



University of Southern California  
Center for Systems and Software Engineering

---

# Assessing Quality Processes with ODC COQUALMO

**Ray Madachy, Barry Boehm**

**USC Center for Systems and Software Engineering**

**{madachy, boehm}@usc.edu**

**2008 International Conference on Software Process**

**May 10, 2008**



# Introduction

- **Cost, schedule and quality are highly correlated factors in software processes.**
- **In recognition, the COConstructive QUALity MOdel (COQUALMO) was created to predict defects as an extension of the COCOMO II software cost model**
  - **Supports cost/schedule/quality tradeoff analyses**
- **ODC COQUALMO is a further extension that predicts software defects introduced and removed classifying them with Orthogonal Defect Classification (ODC) defect types**
- **ODC COQUALMO is initially targeted to support critical NASA flight projects on the Constellation program \***

\* This work was partially sponsored by NASA AMES *Software Risk Advisory Tools* Cooperative Agreement No. NNA06CB29A

# COQUALMO Background

- **COQUALMO is a COCOMO extension that predicts the number of residual defects in a software product [Chulani, Boehm 1999]**
  - Uses COCOMO II cost estimation inputs with defect removal parameters to predict the numbers of generated, detected and remaining defects for requirements, design and code
- **Enables 'what-if' analyses that demonstrate the impact of**
  - Defect removal techniques for automated analysis, peer reviews, and execution testing on defect types
  - Effects of personnel, project, product and platform characteristics on software quality
- **Provides insights into**
  - Assessment of payoffs for quality investments
  - Understanding of interactions amongst quality strategies
  - Probable ship time

# ODC COQUALMO Overview

*COQUALMO extensions to COCOMO II in red*

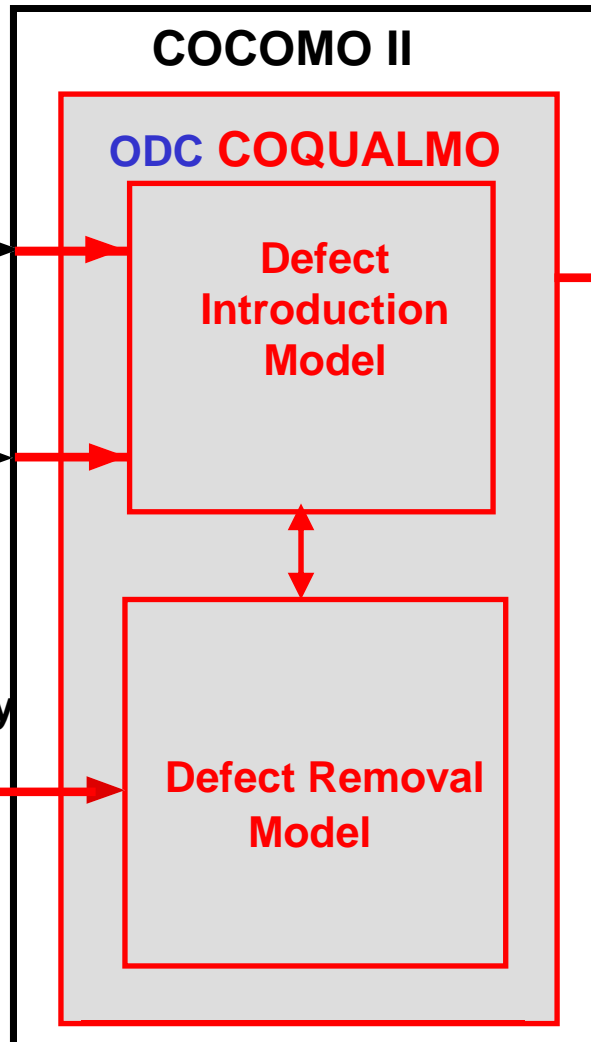
*ODC additions in blue*

Software size

Software product, process, platform and personnel attributes

Defect removal capability levels

- Automated analysis
- Peer reviews
- Execution testing and tools



Software development effort and schedule

Defect density per unit of size

Number of residual defects

• **Requirements**

- Correctness
- Completeness
- Consistency
- Ambiguity/Testability

• **Design**

- Interface
- Timing
- Class/Object/Function
- Method/Logic/Algorithm
- Data Values/Initialization
- Checking

