AFRL-RI-RS-TR-2020-007

X-PLORE: COMBINING MODEL-DRIVEN ENGINEERING, BIO-INSPIRATION AND FORMAL ANALYSIS TO MITIGATE UNCERTAINTY IN HIGH ASSURANCE SOFTWARE SYSTEMS

UNIVERSITY OF MICHIGAN

JANUARY 2020

FINAL TECHNICAL REPORT

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED

STINFO COPY

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

NOTICE AND SIGNATURE PAGE

AFRL-RI-RS-TR-2020-007 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE CHIEF ENGINEER:

/ S /
CARL R. THOMAS
Work Unit Manager

/ S /
LAUREN HUIE-SEVERSKY
Technical Advisor, Computing
and Communications Division
Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
JANUARY 2020	FINAL TECHNICAL REPORT	AUG 2016 – JUN 2019

4. TITLE AND SUBTITLE

X-PLORE: COMBINING MODEL-DRIVEN ENGINEERING, BIO-INSPIRATION AND FORMAL ANALYSIS TO MITIGATE UNCERTAINTY IN HIGH ASSURANCE SOFTWARE SYSTEMS

5a. CONTRACT NUMBER FA8750-16-2-0284

5b. GRANT NUMBER N/A

5c. PROGRAM ELEMENT NUMBER 62788F

6. AUTHOR(S)

Betty Cheng
Philip McKinley

5d. PROJECT NUMBER T2RS

5e. TASK NUMBER AM

5f. WORK UNIT NUMBER SU

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

University of Michigan
426 Auditorium Rd Rm 2
Lansing, MI 48824-2600

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Research Laboratory/RITA
525 Brooks Road
Rome NY 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S) AFRL/RI

11. SPONSOR/MONITOR'S REPORT NUMBER AFRL-RI-RS-TR-2020-007

12. DISTRIBUTION AVAILABILITY STATEMENT

Approved for Public Release; Distribution Unlimited. This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Increasingly, cyber-physical systems are expected to deliver acceptable and trusted behavior despite highly dynamic and uncertain operating conditions. The X-PLORE project investigated the integration of evolutionary search algorithms with formal analysis methods in order to enhance system robustness and resiliency, while identifying corner cases that might lead to system failure under certain conditions. Evolutionary search algorithms operate in an open-ended manner, unconstrained by human bias and preconceptions. Combining this capability with formal analysis enables discovery of unintuitive solutions to design problems as well as situations that might cause the system to behave in an unintended manner after deployment. This report describes the main capabilities developed in the project along with the results of studies in applying those methods to autonomous vehicles of different scales. A list of publications and presentations resulting from this research is also provided.

15. SUBJECT TERMS

cyber-physical systems, autonomous systems, resilient systems, search-based software engineering, formal analysis, external and internal uncertainty

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			**CARL R. THOMAS**
U	U	U	UU	22	19b. TELEPHONE NUMBER (*Include area code*) **N/A**

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. Z39.18

TABLE OF CONTENTS

Section **Page**

1 SUMMARY

Increasingly, cyber-physical systems are expected to deliver acceptable and trusted behavior despite highly dynamic and uncertain operating conditions. The X-PLORE project investigated the integration of evolutionary search algorithms with formal analysis methods in order to enhance system robustness and resiliency, while identifying corner cases that might lead to system failure under certain conditions. Evolutionary search algorithms operate in an open-ended manner, unconstrained by human bias and preconceptions. Combining this capability with formal analysis enables discovery of unintuitive solutions to design problems as well as situations that might cause the system to behave in an unintended manner after deployment. This report describes the main capabilities developed in the project along with the results of studies in applying those methods to autonomous vehicles of different scales. A list of publications and presentations resulting from this research is also provided.

2 INTRODUCTION

A major problem facing software intensive systems is that current software engineering tools do not adequately support the discovery of "unlikely-but-possible" situations that might produce unexpected behavior in the deployed system. To address these needs, it is necessary to leverage automated search technology to enable the software developer to foresee potential problems and enhance system requirements accordingly. The X-PLORE project investigated and developed approaches to manage uncertainty in order to enhance software assurance. In order to maximize assurance impact, while reducing development and maintenance costs, the project focused on requirements analysis throughout the development process. These methods enable the developer to detect, identify, and mitigate situations (arising from unanticipated environmental conditions, unwanted feature interactions, and combinations thereof) that would otherwise lead to system malfunction and possible failure.

3 METHODS, ASSUMPTIONS, AND PROCEDURES

A robust computing-based system must be able to monitor its environment, adapt to changing conditions, withstand component failures and attacks, and continuously deliver acceptable and trusted behavior. However, the designer of such a system is faced with a challenging set of tasks: anticipating and characterizing conditions in which the system will operate; enabling the system to accommodate changing requirements, both functional and non-functional; and ensuring that existing and newly added functionality (possibly from third-party providers) will produce expected overall system behavior. These tasks are particularly difficult for systems that must operate safely and securely in the face of uncertainty.

The primary objective of the X-PLORE project has been to harness the power of evolutionary computing (EC) and symbolic analysis (SA) in software engineering in order to advance the field to be able to support the modeling, analysis, and mitigation of two broad categories of uncertainty in software systems. *External uncertainty* includes aspects of the operating environment (including adverse conditions in the physical environment and unexpected human interaction) that can lead to suboptimal, and possibly catastrophic, results as the system tries to adapt to mischaracterized conditions. *Internal uncertainty* includes (unwanted) feature interaction among system components as well as unexpected component behavior due to software updates and reconfiguration.

Increasingly, software systems operate in highly dynamic and unpredictable environments. This situation is self-evident with the proliferation of cyber-physical systems, which combine computing technology with sensors and actuators that interact with the physical world. However, even systems that operate entirely within cyberspace need to account for changing network and load conditions, component failures, and exposure to a wide variety of cyberattacks. Moreover, the complexity of software in terms of features and their interactions is ever growing. The challenge of guaranteeing the integrity of the system is exacerbated by the fluid nature of software, where functionality may necessarily need to change in response to new requirements, patches, and other updates. Uncertainty has been defined as the difference between information that exists in an executing system and the information that is both measurable and available to the system at a given point in time. The challenge for the software developer is to (1) foresee situations in which the system will execute and (2) design the system to operate as intended despite internal uncertainty (including unwanted feature interactions) and external uncertainty (including unlikely-but-possible environmental conditions).

