

Program Specialization for Wireless Sensor Networks

Background:

Wireless sensors networks collect information from remote locations in a cost effective and reliable manner. Examples include Harvard's volcano eruption monitoring project [7] and the SWE project at the University of Vermont to measure the snow-water equivalent in a snow pack [6]. These autonomous sensors are extremely efficient because memory, speed, and power are limited. The devices consume as little energy as possible in order to run off a small battery for an extended period of time. The goal of wireless sensor software is to run quickly and utilize the smallest amount of resources and energy [2].

TinyOS is an event-driven operating system specialized for memory constrained embedded systems, specifically wireless sensors. The language that is most commonly used to develop wireless sensor network applications on the TinyOS platform is nesC, which has the low-level control benefits of C, while optimizing the resource limitations of a wireless device. The flexible, component-based design of nesC uses tasks and events to support the particular requirements of this domain. NesC allows several software components to be statically wired together to form node-level applications [3].

The goal of this research project is to improve efficiency of a wireless sensor network in three different ways: first to increase the ease of programming with an improved programming abstraction, second to reduce the overall code footprint, and third to allow for better memory utilization. Benchmark testing will finally be performed and results will be processed and published with the assistance of Dr. Christian Skalka (Associate Professor, Computer Science) as part of his grant for Programming Language-Based Access Control for Wireless Sensor Networks [5].

The first phase of the implementation will be accomplished using staged programming. Multi-stage programming provides, in addition to the usual constructs of a general-purpose language, both partial evaluation and program specialization techniques. This causes a significant increase in efficiency, because code generation, reuse, and execution are supported [4].

Partial evaluation is a technique used to compact code for optimization by reducing computational effort for situations that have an unchanging temperament. The compiler performs pre-computations of these common situations in order to save time during execution. An example would be calculating a routing table in the first stage of computation for the wireless sensor network and using this as a constant value in a separate section of the application. In addition, constant values may then be stored in ROM, freeing up the limited availability of RAM (2kb on our devices). Releasing RAM for individual wireless devices will increase the efficiency of the program significantly allowing wireless sensors networks to not only run longer and more effectively, but also increase the sophistication of the programs they are running [1]. Giving otherwise impossible applications, such as crypto protocols using multiple keys, the ability play a part in the security of wireless devices.

Hypothesis:

As noted above, we hypothesize that the advantages of staged programming include ease of programming, code reduction, and greater memory utilization. Given that efficiency with wireless sensors is extremely important, this cannot be a purely theoretical issue. Implementation is essential. The language extension to nesC (named nesT) will consist of a parser, compiler and run-time support, including static analysis to ensure any generated program is syntactically correct and well typed. The multi-staged program built with the extension will allow for program optimization by runtime algorithm specialization. We also hypothesize that quantitative efficiency gains of the system will be revealed by benchmarking and empirical testing the application.

Specific aims:

Previous work with partial evaluation has been shown to benefit embedded systems as demonstrated [1], however very little has been done in the area of wireless sensor program specialization. Our objective is to realize this theory in practice as an extension of nesC. For this to happen we must implement the language so that compilation and over-the-air programming of communicating devices is supported at runtime. Our goal is to make this process part of nesT itself, rather than an ad-hoc process. Instead of compiling a file, saving it to a text file, then opening it with a shell command, we envision an

entirely new communication scheme. The new extension will combine all these steps into a single application that runs on a *hub* (a more powerful “master” processor such as a laptop) as the initial stage of the program. The second stage occurs on the wireless sensors after the code has been sent over the network and loaded with a bootloader on the low-power network *nodes*.

Herein lies the most interesting and novel part of this project, which is how we combine these actions into a single idiom. The basic syntax running on the hub is *run <p> at m* where *p* is a program run on a specific wireless sensor *m*. The hub will run its program normally until it comes to *<p>*. Only when the keyword *run* is used before *<p>*, will *p* be compiled, and sent over the network to the sensor. Furthermore, *p* may be dynamically constructed as part of the hub program computation- so there are *two-stages* of computation- one on the hub and one on the node(s). After the hub ships the code to the wireless sensor node, the hub will continue on where it left off in the program.

