



Research Issues in Deployed Adaptive Systems

NIPS 2006

Tom Dieterich

School of EECS

Oregon State University

<http://web.engr.oregonstate.edu/~tgd>

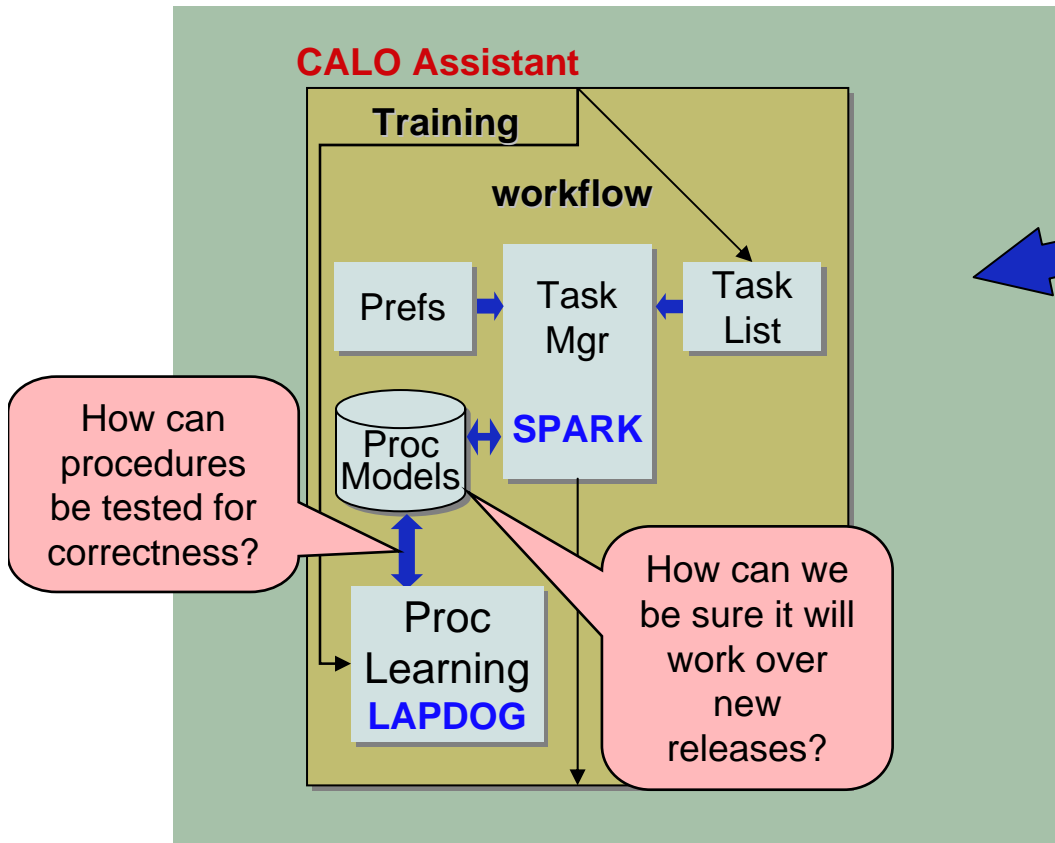
Overview

- Machine learning systems are beginning to be deployed in end-user systems
 - continue to learn and adapt after being deployed
- This raises two major challenges:
 - Need to integrate adaptive software into the software engineering life cycle
 - Need to have confidence in the system's behavior over time
- Key technical challenges:
 - testing the adaptive capabilities of the system
 - integrating adaptations into new system releases
 - sharing and integrating adaptations
 - tracking the region of competence
 - constraining adaptation