The hallmark of the X-PLORE project is to introduce a fundamental biological principle, evolution, into the development process for cyber-physical systems that explicitly addresses different types and levels of uncertainty. Evolutionary search algorithms operate in an open-ended manner, unconstrained by human bias and preconceptions. As a result, such algorithms are able to discover unintuitive solutions to engineering problems, as well as reveal potential problems the developer might not have considered. This is a paradigm shift from the traditional approach to design, which is limited to a developer's prior knowledge and experience, in order to enhance the robustness and resilience of the target cyber-physical systems. The basic idea is to combine evolutionary search with formal analysis to discover, at development time, situations that might cause the system to behave in an unintended manner after deployment, as well as ways to mitigate those situations. In order to maximize the benefits of this approach and manage the complexity of the software artifacts under analysis, we focus on the requirements stage of model-driven engineering, thereby ensuring traceability of system behavior to design decisions and functional objectives. When an unwanted behavior is revealed, the requirements engineer can address the problem by amending the goals and/or requirements of the system, including identifying and characterizing conditions that will trigger software adaptation during execution. For high-assurance systems that require certification, adaptation comes in the form of mode changes, where each mode handles different environmental and operating conditions.

4 RESULTS AND DISCUSSION

This section describes the key capabilities that we developed for the X-PLORE project to address internal and external uncertainty for trusted and resilient systems. We describe the specific techniques that we developed to address internal uncertainty and external uncertainty, respectively. We conclude the section with descriptions of the evolutionary-based simulation environment and the physical demonstration platforms that we used to validate our techniques.

4.1 Internal Uncertainty

This section describes several activities that focused on addressing internal sources of uncertainty: incomplete requirements and unwanted feature interactions. In most cases, we started with a formal analysis approach of goal-based model representations of requirements specifications, proceeding to the use of evolutionary-based techniques to identify multiple counterexamples, and finally progressing to the use of utility function-based monitoring for run-time detection and analysis of self-adaptive systems. In all cases, we leveraged our collaborations with the automotive industry to obtain real-world, industrial-strength specifications of applications to illustrate and validate our techniques.

4.1.1. Automated Detection of Incomplete Requirements. We developed a technique, *Ares-EC* that makes use of evolutionary computation and symbolic analysis to automatically assess requirements completeness. Ares-EC leverages the previously developed Ares technique that uses symbolic analysis to detect the existence of incomplete requirements decomposition, where Ares returned a single counterexample if a given requirement had been incompletely decomposed. With Ares-EC, we empirically explored how search-based techniques can provide a *suite* of counterexamples to account for different environmental conditions that help to uncover requirements gaps. We explored multiple treatments for comparison in order to address the "needle in the haystack problem" and "multiple optima" in the solution space. The treatments included symbolic analysis (SA) that would return a single counterexample if an incompleteness exists; using EC alone (EC) that returned no counterexamples in the time allotted due to the "needle in the haystack" problem; symbolic analysis and then EC (SAEC) to look for multiple counterexamples close to the counterexample found by SA; and then EC with periodic introduction of SA-identified counterexamples (PSAEC) to address the situation where multiple optima may be possible in the solution space. Our conclusion from this study is that PSAEC is the best approach to identifying a suite of counterexamples for incomplete requirements. The suite of counterexamples provides much more information to the developer in *how* to revise the requirements and environmental assumptions to improve overall system resilience and trustworthiness.

> This work was published and presented at the *9th International Symposium on Search-based Software Engineering (SSBSE)*, held at Paderborn, Germany, co-located with the *Foundations of Software Engineering/European Software Engineering Conference (FSE/ESEC), September 2017, pp. 49-64.*

We then extended our work with automatic detection of incomplete requirements (i.e., Ares and Ares-EC) to perform the analysis at run time. The validity of run-time monitoring of system goals and requirements depends on both the completeness of the requirements, as well as the correctness of the environmental assumptions. Often specifications are built with an idealized view of the environment that leads to incomplete and inconsistent requirements related to non-idealized behavior. Worse yet, requirements may be measured as satisfied at run time despite an incomplete or inconsistent decomposition of requirements due to violated environmental assumptions. While methods exist to detect incomplete requirements at design time, environmental assumptions may be invalidated in unexpected run-time environments causing undetected incomplete decompositions. We developed *Lykus*,* an approach for using models at run time to detect incomplete and inconsistent requirements decompositions at run time. We demonstrated this approach by applying Lykus to a requirements model of an adaptive cruise control system from one of our industrial collaborators. Lykus was able to automatically detect instances of incomplete and inconsistent requirements decompositions at run time.

> This work was described in a paper published and presented at the *12th International Workshop on Models at Run Time (M@RT)*, as part of the *International MODELS (Model Driven Engineering Languages and Systems) Conference*, held in Austin, Texas, September 2017, pp. 201-209.

4.1.2. Detecting n-way unwanted feature interactions. We have developed Phorcys, an approach to use symbolic analysis to detect n-way feature interactions (FIs) at the requirements level. This capability is increasingly important with autonomous systems, where increasing numbers of onboard features are being added to provide a broad spectrum of capabilities. When individually tested and analyzed, a given feature may provide the appropriate behavior. But as the feature is deployed "in the wild," unexpected and/or adverse environmental conditions may behave unexpectedly. This problem is exacerbated and compounded when considering multiple features operating collectively in potentially unknown conditions. Phorcys tackles this problem from a design-time point of view by applying symbolic analysis to feature requirements based on expected environmental conditions. Here, we are exploring how the composition of features may reveal unknown and unwanted n-way FIs. Phorcys differs from other FI techniques in that it detects a possible FI as well as the *cause* of the FI. In addition, due to the computational complexity typically associated with the analysis, most FI detection techniques focus on pairwise FI. However, studies have shown that in complex systems, many FIs only manifest in three or more features. Therefore, Phorcys supports the detection of n-way FI. We applied Phorcys to a composite braking subsystem that illustrates problems similar to those that caused the unintended acceleration associated with the Toyota Prius and other Toyota models.

