Fall 2014
SEI Research Review
Value-Driven Iterative and Incremental Development

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Ipek Ozkaya
October 28, 2014

© 2014 Carnegie Mellon University

Report Documentation Page

Form Approved
OMB No. 0704-0188

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE **28 OCT 2014**	2. REPORT TYPE **N/A**	3. DATES COVERED
4. TITLE AND SUBTITLE **Value-Driven Iterative and Incremental Development**		5a. CONTRACT NUMBER 5b. GRANT NUMBER 5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) **Ozkaya /Ipek**		5d. PROJECT NUMBER 5e. TASK NUMBER 5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213**		8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)		10. SPONSOR/MONITOR’S ACRONYM(S) 11. SPONSOR/MONITOR’S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT **Approved for public release, distribution unlimited.**		
13. SUPPLEMENTARY NOTES **The original document contains color images.**		
14. ABSTRACT		
15. SUBJECT TERMS		

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**SAR**	**17**	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std Z39-18

Copyright 2014 Carnegie Mellon University

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Department of Defense.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

This material has been approved for public release and unlimited distribution except as restricted below.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

DM-0001795

Value-Driven Incremental Development

The current approach in highly-regulated domains, such as DoD, **still** depends on lengthy requirements, design, test, and evaluation cycles

- Excessive documentation without analysis
- Monolithic architecting, modeling, or assurance activities result in rework

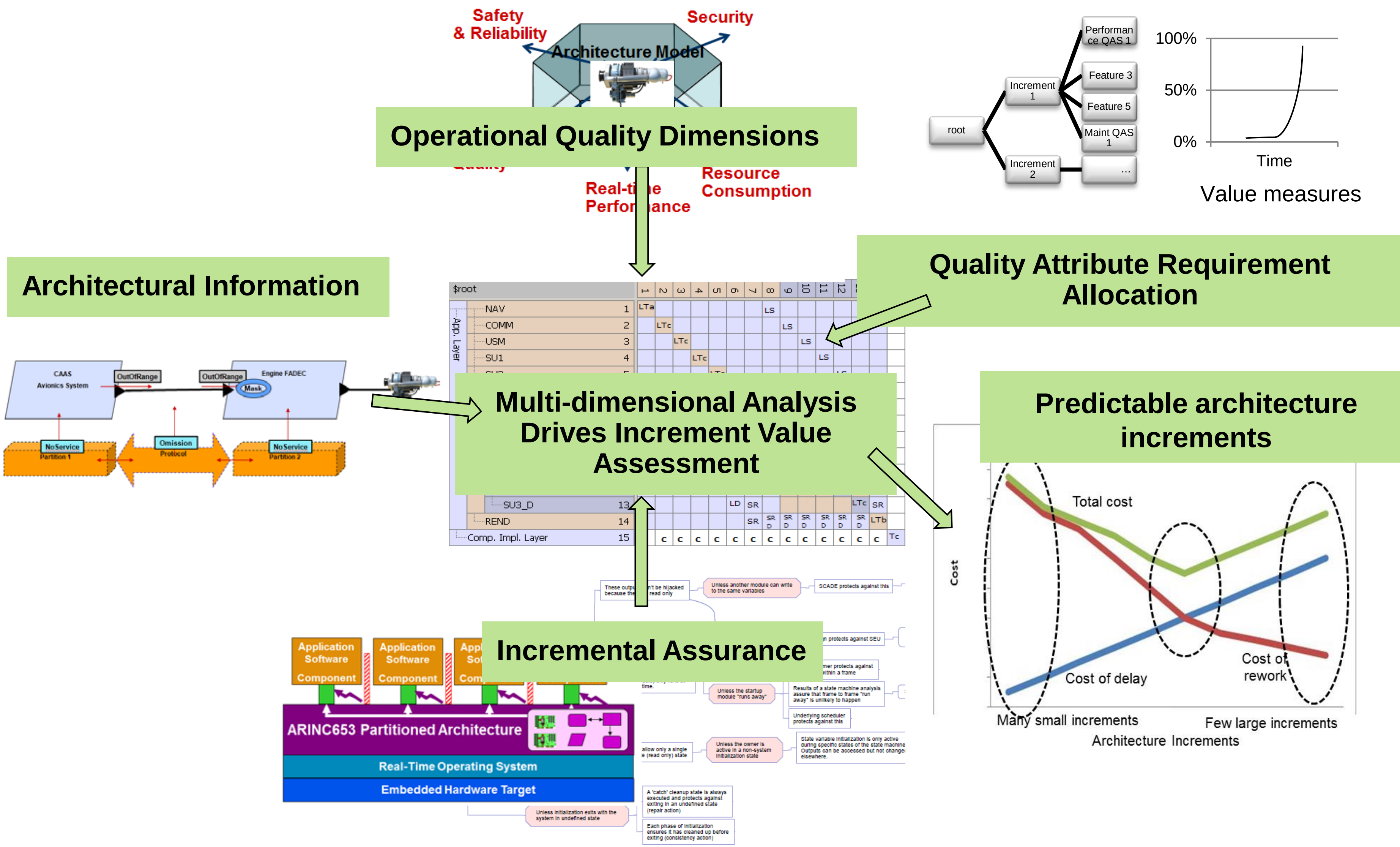
The goal of this project is to develop architecture dependency analysis focused techniques to integrate architecture analysis with development efforts early-on and continuously:

Our approach includes:

- Architecture dependency management
- Incremental assurance structuring
- Quality attribute allocation techniques

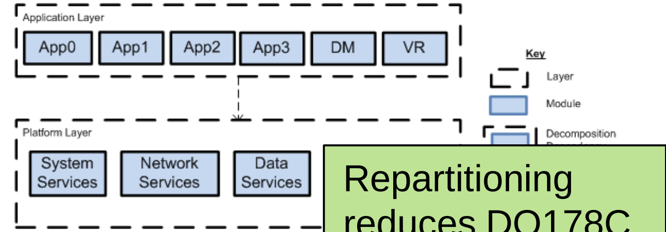
© 2014 Carnegie Mellon University

Technical Approach



© 2014 Carnegie Mellon University

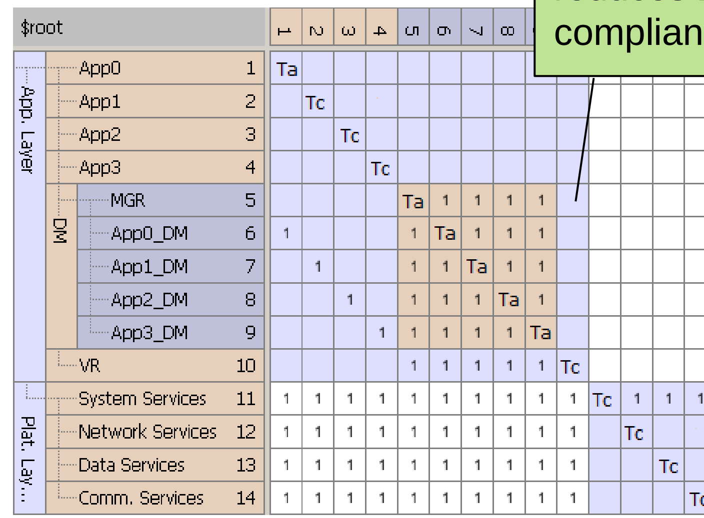
Architectural Dependencies

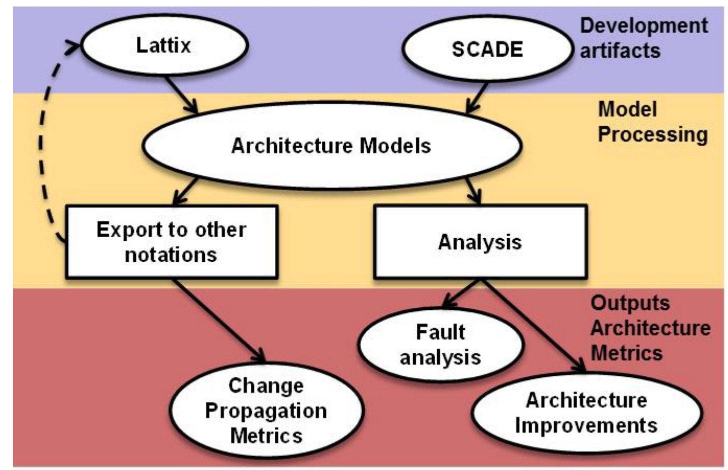


Repartitioning reduces DO178C compliant testing

Technical approach

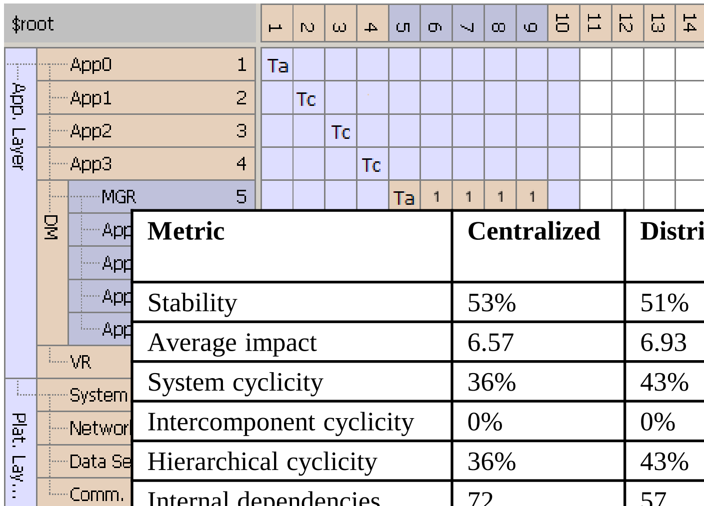
- Track additional information (e.g., safety critical testing level) using a DSM
- Extract fault ontology and propagation-related information from architecture analysis tools (e.g., AADL)
- Apply structural metrics (e.g., stability) on the augmented DSM and check on collaborator data





© 2014 Carnegie Mellon University

Architectural Dependencies



Metric	Centralized	Distributed	Health *
Stability	53%	51%	–
Average impact	6.57	6.93	–
System cyclicity	36%	43%	–
Intercomponent cyclicity	0%	0%	=
Hierarchical cyclicity	36%	43%	–
Internal dependencies	72	57	+
Average dependency	5.14	4.07	+
Average cumulative dependency	7.57	7.93	–
Normalized cumulative dependency	2.356	2.467	–
Connectedness	51%	53%	–
Connectedness enrichment	1.00	1.37	–
Connectedness strength	5.86	5.10	+
Coupling	11%	16%	–
Coupling enrichment	1.00	15.00	–
Coupling strength	0.36	0.30	+

	1	2	3	4	5	6	7	8	9	10
App3_DM (1)		OK			OK					OK
VR (2)	OK			OK				OK	KO	OK
App0 (3)									OK	
App2_DM (4)		OK					OK			OK
App3 (5)	OK									
App1 (6)								OK		
App2 (7)				OK						
App1_DM (8)		OK				OK				OK
App0_DM (9)		KO	OK							KO
MGR (10)	OK	OK		OK				OK	KO	

Criticality-level interaction across partitions cannot be captured with code-based analysis