Example: Task Learning in CALO

Goal: Teach CALO workflows that it can perform automatically

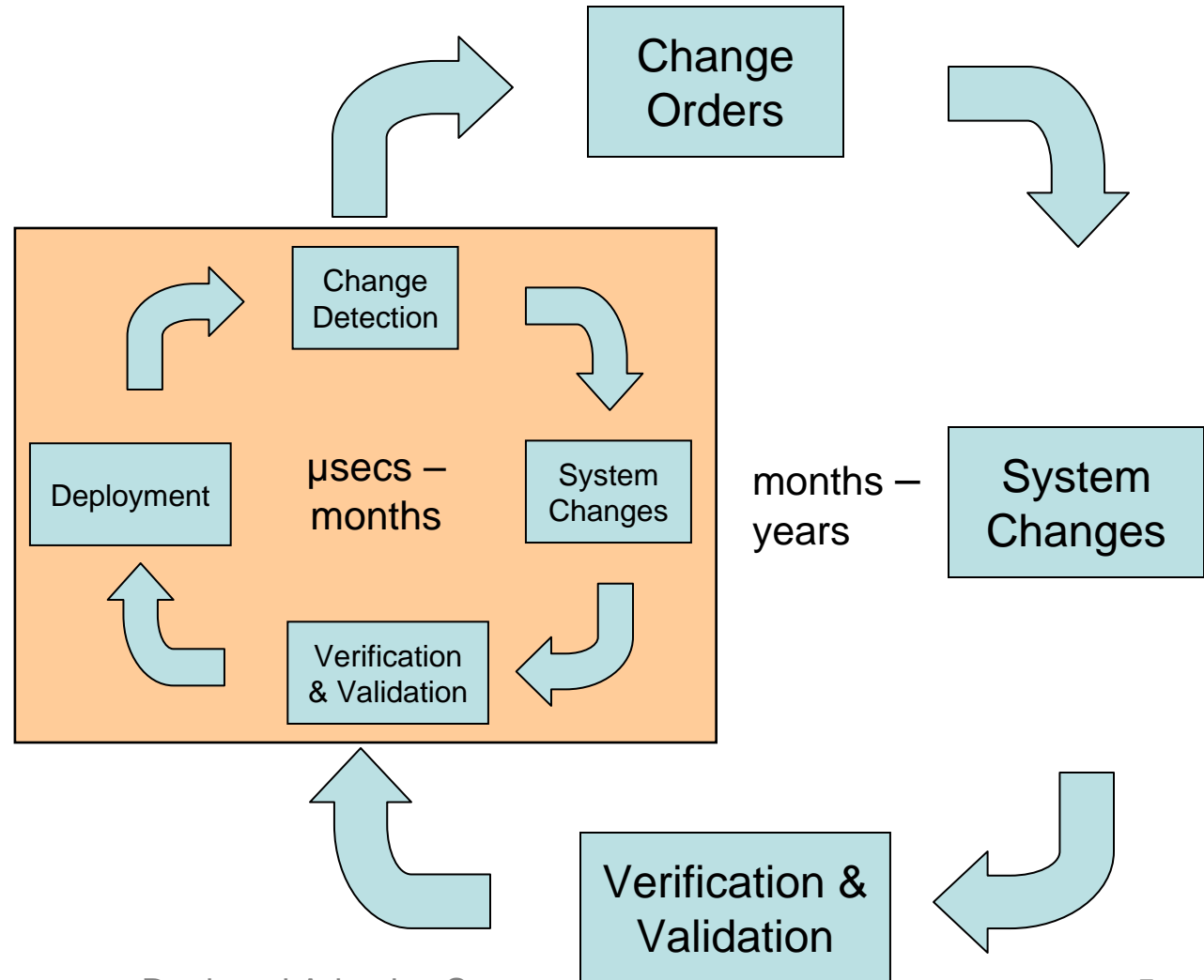
How can users gain trust in procedures created by other users?



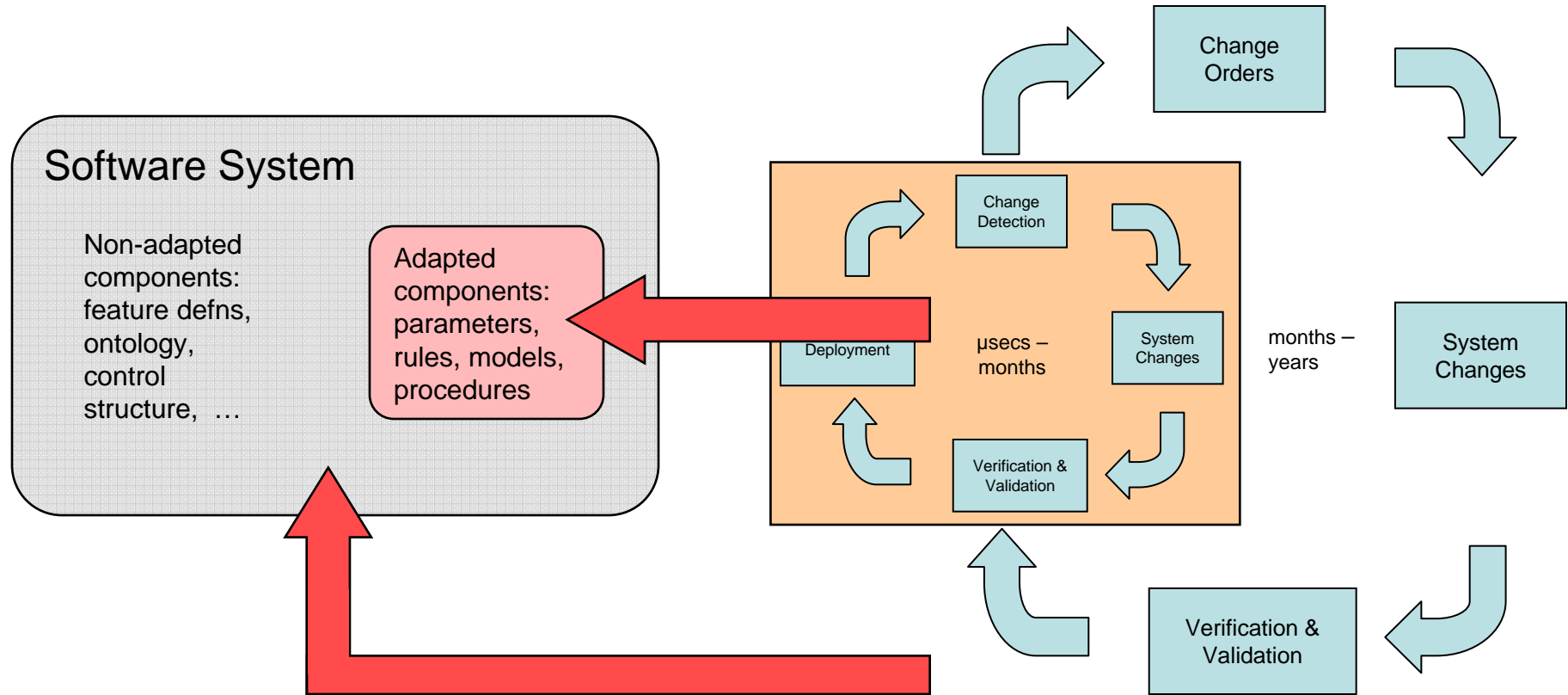
- Learn to file purchase requisitions
- Learn to file travel reimbursements