> This work was published and presented as: Byron DeVries, Betty H. C. Cheng: "Automatic Detection of Feature Interactions Using Symbolic Analysis and Evolutionary Computation" at Proceedings of *the 18th IEEE International Conference on Software Quality, Reliability, and Security (QRS)*, July 2018, Lisbon, Portugal, pp. 257-268.

Next, we studied how evolutionary computation (EC) could be used to enhance the FI detection technique to better capture the impact of uncertainty on cyber-physical and autonomous systems. Specifically, Phorcys-EC leverages our experience with using EC to detect requirements incompleteness, where we explored how EC could be used to identify multiple FI counterexamples that better identify the range of environmental conditions that cause an unwanted n-way FI. As before, we conducted numerous empirical studies to explore how enhancing our symbolic analysis-based approach for detecting FI can be enhanced by use of EC techniques. Specifically, we compared the following techniques: using symbolic analysis only (SA), using EC only (EC), and

* *Lykus* is the mortal son of *Ares*, who sacrificed strangers to his father.

using a combination of SA and EC (SA+EC). When using EC-based techniques, common challenges to overcome include: avoiding the 'needle in a haystack problem' (looking for one solution within a large solution space); lack of a gradient between optimal and non-optimal results, thus providing no basis for improvement during evolution and degrading to random search; and dependencies between variables may necessitate changes with additional variables to maintain optimality. In order to overcome these challenges, we combined SA and EC. Specifically, we initially use SA to seed the evolutionary process and then periodically inject SA-detected FIs to the evolutionary process, with a particular focus on diversity of the FI results. Using this approach, we were able to obtain diversity along two dimensions and identify FIs amongst the most diverse collection of features and environmental conditions.

4.1.3. Run-time Monitoring and Detection of Feature Interactions. The validity of systems at run time depends on the features included in those systems operating as specified. However, when feature interactions occur, the specifications no longer reflect the state of the run-time system due to the conflict. While methods exist to detect feature interactions at design time, conflicts that cause features to fail may still arise when new detected feature interactions are considered unreachable, new features are added, or an exhaustive design-time detection approach is impractical due to computational costs. We have developed *Thoosa*,[1] an approach for using models at run time to detect features that can fail due to n-way feature interactions at run time and thereby trigger mitigating adaptations and/or updates to the requirements. We demonstrated this approach by applying *Thoosa* to an industry-based automotive braking system comprising multiple subsystems.

Thoosa analyzes features represented in goal models that hierarchically decompose a high-level goal down to individual requirements. ***Thoosa*** executes generated logic, in the form of C++ code that analyzes each feature with respect to the current feature combinations of the goal model and identifies if the analyzed feature can fail due to a conflict in one or more requirements to be satisfied at run time. Each feature is analyzed for failure due to a feature interaction. Where previous run-time feature interaction detection techniques indicate that a feature interaction exists, ***Thoosa*** identifies which features fail due to the feature interaction.

> This work was published and presented as: Byron DeVries, Betty H. C. Cheng: ``Run-time monitoring of self-adaptive systems to detect N-way feature interactions and their causes'' in the *Proc. 13th IEEE International Symposium on Software Engineering and Self-Managing Systems (SEAMS),* May 2018, Gothenburg, Sweden, pp. 94-100.

[1]Thoosa is a Greek sea nymph associated with swiftness and the daughter of Phorcys.

4.1.4. Detecting Unwanted Interactions between Non-functional and Functional Properties. Non-functional requirements are intended to ensure the non-functional properties of the system under development. However, non-functional properties of the system often crosscut functional and non-functional requirements. These cross-cutting concerns are dispersed throughout the requirements. This dispersion renders manual insertion of the non-functional concerns difficult and error prone. We have developed *Soter*[2], a method for aspect-oriented modeling of non-functional requirements and properties, which applies a symbolic analysis-based approach to detect unwanted interactions between non-functional properties and/or functional requirements. We demonstrated this approach by applying *Soter* to detect unwanted interactions among aspect-oriented safety and performance models and the requirements of an industry-based automotive braking system.

Soter leverages the similarity between aspect and feature interactions by translating the aspect-oriented safety requirements into features representing the safety requirements. These additional safety features are woven into the existing GORE (goal-oriented requirements engineering) model. Feature interaction detection analysis is applied to each of the functional features and safety features to determine if they cause an interaction or safety violation. The counterexamples for each detected interaction amongst the safety features are classified according to the safety model properties and provided to the system designer to guide the revision of the functional and safety requirements. For example, an Adaptive Cruise Control (ACC) may have a safety requirement to avoid collisions with other cars depending on a specific proximity, while the requirements model for the ACC also has requirements to maintain the driver's desired speed. The safety requirement to avoid a collision can be violated when the driver's desired speed is maintained and therefore an interaction exists between the requirements to avoid collision and maintain the driver's desired speed. ***Soter*** provides a method for modeling aspect-oriented safety goals that includes goal decomposition strategies that provide functional mitigations. ***Soter*** recombines the safety property and an optional mitigation decomposition to create safety features that represent one of the following cases:

- **Safety Properties:** safety invariants of the system with no mitigation,
- **Weak Mitigations:** mitigations that are applied when safety properties are violated, and
- **Strong Mitigations**: mitigations that are applied to ensure safety properties are never violated.

These safety features are defined separately (i.e., as aspect models) from the traditional requirements decomposition hierarchy and are subsequently woven into the relevant portions of the requirements model automatically. Counterexamples representing interactions between functional and non-functional features are generated by a feature interaction detection tool, ***Phorcys***, and categorized by ***Soter*** as to whether the safety properties, the mitigation objectives, or both were violated.

[2] ***Soter***: Greek god for safety, deliverance from harm.

This work was published and presented as: Byron DeVries and Betty H.C. Cheng, "Goal-Based Modeling and Analysis of Non-Functional Requirements" in the *Proc. of* IEEE/ACM *International Conference (Model Driven Engineering Languages and Systems (MODELS)*, held in Munich, Germany, September 2019.

