The Lambda Protocol for Synthesizing Trustworthy Requirements

Larry Bernstein
Stevens Institute of Technology
Castle Point, Hoboken, NJ 07030
USA
QSE Lambda Protocol

- Prospectus
- Measurable Operational Value
- Prototyping or Modeling
- sQFD
- Schedule, Staffing, Quality Estimates
- ICED-T
- Trade-off Analysis
Trustworthy Software is:

- Safe: Does no harm
- Reliable: No crash or hang.
- Secure: No Hacking Possible
What is a Requirement?

• A property that must be exhibited by a system to solve some problem.

• Requirements may be
  – Functional providing product capabilities
  – Non-Functional constraining the implementation
System Performance Resulting from Robust Requirements vs. Discrete Specifications

- **Volume**
- **Dynamic Range**

- **Discrete Specifications**
- **Ideal**

- **Agile Requirements**
# Top Ten Software Risk Items

<table>
<thead>
<tr>
<th>Category</th>
<th>Risk Item</th>
</tr>
</thead>
<tbody>
<tr>
<td>People</td>
<td>1. Personnel Shortfalls</td>
</tr>
<tr>
<td></td>
<td>2. Unrealistic Schedules and Budgets</td>
</tr>
<tr>
<td>Requirements</td>
<td>3. Developing the Wrong Software Functions</td>
</tr>
<tr>
<td></td>
<td>4. Developing the Wrong User Interface</td>
</tr>
<tr>
<td></td>
<td>5. Gold Plating</td>
</tr>
<tr>
<td></td>
<td>6. Continuing Stream of Requirements Changes</td>
</tr>
<tr>
<td>Externalities</td>
<td>7. Shortfalls in Externally-Furnished Component</td>
</tr>
<tr>
<td></td>
<td>8. Shortfalls in Externally-Performed Tasks</td>
</tr>
<tr>
<td>Technology</td>
<td>9. Real-Time Performance Shortfalls</td>
</tr>
<tr>
<td></td>
<td>10. Straining Computer Science Capabilities</td>
</tr>
</tbody>
</table>
QSE Characteristics

- Solving the right problem the right way
- Tested against requirements.
- Certified against problem
- Bounded execution domain
- Industrial Strength Requirements for Software Intensive Systems-of-Systems
Universal Software Engineering Equation

Reliability \( (t) = e^{-k \lambda t} \)

when the error rate is constant and where \( k \) is a normalizing constant for your software shop and

\( \lambda = \text{Complexity/ [effectiveness x staffing]} \)
Boundary Conditions

\[ \text{Reliability } (0) = 1 \]
\[ \text{Reliability } (T) = e^{-k \lambda T} \]
\[ \text{Reliability } (\infty) = 0 \]
Software Testing Footprint

Tests Completed vs. Time

- Tests run successfully
- Rejection point
- Poor Module Quality
- Planned
QSE Lambda Protocol

- Prospectus
- Measurable Operational Value
- Prototyping or Modeling
- sQFD
- Schedule, Staffing, Quality Estimates
- ICED-T
- Trade-off Analysis
Prospectus

• Description of the problem domain
• Scope of solution
• Specific project goals
• Constraints on the behavior or structure of the software:
  – For example, Trustworthiness
Case Study: SchedulerPro

Prospectus

User friendly, efficient interface for students to create and modify class schedules.