• **Code** (same 6 types as Design)

# COQUALMO Defect Removal Rating Scales - Top Level

	Very Low	Low	Nominal	High	Very High	Extra High
<b>Automated Analysis</b>	Simple compiler syntax checking	Basic compiler capabilities	Compiler extension Basic req. and design consistency	Intermediate-level module Simple req./design	More elaborate req./design Basic dist-processing	Formalized specification, verification. Advanced dist-processing
<b>Peer Reviews</b>	No peer review	Ad-hoc informal walk-through	Well-defined preparation, review, minimal follow-up	Formal review roles and Well-trained people and basic checklist	Root cause analysis, formal follow Using historical data	Extensive review checklist Statistical control
<b>Execution Testing and Tools</b>	No testing	Ad-hoc test and debug	Basic test Test criteria based on checklist	Well-defined test seq. and basic test coverage tool system	More advance test tools, preparation. Dist-monitoring	Highly advanced tools, model-based test

# ODC Extension

- **ODC COQUALMO decomposes defects using ODC categories [Chillarege et al. 1992]**
  - Enables tradeoffs of different detection efficiencies for the removal practices per type of defect
- **The ODC taxonomy provides well-defined criteria for the defect types and has been successfully applied on NASA projects**
  - ODC defects are mapped to operational flight risks
- **With more granular defect definitions, ODC COQUALMO enables tradeoffs of different detection efficiencies for the removal practices per type of defect.**



# ODC Defect Categories

## Requirements

- Correctness
- Completeness
- Consistency
- Ambiguity/Testability

## Design/Code

- Interface
- Timing
- Class/Object/Function
- Method/Logic/Algorithm
- Data Values/Initialization
- Checking

- Data collection forms with detailed definitions and examples available from USC

# Defect Types Matter to User



**Aerobic Mode:**  
2 defects / 7 machines



**Running Mode:**  
2 defects / 2 machines

**Biking Mode:**  
no defects affect operations

# Supports Value-Based Perspective

- **Motivations**
  - V&V techniques have different detection efficiencies for different types of defects
  - Effectiveness may vary over the lifecycle
  - Different defect types have different flight risk impacts
  - Scarce V&V resources to optimize
  - V&V methods may have overlapping capabilities for detecting defects
- **ODC COQUALMO with other tools can help quantify the best combination of V&V techniques, their optimal order and timing**