4.2 External Uncertainty

In order to investigate the impact of uncertainty on adaptive systems, we explored the use of evolutionary techniques to generate a broad range of environmental conditions, with an emphasis on discovering a diverse range of system behavior. The behavior discovered ranged from "perfect" (expected behavior) to unacceptable, failing behavior, including behavior in the "grey" area that requires additional analysis to determine whether the behavior is acceptable or needs to be mitigated. We looked at how natural environmental conditions impact both functional and non-functional requirements. And we initiated preliminary investigations into security vulnerabilities on adaptive systems, as part of the TRSYS Phase 3, integration projects with Gamble from Tulsa.

4.2.1. Enki[3]: a configurable platform for exploring behavioral uncertainty due to environment and system conditions. Our approach to identifying unlikely-but-possible conditions that can lead to system failure is based on *novelty search*. Unlike evolutionary algorithms that evaluate individuals based on fitness with regard to one or more tasks, novelty search proportionately rewards individuals whose phenotypes (e.g., behavior in the case of cyber-physical systems) are *most different* from those previously discovered. An archive is maintained to record such individuals from each generation, ultimately producing a set of solutions with widely diverse phenotypes. We previously developed a novelty search tool called LOKI and used it to generate operating contexts for industry-provided automotive software. In the X-PLORE project, we developed a more general tool called Enki. Whereas LOKI is tightly integrated into the target system, Enki is completely standalone and applicable to any target system. Not only can Enki discover dangerous combinations of conditions, unwanted feature interactions, and new attack vectors, but it can improve machine learning performance by generating training data that reflects conditions not well represented in the original training data. Enki is a configurable and parameterized framework that enables users to specify sources of uncertainty (and range of values), define a plant models (e.g., a simulation frameworks), and parameterize their diversity needs (e.g., for both environmental conditions and system behavior). We have validated the framework with in-house projects, some of which previously studied using LOKI, whose uncertainty exploration capabilities were woven into the application under study.

One of our original project objectives was to use LOKI-like search to assist in the discovery of mode boundaries and modes of autonomous vehicles. Enki development was motivated by discussion at an early TRSYS PI meeting on ways to make the capability applicable to many types of cyber-physical systems. The nuances in the environmental conditions (i.e., external uncertainty) and onboard features behind the rover and quad copter that prompted us to make transform LOKI into a more generalized framework. Enki development focused on extracting the LOKI functionality out of the application, developing well-defined interfaces for using LOKI

[3] Sumerian god for mischief.

capabilities, such as sources of uncertainty, execution environment for application, and configuration of the evolutionary search process for identify diverse combinations of environmental and onboard conditions. Enki is designed to apply novelty search within a blackbox automation tool to help assess and improve an SUT (system under test) in the presence of environmental uncertainty. Enki requires a simulation executable to evaluate the SUT and a specification of the simulation's input and output parameters.

Enki is designed to be a general technique to be used with any type of system. In one of our first studies, we applied Enki to generate novel training data for ***machine learning (ML) algorithms used to classify images for autonomous vehicles***. Indeed, one of the biggest challenges facing the autonomous vehicles area is the resiliency of the classification algorithms in the face of (environmental) uncertainty,that is by definition limited by the scope and coverage of training data. We used Enki to explore two important research questions: 1) Can Enki generate useful test data that detects weaknesses in existing ML for handling uncertainty conditions; 2) Can Enki generate synthetic training data that can be used to train existing MLs to improve their behavior and improve their overall resilience to uncertainty in the environment. The current target for the ML is deep neural networks (DNNs) given their common use for classification problems, such as what is needed for obstacle avoidance and other onboard camera-based functions for autonomous vehicles.

We started by using Enki to support the generation of *synthetic testing data* to assess the utility of machine learning algorithms (currently focusing on image classification techniques, similar to those used for autonomous vehicles) in the face of uncertain environmental conditions. Applying novelty search synthetically generate testing data enables exploration of the broadest diversity in the behavior of the ML algorithms. This diversity assists in uncovering weaknesses and gaps in the ML behavior that have not been uncovered by the existing training data. Results indicate that Enki can generate a broad range of test data that detrimentally decreases the accuracy of the ML algorithm when applied to the CIFAR-10 benchmark data set. Even when compared to randomly generated test data, Enki is still better able to increase the accuracy of the ML algorithms. Our results show that by using Enki, we can generate environmental conditions such as decreased lighting, haze, and the presence of rain to construct a set of conditions that negatively impact the performance of a CIFAR-10 DNN. Specifically, we showed that a DNN that demonstrates a 91% accuracy on the default benchmark test data can be shown to only have a 10% accuracy when exposed to environmental effects introduced by Enki. **This result indicates that Enki can be effective in identifying weaknesses in existing ML algorithms when applied to uncertainty conditions.** This capability comes from the novelty-based technique used to generate test data that are not covered by the existing test data used to validate ML techniques.

We next explored whether Enki could be leveraged to improve the performance of existing ML techniques to make them more resilient to uncertainty. Towards this end, we used Enki to generate novel training data that focused on diverse behavior in the ML. Again, through empirical analysis, we found that the ML that was trained against Enki-generated data and the existing training data performed much better than the MLs that were only trained with the original training data or those that were only trained against random training data. Specifically, the synthetic training data generated by Enki was able to improve the accuracy of the DNN up to 76% in the presence of comparable adverse environmental effects. From a regression point of view, the Enki-trained ML also performed comparably to the original ML when applied to the original testing data and to the random test data. As such, the Enki-trained ML performed better for the original test data, random test data, and the Enki test data, thus making the overall autonomous system more robust and resilient to environmental uncertainty.

> The results describing the Enki work were published and presented as: M. Langford, B. H. C. Cheng: "Enhancing Learning-"Enabled Software Systems to Address Environmental Uncertainty", at the *IEEE International Conference on Autonomic Computing (ICAC2019*), in Umea, Sweden, June 2019, pp. 115-124

4.2.2. Run-Time Adaptation. We also explored an evolutionary-based approach to support run-time adaptation when making tradeoffs with non-functional properties (e.g., performance, cost, and reliability) in response to environmental uncertainty. This work, performed in conjunction with collaborators at Oakland University, developed an automated technique to use search-based techniques to identify reconfiguration options when analyzing goal-based models that explicitly represent the contribution of functional requirements to the satisficement of non-functional properties. This type of modeling enabled us to make tradeoffs between different adaptation configurations that are functionally comparable in satisfying system requirements, but differ in their non-functional impact. This work can be leveraged when defining modes of behavior and associating their corresponding impact on non-functional properties.

