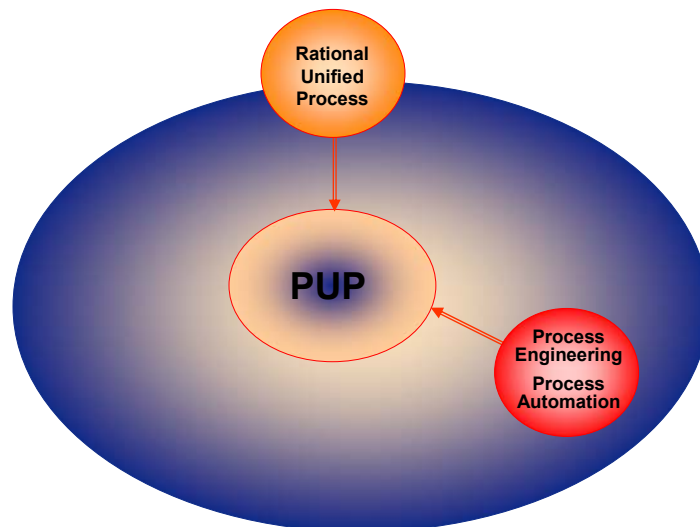


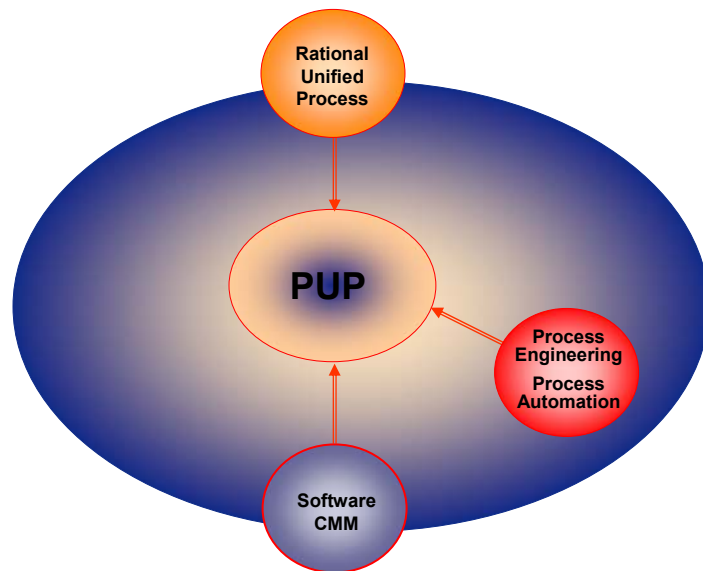
# Making the Role Your Own

## Notes on Process Adoption at Pitney Bowes

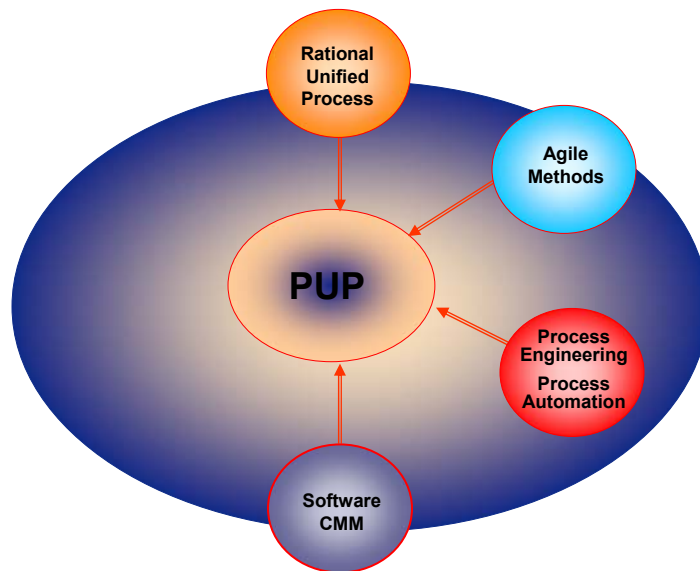
Nanette Brown, Director Architecture & Quality Management



Should Pitney Bowes get CMM Certification?



When and where should we look at streamlining our processes?