# Complementary Risk Techniques

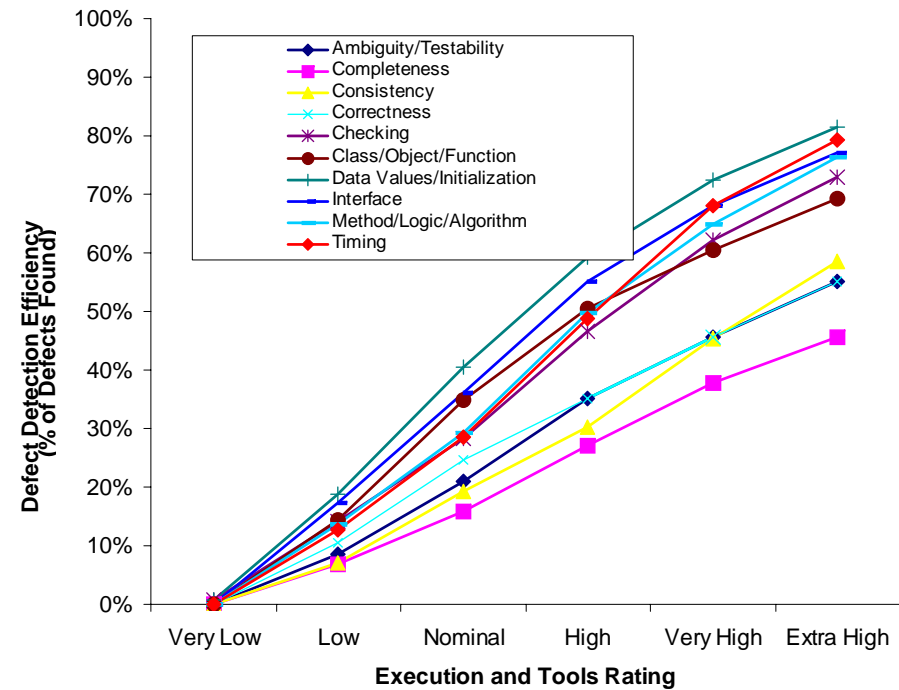
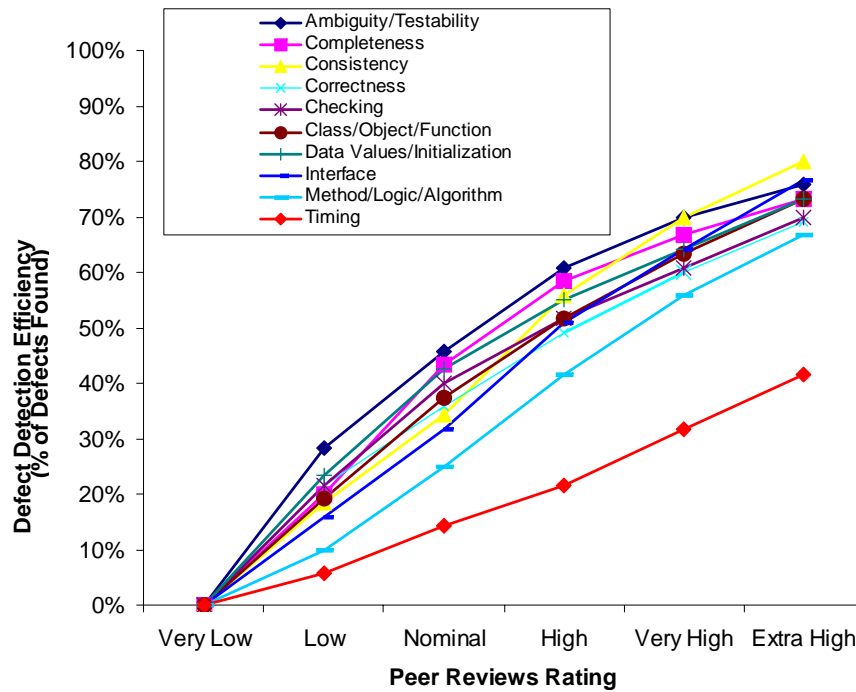
- **V&V processes are designed and optimized with ODC COQUALMO by integrating it with different risk minimization techniques**
- **For NASA we have demonstrated linkage with machine learning [Menzies, Richardson 2005] , strategic optimization and JPL's Defect Detection and Prevention (DDP) risk management method**
  - **Can determine the impact of V&V techniques on specific risks and overall flight risk**

# Defect Calibrations

- The initial ODC defect distribution pattern is per JPL rover studies [Lutz and Mikulski 2003]
- A rigorous two round Delphi survey was completed to quantify ODC defect detection efficiencies, gauging the effect of different defect removal techniques against the ODC categories (example defect detection efficiencies below)

V&V Technique and Defect Type		Very Low	Low	Nominal	High	Very High	Extra High
<b>Automated Analysis</b>							
Requirements	Ambiguity/Testability	0.021	0.057	0.293	0.400	0.514	0.621
	Completeness	0.014	0.079	0.314	0.400	0.500	0.593
	Consistency	0.014	0.043	0.316	0.451	0.581	0.736
	Correctness	0.007	0.086	0.279	0.386	0.486	0.571
Design/Code	Checking	0.014	0.119	0.347	0.504	0.646	0.779
	Class/Object/Function	0.064	0.197	0.330	0.470	0.589	0.707
	Data Values/Initialization	0.086	0.246	0.371	0.533	0.656	0.743
	Interface	0.043	0.159	0.296	0.430	0.544	0.636

# Sample Defect Removal Profiles



- Note differences between methods for specific defect types (e.g. completeness, timing)