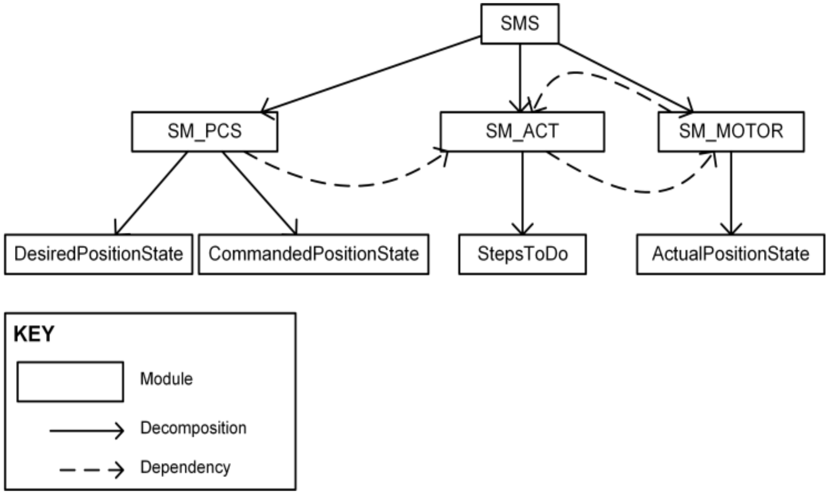
Changes propagate beyond models and implementation

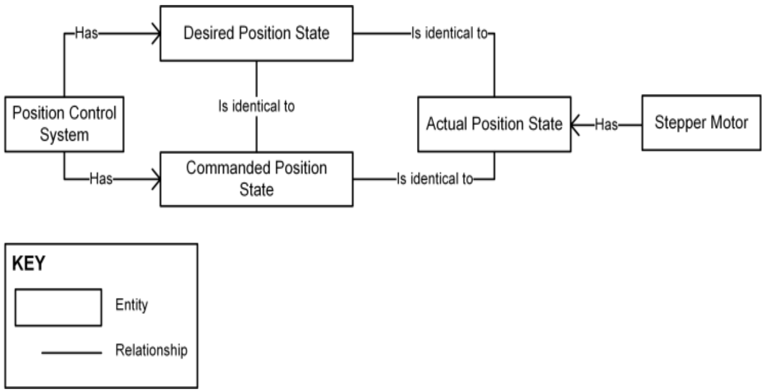
© 2014 Carnegie Mellon University

Stepper Motor Example

A stepper motor systems is an open loop system with no feedback on the successful execution of the steps it must take for a position change command.

- How can we ensure that steps are not missed during execution?
- How can we ensure that when change are made testing resources are spent on target?





© 2014 Carnegie Mellon University

Dependency Type Guide

Dependency type		Description
A	Aggregation	Data element A and Data element B have a semantic coherence that can be aggregated as Module AB
C	Control	Module A depends on the presence of a correct functioning module B.
D	Data	For a module B to execute correctly, the syntax (type or format)/semantics of the data produced by module A must be consistent with the assumptions of module B.
L	Location	For B to execute correctly, the runtime location of A must be consistent with the assumptions of B.
R	Allocation of responsibilities	Behavior and functionality assigned to design time elements, used to separate concerns, e.g. safety criticality.
S	Sequence of flow	For B to execute correctly, it must receive the data produced by A in a fixed sequence (data flow). For B to execute correctly, A must have executed previously within certain timing constraints (control flow).
P	Physical resource behavior	For B to execute correctly, the resource behavior of A must be consistent with B's assumptions about physical resource (such as bandwidth, memory, storage capacity, CPU, etc.) usage or ownership,
Q	Quality of service	For B to execute correctly, some property involving the quality of the data or service provided by A must be consistent with B's assumptions.
V	Virtual resource behavior	For B to execute correctly, the resource behavior of A must be consistent with B's assumptions about virtual resource usage or ownership

© 2014 Carnegie Mellon University

Missing information

Module-view dependencies

	SMS-Arch-1	SMS-Arch-2	SMS-Arch-3	SMS-Arch-4	SMS-Arch-5	SMS-Arch-6	SMS-Arch-7	SMS-Arch-8
SMS-Arch-1: SMS.SM_PCS	.			CD				
SMS-Arch-2: SMS.SM_PCS.DesiredPositionState	D	.						
SMS-Arch-3: SMS.SM_PCS.CommandedPositionState	D		.					
SMS-Arch-4: SMS.SM_ACT				.		CD		
SMS-Arch-5: SMS.SM_ACT.StepsToDo				D	.			
SMS-Arch-6: SMS.SM_MOTOR						.		
SMS-Arch-7: SMS.SM_MOTOR.ActualPositionState						D	.	
SMS-Arch-8: SMS.SM_HM								.