Deploying Adaptivity Requires Deploying the Software Process

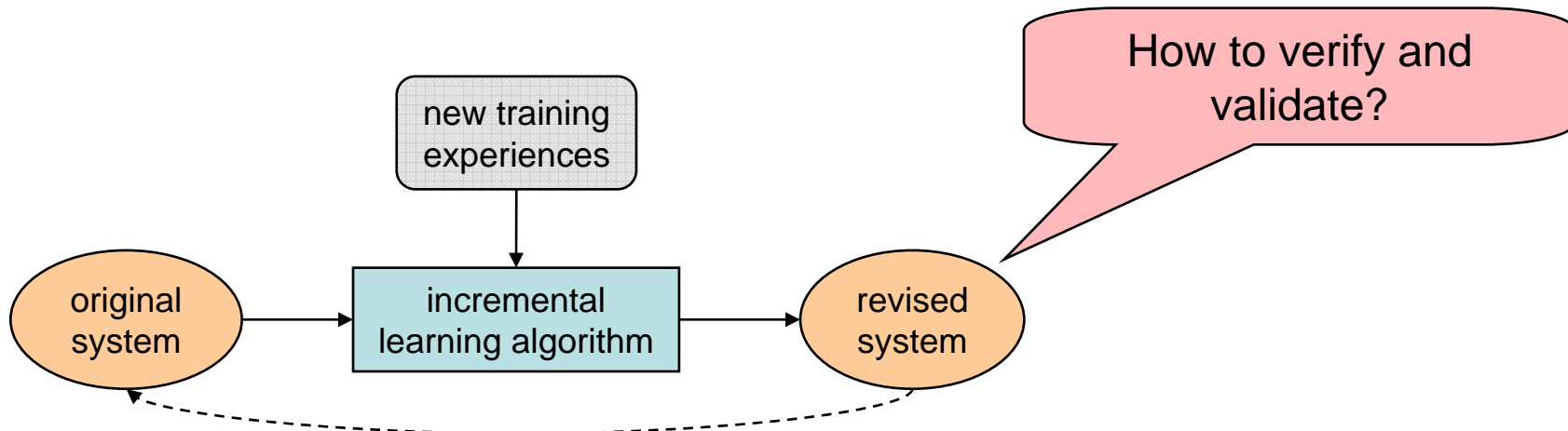
- System Autonomously detects need for change (e.g., prediction errors)
- Autonomously proposes candidate changes (e.g., via learning algorithm)
- Autonomously verifies and validates
- Autonomously deploys the changes



