



Technology Opportunities

Useful things from DARPA's Evolutionary Design of Complex Software (EDCS) Program

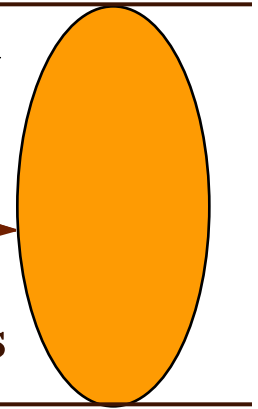
John Salasin, Ph.D.



Program Aims

- Paradigm-shifting, theoretical breakthrough
- Paradigm-shifting, useful, theoretical breakthrough (implies usage context)
- Significant, measurable improvement
- Everyone wants it (mass market products)

EDCS
Emphasis

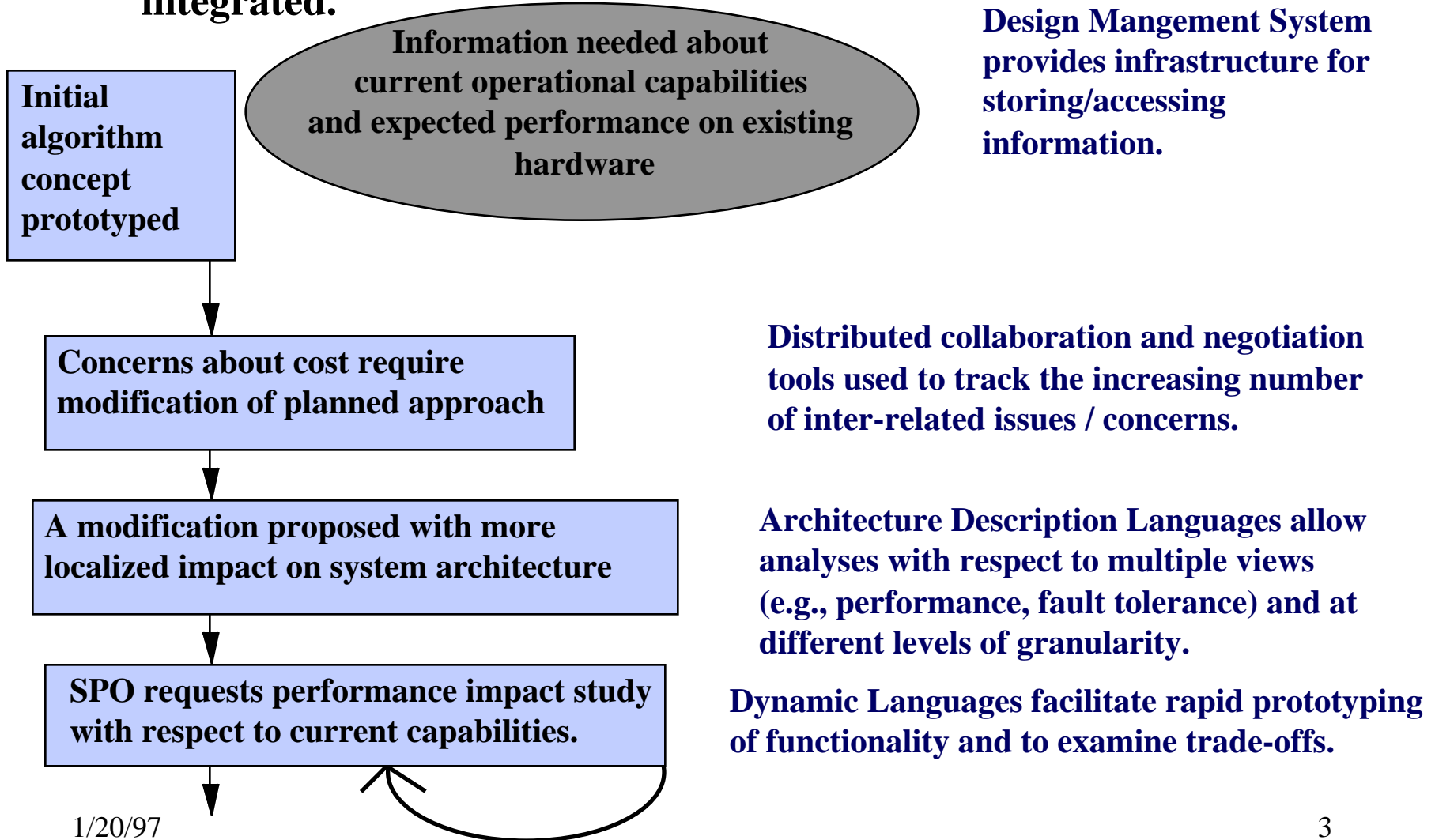


Goal: Major advances with measurable improvements in our ability to evolve software



Evolutionary Software

- **Need: Change in satellite coverage patterns with new sensor data integrated.**





Evolutionary Software

Need to understand why a module was designed the way it was (concerns about breakage)

Explanation from reverse engineering and Design Management system is immediately available.

Automated code generation based on Dynamic Language Prototype, optimized based on architectural model.

Architecture representations used to automatically generate simulations of SGS in context of the global digital infrastructure

Initial system debug and test

Dynamic system visualization tool provides view of the entire global intelligence and command and control infrastructure of which the updated sub-system will be a part

Detailed static test

Prototype code optimized for deployment and/or deployed code generated from architecture description

Dynamic test to specified level of assurance

Code analysis tools guarantee user-defined properties. Architectural representations used to generate test data and "oracles".