Features:

– Visual schedule creation and editing
– Schedule suggestion
– Schedule comparison view
– Monitor closed-out sections
SchedulerPro Prototype Screen
SchedulerPro Prototype Screen
SchedulerPro Notification Emails

From: schedulerpro@stevens.edu
Sent: Tuesday, April 19, 2005 12:15 PM
To: gdeangel@stevens.edu
Subject: Notification From Scheduler Pro

Scheduler PRO

This is an automated notification from Scheduler Pro. The following class is available for registration:

Notification Class Details

Title: Microprocessor Sys. Lab
Section: CS391C
Call Number: 10239
Instructor: STAFF
Scheduled Meetings: Thursday
Time: 11:00 AM-1:50 PM

Section: CS391D
Call Number: 10240
Instructor: STAFF
Scheduled Meetings: Thursday
Time: 2:00 PM-4:50 PM

Section: CS391A
Call Number: 10237
Instructor: STAFF
Scheduled Meetings: Tuesday
Time: 2:00 PM-4:50 PM

Section: CS391B
Call Number: 10238
Instructor: STAFF
Scheduled Meetings: Wednesday
Time: 10:00 AM-12:50 PM

This address is not monitored so please do not respond to this message. To discontinue this notification or to manage your schedule please visit the Scheduler Pro Homepage.
Measurable Operational Value
SchedulerPro MOV

Reduce student withdrawals by 20%
SchedulerPro Functional Goals

Schedule Classes and Personal Time

→ Searching
→ Course Placement
→ Course Detail Viewing
→ Course Removal
→ Scheduling Personal Blocks
→ Notification (optional)
→ Course Suggestions (optional)
Student Directed Features

• Search available classes by:
  ✓ Same professor
  ✓ Similar time
  ✓ Same or equivalent class but different sections

• Register and track registrations

• Color classes and arbitrary time-blocks by user choice
SchedulerPro Nonfunctional Requirements

• Integrate with “Web for Students’ and existing authentication systems and avoid incompatibilities
• Allow schedules to be saved/accessed from a server or local file
• Provide a scaled time-accurate visual representation of the schedule
More Non-functional requirements

- Make schedules available even if the application is down, provided an internet connection is available
- Perform some functions without a live connection to the ‘Web for Students’ registrar web site
- Make compatible with all popular browsers
- Display section states and print schedules without loss of detail
## sQFD

<table>
<thead>
<tr>
<th>Functions/Features</th>
<th>Class Filters</th>
<th>Allocate non-class time</th>
<th>Long term information availability</th>
<th>Authenticate</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Makes scheduling classes easier</td>
<td>8</td>
<td>3</td>
<td>6</td>
<td>2</td>
<td>19</td>
</tr>
<tr>
<td>Makes scheduling a semester easier</td>
<td>7</td>
<td>9</td>
<td>8</td>
<td>2</td>
<td>26</td>
</tr>
<tr>
<td>Find schedules in one place</td>
<td>1</td>
<td>1</td>
<td>5</td>
<td>7</td>
<td>14</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>16</strong></td>
<td><strong>13</strong></td>
<td><strong>19</strong></td>
<td><strong>11</strong></td>
<td><strong>59</strong></td>
</tr>
</tbody>
</table>
SchedulerPro Product Reliability

- Two hours of unavailability allows for daily backups, service, and reboots of the system
- Connections to server are minimized, reducing overall activity on the server
SchedulerPro
Estimate of Reliability

\[ R(t) = 1 - F(t) \quad \text{F(t) = P(T \leq t)} \]

- During load testing, we discovered the test server can support 1500 user queries a minute.
- \[ P(\text{failures/query}) = \frac{55}{1500} = 0.036 \]
- Thus, \[ F(t) = 3.6\% \], which means the software is **96.4\% reliable**
SchedulerPro
Reliability Estimate

\[
\frac{1}{\lambda} = \text{MTTF} = \varepsilon E / kC
\]

\[
k = \text{scaling constant} = 1
\]

\[
C = \text{complexity} = 2.78
\]

\[
E = \text{the development effort} = 36.4
\]

\[
\varepsilon = \text{the expansion factor} = 1.5
\]

\[
\lambda = 0.05
\]

t is the continuous execution time for the software

\[
R(t) = 95.12\%
\]
Complexity Chart - Client

- Project Type: online transaction
- Problem Domain: 2
- Architecture Complexity: 3
- Logic Design – Data: 2
- Logic Design – Code: 3
  - Total Score: 10
  - Complexity = (10/18) * 5 = 2.78
Complexity Chart - Server