Division of labor between inner and outer software engineering processes



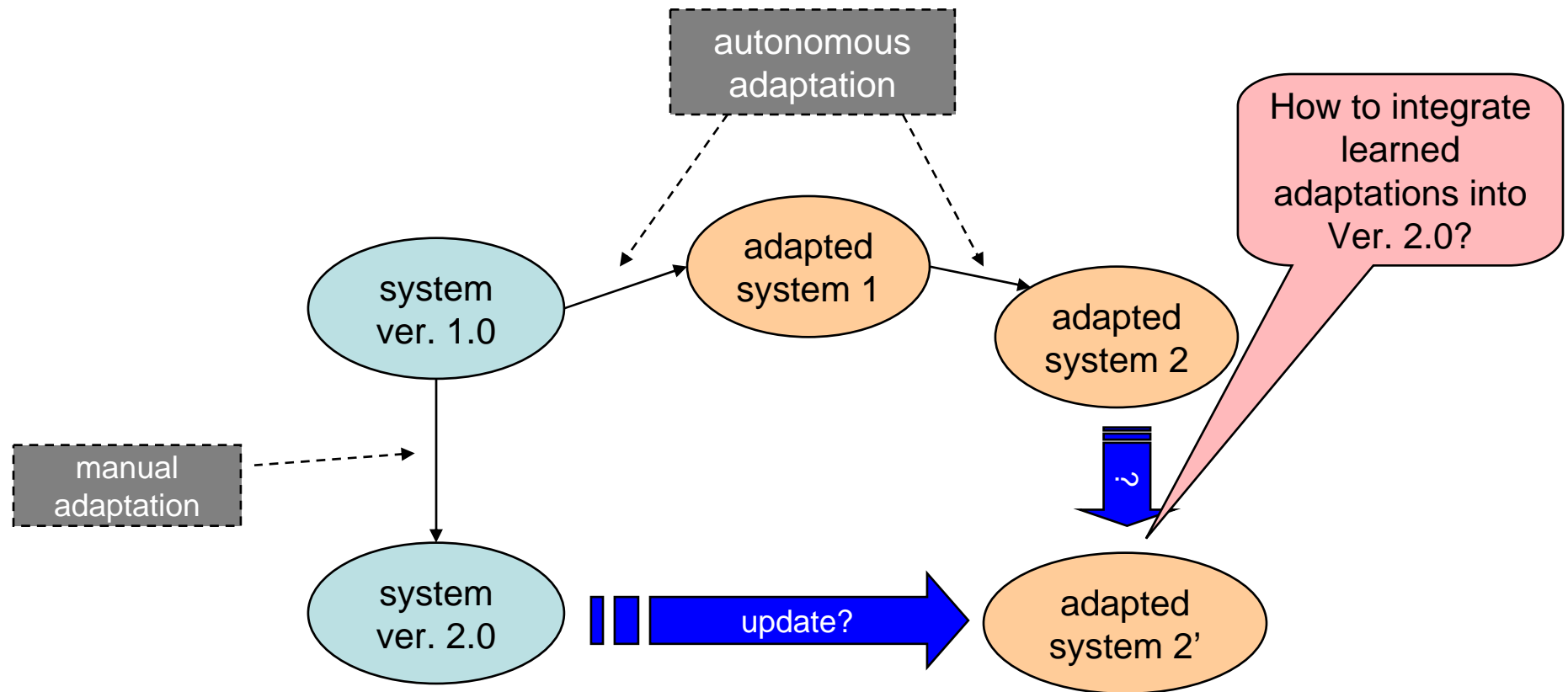
Technical Challenge 1: Autonomous Verification and Validation



Challenges:

- Source of test cases?
- Need models of the revised system and environment
- Need formal specification of properties to verify

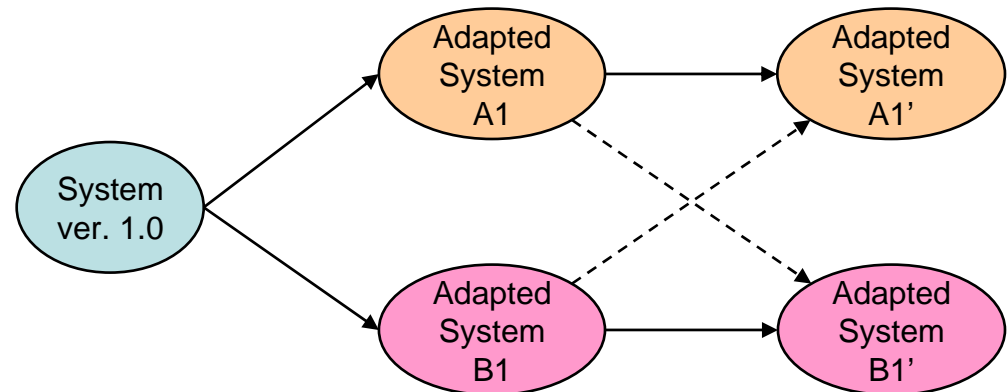
Technical Challenge 2: Integrating New Releases Without Losing Learned Knowledge?



Technical Challenge 3: Integration and Sharing of Adaptations

Multiple copies of an adaptive system will diverge over time

- “field discoveries” by one system should be shared, where appropriate, with others
- should occur at multiple time scales
 - immediately: new phishing tactics
 - days → weeks: improved detectors, learned procedures
 - annual: integrate general lessons learned into next software release