Multi-staging creates these generic programs, in which *<p>* can have any value, be called multiple times in the program and be sent to several sensors at the same time. With this programming technique, the more powerful processor can do all the memory tasking calculations and lift these calculations as a constant value into the program *p*. With this syntax we can calculate certain values only when we need and these pre-computed values can be brought into expressions with ease.

Following the implementation of this staging procedure, the components of nesT will undergo controlled tests for relative performance. Conservation of memory and speed will be analyzed, particularly for efficiency in the new extension and compiler versus systems that do not use multi-staging. Benchmark testing against systems that do not use staging will be created for the application and hub to node communication to determine if any additional performance gains are obtained via our methods. Using TinyDB [9] we can gather information about a network of sensors to benchmark the runtime efficiency of the program, memory usage and the code size. Since TinyDB is a common application and many benchmarking tests have been recorded already in wireless sensor network platforms, it will provide a good point of comparison.. Tests will also be created to stress the system for efficiency in other representative scenarios and any bugs will be ironed out. The data will then be empirically analyzed and assembled into at least one article, submitted for publication at a prestigious international conference such as the International Conference on Embedded Software (EMSOFT).

Methods:

In addition to the direct mentorship of Dr. Skalka, this project will be carried out under the student leadership of graduate students under the direction of Professor and Chair Dr. Scott Smith at John Hopkins University Department of Computer Science. We will use standard nesC and TinyOS development implementation techniques as specified in the documentation [3]. Any new syntax created within the extension nesT will be handled within our new compiler.. The staging deployment in our application relies on wireless communication from the hub to the sensors. The leveraging for this “over air programming” will come from Deluge [8], which is the current prevailing method to disseminate code to wireless sensors. Implementation of the application will be tested on the wireless sensor network within the University of Vermont on the third floor of the Votey building.

Interpretation of results:

The results will be analyzed by teammates at University of Vermont and John Hopkins University to test the results of the work. The efficiency of the new extension will be observed by the empirical data obtained through various benchmarks. We will observe the reduction in the code footprint as we combine compilation, shipping the code over the network, and building with the boot loader process into one application. Memory consumption can be easily measured by comparing the values of usage before and after the implementation. Finally, the program abstraction will be compared with past programs to test the ease of the use. Researchers and developers from academia, industry, and government from within the embedded software development community will also be able to interpret the results reported in publications and technical reports.

References:

- [1] E. Chung, L. Benini, G. DeMicheli, G. Luculli, and M. Carilli, "Value-Sensitive Automatic Code Specialization for Embedded Software," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 21, No. 9, Sept. 2002.
- [2] H. Karl and A. Willig, *Protocols and Architectures for Wireless Sensor Networks*. West Sussex, England: Wiley 2005.
- [3] P. Levis, *TinyOS Programming*. <http://csl.stanford.edu/~pal/pubs/tinyos-programming.pdf>. June 28, 2006.
- [4] W. Taha and T. Sheard: *MetaML and Multi-stage Programming with Explicit Annotations*, *Theory of Computer Science*, 248(1-2): 211-242, 2000.
- [5] Skalka, C. (2008). *Programming Language-Based Access Control for Wireless Sensor Networks*. The Young Investigators Award (YIP), USAF, AFRL, Air Force Office of Scientific Research from 2008-2012. Available on line at http://uvm.edu/~cems/?Page=alumni/faculty/cskalka.php&SM=alumni/_alumnimenu.html.
- [6] *Snow Water Equivalent Monitoring with Wireless Sensor Networks*. Sensor Networks and Wireless Workgroup: <http://www.cems.uvm.edu/research/cems/snow/swe.php>.
- [7] *Volcano monitoring*. Harvard Sensor Networks Lab: <http://fiji.eecs.harvard.edu/>.
- [8] A. Hagedorn, D. Starobinski, A. Trachtenberg. "Rateless Deluge: Over-the-Air Programming of Wireless Sensor Networks Using Random Linear Codes," *Information Processing in Sensor Networks*, vol. 22-24, pg. 457-466, 2008.
- [9] S. Madden, M. Franklin, J. Hellerstein, W. Hong. *TinyDB: An Acquisitional Query Processing System for Sensor Networks*. *CM Transactions on Database Systems*: <http://telegraph.cs.berkeley.edu/tinydb/>.