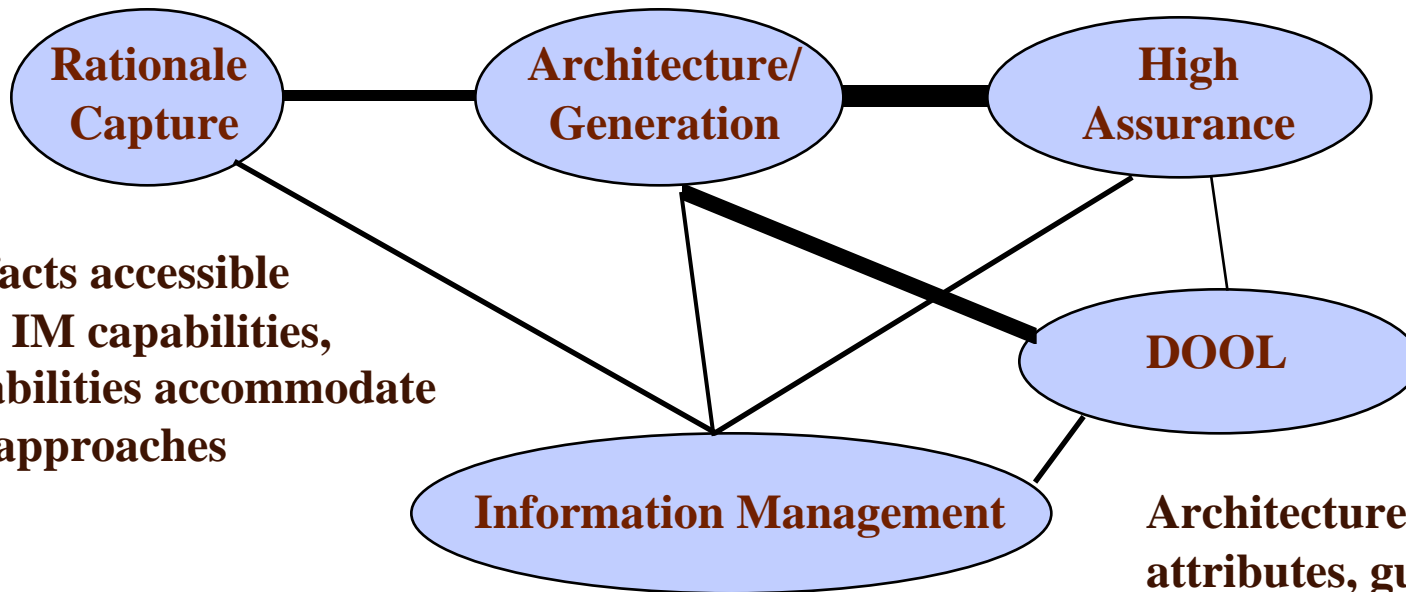
Old test cases/data reused where appropriate. New test cases generated.



“Cluster” organization

Architecture expresses rationale,
validated against rationale

Tests, analyses, proofs based
on architectural specifications



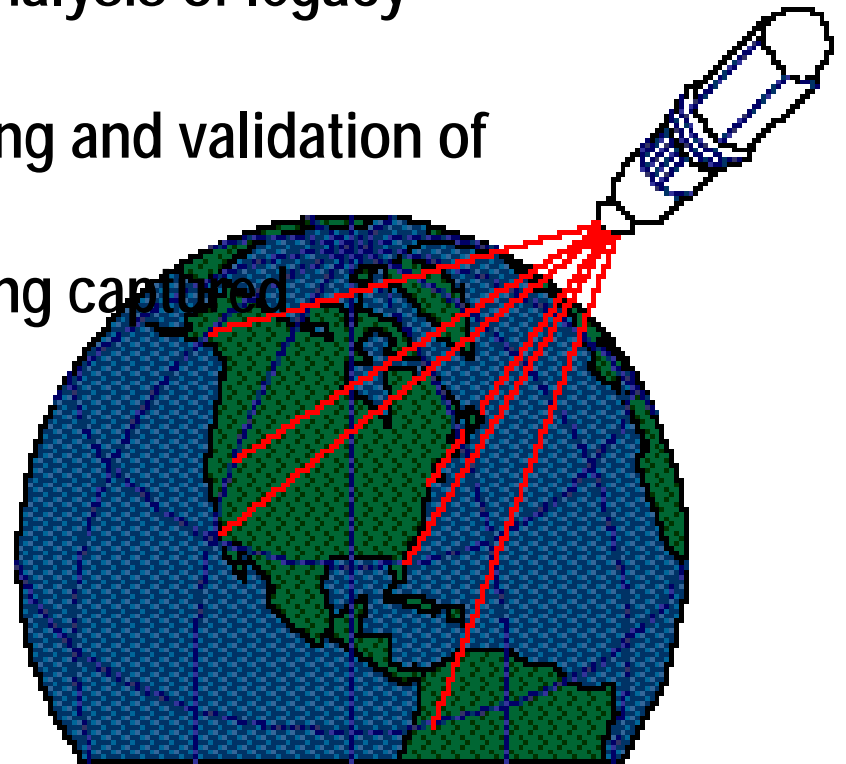
All artifacts accessible
through IM capabilities,
IM capabilities accommodate
diverse approaches

Architecture expresses
attributes, guides
optimizations



Current Efforts: USC/CSE, TRW & Aerospace

- 1) Demonstrate alternative architecture analysis and architecture modification using development rationale
- 2) Demonstrate re-engineering and analysis of legacy code
- 3) Demonstrate the design, prototyping and validation of new algorithms
- 4) Demonstrate rationale actively being captured and adopted into the evolution process





Future Efforts

Transition program
being proposed.