# ODC COQUALMO Inputs

Size      Sizing Method  ▼

	<u>SLOC</u>	% Design Modified	% Code Modified	% Integration Required	Assessment and Assimilation (0% - 8%)	Software Understanding (0% - 50%)	Unfamiliarity (0-1)
New	<input type="text" value="124000"/>						
Reused	<input type="text" value="36000"/>	<input type="text" value="0"/>	<input type="text" value="0"/>	<input type="text" value="30"/>	<input type="text" value="4"/>		
Modified	<input type="text" value="45300"/>	<input type="text" value="20"/>	<input type="text" value="35"/>	<input type="text" value="80"/>	<input type="text" value="2"/>	<input type="text" value="25"/>	<input type="text" value=".5"/>

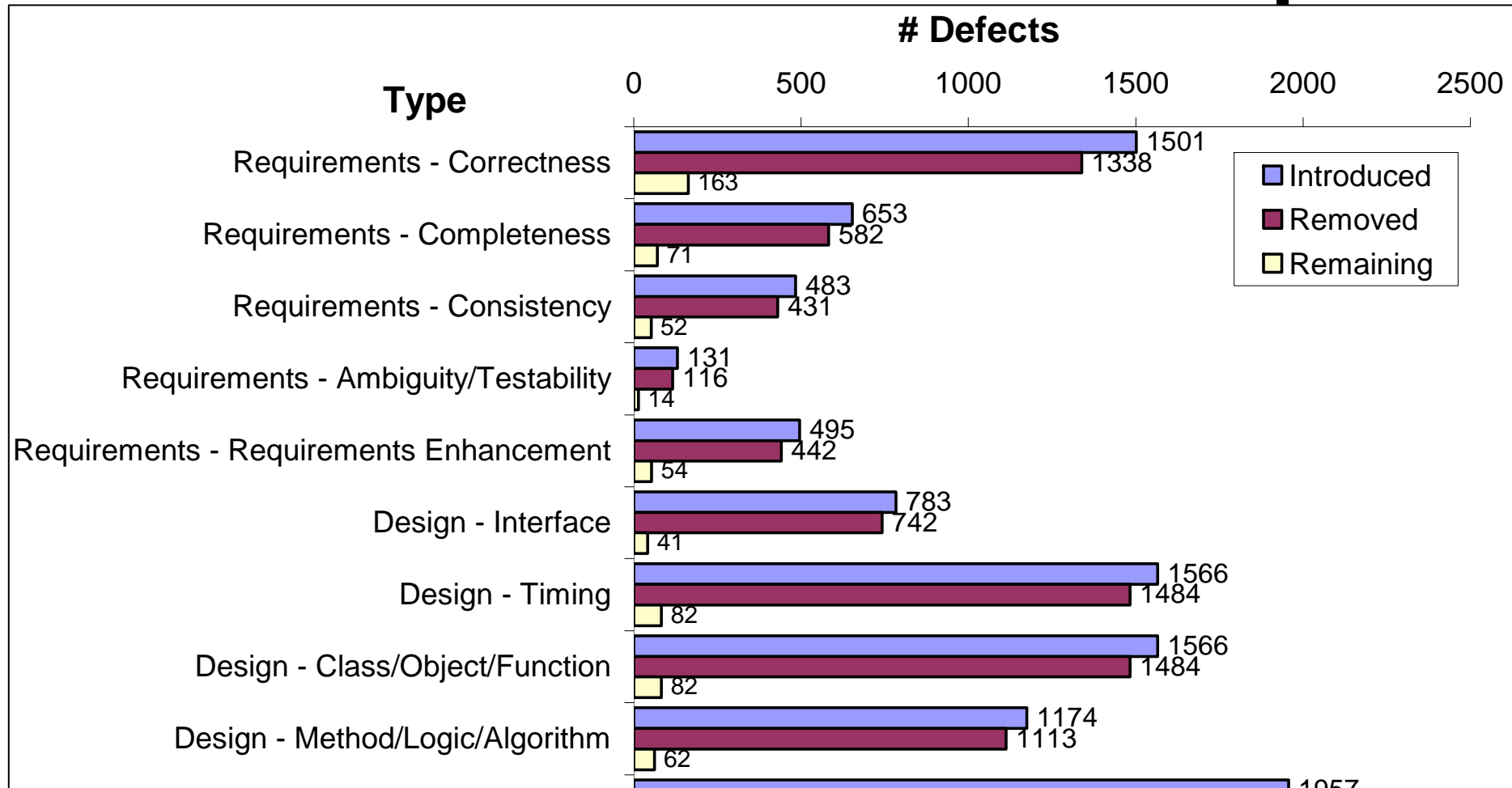
### Scale Drivers

Precedentedness	<input type="text" value="Low"/> ▼	Architecture / Risk Resolution	<input type="text" value="Very High"/> ▼	Process Maturity	<input type="text" value="High"/> ▼
Development Flexibility	<input type="text" value="Low"/> ▼	Team Cohesion	<input type="text" value="High"/> ▼		
<b>Cost Drivers</b>		<b>Personnel</b>		<b>Platform</b>	
<b>Product</b>		Analyst Capability	<input type="text" value="High"/> ▼	Time Constraint	<input type="text" value="Nominal"/> ▼
Required Software Reliability	<input type="text" value="Very High"/> ▼	Programmer Capability	<input type="text" value="High"/> ▼	Storage Constraint	<input type="text" value="Nominal"/> ▼
Data Base Size	<input type="text" value="High"/> ▼	Personnel Continuity	<input type="text" value="High"/> ▼	Platform Volatility	<input type="text" value="Nominal"/> ▼
Product Complexity	<input type="text" value="Very High"/> ▼	Application Experience	<input type="text" value="Nominal"/> ▼	<b>Project</b>	
Developed for Reusability	<input type="text" value="Low"/> ▼	Platform Experience	<input type="text" value="Nominal"/> ▼	Use of Software Tools	<input type="text" value="High"/> ▼
Documentation Match to Lifecycle Needs	<input type="text" value="Nominal"/> ▼	Language and Toolset Experience	<input type="text" value="Nominal"/> ▼	Multisite Development	<input type="text" value="High"/> ▼
				Required Development Schedule	<input type="text" value="Nominal"/> ▼