Budget and Budget Justification

Funds are requested as follows:

2 visits to Johns Hopkins University: \$800 x 2
Domestic conference attendance: \$1200
Total funds requested: \$2800

All funds requested are to support travel of Simone Willet. Visits to Johns Hopkins University (JHU) will be necessary to allow collaboration between project teams at UVM and JHU; funds will cover transportation, food, and lodging. Funds are also requested for attendance at the conference at which we envision publishing the results of our work; funds will cover conference registration, transportation, food, and lodging.

Simone G. Willett

87 S. Willard St. (207) 5515578
Burlington, VT 0540 simone.willett@uvm

Objective

Research with a professor to lead towards my Masters degree thesis.

Graduation Date: 2009

University of Vermont in Burlington, Vermont
Double Major Computer Science and Mathematics
GPA: 3.3

Relevant Coursework

Wireless Sensor Network Apps.	Differential Equations	Discrete Math
Advanced Web Design	Physics	Mathematics of Biology
Computer Organization	Hands-On-Robotics	Expository Writing
Software Engineering	Bioinformatics	Abstract Algebra
Programming Languages	Data Structures	Network Security & Cryptography
Linear Algebra	Visual Basic Programming	

Programming Languages: Visual Basic, C++, Java, OCaml, nesC, Mips, Html, CSS

Teaching Assistantship: for Wireless Sensors and Java Programming

Work Experience

MicroStrain: Wireless Sensors Software Engineering in Williston, VT August 2008-present
Developing 3D visualizations demos using Eye-Sys. Using the Eye-Sys SDK integrated the inertial link sensors to retrieve live data. Researching the use of Kalman filters to combine inertial sensors with GPS.

Mathematics Tutor: Learn Without Limits Internet Startup in Potomac, MD May 2007-May 2008

Recruited by early-stage venture to help build the business infrastructure as the company has grown from 3 to 100's of tutors. I helped with decision-making, problem solving and technology upgrading (drawing board, tablets). With my major in math, I helped by tutoring high school and college math students over the Internet.

Assistant Property Manager with East Brown Cow in Portland, Maine May 2005–January 2006
At EBC, I reviewed property leases, organized records, prepared bank deposits, answered the phone, and performed other office tasks as required. I used my software background to create an organizational computer filing system for the company.

Christian Skalka., Ph.D. Associate Professor, University of Vermont (802) 656-1920

Dave L. Churchill, Ph.D. VP Engineering: Microstrain Inc., 459 Hurricane Lane, Williston, VT 05495

David A. Soley, Attorney-at-Law: Bernstein Shur: 100 Middle St, Portland, ME 04101 (207) 774-1200

Dr. Christian E. Skalka, PhD

Contact Information

Votey Hall 379
Department of Computer Science
University of Vermont
Burlington, VT 05405 USA

Voice: (802) 656 1920
Fax: (802) 656 0696
Email: skalka@cs.uvm.edu
WWW: <http://www.cs.uvm.edu/~skalka>

Current Position

Assistant Professor, Department of Computer Science, University of Vermont.

Research Interests

Programming languages, especially type theory and programming language-based security. Logic in computer science, especially logic programming and trust management logics. Wireless sensor networks, especially software systems, security architectures, and environmental monitoring applications.

Education

Johns Hopkins University, Baltimore, Maryland USA

- PhD, Computer Science, September 2002
- Advisor: Scott Smith, Dept. of Computer Science
- Thesis: *Types for Programming Language-Based Security*

Carnegie Mellon University, Pittsburgh, Pennsylvania USA

- MS, Logic and Computation, May 1997
- Advisor: Frank Pfenning, Dept. of Computer Science
- Thesis: *Some Decision Problems for ML Refinement Types*

St. John's College, Santa Fe, New Mexico USA

- BA, Philosophy and Mathematics, May 1991

Honors and Awards

Best Paper Award, Conference on European Theory and Practice of Software (ETAPS), 2001.
ARCS Foundation Fellowship, 2000-2001.

Research Funding History

Project: *Context Based Security in Programming Languages*

PI: Christian Skalka

Funding Agency: VT EPSCoR.

Award: \$18,750, 9/1/03-5/31/04

Status: Expired.

Project: *Context Based Security in Programming Languages*

PI: Christian Skalka

Funding Agency: VT EPSCoR.

Award: \$18,750, 9/1/04-5/31/05

Status: Expired.