- Project Type: online transaction
- Problem Domain: 1
- Architecture Complexity: 2
- Logic Design – Data: 2
- Logic Design – Code: 2
  - Total Score: 7
  - Complexity = (7/18) * 5 = 1.94
Complexity Chart - Overall

- Project Type: client/server
- Problem Domain: 2
- Architecture Complexity: 3
- Logic Design – Data: 2
- Logic Design – Code: 3
  - Total Score: 10
  - Complexity = (10/18) * 5 = 2.78
# Jan. Function Point Est.

<table>
<thead>
<tr>
<th>Function</th>
<th>Low (L)</th>
<th>Average (A)</th>
<th>High (H)</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outputs</td>
<td>1</td>
<td>3</td>
<td>0</td>
<td>19</td>
</tr>
<tr>
<td>Inquiries</td>
<td>8</td>
<td>4</td>
<td>1</td>
<td>49</td>
</tr>
<tr>
<td>Inputs</td>
<td>5</td>
<td>7</td>
<td>1</td>
<td>41</td>
</tr>
<tr>
<td>Internal Files</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>24</td>
</tr>
<tr>
<td>External Interfaces</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td><strong>Total UFP</strong></td>
<td></td>
<td></td>
<td></td>
<td>143</td>
</tr>
<tr>
<td>Adjustment Factor</td>
<td></td>
<td></td>
<td></td>
<td>0.99</td>
</tr>
<tr>
<td><strong>Total AFP</strong></td>
<td></td>
<td></td>
<td></td>
<td>141</td>
</tr>
</tbody>
</table>
# April Function Points Est.

<table>
<thead>
<tr>
<th>Function</th>
<th>Low</th>
<th>Average</th>
<th>High</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outputs</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>Inquiries</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>Inputs</td>
<td>2</td>
<td>3</td>
<td>0</td>
<td>18</td>
</tr>
<tr>
<td>Internal Files</td>
<td>3</td>
<td>1</td>
<td>0</td>
<td>31</td>
</tr>
<tr>
<td>External Interfaces</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>12</td>
</tr>
<tr>
<td><strong>Total UFP</strong></td>
<td></td>
<td></td>
<td></td>
<td>79</td>
</tr>
<tr>
<td><strong>AFP</strong></td>
<td></td>
<td></td>
<td></td>
<td>82</td>
</tr>
</tbody>
</table>
## History of Function Points

<table>
<thead>
<tr>
<th>Date</th>
<th>AFP</th>
<th>Project Length*</th>
<th>Projected Finish*</th>
</tr>
</thead>
<tbody>
<tr>
<td>January 27</td>
<td>141</td>
<td>19.7 staff months</td>
<td>August 2006</td>
</tr>
<tr>
<td>February 24</td>
<td>104</td>
<td>14.4 staff months</td>
<td>March 2006</td>
</tr>
<tr>
<td>April 17</td>
<td>82</td>
<td>8.5 staff months</td>
<td>May 2006</td>
</tr>
</tbody>
</table>

*Using COCOMO Model*
## ICED-T

<table>
<thead>
<tr>
<th>Scheduling by:</th>
<th>Intuitive</th>
<th>Consistent</th>
<th>Efficient</th>
<th>Durable</th>
<th>Thoughtful</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paper</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>Excel</td>
<td>3</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>School Scheduler</td>
<td>3</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>SchedulerPro</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>
Missing: An Installation Plan

Installation

1. Third Party Software Required

Scheduler Pro requires the following products to be already installed on the target machine. Please consult the documentation of each product for installation instructions specific to each.