### Defect Removal Practices

Automated Analysis  ▼      Peer Reviews  ▼      Execution Testing and Tools  ▼

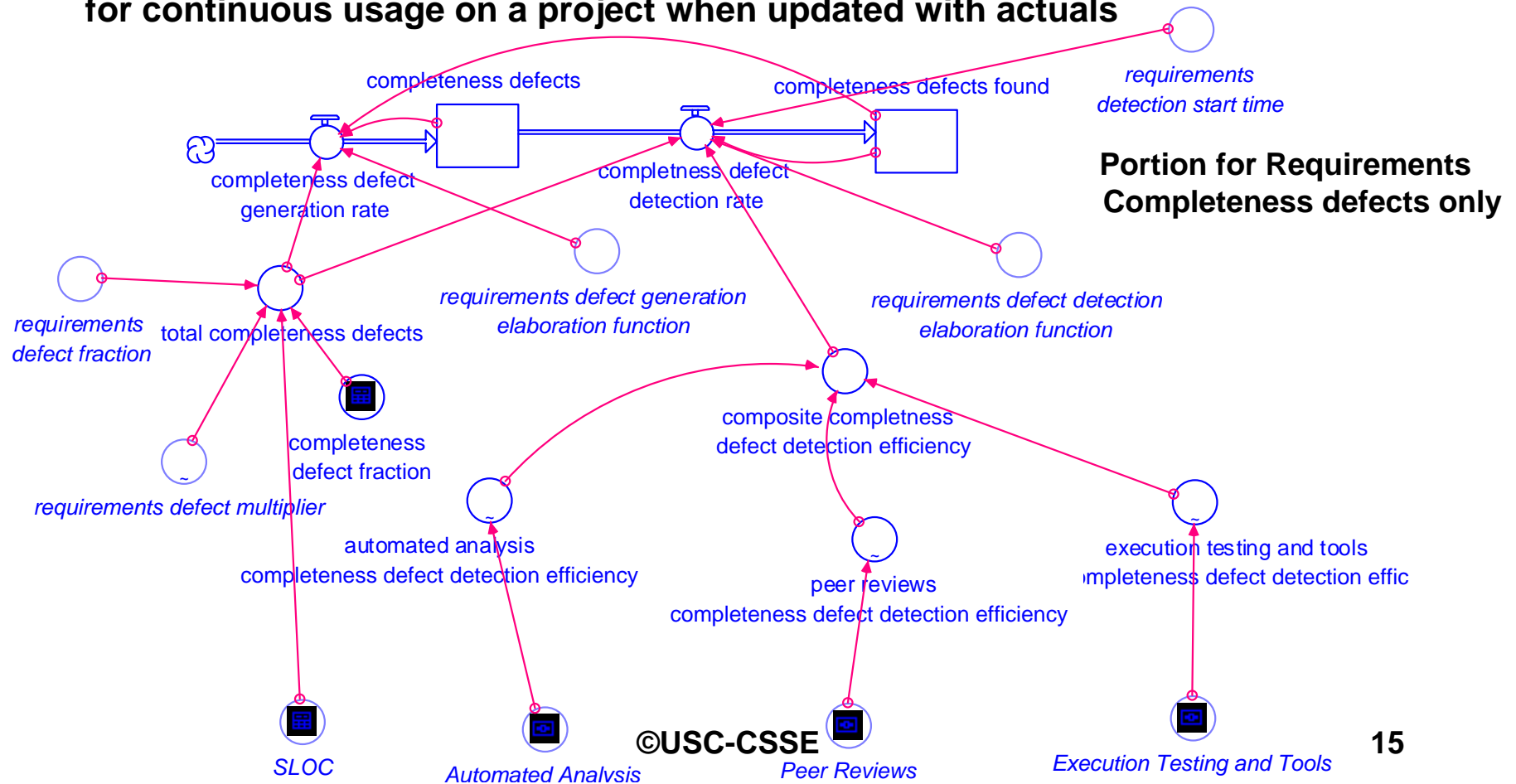
# ODC COQUALMO Defect Outputs



**Effort and schedule outputs not shown**

# Dynamic ODC COQUALMO

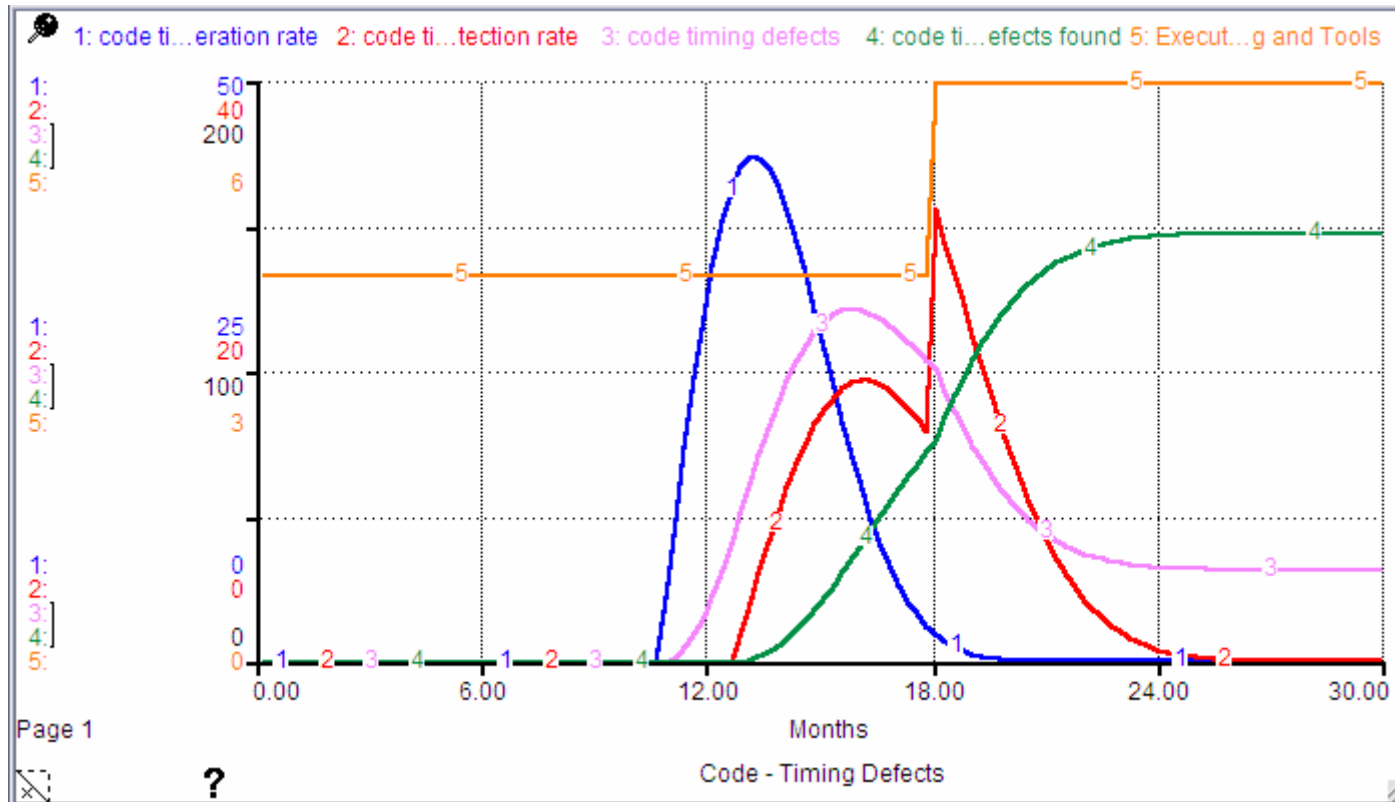
- Continuous simulation version using system dynamics models defect generation and detection rates over time
- Can be used for interactive training to assess decisions midstream or suitable for continuous usage on a project when updated with actuals



# Dynamic ODC COQUALMO

## Sample Output

- Effect on *Code Timing Defects* when applying increased V&V for *Execution Testing and Tools* at 18 months:



# Conclusions and Ongoing Work

- **Software estimation models can play an important role in facilitating the right balance of activities to meet project goals**
  - ODC COQUALMO can be used in different ways to reason about and optimize V&V processes
- **We are recalibrating ODC COQUALMO with a combination of empirical data and Delphi survey results**
  - Includes specialized calibrations with some CSSE industrial affiliates for other domains
- **Continuing to tailor the model for NASA flight projects**