Data and control relationships can be captured

Multi-view dependencies

	SMS-Arch-1	SMS-Arch-2	SMS-Arch-3	SMS-Arch-4	SMS-Arch-5	SMS-Arch-6	SMS-Arch-7	SMS-Arch-8
SMS-Arch-1: SMS.SM_PCS	.			CDS				L
SMS-Arch-2: SMS.SM_PCS.DesiredPositionState	D	.	A				A	
SMS-Arch-3: SMS.SM_PCS.CommandedPositionState	D	A	.				A	
SMS-Arch-4: SMS.SM_ACT	S			.		CD		
SMS-Arch-5: SMS.SM_ACT.StepsToDo				D	.			
SMS-Arch-6: SMS.SM_MOTOR				S		.		
SMS-Arch-7: SMS.SM_MOTOR.ActualPositionState		A	A			D	.	
SMS-Arch-8: SMS.SM_HM	LP							.

Aggregation, sequence of flow, location and physical resource dependencies can be captured when model-based analysis is conducted

© 2014 Carnegie Mellon University

Implication on testing resources

Using clustering algorithms we can locate the most connected areas that need to be tested further.

	SMS-Arch-8	SMS-Arch-6	SMS-Arch-4	SMS-Arch-1	SMS-Arch-5	SMS-Arch-2	SMS-Arch-7	SMS-Arch-3
SMS-Arch-8: SMS.SM_HM	.							
SMS-Arch-6: SMS.SM_MOTOR		.						
SMS-Arch-4: SMS.SM_ACT		CD	.					
SMS-Arch-1: SMS.SM_PCS			CD	.				
SMS-Arch-5: SMS.SM_ACT.StepsToDo			D		.			
SMS-Arch-2: SMS.SM_PCS.DesiredPositionState				D		.		
SMS-Arch-7: SMS.SM_MOTOR.ActualPositionState		D					.	
SMS-Arch-3: SMS.SM_PCS.CommandedPositionState				D				.

	SMS-Arch-6	SMS-Arch-8	SMS-Arch-4	SMS-Arch-1	SMS-Arch-5	SMS-Arch-7	SMS-Arch-3	SMS-Arch-2
SMS-Arch-6: SMS.SM_MOTOR	.		S					
SMS-Arch-8: SMS.SM_HM		.		LP				
SMS-Arch-4: SMS.SM_ACT	CD		.	S				
SMS-Arch-1: SMS.SM_PCS		L	CDS	.				
SMS-Arch-5: SMS.SM_ACT.StepsToDo			D		.			
SMS-Arch-7: SMS.SM_MOTOR.ActualPositionState	D					.	A	A
SMS-Arch-3: SMS.SM_PCS.CommandedPositionState				D		A	.	A
SMS-Arch-2: SMS.SM_PCS.DesiredPositionState				D		A	A	.