> The results of this work were published and presented as: Kate M. Bowers, Erik M. Fredericks, Betty H. C. Cheng: "Automated Optimization of Weighted Non-functional Objectives in Self-adaptive Systems," *Proceedings of 10th International Symposium on Search-Based Software Engineering, (SSBSE 2018),* pp. 182-197.

4.2.3. MAPE-SAC Framework: As part of the Phase 3 and a TRSYS Program Integration effort, in collaboration with researchers at the University of Tulsa, we initiated work into developing an adaptive framework to manage security assurance cases – termed MAPE-SAC. The work is leveraging the adaptive framework developed by Kephart and Chess on managing adaptive systems termed MAPE-K loop (to capture the monitoring, analysis, planning, and execution process used to dynamically adapt a self-adaptive system. The key insight with this project is to develop a synergistic relationship between managing assurance cases for the functional aspects (MAPE-K) and the security aspects (MAPE-SAC) of a trusted and resilient system. This early work illustrated how to manage the adaptations of security assurance cases (SACs) that are based on NIST communication protocols for secure communication, such as that used for autonomous systems.

> The preliminary results were published and presented as: Sharmin Jahan, Matthew Pasco, Rose F. Gamble, Philip K. McKinley, Betty H. C. Cheng: "MAPE-SAC: A Framework to Dynamically Manage Security Assurance Cases." *Proc. IEEE 4th International Workshops on Foundations and Applications of Self* Systems FAS*W@SASO/ICAC 2019*: 146-151.

Extended version of the work has been submitted: Sharmin Jahan, Ian Riley, Charles Walter, Rose F. Gamble, Matthew Pasco, Betty H.C. Cheng, and Philip K. McKinley, "MAPE-K/MAPE-SAC:Interaction Framework for Adaptive Security Assurance Cases,", submitted to special issue *Special Issue on Self- Protecting Systems in Journal of Future Generation Computer Systems*. July 31, 2019

4.3 Demonstration/Validation

Many of our investigations in the X-PLORE project fall within the field of evolutionary robotics (ER), which harnesses the open-ended search capabilities of evolutionary algorithms to optimize (or, in the case of Enki, detect problems in) robot design and operation. In ER, an artificial genome specifies the robot's control system and possibly aspects of its morphology. Individuals in a population are evaluated with respect to one or more tasks, with the best performing individuals selected to pass their genes to the next generation. Simulation is typically used to evaluate individuals, greatly reducing the time to evolve solutions while avoiding possible damage to physical robots. Over the past two decades, evolutionary approaches have yielded effective controllers and physical structures for a variety of crawling, swimming, and flying robots. Despite these advances, ER has had little effect on mainstream robotics, in part due to the complexity of the evolutionary algorithms and the corresponding use of simple models of robots and their environments.

A goal of the X-PLORE project was to ensure that the techniques developed can be applied to state-of-the-art cyber-physical systems. The broader robotics community is developing autonomous systems that integrate advanced computing technology, machine learning algorithms, and multimodal sensing (e.g., radar, lidar, GPS) to interpret and help navigate complex real-world environments. Such systems are typically controlled by complex software infrastructures, such as the Robot Operating System (ROS). ROS is often coupled with the Gazebo physics- based simulator, which provides tested models of many commercially available hardware components. An advantage of ROS is that code from a simulated robot can be inserted directly into the corresponding physical robot. Ensuring the applicability of X-PLORE technologies to such complex systems required development of physical vehicles as well as an evolutionary robotics framework to enable exploration of the design space for both hardware and software configuration.

4.3.1. Physical Vehicles. In the X-PLORE project we developed two physical vehicles. The first is a 1:16-scale autonomous rover based on a design from a group at BBN. The rover is modeled after the Erle-Rover, a commercially available 1:16-scale car-like robot controlled by ROS, with a maximum speed of approximately 25 mph. The rover was assembled in-house and comprises a commercial chassis augmented with a custom mounting board to hold sensors, instruments, and battery packs. To protect the on-board electronics from impact, a roll cage has also been installed. The 4-wheel drive rover is governed by a Pixhawk Mini autopilot unit (with 32-bit ARM processor, 2MB flash memory, GPS unit, dual IMUs) and includes a 2040 brushless Inrunner motor, high-torque steering servo, and Maxbotix sonar sensors. This platform enables the investigation of several questions related to resilient sensing, navigation and locomotion**.**

The second platform is the MSU version of AutoRally, which was constructed in-house over several months. MSU AutoRally is based on a design provided by researchers at Georgia Tech. The completed platform weighs 46 lbs and has a top speed of 60 mph. Default sensors include a high-precision IMU, GPS, Hall-effect rotation sensor on each wheel, and two front-facing machine vision cameras. The computing resources are housed in a custom compute box and include an Intel Skylake Quad-core i7, 32GB DDR4 RAM, 2TB SSD storage, an Nvidia GTX 1050 Ti GPU for real-time image processing, and WiFi and XBee network interfaces. In the future, we plan to add at least one lidar unit to the vehicle. The MSU version of AutoRally is fully compatible with the ROS navigation stack, which takes in sensor and odometry data and outputs throttle and steering commands to produce navigation from the platform's current location to a goal pose (consisting of location and heading). The ROS navigation stack maintains both a global and local cost map of the environment, updating the presence of obstacles in real-time, allowing it to plan paths to goal locations while avoiding both stationary and moving obstacles. This vehicle and the simulation of it in Evo-ROS enables studies of navigation, obstacle avoidance, operation under adverse conditions, and mitigation of internal and external uncertainty.