How can we improve our architecture review process?



## Improving the Architecture Review Process

- Goals
  - More Participative
  - More Rigorous
  - Legitimize Architecture in Discussions with Lines of Business



## **Improving the Architecture Review Process Revision #1**

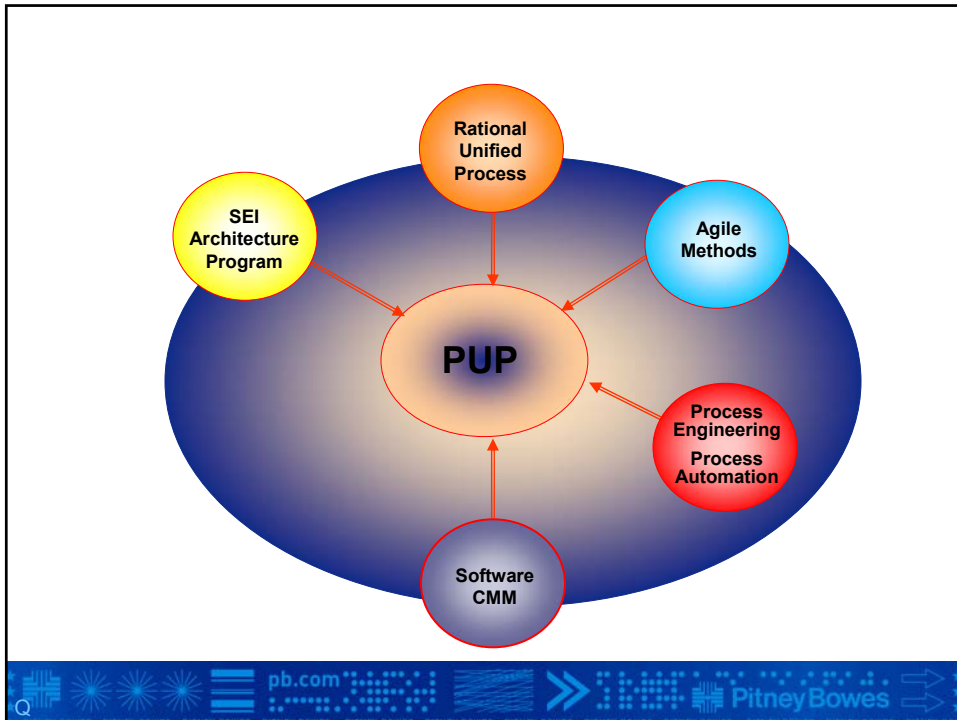
- Pre-Review
  - Architecture Manager coaches Project Team on presentation
- Day 1
  - Project Team presentation
  - Small teams of reviewers create lists of “challenge” questions
- Offline
  - Project Team creates response with coaching from Architecture Manager
- Day 2
  - Project addresses “challenge” questions
  - Small teams of reviewers prepare and read out conclusions
- Post-Review
  - Architecture Manager creates and distributes formal architecture review report



## **Improving the Architecture Review Process**

- Lessons Learned
  - Participation and Better Review Quality Much Better
  - Focus and Rigor Could Still be Improved





## Improving the Architecture Review Process

Adopt Quality Attributes & Quality Attributes scenarios and incorporate them into the Architecture Review Process

## Adopting Quality Attributes

- Process Activities
- Evangelical Activities



## Adopting Quality Attributes

- Process Activities
  - Selected key quality attributes and established formal definitions
  - Defined templates for quality attribute scenarios
  - Incorporated quality attributes into requirements & architecture process specifications



## Customer Perspective Quality Attributes

- Performance
- Robustness
- Availability
- Integrity
- Confidentiality
- Non-repudiation