© 2014 Carnegie Mellon University

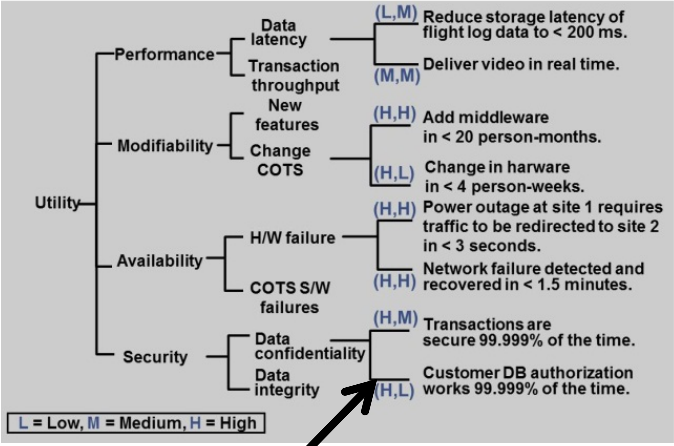
Incremental assurance

Technical approach

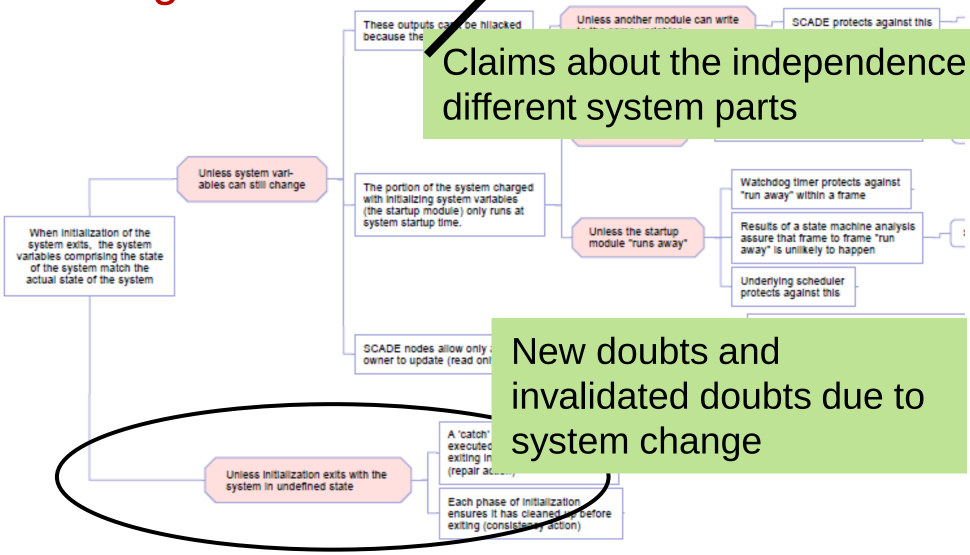
- Use quality attribute utility trees and architectural dependency analysis to structure the system's architecture and its assurance argument

FY14 results

- confidence map notation and theory
- generation capability of assurance cases from requirements



generate the assurance case



© 2014 Carnegie Mellon University

Quality Attribute Allocation to Iterations

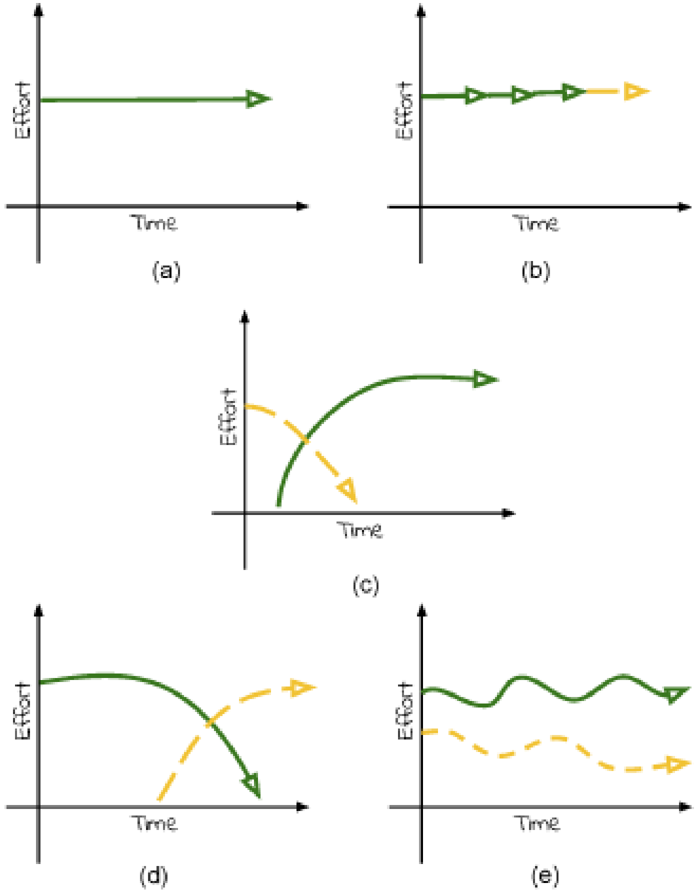
Technical approach

- Use architecture tactics-based and story slicing techniques to link architectural tasks to backlog management tools