4.3.2. Evo-ROS Evolutionary Robotics Platform. To apply evolution to these platforms, we needed simulated versions, which we developed by extending Gazebo simulations of the Erle-rover and the Georgia Tech AutoRally vehicle. To enable evolution for ROS-based robots, we developed Evo-ROS, which is intended as a bridge between evolutionary robotics and the broader robotics community. Specifically, Evo-ROS (1) provides researchers in evolutionary robotics with access to the extensive support for real-world systems and components available with ROS, and (2) enables ROS developers, and more broadly robotics researchers, to take advantage of evolutionary search during design and testing. To address the execution time needed for multiple high-fidelity simulations, Evo-ROS provides an interface to parallelize evolutionary runs across multiple physical and virtual machines. Evo-ROS runs are executed on 448-core compute cluster. The fitness of individual configurations is based on the time and space efficiency, while avoiding obstacles, in following waypoints in multiple environments.

Our initial work on Evo-ROS (version 1.0) combined the Ardupilot control software along with the MAVLink protocol for communication with ground control, ROS, and Gazebo. Our initial experiments focused on the GRover platform described above. (We also extended Evo-ROS to support the ErleCopter aerial platform.) Specifically, we conducted a study with the rover that applies evolutionary search to determine the optimal number, placement and configuration of sonar sensors, given the possibility that one or more of the sensors might fail during operation. The primary purpose of that study was to demonstrate the operation of Evo-ROS, but the experiments also revealed interesting characteristics in how evolution realized sensor redundancy in the presence of failures.

A full paper on the Evo-ROS platform and our experiments with sensor failures and placement was published and presented as follows: Evo-ROS: Integrating Evolution and the Robot Operating System, Glen A. Simon, Anthony J. Clark, Jared M. Moore, and Philip K. McKinley, in Proceedings of the Genetic and Evolutionary Computation Conference Companion (Workshop on Evolutionary Computation Software Systems), Kyoto, Japan, pages 1386-1393, July 2018.

In the second year of the project, we conducted a major refactoring of Evo-ROS. A primary difference in the newer version, Evo-ROS 2.0, is to remove the dependence on Ardupilot in the simulation software stack. While Ardupilot is needed for simulating certain platforms, such as the Erle-Rover, it also limits the number of platforms to which Evo-ROS can be applied. In addition, Ardupilot adds an unnecessary level of complexity to the software and contains hard-coded control loops that prevent faster-than-real-time simulation. Finally, Ardupilot introduces significant overhead in the time required to prepare a simulation instance. In contrast, Evo-ROS 2.0 focuses on serving purely ROS-based systems, with the inclusion of Ardupilot as an option. This design allows the software management code of Evo-ROS 2.0 to be more efficient and user-friendly. In addition, the reduction in complexity of the simulation software stack and the removal of the hard-coded control loops has produced close to an order of magnitude performance improvement in evaluation time of intermediate solutions. Lastly, Evo-ROS 2.0 provides a cleaner interface for the front-end evolution algorithm (EA), **enabling users to easily configure Evo-ROS with different EA front-ends.**

4.3.3. Integration between Enki and Evo-ROS. This last feature listed above enables us to use Evo-ROS with either a traditional evolutionary algorithm or with a novelty search algorithm to enhance system robustness and resiliency, while identifying corner cases. Specifically, alternating between these two search types provides a means to discover a solution that performs well and is hardened against uncertainty. For example, if the front-end is a traditional genetic algorithm and individuals in the population represent the target platform or one of its subsystems, then Evo-ROS will optimize the target according to a fitness function, such as performance on a task or set of tasks. On the other hand, if the front-end is a novelty search algorithm such as Enki and individuals represent combinations of environmental conditions and internal states, then Evo-ROS will generate a set of scenarios that produce diverse, and possibly erroneous, behaviors in the target system. These two Evo-ROS configurations operate synergistically: Enki challenges the system by finding “difficult” scenarios, while the GA works to optimize the performance of the target system against those scenarios.

We implemented and applied this approach to improve the tracking ability of the throttle controller for MSU AutoRally. First, we demonstrated that Evo-ROS is able to evolve PID controller settings that yield a better performing controller than the default PID settings. Then we used Enki to generate environmental conditions that reveal the most diverse controller behavior, ranging from near failure to unexpected behavior. Based on this information, we were able to use the Enki-discovered environmental conditions to evolve better overall resilient behavior. Specifically, results demonstrated that the re-evolved controller outperformed the original PID controller in terms of tracking a reference signal for desired speed of the vehicle.

This work was published and presented as: "Applying evolution and novelty search to enhance the resilience of autonomous systems," Michael Austin Langford, Glen A. Simon, Philip K. McKinley, Betty H. C. Cheng, Proceedings of the 11th International Symposium on software Engineering for Adaptive and Self-managing Systems (SEAMS), pp. 63-69, 2019.

4.3.4. Evolving for Robust Localization. Modern localization techniques allow ground vehicle robots to determine their position with centimeter-level uncertainty under nominal conditions. This capability enables the robots to utilize fixed maps to navigate their environments, reducing overall system and software complexity. However, these techniques typically rely on measurements such as those from Global Navigation Satellite System (GNSS) that may be unavailable under certain conditions. While research and development on localization estimation seeks to reduce the severity of these outages, the question of what actions a robot should take under high localization uncertainty is still unresolved.

In this work we explored the possible role of evolutionary search to identify potential actuation adaptations when a robot detects its localization uncertainty is too high. We modeled the uncertainty as a Gaussian covariance matrix that correlates the robot's measured Cartesian position and Euler angle rotation, and model potential adaptation as weighted factors related to individual components of and/or norms of this covariance matrix. We then applied Evo-ROS to search for combinations of these weights that allow the vehicle to follow a path defined in the global reference frame while reducing the cross-track error. We applied these methods to the simulated MSU AutoRally. In simulation we mimicked both time-transient and location-transient localization, providing the evolution framework an environment by which to modify the weights and find potential adaptation factors. Evolutionary search produced a two-fold increase in fitness compared to a baseline path following framework without adaptation, with results validated on the physical robot platform.

A paper on this work is currently under review for publication: "Evolving Localization Uncertainty Adaptation for Globally Defined Path Following in Ground Vehicle Robots," (Daniel Kent, Philip K. McKinley, and Hayder Radha, submitted for publication, 2019.