## Deployment Perspective Quality Attributes

- Scalability



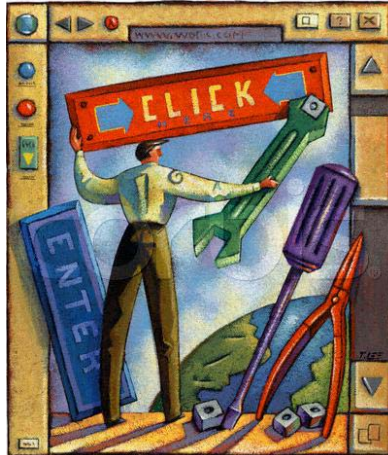
- Interoperability





## Development Perspective Quality Attributes

- Modifiability
- Testability
- Reusability



## Robustness (Fault Tolerance / Reliability)

Robustness is the degree to which a system continues to function properly when confronted with invalid inputs, defects in connected software or hardware components or unexpected operating conditions.<sup>1</sup>

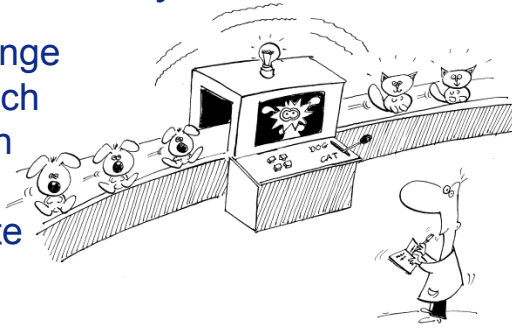


<sup>1</sup> Wiegers, Karl, 2003, *Software Requirements*, 2<sup>nd</sup> Edition. Redmond, Washington: Microsoft Press



# Modifiability

Refers to the cost of change and the ease with which a software system can accommodate changes.<sup>2</sup> This attribute incorporates changes related to extensibility, maintainability and portability.



<sup>2</sup> Carnegie Bosch Institute, "Software Architecture for Product Excellence, Course Materials



## PB Scenario Format - Robustness

Source of Stimulus	<ul style="list-style-type: none"><li>• External Actor</li><li>• Internal System Component</li></ul>
Stimulus Description	<ul style="list-style-type: none"><li>• Unexpected Event(s)</li><li>• Internal Failure(s)</li></ul>
Response	<ul style="list-style-type: none"><li>• Log Error</li><li>• Notify User</li><li>• Terminate Process Normally</li><li>• Terminate process abnormally</li><li>• Switch to backup system component(s)</li><li>• Data Loss / Data Corruption</li></ul>
Environment after Stimulus	<ul style="list-style-type: none"><li>• Normal Operation</li><li>• Degraded operation</li><li>• Unavailable</li></ul>



## PB Scenario Format - Modifiability

Source of Change Request	<ul style="list-style-type: none"> <li>• Stakeholder</li> </ul>
Change Request Description	<ul style="list-style-type: none"> <li>• Request to modify functionality</li> <li>• Request to modify quality attributes</li> <li>• Request to modify technical platform</li> </ul>
Change Request Implementer	<ul style="list-style-type: none"> <li>• Engineering</li> <li>• Service</li> <li>• Deployment</li> </ul>
System Elements to Modify	<ul style="list-style-type: none"> <li>• Components</li> <li>• Database Tables</li> <li>• Configuration Tables</li> <li>• Resource Files</li> <li>• Published Interfaces</li> </ul>
Change Request Response Measure	<ul style="list-style-type: none"> <li>• Cost in Time, Effort, Money</li> <li>• Difficulty: Easy, Medium, Complex</li> <li>• Cost to upgrade Existing Customers</li> <li>• Degradation of other system functions or quality attributes</li> </ul>



## Adopting Quality Attributes

- Evangelical Activities
  - Put together an internal “road show” to introduce the organization to the quality attributes and quality attribute scenarios



## **Improving the Architecture Review Process Revision #2**

- Retained the same 2-day format of Revision #1
- Made Quality Attributes the organizing principle underlying
  - Project team presentation
  - Small team “challenge” questions
  - Architecture Review Report



