Software and Systems

05 MAR 2012

Robert J. Bonneau, Ph.D.
Program Manager
AFOSR/RSL

Air Force Research Laboratory

Integrity ★ Service ★ Excellence
<table>
<thead>
<tr>
<th>1. REPORT DATE</th>
<th>2. REPORT TYPE</th>
<th>3. DATES COVERED</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>05 MAR 2012</strong></td>
<td></td>
<td><strong>00-00-2012 to 00-00-2012</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>4. TITLE AND SUBTITLE</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Software And Systems</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>5a. CONTRACT NUMBER</th>
<th>5b. GRANT NUMBER</th>
<th>5c. PROGRAM ELEMENT NUMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>6. AUTHOR(S)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>5d. PROJECT NUMBER</th>
<th>5e. TASK NUMBER</th>
<th>5f. WORK UNIT NUMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Air Force Research Laboratory, Wright Patterson AFB, OH, 45433</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>8. PERFORMING ORGANIZATION REPORT NUMBER</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>10. SPONSOR/MONITOR’S ACRONYM(S)</th>
<th>11. SPONSOR/MONITOR’S REPORT NUMBER(S)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>12. DISTRIBUTION/AVAILABILITY STATEMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Approved for public release; distribution unlimited</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>13. SUPPLEMENTARY NOTES</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Presented at the Air Force Office of Scientific Research (AFOSR) Spring Review Arlington, VA 5 through 9 March, 2012</strong></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>14. ABSTRACT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>15. SUBJECT TERMS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>16. SECURITY CLASSIFICATION OF:</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>a. REPORT</strong></td>
</tr>
<tr>
<td>unclassified</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>17. LIMITATION OF ABSTRACT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Same as Report (SAR)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>18. NUMBER OF PAGES</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>19a. NAME OF RESPONSIBLE PERSON</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
</tbody>
</table>
NAME: Software and Systems

BRIEF DESCRIPTION OF PORTFOLIO:

• Enable quantifiable performance evaluation of critical software systems
• Manage software environments in order to preserve vital mission functions
• Comprehensively understand distributed effects in large software infrastructures to predict global system failures

LIST SUB-AREAS IN PORTFOLIO:

• Models for Composeable Dynamic Software
• Dynamic Formal Analysis and Verification
• Online Assessment and Repair of Failure
Unified Approach to Software

- Many current problems in software can be addressed in a more rigorous unified way by casting the software problem as a dynamic processes that can be measured and online management of software into existing and future systems.
Current Program Scope

- Models for Composeable Dynamic Software
  - New programming languages or language constructs reduce errors at run-time
  - Domain-specific languages enhance capabilities for code generation

- Dynamic Formal Analysis and Verification
  - Verification of system properties based on formal specifications

- Online Assessment and Repair of Failure
  - Abstract models of systems and their interactions facilitate automated generation of code
Systems and Software
Agency Interaction

- OSTP/NITRD Coordinating Group
  - High Confidence Systems and Software (HCSS) Member
- ASDR&E
  - Software Producibility Initiative
- Secretary of the Air Force
  - Air Force Software and systems Overview Study
- NSF
  - Cyber Physical Systems
    - Panelist and guest speaker at 2011 meeting
- NASA
  - V&V of Flight Critical Systems
  - Ames Research Laboratory
    - Human Systems Integration Division
    - Intelligent Systems Division
Systems and Software

Other funding agencies

- Army Research Office
  - Software investment mostly directed toward information assurance
- ONR
  - Software and Computing Systems
    - Principles for Correctness and Security Properties
    - Human Robot Interaction
    - Perception and Cognitive Control
- NSF
  - Cyber Physical Systems – focused on interaction with physical environment and sensing systems
- DARPA: Software Producibility
• Software Models Using Adaptive Feedback and Complexity Reduction
• Feedback in Formal Analysis and Verification
• Adaptive repair and assessment of distributed software infrastructures
• Language-based approaches
• Modeling Human-Machine Interaction
• Agent-based approaches
Approach: Software contracts incorporate feedback into models of online software assessment and require analysis of data type representation and meaning of data types to software performance.