5 References

The results of this work have been published and presented at a number of venues over the course of the project. In addition, due to the timely nature of the research and its relevance to a number of critical challenges faced by the computing and cyber-physical systems community, the PIs were invited to give a number of invited presentations and international conference keynotes relating to this work. In addition to disseminating results and providing visibility to the TRSYS program, the project team benefited from the feedback from academia and industry regarding this work.

5.1. Peer-reviewed Publications. This section contains a chronologically ordered list of peer-reviewed papers that describe the project results.

- Anthony J. Clark, Byron DeVries, Jared M. Moore, Betty H.C. Cheng, and Philip K. McKinley, "An Evolutionary Approach to Discovering Execution Mode Boundaries for Adaptive Controllers,", in P*roceedings of the IEEE Symposium Series on Computational Intelligence* (SSCI), Athens, Greece, pages 1-8, December 2016.

- B. DeVries and B.H.C. Cheng: Using Models at Run Time to Detect Incomplete and Inconsistent Requirements (full paper and presentation), Proc. of *The 12th International Workshop on* Models@run.time, September 2017, Austin, Texas.

- J. M. Moore, A. J. Clark, G. A. Simon and P. K. McKinley, Evo-ROS: Integrating Evolutionary Robotic and ROS (poster summary), *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vancouver, BC, Canada, September 2017.

- J. M. Moore, A. J. Clark and P. K. McKinley, Effect of Animat Complexity on the Evolution of Hierarchical Control, *Proceedings of the 2017 Genetic and Evolutionary Computation Conference (GECCO)*, Berlin, Germany, July 2017.

- B. DeVries and B.H.C. Cheng, "Automatic Detection of Incomplete Requirements using Symbolic Analysis and Evolutionary Computation (full paper and presentation)," *Proceedings of International Research Symposium on Search-based Software Engineering (SSBSE'17*), Paderborn, Germany, September 2017*, pp. 49-64.*

- Chung-Ling Len, Wuwei Shen, Steven Drager, and Betty H. C. Cheng, ``Measure Confidence of Assurance Cases in Safety-Critical Domains,*'' Proceedings at IEEE International Conference on Software Engineering (ICSE): in New Ideas and Emerging Results (NIER) Track*, May 2018, Gothenburg, Sweden.

- Byron DeVries, Betty H. C. Cheng: ``Run-time monitoring of self-adaptive systems to detect N-way feature interactions and their causes'' in the *Proc. 13th IEEE International Symposium on Software Engineering and Self-Managing Systems (SEAMS)*, May 2018, Gothenburg, Sweden, pp. 94-100.

- Glen A. Simon, Anthony J. Clark, Jared M. Moore, and Philip K. McKinley, "Evo-ROS: Integrating Evolution and the Robot Operating System,", in *Proceedings of the Genetic and Evolutionary Computation Conference Companion (Workshop on Evolutionary Computation Software Systems)*, Kyoto, Japan, pages 1386--1393, July 2018.

- Byron DeVries, Betty H. C. Cheng: "Automatic Detection of Feature Interactions Using Symbolic Analysis and Evolutionary Computation" at Proceedings of *the 18th IEEE International Conference on Software Quality, Reliability, and Security (QRS)*, July 2018, Lisbon, Portugal, pp. 257-268.

- Kate M. Bowers, Erik M. Fredericks, Betty H. C. Cheng: "Automated Optimization of Weighted Non-functional Objectives in Self-adaptive Systems**."** **–** *Proceedings of 10th International Symposium on Search-Based Software Engineering*, (SSBSE 2018), pp. 182-197.

- Byron DeVries, Betty H. C. Cheng: "Towards the detection of partial feature interactions." *Proc. 14th IEEE International Symposium on Software Engineering and Self-Managing Systems* SEAMS@ICSE 2019: 146-152

- Michael Austin Langford, Glen A. Simon, Philip K. McKinley, Betty H. C. Cheng: Applying evolution and novelty search to enhance the resilience of autonomous systems. *Proc. 13th IEEE International Symposium on Software Engineering and Self-Managing Systems,* SEAMS@ICSE 2019: 63-69

- Sharmin Jahan, Matthew Pasco, Rose F. Gamble, Philip K. McKinley, Betty H. C. Cheng: "MAPE-SAC: A Framework to Dynamically Manage Security Assurance Cases." *Proc. IEEE 4th International Workshops on Foundations and Applications of Self* Systems FAS*W@SASO/ICAC 2019*: 146-151.

- Michael Austin Langford, Betty H. C. Cheng: "Enhancing Learning-"Enabled Software Systems to Address Environmental Uncertainty.", at the IEEE International Conference on Autonomic Computing (ICAC2019), in Umea, Sweden, June 2019, pp. 115-124

- Byron DeVries and Betty H.C. Cheng, "Goal-Based Modeling and Analysis of Non-Functional Requirements" in the *Proc. of* IEEE/ACM *International Conference Model Driven Engineering Languages and Systems (MODELS)*, held in Munich, Germany, September 2019, .

- Daniel Kent, Philip K. McKinley and Hayder Radha, "Evo-LUCK: Evolutionary-algorithm based Localization UnCertainty Kernel for Ground Vehicle Robots", submitted for publication, 2019.

5.2. Invited Keynotes and Presentations (not including conference presentations). This section provides a list of invited presentations given by the PIs describing various aspects of the project to a broad range of audiences from academia and industry.

- B. H.C. Cheng, "Addressing Assurance for Self-Adaptive Systems in the Context of Uncertainty," Computer Science and Engineering Seminar, Oakland University, March 24, 2017.

- P. McKinley and Glen Simon, Evo-ROS: Combining Evolutionary Robotics and the Robot Operating System presented at the BEACON Congress, East Lansing, MI, August 2017.

- B. H.C. Cheng, "Requirements Engineering for High Assurance Autonomous Systems in the Face of Uncertainty: A Multidisciplinary Perspective," **Opening Keynote Speech** for *IEEE 25th International Requirements Engineering Conference (RE)*, held in Lisbon, Portugal, September 2017.