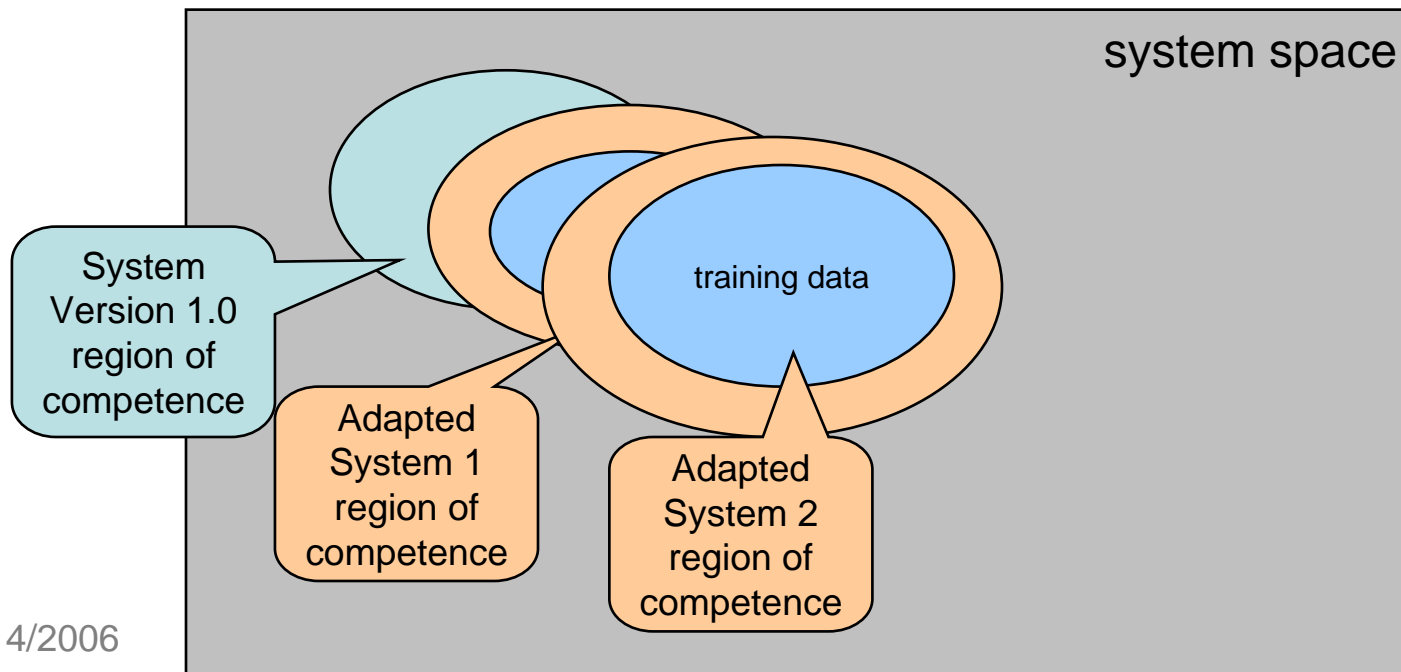


Challenges:

- How to decide which field discoveries to adopt?
- How to incorporate them without damaging ongoing performance?

Technical Challenge 4: Adaptive Region of Competence

- Standard software systems operate within a fixed region of competence:
 - they perform run-time checks of inputs/context to ensure correctness
- The region of competence for adaptive systems is changing dynamically
 - system must simultaneously learn its region of competence so that it can also perform these run-time checks



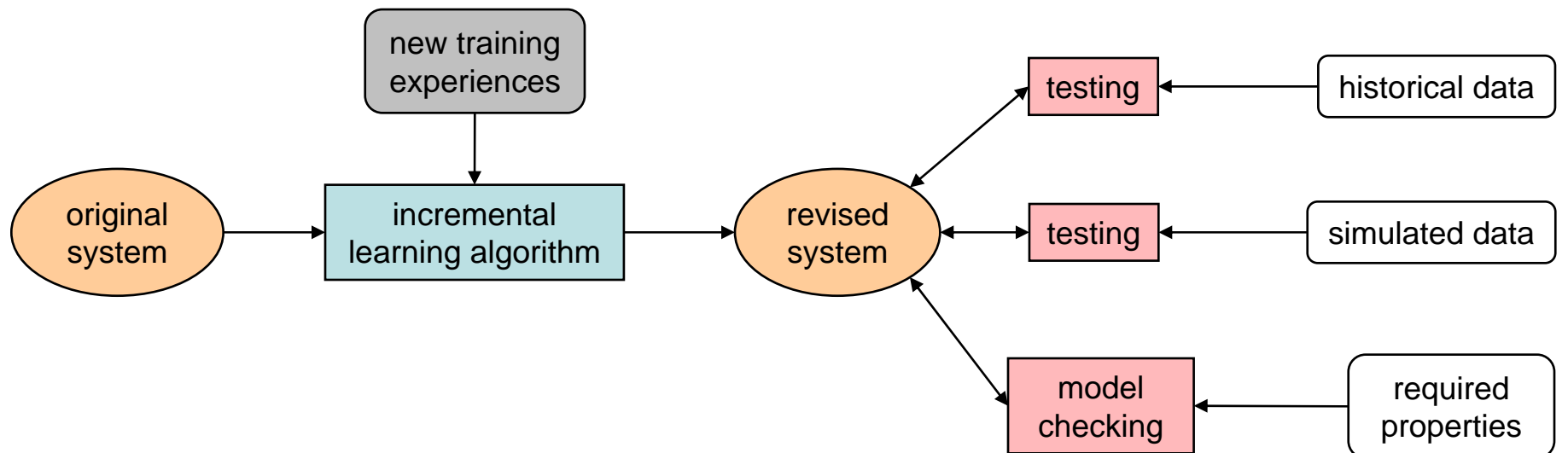
Technical Challenge 5: Constrained Adaptation

- Humans must be able to specify constraints that the system will not violate as it adapts
- The adaptive system should avoid blunders:
 - “Never delete most recent version of file”
 - “Never bring down the network with too much traffic”
- Technical challenges:
 - How can the user express these constraints to the system?
 - How can the system enforce these constraints?