Payoff: Real time assessment of registers and data types in hardware software infrastructures can be performed.

Contracts Create Ability To Trace Logical Errors

Trace of Logical Outcome of Mathematical Computation

Contract With Feedback

Contract + Feedback Allows Identification Of Logical Process Failure in Real Time
Scalable Model Checking
C. Tinelli U Iowa, C. Barret, NYU

Approach: Formal verification suffers from state space explosion. Compactly represent logical symbols in scalable nested satisfiability modulo theory (SMT)

Payoff: More automated more scalable verification to handle large heterogeneous systems

Compact SMT Language

- **Valid:**
  - satisfied by all states in Q
- **Inductive:**
  - \( I(s_0) \models P(s_0) \),
  - \( P(s_n), T(s_n, s_{n+1}) \models P(s_{n+1}) \)
- **k-inductive:**
  - \( I(s_0), T(s_0, s_1), ..., T(s_{k-1}, s_k) \models P(s_0), ..., P(s_k) \),
  - \( T(s_n, s_{n+1}), ..., T(s_{n+k}, s_{n+k+1}), P(s_n), ..., P(s_{n+k}) \models P(s_{n+k+1}) \)
- **Invariant:**
  - satisfied by all reachable states of S

Improved Lower Dimensional Model

DISTRIBUTION A: Approved for public release; distribution is unlimited.
Approach: Understanding how to statistically represent a software model for software testing requires accurate models of mapping what to measure to performance.

Payoff: Using a principled approach that captures the right level of software and abstraction statistically enables accurate statistical representation of failure modes.
Mission Verification

**Approach:** Develop a language to represent mission scenarios tied to integrated distributed software architecture.

**Payoff:** Verify global mission properties as function of lower level software constructs for quantifiable fault tolerance in achieving mission objectives.

**Mission Analysis**
Language Architecture

**Fault Tolerant Mission Design**

Program Analysis

DISTRIBUTION A: Approved for public release; distribution is unlimited.
Approach: Many software systems are introduced into environments that have uncertain conditions that result in unforeseen failures. Feedback failure correction mechanisms can augment software to adapt to failures.

Payoff: Systems such as those on networks or those subject to uncertain physical environments can adapt to conditions using binary runtime repair of errors or faults based on automata theory and algebraic proofs of correctness.

Robust Architecture with Feedback

Mathematical Formalism

Automata Description (feedback)

\[ r_t(t) = \sum_{e \in O \cup \{\tau\}} \sum_{t'} \Delta_e^O(t, t'). \]

Formal Logic (constraints)

\[ P ::= X \mid \text{nil} \mid a_w?t \mid b_r!t \mid \tau_r \cdot t \mid t_1 + t_2 \mid t_1 \circ_2 t_2 \mid t\{0\} \mid t\{a \leftarrow a'\} \mid \mu X.t \]

NASA Slated to Use Technology in Next Generation Mars Rover
**Runtime Repair**

**S. Khurshid, UT Austin**

**Approach:** A functional approach can be developed for real time software runtime repair using new paradigms for online verification

**Payoff:** Faults in software can be corrected in real time and tracked rigorously

---

**Real Time Runtime Software Repair Architecture**

**Results in Corrections of Multiple Faults**

<table>
<thead>
<tr>
<th>Measure</th>
<th>Alg.</th>
<th>List with 10 nodes</th>
<th>List with 20 nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Fail in fix the size</td>
<td>Delete one element too</td>
</tr>
<tr>
<td>Time (ms)</td>
<td></td>
<td>7557</td>
<td>7314</td>
</tr>
<tr>
<td>IR</td>
<td></td>
<td>726</td>
<td>851</td>
</tr>
<tr>
<td>EL</td>
<td></td>
<td>614</td>
<td>13113</td>
</tr>
<tr>
<td>GL</td>
<td></td>
<td>598</td>
<td>1060</td>
</tr>
<tr>
<td>Edit distance</td>
<td></td>
<td>38</td>
<td>33</td>
</tr>
<tr>
<td>BM</td>
<td></td>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>IR</td>
<td></td>
<td>2</td>
<td>12</td>
</tr>
<tr>
<td>EL</td>
<td></td>
<td>2</td>
<td>13</td>
</tr>
<tr>
<td>GL</td>
<td></td>
<td>2</td>
<td>15</td>
</tr>
</tbody>
</table>