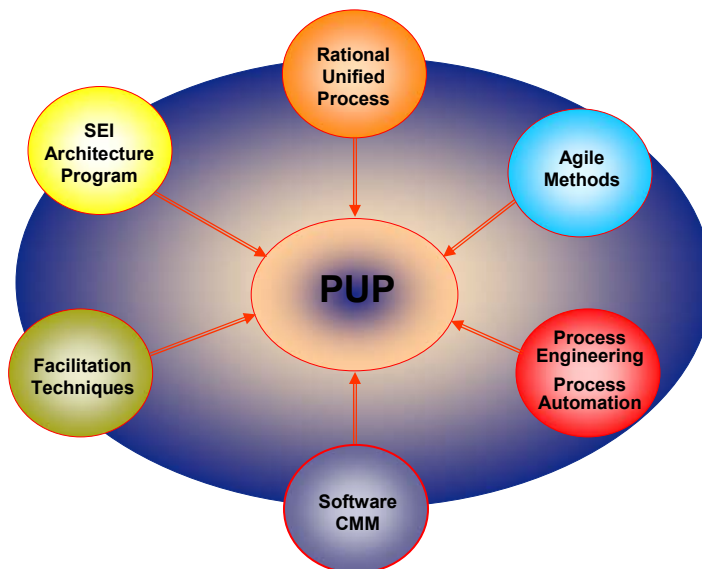
## **Improving the Architecture Review Process**

- Lessons Learned
  - Quality Attributes added much more rigor and focus
  - Quality Attribute scenario generation not as successful



## Reactions to Quality Attribute Scenarios

- They Elicited Immediate Acceptance and Enthusiasm
- They were Harder Than They Looked



## Application of Facilitation Techniques

- Applied Facilitation Techniques to Design of Quality Attribute Workshops
  - Optimize individual work, small team work, “wall” work, large group work
  - Tailor Workshop Design to Specific Workshop Goals
    - Brainstorming workshop to generate scenarios
    - Offline effort to “rigor” scenarios
    - Analysis workshop to review scenarios and generate tradeoff matrix



## Application of Facilitation Techniques

- Used facilitation techniques to hold a feedback workshop on quality attributes and architecture reviews
  - Participants wanted to see Quality Attributes moved up in the lifecycle
    - Used to better understand customer needs and expectations
    - Used to better understand the impact of technical constraints
    - Used to integrate the lifecycle
  - Participants wanted to see ongoing mini architecture reviews earlier in the lifecycle rather than a large review at the end



# Full Lifecycle Application of Quality Attribute Scenarios

## Analyze

- Use scenarios to analyze Quality Attributes and Quality Attribute tradeoffs
- Identify technology requirements

## Test / Deploy

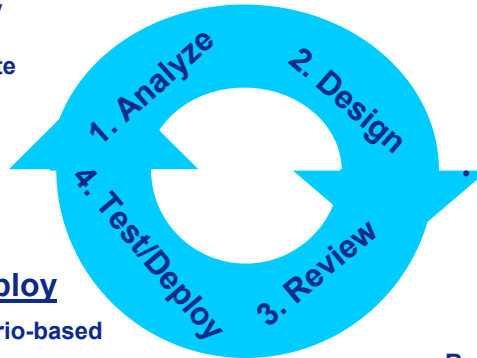
- Execute scenario-based test cases

## Design

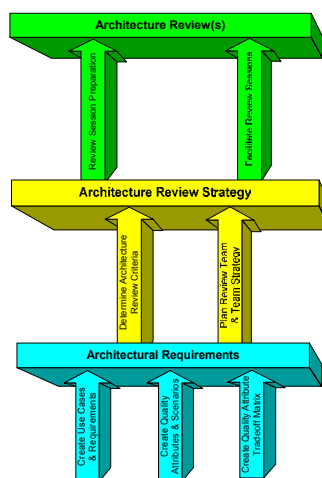
- Choose architectural mechanisms to address scenarios & technology requirements
- Generate test cases from Quality Attribute scenarios

## Review

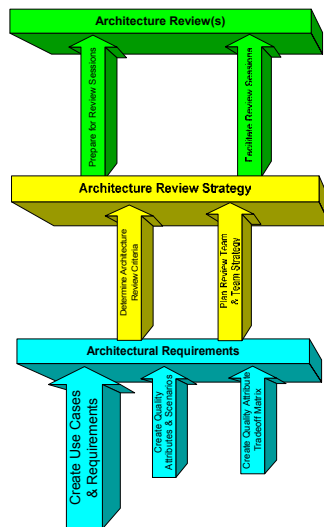
- Review proposed architectural mechanisms using scenarios