- Windows 2000, XP, or 2003 Server
- Microsoft IIS, version 5.0 or higher
- Microsoft .NET, version 1.1
- Microsoft SQL Server 2000
- Message Queuing Service (Windows component)
- ASP.NET State Service
Software Requirements Process

- Requirements Elicitation
- Requirements Analysis
- Use Cases
- Requirements Specification
- **Prototype/Modeling**
- Requirements Management
Creeping Featurism

• Endemic to the Software Industry
  – Occurs on more than 70% of all applications of over 1000 function points

• From a 60 project sample
  – Average creep was 35%
  – Maximum observed was 200%
  – Creeping requirements change about 1% per month
    • For a 3 year project, 1/3 of the delivered requirements would have been added after requirements were initially defined

• Rate of Requirements change is higher than for other forms of engineering (electrical, mechanical, civil)
Root Causes of Creeping Requirements

• Uncertainty in resolving true user needs
• For multi-year projects, changes in normal business environment
• Failure to adopt methodologies that limit the risk associated with creeping requirements
• Primitive fundamental technologies for exploring and modeling requirements
• Failure to use technology to measure the impact of creeping requirements
• Engineering trade-off analysis is impossible
Requirements Management

• Establish and maintain a business case to support funding
• Strategic linkages to business and technology organizations – AVOID SHELFWARE
• Continuous customer agreement on requirements
• Requirements agreement used as a basis for estimating, planning, implementing and tracking
• FORMAL COMMITMENT PROCESS
Requirements Engineering Process

- Process Models
- Process Actors and Stakeholders
- Process Support and Management
- Process Quality and Improvements
- Relationship to the Business Decision
Real-time Requirements

• Computer uses only past and present data
• Data is sampled at a constant rate, the pulse repetition rate of the radar,
• The calculations are completed in time to adjust the radar for the next sample
• The equations are stable
Requirements Process

- Elicitation
  - Request Analysis
  - Definition
    - Purposeful
    - Understand value
- Analysis
  - Interrelationships
  - Prioritization
  - Risk & Cost Assessment
- Specification
  - Modeling
- Validation
  - Agreement
- Change Management
Requirements Analysis

- Requirements Classification
  - Product/Process
  - Priority/Risk
  - Scope/Allocation
  - Volatility/Stability

- Conceptual Modeling
  - Understanding & Communication
  - Functional Architecture

- Requirements Negotiation
  - Trade Offs
  - Consensus with Stakeholder
Example

• Develop Use Cases
  – Focus on Goals
  – Identify Actors
  – Identify Main Tasks

• Use Case Concept
  – Complete, orthogonal, externally visible functionality
  – Initiated by an actor
  – Identifiable value to the actor
Software Requirements Spec.

- Concept of Operations
  - System Characteristics
  - User Operational Needs
  - Domain Perspective
  - Constraints
  - Trade-Off Analysis

- Software Requirements Specification
  - Basis for Agreement
  - Reduce Development
  - Provide Basis for Estimation
  - Baseline for Validation & Verification
  - Basis for Enhancement
Requirements Specification
Spec

1. Project Title, Revision Number and Author
2. Scope and Purpose of the system
3. Measurable Operational Value
4. Description
5. Feature List including ICED T and Simplified QFD analysis
6. Interfaces
7. Constraints
8. Change Log and Expected Changes
9. Responses to the unexpected
10. Measurements
11. Glossary
12. References
Requirements Validation