DISTRIBUTION A: Approved for public release; distribution is unlimited.
Automated Model Revision
Kulkarni, Mich State

Approach: Verification tends to use approaches that are fixed based on the notion of pre-existing code and logical structures. In order to adapt to unanticipated conditions it is necessary to be able to revise models if conditions change.

Payoff: In dynamic heterogeneous systems, it is necessary to update the verification of the system as it evolves

Adaptive Verification

**Question**: Is it possible to revise the model automatically such that it satisfies the failed property while preserving the other properties?
Systems and Software
AFRL Tech Directorate Interest/Coordination

• Information Directorate
  – Systems and Software Producibility
  – Multi-core Computing

• Air Vehicles
  – Flight-critical systems and software
  – Mixed-criticality architectures

• Human Effectiveness
  – Modeling of human-machine systems
  – Meta-information portrayal STTR

• Robust Decision Making STT
  – Large Scale Cognitive Modeling/C2WT
Increased Scale/Integration via DSMLs Anchored in DEVS (Douglass, 711th HPW/RH)

DEVS (discrete event system specification)
- Formal rigor
- Model reusability
- Interoperability

A discrete event system specification (DEVS) is a mathematical structure (7-tuple)

$$M = <X, S, Y, \delta_{int}, \delta_{ext}, \lambda, ta>$$

where
- $X$ is the set of input values
- $S$ is a set of states
- $Y$ is the set of output values
- $\delta_{int}: S \rightarrow S$ is the internal transition function
- $\delta_{ext}: Q \times X \rightarrow S$ is the external transition function
- $\lambda: S \rightarrow Y$ is the output function
- $ta: S \rightarrow R_{0,\infty}$ is the time advance function

Navigator
Plans routes from targets to targets under constraints

Domain-Specific Languages
- Tailored for cognitive modeling
- Semantically anchored in DEVS

High-Performance Computing
- Scalable simulation infrastructure
- Exploiting 25 years of DEVS
Approach: Use parallel processing resources and network infrastructure as means of emulating and detecting system faults in new software deployment.

Payoff: Deployment of new software tools has far fewer defects and more detailed assessment of integrated system performance.
Software Collaborations at AFOSR

- **Information Operations and Security**
  - Fundamental software constructs for software and system security

- **Information Fusion**
  - Signal and sensor processing for integration of large data into systems architectures

- **Complex Networks**
  - Mathematical and statistical methods for network and networked systems

- **Foundations of Information Systems**
  - Measurement and statistical verification for software, network, and hardware

- **Computational Mathematics**
  - Methods of computational modeling of large complex physical processes

- **Dynamic Data Driven Applications Systems**
  - Strategies for real time feedback of data into distributed computational processes

- **Optimization and Discrete Mathematics**
  - Optimization strategies and algorithms for discrete computational processes

- **Dynamics and Control**
  - Dynamical systems theory for assessment of performance of control architectures

DISTRIBUTION A: Approved for public release; distribution is unlimited.
Transitions

• Smolka/Havelund (Stony Brook/JPL)
  – *JPL Mars Science Laboratory* using rule-based specification language to ensure correct execution of software on next Mars Rover

• Harmonia STTR with AFRL/RI
  – using a modified version of Hadoop data analysis API for distributed parallel load balancing and computation over cloud architectures

• Tinelli/Barrett (Iowa/NYU)
  – *Rockwell-Collins* interested in transitioning SMT-based verifier research into formal methods toolkits for avionics systems

• Durfee (Univ of Michigan)
  – Collaboration on SBIR with *Intelligent Automation Inc.*, applying hybrid scheduling techniques to large-scale human expert teaming problems involving dozens of teams, hundreds of experts, and thousands of constraints.