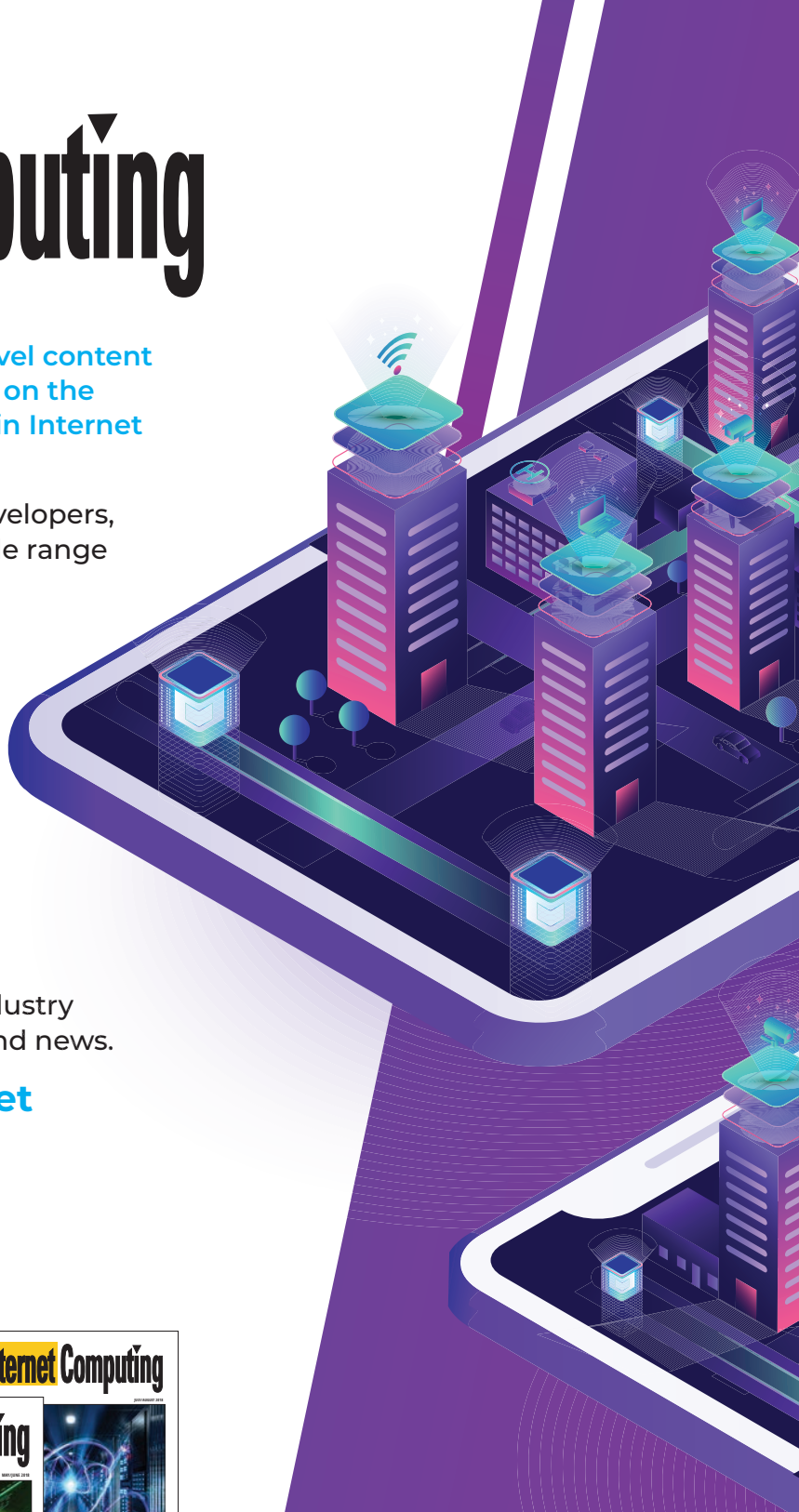
IEEE Internet Computing delivers novel content from academic and industry experts on the latest developments and key trends in Internet technologies and applications.