Project: *Trace Effect Analysis for Software Security*

PI: Christian Skalka

Funding Agency: USAF, AFRL, Air Force Office of Scientific Research.

Award: \$358,185, 06/01/06-05/31/09

Status: Current.

Project: *An Exploratory Project to Develop an In Situ Water Equivalent Monitoring System with Improved*

Spatial Resolution

PIs: Christian Skalka, Jeff Frolik (UVM), Beverley Wemple (UVM), Tom Neumann (UVM)

Funding Agency: NASA DEPSCoR

Award: \$25,000, 07/01/08-6/30/09

Status: Current.

Project: *Programming Language-Based Access Control for Wireless Sensor Networks*

PI: Christian Skalka

Funding Agency: USAF, AFRL, Air Force Office of Scientific Research.

Requested Funds: \$599,885, 04/01/09-12/31/13

Status: Current.

Fully Refereed International Journal Publications

Peter Chapin, Christian Skalka, and X. Sean Wang. *Authorization in Trust Management: Features and Foundations.*

ACM Computing Computing Surveys 40(3): 1-48, 2008.

Christian Skalka. *Types and Trace Effects for Object Orientation.* Journal of Higher Order and Symbolic Computation 21(3): 239-282, 2008.

Christian Skalka, Scott Smith, and David Van Horn. *Types and Trace Effects of Higher-Order Programs.* Journal of

Functional Programming 18(2): 179-249, 2008.

Christian Skalka, X. Sean Wang and Peter Chapin. *Risk Management for Distributed Authorization.* Journal of

Computer Security 15(4): 447-489, 2007.

Christian Skalka, X. Sean Wang. *Trust but Verify: Authorization for Web Services.* Journal of Computer Systems Science and Engineering 21(5): 381-392, 2006.

Christian Skalka and Scott Smith and David Van Horn. *A Type and Effect System for Flexible Abstract Interpretation*

of Java. Electronic Notes in Theoretical Computer Science 131: 111-124, 2005.

François Pottier, Christian Skalka, and Scott Smith. *A Systematic Approach to Static Access Control.* ACM

Transactions on Programming Languages and Systems, 27(2): 344-382, 2005.

Christian Skalka and Scott Smith. *Static Use-Based Object Confinement.* Springer International Journal of

Information Security, 4(1-2): 87-104, 2005.

Christian Skalka and Scott Smith. *Set Types and Applications.* Electronic Notes in Theoretical Computer Science 75:75-94, 2003.

Christian Skalka and François Pottier. *Syntactic Type Soundness for $HM(X)$.* Electronic Notes in Theoretical

Computer Science 75: 61-74, 2003.

Fully Refereed International Conference and Workshop Publications

Paritosh Shroff, Scott Smith and Christian Skalka. *The Nuggetizer: Abstracting Away Higher Orderness for*

Program Verification. Asian Symposium on Programming Languages and Systems, APLAS 2007.

Christian Skalka. *Type Safe Dynamic Linking for JVM Access Control*. ACM Conference on Principles and Practice of Declarative Programming, PPDP 2007: 51-62.

Christian Skalka and Jeff Polakow. *A LolliMon Specification of RT*. ACM Workshop on Programming Languages and Analysis for Security, PLAS 2006: 37-46.

Christian Skalka, X. Sean Wang and Peter Chapin. *Risk Assessment in Distributed Authorization*. ACM Workshop on Formal Methods in Security Engineering, FMSE 2005: 33-42.

Christian Skalka. *Trace Effects and Object Orientation*. ACM Conference on Principles and Practice of Declarative Programming, PPDP 2005: 139-150.

Christian Skalka and X. Sean Wang. *Trust but Verify: Authorization for Web Services*. ACM Workshop on Secure Web Services, SWS 2004: 47-55.

Christian Skalka and Scott Smith. *History Effects and Verification*. Asian Programming Languages Symposium, APLAS 2004: 107-128.

Christian Skalka and Scott Smith. *Static Use-Based Object Confinement*. Workshop on Foundations of Computer Security, FCS 2002: 117-126.

François Pottier, Christian Skalka, and Scott Smith. *A Systematic Approach to Static Access Control*, European Symposium on Programming, ESOP 2001: 30-45.

Christian Skalka and Scott Smith. *Static Enforcement of Security with Types*. ACM International Conference on Functional Programming, ICFP 2000: 35-45.