- B. H.C. Cheng, "Goal-Driven Approach to MDE-Based Research: Sharing some highlights and lessons," Keynote for Doctoral Symposium at *IEEE Int. Model-Driven Engineering Languages and Systems (MODELS) Conference*, held in Austin, TX, October 2017.

- B. H.C. Cheng, "MSU helping to create safer self-driving cars," Betty H.C. Cheng, Interview with Jorma Duran, WLNS, November 3, 2017, (URL: http://wlns.com/2017/11/03/msu-helping-to-create- safer-self-driving-cars/).

- B. H.C. Cheng, "Dealing with Uncertainty for High-Assurance Self-Adaptive Systems" Colloquium as part of "Feed your Brain Seminar Series", Medtronic, March 2018.

- B. H.C. Cheng, "Dealing with Uncertainty for High-Assurance Self-Adaptive Systems" Interdisciplinary Centre for Security, Reliability and Trust, Universit du Luxembourg, August 2018.

- B. H.C. Cheng, "Addressing Uncertainty for High-Assurance Self-Adaptive Systems," University of Toronto, Department of Computer Science Research Seminar, November 2018.
- B. H.C. Cheng, "A Multidisciplinary Approach to Requirements-based Adaptive Testing of Autonomous Systems" **Invited Keynote** for *6th International Workshop on Requirements Engineering and Testing, co-located with 41st IEEE International Conference on Software Engineering (ICSE),* May 25 - 31, 2019. Montreal, QC, Canada.

- B. H.C. Cheng, "A Multidisciplinary Approach to Developing Community-Based Research Infrastructure" **Invited Keynote** *for Second International Workshop on Establishing a Community-Wide Infrastructure for Architecture-Based Software Engineering (ECASE'19), co-located with The 41st IEEE International Conference on Software Engineering*, May 25 - 31, 2019. Montreal, QC, Canada.

- B. H.C. Cheng, "Goal-Driven Approach to High-Assurance Systems Research: Sharing some highlights and lessons," Faculty Research Seminar for MSU Summer Research Opportunities Program (SROP), June 2019.

- B. H.C. Cheng, "A Requirements-Driven and Context-Aware Approach to Assurance of Autonomous Systems" **Invited Keynote** for *Joint workshops for IEEE International Conference on Autonomic Computing (ICAC) and Self Adaptive Self-Organizing, Self-protecting Systems (SPS), 1st Workshop on Evaluations and Measurements in Self-*

Aware Computing Systems (EMSAC?19), 3rd Workshop on Self-Aware Computing (SeAC 2019), June 2019, Umea, Sweden.

- B. H.C. Cheng, "A Multi-Disciplinary Approach to Addressing Uncertainty for High-Assurance Self-Adaptive Systems," **Invited presentation** for *Assurance of Autonomy for Robotic Space Missions workshop at 7th International Conference on Space Missions Challenges for Information Technology (SMC-IT),* Pasadena, California, July 2019.

6 CONCLUSIONS

The X-PLORE project has developed a set of enabling technologies that can be applied during software development (specifically requirements engineering) to support the run-time adaptation of cyber-physical systems that are trusted and resilient to uncertainty. Different sources of uncertainty and their corresponding impact have been identified and used to enhance requirements in terms of system execution modes. In order to manage the complexity of these systems, a model-based approach has been used throughout, where automated formal (e.g., symbolic analysis) and lightweight (e.g., utility functions) techniques were used to capture and reason about the discrete and continuous data of cyber-physical systems. The scale and increasing complexity of CPSs make it necessary to go beyond the boundaries of traditional software engineering techniques in order to deal effectively with the numerous sources of uncertainty. Given the versatility and robustness of biological organisms to handle uncertainty, this project has demonstrated how evolutionary search can be harnessed and integrated with traditional software development techniques to support the proposed project objectives. The open-ended nature of evolutionary algorithms enables detection of unlikely-but-possible scenarios, unconstrained by human preconception. The proposed project will be guided and validated by two complementary CPSs: in-house autonomous robots (enabling analysis of both simulated and physical systems) and onboard software for intelligent vehicle systems, obtained from industrial collaborators in the automotive domain.

7 List of Acronyms

ACC	Adaptive Cruise Control
Ares-EC	Evolutionary Computation and Symbolic Analysis to Assess Requirements Completeness tool
CIFAR-10	Machine Learning Deep Neural Network Training Set Data
CPS	Cyber-Physical System
DNN	Deep Neural Network
EA	Evolutionary Algorithm
EC	Evolutionary Computation
Enki	Configurable Platform for Exploring Behavioral Uncertainty due to Environment and System Conditions tool
ER	Evolutionary Robotics
Evo-ROS	Evolutionary Search Framework for the Robot Operating System
FI	Feature Interactions
GA	Genetic Algorithms
GNSS	Global Navigation Satellite System
GORE	Goal-Oriented Requirements Engineering
GPS	Global Positioning System
IMU	Inertial Measurement Unit
Lidar	Light Detection And Ranging
LOKI	General Behavior Uncertainty Search tool
Lykus	Models at Run Time to Detect Requirements Incompleteness tool
MAPE	Monitor – Analyze – Plan – Execute Feedback Loop
MAPE-K	Monitor – Analyze – Plan – Execute Feedback Loop Plus Knowledge
MAPE-SAC	Monitor – Analyze – Plan – Execute Feedback Loop for Security Assurance Cases
ML	Machine Learning
NIST	National Institute of Standards and Technology
Phorcys	Symbolic Analysis to Detect n-way Feature Interactions tool
Phorcys-EC	Symbolic Analysis to Detect n-way Feature Interactions with Evolutionary Computation tool
PID	Proportional-Integrative-Derivative
PSAEC	Periodic Symbolic Analysis with Evolutionary Computing to find requirements completeness counterexamples tool
ROS	Robot Operating System
SA	Symbolic Analysis
SA+EC	Symbolic Analysis and Evolutionary Computation combined
SAC	Security Assurance Case
SAEC	Symbolic Analysis and Evolutionary Computing to find requirements completeness counterexamples tool
Soter	Aspect-oriented modeling of non-functional requirements and properties tool
Thoosa	Run-time detection of n-way feature interactions tool
TRSYS	AFRL Trusted and Resilient Systems Program
WiFi	Wireless Network