Written by and for both users and developers, the bimonthly magazine covers a wide range of topics, including:

- Applications
- Architectures
- Big data analytics
- Cloud and edge computing
- Information management
- Middleware
- Security and privacy
- Standards
- And much more

In addition to peer-reviewed articles, *IEEE Internet Computing* features industry reports, surveys, tutorials, columns, and news.

www.computer.org/internet



Join the IEEE Computer Society
for subscription discounts today!

www.computer.org/product/magazines/internet-computing





Conference Calendar



Questions? Contact conferences@computer.org

IEEE Computer Society conferences are valuable forums for learning on broad and dynamically shifting topics from within the computing profession. With over 200 conferences featuring leading experts and thought leaders, we have an event that is right for you.

Find a region:

Africa 
Asia 

Australia 
Europe 


North America 
South America 

SEPTEMBER



13 September

- EWDTTS (IEEE East-West Design & Test Symposium) 





15 September

- MODELS (ACM/IEEE 22nd Int'l Conf. on Model Driven Eng. Languages and Systems) 


19 September

- AVSS (16th IEEE Int'l Conf. on Advanced Video and Signal Based Surveillance) 
- ESEM (ACM/IEEE Int'l Symposium on Empirical Software Eng. and Measurement) 

23 September

- CLUSTER (IEEE Int'l Conf. on Cluster Computing) 
- PACT (28th Int'l Conf. on Parallel Architectures and Compilation Techniques) 
- RE (IEEE 27th Int'l Requirements Eng. Conf.) 
- SecDev (IEEE Secure Development) 

25 September


- HCC (IEEE Int'l Conf. on Humanized Computing and Communication) 

29 September


- ICSME (IEEE Int'l Conf. on Software Maintenance and Evolution) 

OCTOBER

1 October

- MCSoc (IEEE 13th Int'l Symposium on Embedded Multicore/Many-core Systems-on-Chip) 




2 October

- DFT (IEEE Int'l Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems) 


12 October

- MICRO (52nd Annual IEEE/ACM Int'l Symposium on Microarchitecture) 

14 October

- ISMAR (IEEE Int'l Symposium on Mixed and Augmented Reality) 
- LCN (IEEE 44th Conf. on Local Computer Networks) 
- VL/HCC (IEEE Symposium on Visual Languages and Human-Centric Computing) 

15 October

- AIPR (IEEE Applied Imagery Pattern Recognition Workshop) 


16 October

- FIE (IEEE Frontiers in Education Conf.) 

20 October

- VIS (IEEE Visualization Conf.) 

27 October

- ICCV (IEEE/CVF Int'l Conf. on Computer Vision) 

28 October

- EDOC (IEEE 23rd Int'l Enterprise Distributed Object Computing Conf.) ●
- ISSRE (IEEE 30th Int'l Symposium on Software Reliability Eng.) ●

NOVEMBER

4 November

- ICTAI (IEEE 31st Int'l Conf. on Tools with Artificial Intelligence) ▶

7 November

- SEC (IEEE/ACM Symposium on Edge Computing) ▶

8 November

- ICDM (IEEE Int'l Conf. on Data Mining) ▲

9 November

- FOCS (IEEE 60th Annual Symposium on Foundations of Computer Science) ▶

11 November

- ASE (34th IEEE/ACM Int'l Conf. on Automated Software Eng.) ▶

17 November

- ICCD (IEEE 37th Int'l Conf. on Computer Design) ▲
- SC19 (SC19: Int'l Conf. for High Performance Computing, Networking, Storage and Analysis) ▶

18 November

- BIBM (IEEE Int'l Conf. on Bioinformatics and Biomedicine) ▶

DECEMBER

3 December

- RTSS (IEEE Real-Time Systems Symposium) ▲

4 December

- IREHI (IEEE Int'l Rural and Elderly Health Informatics Conf.) ■

9 December

- AIVR (IEEE Int'l Conf. on Artificial Intelligence and Virtual Reality) ▶
- Big Data (IEEE Int'l Conf. on Big Data) ▶
- ISM (IEEE Int'l Symposium on Multimedia) ▶