Technical Approaches

		Approaches				
		deployed model checking	deployed simulation, historical data	self-describing components and APIs	self-aware architecture	constrainable systems
Technical Challenges	testing adaptation					
	divergent adaptation/ update merge					
	integration and sharing of adaptations					
	region of competence					
	control over adaptation					

Technical Approaches 1+2: Deployed Testing and Model Checking



System Properties:

provable properties:

- stability
- no deadlocks

constraints:

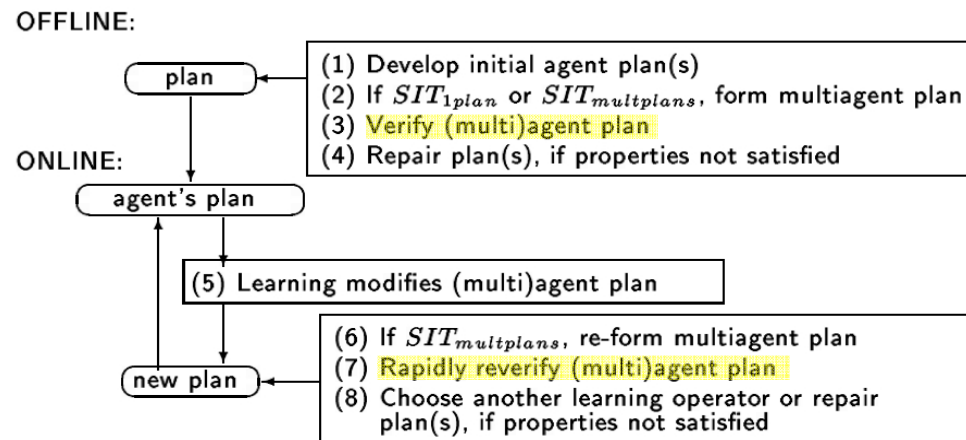
- never delete most recent version

probabilistic properties:

- $\text{Pr}(\text{delay} > 2\text{secs}) < 0.01$

Existing Work

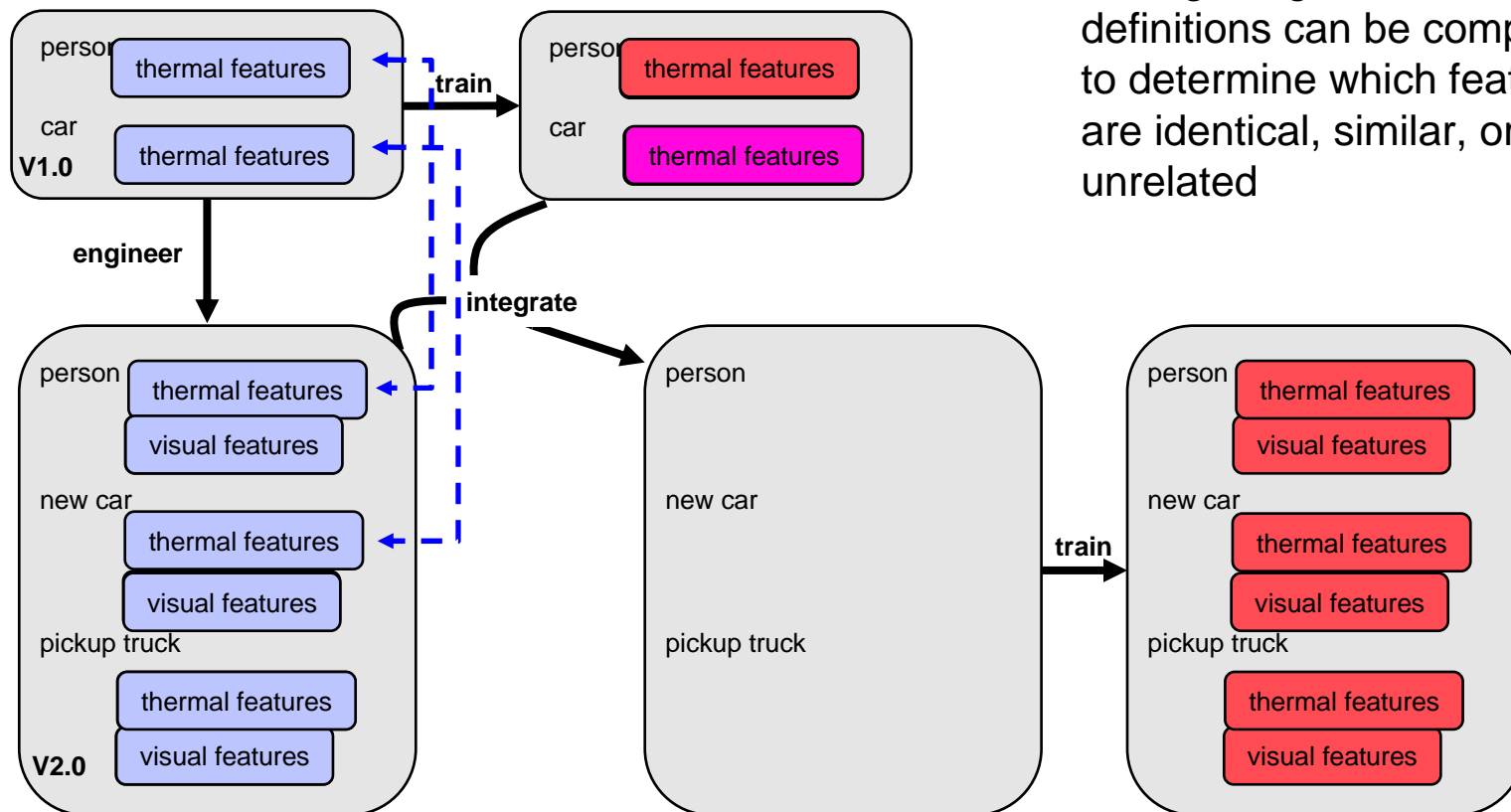
- Deployed verification
 - Gordon-Spears & Kiriakidis (2003, 2004): autonomous model checking of learned robot programs