  - Analyzing empirical case studies and data from manned and unmanned NASA flight projects, as well as other USC-CSSE industrial affiliates who serve as contractors on space projects
- **Integrating the model with complementary techniques to support risk management practices, and to compare their performances.**

# References

- [Chulani, Boehm 1999] Chulani S, Boehm B, “Modeling software defect introduction and removal: COQUALMO (CONstructive QUALity MOdel)”, USC-CSE Technical Report 99-510, 1999
- [Chillarege et al. 1992] R. Chillarege, I. Bhandari, J. Chaar, M. Halliday, D. Moebus, B. Ray, and M. Wong, "Orthogonal Defect Classification -- A Concept for In-Process Measurements," *IEEE Transactions on Software Engineering*, vol. 18, no. 11, pp. 943-956, November 1992
- [Lutz, Mikulski 2003] Lutz R, Mikulski I, “Final Report: Adapting ODC for Empirical Analysis of Pre-Launch Anomalies”, version 1.2, JPL Caltech report, December 2003
- [Menzies, Richardson 2005] Menzies, T., and Richardson J.: “XOMO: Understanding Development Options for Autonomy”, In: 20th International Forum on COCOMO and Software cost Modeling, USC, 2005
- [http://csse.usc.edu/tools/odc\\_coqualmo.php](http://csse.usc.edu/tools/odc_coqualmo.php)