FY14 results

- Patterns of iterative incremental development
- Rework occurs regardless of process followed

Ongoing organization wide-surveys of the patterns of iterative incremental development



© 2014 Carnegie Mellon University

Example: Performance Improvement Evolution

	QAR Parsing		Value	Effort
A-S1	Stimulus: Context: Response:	Customer initiates manual process (multi-user) Users processing transactions with system Process volume of transactions		
A-S2	Stimulus: Context: Response:	Customer initiates automated process System processing transactions (single-user) Process batch transactions; new time less than current time	Enhanced "Autopilot" feature	3x
A-S3	Stimulus: Context: Response:	Order process initiates transaction System processing transaction; single-user Process individual transaction; new time less than current time	Improved order…	1x
A-S4	Stimulus: Context: Response:	Order process initiates transaction System processing transaction; single-user Process individual transaction; processing time less than or equal to 1 s	Further improved order capability	2x
A-S5	Stimulus: Context: Response:	Customer submits orders System processing trans; rotary algorithm; multi-user Process and prioritize transactions	…n feature	

Ratcheting Stimulus

Ratcheting Response Measure

Ratcheting Environment

© 2014 Carnegie Mellon University

Publications

Prototypes:

Semantic wiki to capture architecture-tactics

Assurance case generation tool

Publications:

Architectural dependency analysis to understand rework costs for safety-critical systems – ICSE 2014

Design Rule Spaces: A New Form of Architecture Insight – ICSE 2014

Evolutionary Improvements of Cross-cutting Concerns: Performance in Practice – ICSME 2014

Increasing Confidence by Strengthening an Inference in a Single Argument Leg: An Alternative to Multi-Legged Arguments – Dependable System Networks (DSN)

Using AI to model quality attribute tradeoffs – AI in Requirements Engineering @ RE 2014

Agile in Distress: Architecture to the Rescue - Principles of Large-Scale Agile Development @ XP Conference

Research Workshops Led:

6th International Workshop on Managing Technical Debt @ ICSME 2014

1st International Workshop on Software Architecture & Metrics @ WICSA 2014

© 2014 Carnegie Mellon University

Going Forward in FY15

Improving Software Sustainability through Data-driven Technical Debt Management

What code and design indicators can be discovered in a repeatable way to measure and manage technical debt?

Incremental Life Cycle Assurance of Critical Systems

How can system assurance confidence and cost be improved through requirements coverage and consistency checking and compositional verification evidence?

© 2014 Carnegie Mellon University

Team: Value-Driven Incremental Development

SEI team members

- Ipek Ozkaya, PhD (lead)
- Robert Nord, PhD (co-lead)
- Stephany Bellomo, MSc.
- Julien Delange, PhD
- Neil Ernst, PhD
- Peter Feiler, PhD
- Ian Gorton, PhD
- John Goodenough, PhD
- Rick Kazman, PhD
- Ari Klein, PhD Candidate
- Chuck Weinstock, PhD

Collaborators

- Prof. Philippe Kruchten, PhD
University of British Columbia
- Prof. Raghu Sangwan, PhD
Penn State University
- Prof. David Garlan, PhD
Carnegie Mellon University
- John McGregor, PhD
Clemson University
- And other industry and DoD collaborators

© 2014 Carnegie Mellon University

Contact Information

Ipek Ozkaya, PhD

SSD SEAP Architecture Practices Initiative

ozkaya@sei.cmu.edu

+1 412-268-3551

Web

www.sei.cmu.edu

www.sei.cmu.edu/contact.cfm

www.sei.cmu.edu/architecture/research/

© 2014 Carnegie Mellon University