Collaborative Software Engineering Tools Workshop
Dr. John Penix
Motivation*

- Over 10% of NASA’s civil servant and contractor workforce spend the majority of their time managing, developing, assuring, verifying, and/or maintaining software
- NASA has in operation use (and maintains) at least 200 million lines of source code
- Over $1 billion dollars of NASA’s annual $15 billion budget is software cost

* Based on estimates extrapolated from a 1993 study – source NASA Chief Engineer’s Office
Resilient Software Engineering
Project Overview

- **Fundamental New Technologies**
- **High Dependability Computing**
- **Intelligent Software Engineering Tools**
- **NASA Missions and Aerospace Industry**

<table>
<thead>
<tr>
<th>NASA/CICT NSF</th>
<th>High Dependability Computing</th>
<th>Intellige Software Engineering Tools</th>
<th>NASA Missions and Aerospace Industry</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ECS Resilient Software Engineering</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Intelligent Software Engineering Tools

Goal

- Reduce mission critical risks by developing tools and methods to identify and eliminate software errors

Mishap Cause Classification: 1/3 aerospace mishaps are software related

- Mars Climate Orbiter
- Mars Polar Lander
- Ariane 5

Sources of critical software risk:

- Misunderstanding requirements and hardware software interface
- Poor communication between teams
- Insufficient design and testing
- Inadequate or inappropriate methods and processes
Intelligent Software Engineering Tools Approach

- Mature advanced modeling and analysis tools
  - Advanced Software Verification and Testing
  - Integrated Formal/Informal Requirements Engineering

- Integrate and leverage state of art tool technology
  - Commercial and open source tools
  - Distributed collaboration frameworks

- Work with missions to infuse tools into specific processes:
  - Add early lifecycle requirements analysis capabilities
  - Improve testing effectiveness
  - Enable tool-supported, distributed code reviews
Collaborative Software Engineering Tools

- Problem: Miscommunication between teams is a common source of critical errors
  - NASA software is often developed by distributed multidisciplinary teams, compounding this problem
  - Existing software engineering tools do not provide strong support for collaboration

- Solution: Insert advanced tools into NASA mission processes by integrating with collaborative frameworks:
  - Integration of the Verification and Testing Tools into a collaborative environment to support collaborative software design and code reviews (ARC)
Distributed Collaborative Software Reviews

Workflow-driven review process

Event Server

Research Issue Distributing human-centered review process

Research Issue Integrating analysis tools into process

Local Developers

Automated Tools

Remote Experts
Workshop Goal

- Improve this presentation!
  - What are NASA’s problems?
  - What are some potential solutions?
  - What technology do we have that can play a role?
  - What research needs to be done?
  - How do we do that research?