- Mili, Cukic, Liu & Ben Ayed (2003) Verification and validation of on-line adaptive systems

Technical Approach 3: Self-Describing Components

- Features come with definitions
- During integration, definitions can be compared to determine which features are identical, similar, or unrelated

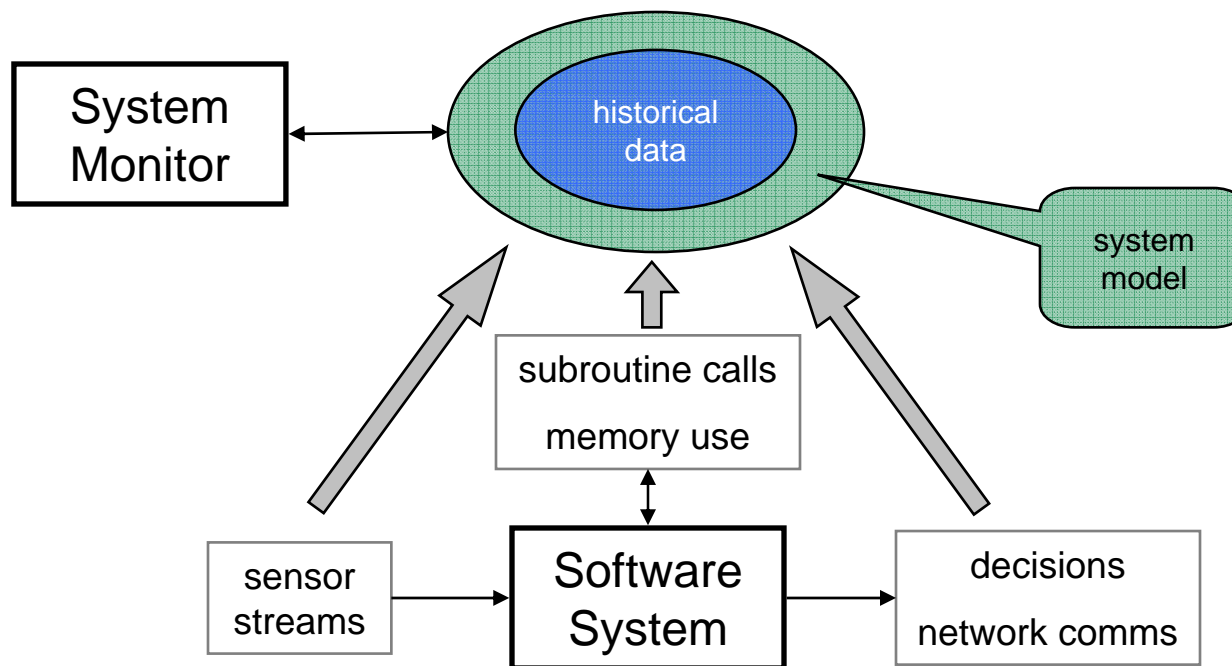


Existing Work

- Cumby & Roth (2002). Feature description logics for machine learning systems
- Harrold, Orso, et al. Selective regression testing via program analysis.
- Large literature and commercial products for migrating data across changes to database schemas.

Technical Approach 4: Self-Aware Software Architecture

- continuously model the region of competence of the system
- continuously profile system behavior to look for anomalies
 - (subroutine calls, memory use, sensor feeds, network communication)