Other Publications

Christian Skalka. *Programming Languages and Systems Security*. IEEE Security and Privacy Magazine, May 2005.

Invited Presentations

- *Type Safe Dynamic Linking for JVM Access Control*. McGill University Computation and Logic Group Summer Seminar Series, 2007
- *Type Safe Dynamic Linking for JVM Access Control*. Harvard University Computer Science Seminar Series, 2007
- *Logic and Practice of Trust Management*. Dartmouth University Computer Science Colloquium, 2006
- *Trace Effects and Object Orientation*. Tufts University Computer Science Seminar Series, 2005
- *Types for Access Control: Foundations and Methodology*. Church Project Seminar, Boston University, 2003

- *A Systematic Approach to Static Access Control*. Carnegie Mellon University Principles of Programming Seminar Series, 2001

Current and Former Graduate Advisees

David Van Horn (MS), graduated 2005
MS Thesis: Algorithmic Trace Effect Analysis

Bridget Zurn (MS), graduated 2006
MS Project: A New Implementation of the MedState Language

Peter Chapin (PhD), current, matriculated Fall 2004
Doctoral Thesis topic: Trust management logics.

Charley Robinson (MS), curren, matriculated Fall 2008.
MS Thesis topic: data reliability in wireless sensor networks.

Sarah Greenberg (MS), current, matriculated Winter 2009.
MS Thesis topic: remote control and data retrival in wireless sensor networks.

Teaching Experience

University of Vermont, Burlington, Vermont USA

Instructor, Department of Computer Science **September 2002-Present**

Conceived and implemented new course designs (CS095, CS103, CS303, CS294, CS361). Responsible for all course

instruction and administration, including lectures, homework, grading, and external help sessions.

- MATH054 Discrete Mathematics
- CS095 Introduction to Computer Science with Wireless Devices
- CS100 Object Oriented Languages
- CS103 Programing Languages
- CS202 Compiler Construction
- CS294 Independent Study in Wireless Network Security
- CS294 Independent Study in iPhone Programming.
- CS294 Independent Study in Natural Language Processing
- CS294 Independent Study in Website Design and Information Retrieval
- CS294 Independent Study in Website Design and Quality Assurance
- CS303 Types in Programming Languages
- CS361 Wireless Sensor Network Applications.
- CS381 Graduate Seminar: Topics in Computer Security

Johns Hopkins University, Baltimore, Maryland USA

Instructor, Department of Computer Science **Fall 2001**

Conceived and implemented new short (12 week) course. Responsible for all course instruction and administration,

including lectures, homework, grading, and external help sessions.

- 600.307 Logic in Computer Science

Computer Science Community Service

National Science Foundation Panelist, 2007.

IFIP TC-11 WG 11.1 & WG 11.5 Joint Working Conference on Security Management, Integrity, and Internal

Control in Information Systems, December 2005, Program committee.

New England Programming Languages Seminar (NEPLS):

- Program committee, February 2004
- Local arrangements, organization, and program committee, June 2004
- Program committee, October 2004

Peer reviewer for the following conferences and journals:

- Journal of Functional Programming
- ACM Transactions on Programming Languages and Systems
- ACM Computing Surveys
- ACM Principles of Programming Languages
- ACM International Conference on Functional Programming
- ACM Workshop on Types in Language Design and Implementation
- IEEE Symposium on Security and Privacy
- IEEE Symposium on Logic in Computer Science
- European Symposium on Programming
- Foundations of Software Science and Computation Structures.
- IFIP TC-11 WG 11.1 & WG 11.5

UVM Service

UVM service positions and committee memberships:

- Computer Science Chair Review Committee for CEMS (2009)
- Computer Science Department Curriculum Committee (2002-Present)
- Computer Science Department Outreach Coordinator (2002-2005)
- Computer Science Department Graduate Committee (2005-Present)
- Computer Science Student Association Faculty Advisor (began Fall 2007)
- College of Engineering and Mathematical Sciences Faculty Advisory Council (began Fall 2007)
- VT DEPSCoR panelist (date withheld)

MS and PhD Thesis Committee memberships:

- David Van Horn
- Bridget Zurn
- Jothi Kuppasamy
- Jason Hill
- Jeremy Gustie
- Peter Chapin
- Diana Tatar