



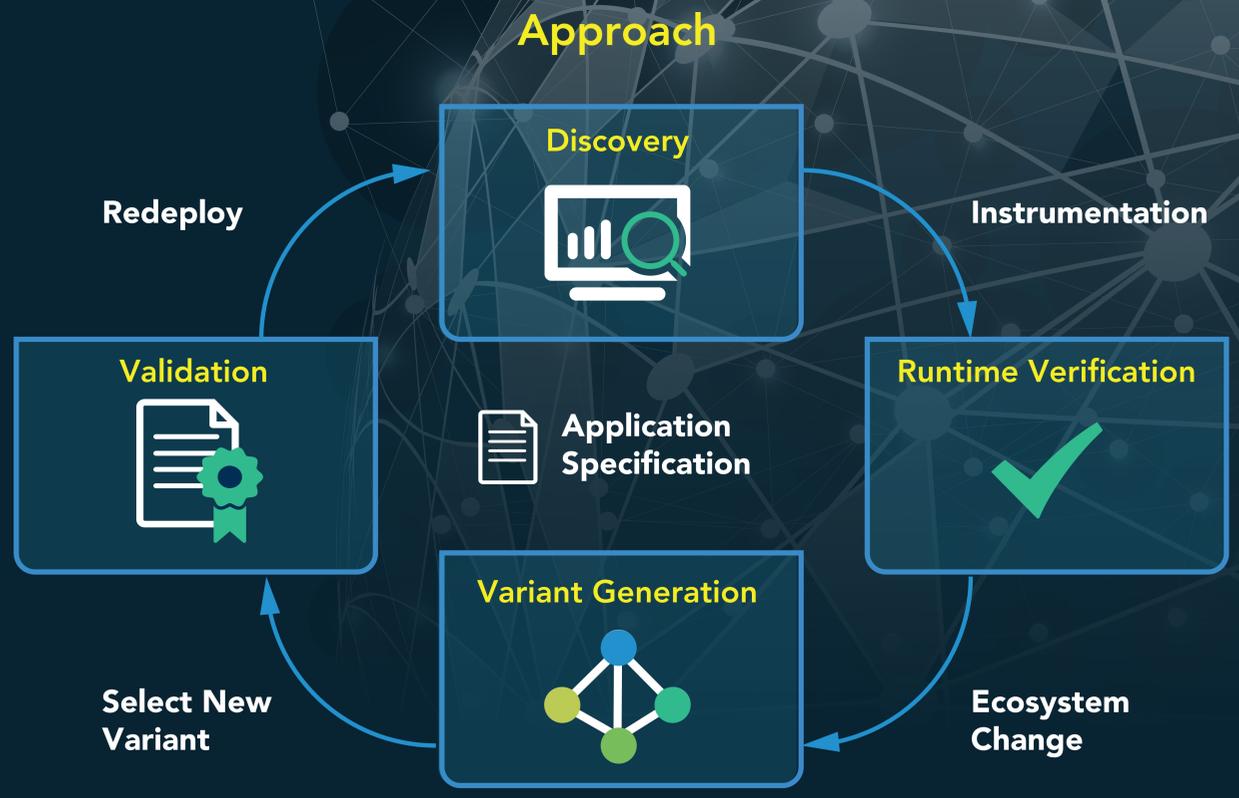
Building Resource Adaptive Software Systems (BRASS)



National Need
Architect, design, and build software systems resilient to changes in their ecosystem

Problem

- Today's software is designed with many hard coded assumptions about the anticipated ecosystem
 - Limits effectiveness of critical DoD systems to survive attack
 - Limits technology upgrades
- Software maintenance is a significant cost driver for long-lived DoD systems



Impact

- Demonstrate various syntactic and semantic forms of adaption to be applied over large code bases
- Demonstrate techniques for software adaptation that are generally applicable for a significant fraction of the code base (application and underlying ecosystem)

Future Work
MIT Lincoln Laboratory is developing a test framework for quantifying software resilience to changes in operational ecosystem

Problem

The diagram shows a transition from 'Logical Resources (libraries, data formats, structure)' and 'Physical Resources (energy, storage, processing, OS)' to a state where 'Resources Change Unexpectedly or Intentionally (e.g., failures, upgrades, patches)'. This leads to a state where 'Functionally correct code may operate incorrectly or inefficiently as the operational environment in which it executes changes'. A 'FRAGILE' warning icon is shown next to a person at a computer, indicating the risk of failure.

Functionally correct code may operate incorrectly or inefficiently as the operational environment in which it executes changes

Discovery and Analytics (DAS) Framework

- TA1 Platform**
 - Explore the applicability of adaptive and transformation analysis techniques to the development of resource adaptive systems
- TA2 Analytics**
 - Monitor system runtime behavior with respect to an application's specification
 - Guide transformation and selection strategies that propose alternative implementations for affected parts of an application
- TA3 Discovery**
 - Provide a detailed understanding of program properties with respect to the resources they depend on in their underlying ecosystems

The DAS Framework architecture diagram shows a flow from 'Platform' (Technical Area 1) through 'Platform APIs' to 'Evolving Ecosystems'. From 'Evolving Ecosystems', the flow goes to 'Models and Specifications', which then feeds into 'Adaptive Variants' (Runtime/VM). 'Adaptive Variants' is influenced by 'Intent' from 'Discovery' (Technical Area 3) and 'TRANS.' from 'Analytics' (Technical Area 2). 'Discovery' includes Abstractions, Analysis, Semantics, and Testing. 'Analytics' includes Monitoring & Profiling, Error & Cost Models, Selection & Transformation, and Validation. 'Programs' are processed by a 'Compiler' and then run on the 'Runtime/VM'.