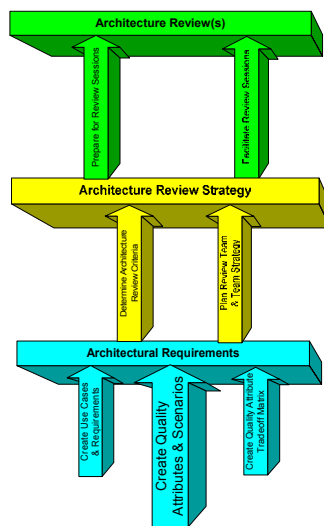
# Revised Architecture Review Process Model



# Revised Architecture Review Process Model

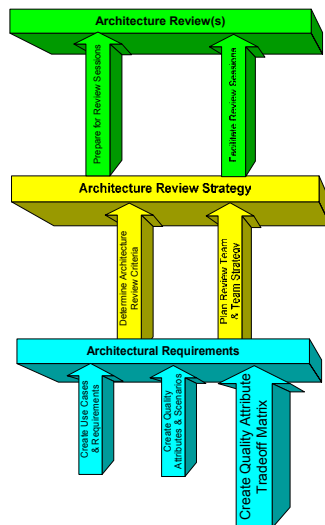


# Revised Architecture Review Process Model

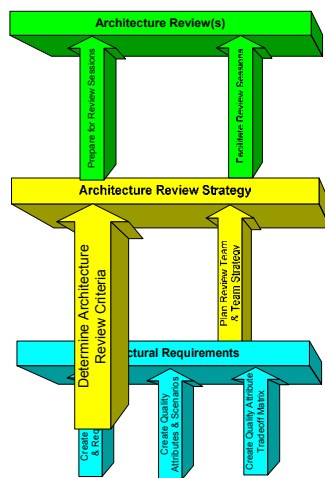




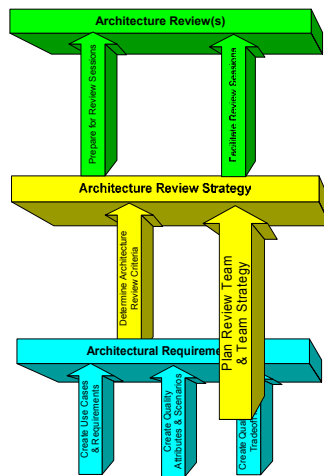
## Revised Architecture Review Process Model



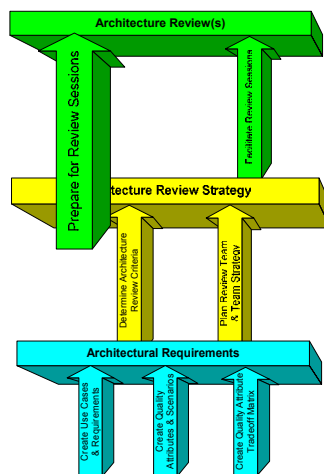
## Revised Architecture Review Process Model



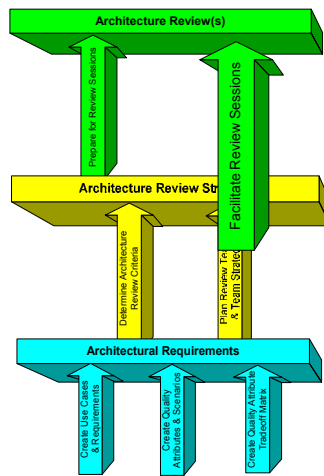
# Revised Architecture Review Process Model



# Revised Architecture Review Process Model



## Revised Architecture Review Process Model



**Where do we  
go from here?**



## Where do we go from here?

Keep improving our skills at holding  
scenario generation workshops



## Where do we go from here?

- Use Quality Attributes to Integrate the Lifecycle
  - Product Requirements
  - Technical Tactics
  - Test Strategy
- Use Quality Attributes to Integrate the Stakeholders
  - Customers
  - Marketing
  - Architects
  - Developers
  - Testers



## Where do we go from here?

Learn what YOU are doing with Quality  
Attribute Scenarios