10 December

- ISSPIT (IEEE Int'l Symposium on Signal Processing and Information Technology) ▲

16 December

- CDKE (IEEE Int'l Conf. on Conversational Data & Knowledge Eng.) ▶

2020

January

13 January

- ICCPS (Int'l Conf. on Cyber-Physical Systems) ●

February

3 February

- ICSC (IEEE 14th Int'l Conf. on Semantic Computing) ▶

18 February

- SANER (IEEE 27th Int'l Conf. on Software Analysis, Evolution and Reengineering) ▶

19 February

- BigComp (IEEE Int'l Conf. on Big Data and Smart Computing) ▲

22 February

- CGO (IEEE/ACM Int'l Symposium on Code Generation and Optimization) ▶

March

2 March

- WACV (IEEE Winter Conf. on Applications of Computer Vision) ▶

9 March

- DATE (Design, Automation & Test in Europe Conf. & Exhibition) ●
- IRC (4th IEEE Int'l Conf. on Robotic Computing) ▲ ●

Learn more about
IEEE Computer
Society Conferences

www.computer.org/conferences

IPDPS 2020 CALL FOR PAPERS

The five-day IPDPS program includes three days of contributed papers, invited speakers, industry participation, and student programs, framed by two days of workshops with peer reviewed papers that complement and broaden the main program. For full details, see www.ipdps.org.

Authors for the main conference are invited to submit manuscripts that present original unpublished research in all areas of parallel and distributed processing, including the development of experimental or commercial systems. Work focusing on emerging technologies and interdisciplinary work covering multiple IPDPS areas are especially welcome. Topics of interest include:

- **Parallel and distributed computing theory and algorithms (Algorithms)**
- **Experiments and practice in parallel and distributed computing (Experiments)**
- **Programming models, compilers and runtimes for parallel applications and systems (Programming Models)**
- **System software and middleware for parallel and distributed systems (System Software)**
- **Architecture**
- **Multidisciplinary**

• Abstracts due	October 7, 2019
• Submissions due	October 14, 2019
• Preliminary decisions	December 9, 2019
• Final submissions due	January 6, 2020
• Final notification	January 20, 2020

SPONSORED BY



IEEE COMPUTER SOCIETY

TCPP

Technical Committee on Parallel Processing

GENERAL CO-CHAIRS

Anu Bourgeois (*Georgia State University, USA*)

Ramachandran Vaidyanathan (*Louisiana State University, USA*)

PROGRAM CHAIR

Yuanyuan Yang (*NSF and Stony Brook University, USA*)

PROGRAM AREA CHAIRS AND VICE CHAIRS

• Algorithms:

Xiaotie Deng (*Peking University, China*) and
Songtao Guo (*Chongqing University, China*)

• Architecture:

Ahmed Louri (*George Washington University, USA*) and
Avinash Karanth (*Ohio University, USA*)

• Experiments:

Xin Yuan (*Florida State University, USA*) and
Scott Pakin (*Los Alamos National Laboratory, USA*)

• System Software:

Alan Sussman (*NSF and University of Maryland, College Park, USA*) and
Zhiling Lan (*Illinois Institute of Technology, USA*)

• Programming Models:

Rudolf Eigenmann (*University of Delaware, USA*)
Zhiyuan Li (*Purdue University, USA*)

• Multidisciplinary:

Manish Parashar (*NSF and Rutgers University, USA*)
Ivona Brandić (*Vienna University of Technology, Austria*)

New Orleans is one of the most eccentric and lively cities in the world. Whatever your interests are, New Orleans has you covered. From its diverse culture, distinctive cuisine, rich history, colorful celebrations, live music, vibrant nightlife, and world-class restaurants, there is something for everyone. It is home to a number of engaging museums, including the World War II Museum, the New Orleans Museum of Art, the Historic Voodoo Museum, Mardi Gras World, and the Pharmacy Museum. A visit wouldn't be complete without a swamp tour, Mississippi river cruise and a stop at Café du Monde for beignets. Join IPDPS at the Hilton New Orleans Riverside in 2020 to find out what makes New Orleans so unique and special.