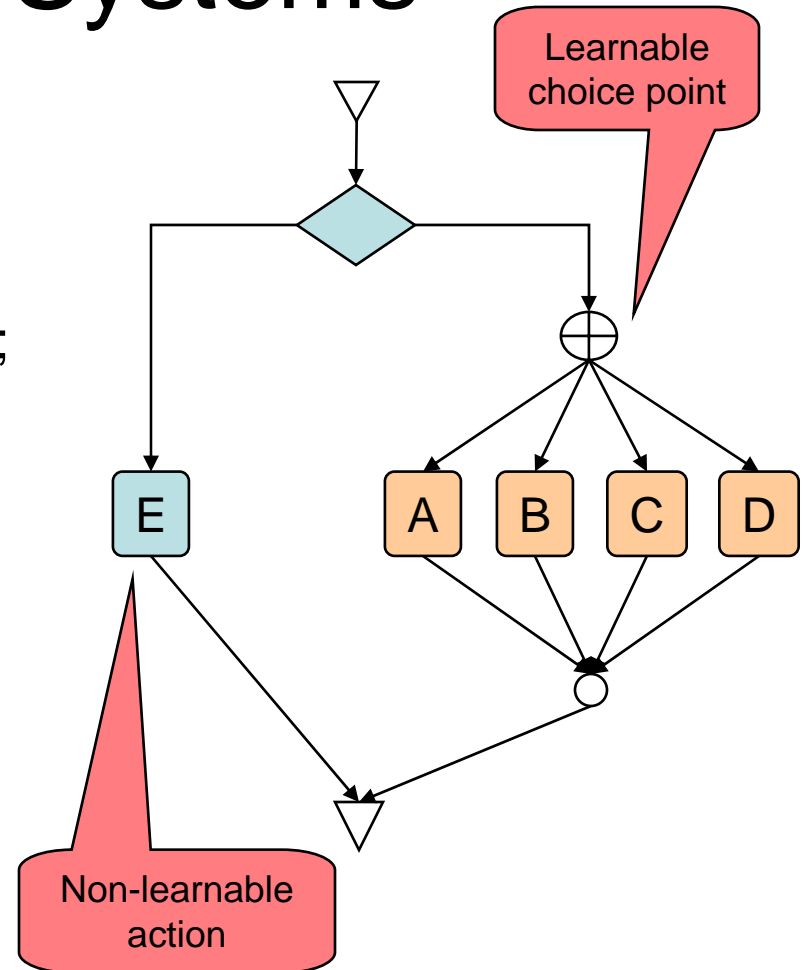


Existing Work

- Deployed self monitoring (e.g., Orso et al.)
 - learn to recognize software malfunctions by monitoring gross program behavior
 - raise alarm and collect detailed trace information when system departs from normal region
- Input space density estimation (e.g., Shen, Li, Herlocker & Dietterich, 2006)
 - modern statistical learning tools for estimating the probability that the system has seen previous cases similar to a new case
- GE coal-fired power generation
 - use region of competence to gradually improve a powerplant

Technical Approach 5: Constrainable Systems

- Future adaptations are guaranteed not to violate constraints
 - Partial Programming (Genesereth; Russell; etc.): Only designated parts of the system are permitted to adapt, and then only permitted to perform certain actions or action combinations.
 - Augmentation-based Learning (Oblinger, Castelli, Bergman, 2006)



Summary

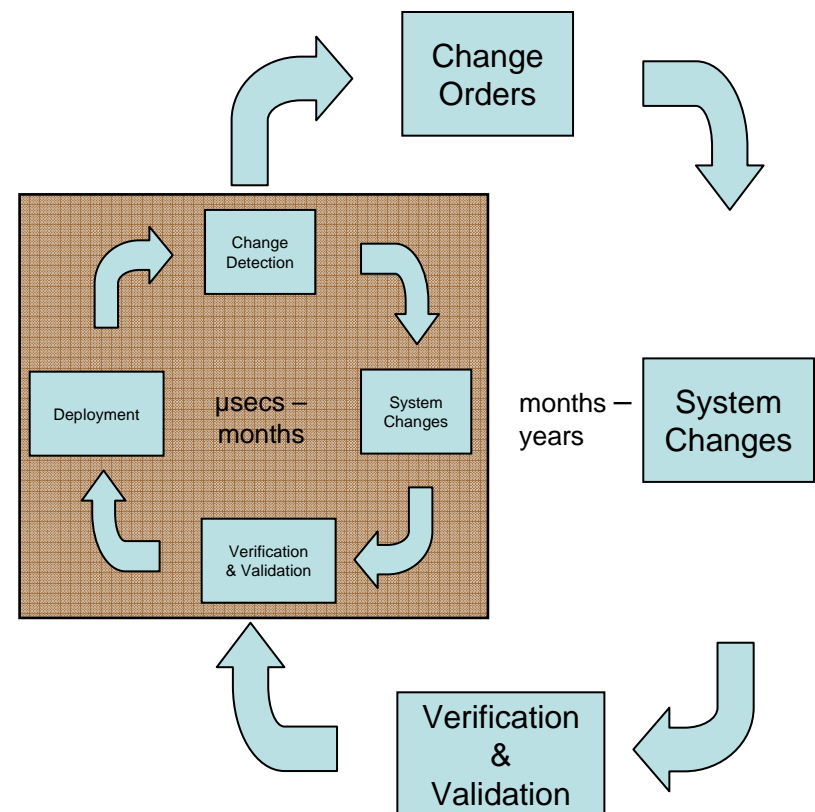
- Need to integrate adaptive software into the software engineering life cycle
 - deployed verification & validation
 - ability to merge new releases, share field discoveries
- Need to have confidence in the system behavior over time
 - deployed verification & validation
 - region of competence
 - constrainable systems

New Software Process for Deployed Adaptive Systems

- testing of adaptive capabilities
 - testing alternative futures under simulation
 - back testing on historical data
 - “red team” exercises
- online audit trail
- human “adaptation supervisor” monitors the systems as they adapt

Even version 0 of system should be built with this process

Essential for certification of adaptive systems



Acknowledgements

- Leslie Kaelbling (MIT)
- Ted Senator (SAIC)

- Discussions with many other people
 - Dragos Margineantu (Boeing)
 - Piero Bonisonne (GE CRD)
 - Kiri Wagstaff (NASA/JPL)