

**James F. Bowring**  
Research Statement - 2006

**Introduction**

The theme of my research is the scientific exploration of software development processes with the twin goals of improving how we model problems and of improving how we implement computer-based solutions. I am using an inter-disciplinary approach in my research by combining conventional software engineering program analysis with statistical machine-learning techniques. To date, my research has focused on modeling and classifying the aggregate behaviors of executing programs. Funding for my research comes in part from a National Defense Science and Engineering Graduate Fellowship and from an NSF grant for Highly Dependable Computing and Communication Systems Research. I am also supervising two undergraduate research assistants.

**Motivation**

Software pervades our environment. The growth in the number of active CPUs and executing programs has outpaced the ability of humans to maintain and manage digital systems without automated assistance. Ideally, digital systems will eventually exhibit some degree of self-awareness and contribute to their own maintenance. Understanding how to specify, model, and implement software with a sense of self is a daunting problem. One promising approach to this problem is to develop meta-strategies for self-awareness by software systems that enable self-diagnosis and self-repair [2]. My own inspiration is the automatic function of a *gimbal*—a mechanical device that supports an object and maintains the orientation of said object with respect to gravity independently of its immediate operating environment. The motivation for my research to date is to extend the foundations of program analysis to provide for software self-awareness for programs. In this context, a gimbal supports the accumulation of historical behavior norms and provides the procedures for both evaluating ongoing behaviors relative to these norms and for proposing corrective actions in the presence of untoward behaviors.

**Inter-Disciplinary Approach**

The implementation of a software gimbal requires the specification of program behaviors, methods for modeling and summarizing behaviors, and techniques for storing, processing, comparing, and evaluating behaviors. One of my goals is to develop effective techniques for automating these features of a software gimbal. I borrow from the recent success of automatic speech recognition systems, which can assess the likelihood of a potentially unbounded set of possible utterances and select the most likely candidate in real-time, given an underlying model of the conversational domain. I see an analogy between estimating an utterance from an acoustic waveform and estimating the behavior of a program from dynamic-analysis data. In both cases, there is an inherently stochastic causal relationship between the quantity of interest and the measurement data. My strategy is to leverage the successful tools and methods of speech recognition in the novel context of program-behavior analysis. For example, I use Markov models and statistical machine-learning techniques.

## **Stochastic Behavior Models and Automated Behavior Classifiers**

I have shown that stochastic models of individual control-flow features can be reliable predictors of behaviors [3]. I have also developed a novel predictive feature---databin-transition-profiles---that abstract data-flows in a program [4]. I have demonstrated the robustness of these stochastic models by developing a machine-learning framework to support the automatic construction of behavior classifiers from these models. Using this framework, I designed a semi-automated testing oracle by applying the machine-learning technique of active learning, where a classifier trains incrementally. I am also interested in how best to combine individual features to leverage the synergy of their discriminatory powers. To that end, I have created and evaluated ensemble classifiers composed of two different stochastic feature classifiers: a control-flow and a data-flow feature. I show empirically that these ensemble classifiers can outperform their component classifiers, a result consistent with results from machine learning. This finding suggests that control-flow and data-flow features of program executions may capture divergent statistical views of behaviors. I am also exploring new techniques that will recognize behaviors on-line by comparing them to learned historical models of behaviors.

### **Contributions and Future Work**

My doctoral research thesis is that statistical and probabilistic models of individual and joint characteristic features of program executions can describe program behaviors both usefully and efficiently, and that machine-learning techniques can train effective behavior classifiers using these models [1, 3, 2, 4]. My results represent a start toward the implementation of practical software self-awareness and toward the implementation of a software gimbal. A roadmap for this work includes further explorations of aggregate features of executions to extend the foundations of program analysis to include statistical models of behavior. As researchers identify additional features and the exact roles they play in characterizing specific behaviors, the work will move toward establishing the fundamentals of self-awareness.

My future research will concentrate on a more global aspect of the self-awareness problem: exploring the characteristics of software development that directly support program self-awareness. My motivation is that we may need entirely new software constructs to support effectively software self-awareness processes. I will begin by researching how to specify self-awareness using the paradigm of model-driven architecture (MDA), which is an industry initiative sponsored by the Object Management Group (OMG).

At Georgia Tech, I collaborate with two advisors: Professor Mary Jean Harrold in software engineering and Professor James Rehg in intelligent systems. I also supervise the work of two undergraduates as research assistants in my investigation of techniques to provide for real-time behavior detection. I am working with a Master of Science student to apply my techniques to the automated generation of test suites for legacy systems. I intend to continue this collaborative approach in my on-going research efforts. I want to inspire other researchers and students to explore software development in new ways.

## References

- [1] J. Bowring, A. Orso, and M. J. Harrold, "Monitoring deployed software using software tomography," presented at ACM SIGPLAN-SIGSOFT Workshop on Program Analysis for Software Tools and Engineering (PASTE'02), 2002.2-9.
- [2] J. F. Bowring, J. M. Rehg, and M. J. Harrold, "TRIPWIRE: Mediating Software Self-Awareness," presented at SE Workshop on Remote Analysis and Measurement of Software Systems (RAMSS '04), Edinburgh, Scotland, 2004.11-14.
- [3] J. F. Bowring, J. M. Rehg, and M. J. Harrold, "Active Learning for Automatic Classification of Software Behavior," presented at International Symposium on Software Testing and Analysis (ISSTA 2004), Boston, Mass., 2004.195-205.
- [4] J. F. Bowring, J. M. Rehg, and M. J. Harrold, "Improving the Classification of Software Behaviors using Ensembles of Control-Flow and Data-Flow Classifiers," College of Computing, Georgia Institute of Technology Technical Report GIT-CERCS-05-10, 2005.