• Requirements Reviews
  – Formal
  – Customer Representative

• Prototyping

• Model Validation
  – Scenario Reviews with Customers
  – Model Consistency

• Acceptance Tests
  – Verifiable Requirements

As of 9/9/04
Use Cases Drive Development

Use Cases

- Architecture and Design
- Test Case Design
<table>
<thead>
<tr>
<th>Feature</th>
<th>Use Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>The customer can order on the web.</td>
<td>UC 1</td>
</tr>
<tr>
<td>The customer builds the order by selecting items from the on-line catalog and specifying a quantity.</td>
<td>UC 1</td>
</tr>
<tr>
<td>Only customers that have an account can create an order.</td>
<td>UC 1</td>
</tr>
<tr>
<td>At any time during the process of creating an order, the customer can determine the current price of the order.</td>
<td>UC 1</td>
</tr>
<tr>
<td>The customer signifies that the order is complete by submitting the order. When an order is submitted, it is assigned an order number.</td>
<td>UC 1</td>
</tr>
<tr>
<td>Customers with the priority privilege may designate an order as priority.</td>
<td>UC 1a</td>
</tr>
<tr>
<td>The customer can view the status of an order at any time by logging on to web site and requesting status on all open orders.</td>
<td>UC 2</td>
</tr>
<tr>
<td>Once an order is submitted, it is checked to see if it is pre-paid or whether the customer has an account in good standing. If these conditions are not met, the order is held until the conditions are met or the order is cancelled.</td>
<td>UC 1</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
## Use Case Documentation

<table>
<thead>
<tr>
<th>Use Case 1</th>
<th>Create Order &amp; Submit</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Brief Description</strong></td>
<td>A customer wishes to order. Provided that the customer has a non-delinquent account or has pre-paid, the product is removed from inventory and delivered to the customer.</td>
</tr>
<tr>
<td><strong>Actors</strong></td>
<td>Customer, Inventory, Shipping Clerk, Account System</td>
</tr>
<tr>
<td><strong>Trigger</strong></td>
<td>Customer visits web site &amp; creates an order.</td>
</tr>
<tr>
<td><strong>Preconditions</strong></td>
<td>Customer has established and account. Customer email address is known. Customers are pre-designated to enter priority orders.</td>
</tr>
<tr>
<td><strong>Main flow</strong></td>
<td>Customer visits web site, signs on and is validated. Customer selects items from the online catalog and builds an order. Customer is appraised of current cost of order. Customer may denote that the order is a priority. Customer submits order when done. A customer order number is assigned and the customer’s credit and account status are checked. If credit is OK or the account shows pre-payment, then the order is sent to the inventory system. .....</td>
</tr>
<tr>
<td><strong>Alternative flows</strong></td>
<td>Priority Order Account is delinquent. <strong>Action taken? Cancelled?</strong> Changes to or cancellation of the order? Order cannot be fulfilled?</td>
</tr>
<tr>
<td><strong>Postconditions</strong></td>
<td>Order has been created and is either been cancelled or been fulfilled.</td>
</tr>
</tbody>
</table>
Package Diagram

- Groups related use cases
- Forms basis for a functional partitioning from the users point of view.
- Shorthand for tracking within the project
**Activity Chart**

- **Order Entry**
  - Enter Order
    - [submitted]
  - [aborted]

- **Finance**
  - Check Credit
    - [approved]
    - [denied]
  - Allocate Inventory

- **Inventory Management**
  - Prepare Delivery

- **Shipping**
  - Receive Payment

**Create Order & Submit**

<<trace>>
Activity Diagram

Allocate
Inventory

Assign Held
Orders First

Inventory
Arrived

Request
Open Items

Items Not Available

Items Available

Update Order Item

Update Order Item

Orders First

For each order item

Hold Order

Priority Order?

Order Assigned

Post to Delivery

For each priority order

For each order item

Held Orders Done?

For all unfulfilled orders

New Items Assigned?

For all fulfilled orders

<<trace>>
Mapping Requirements to a Framework

- **ICED T**
  - Intuition
  - Consistent
  - Efficient
  - Durable
  - Thoughtful

- **UML Framework**
  - Use Cases
  - Structure
  - Business Rules
Case History: Cardiac Data Analysis

Propectus: Create a graphical interface that displays a time series graph with selected points of inflection, and allows for user modification of points.
MOV

Background: Drs. determine points manually taking 20-30 minutes, or with tools that take 2 – 10 minutes.

MOV: Our software allows points to be chosen, on average, 4 times faster than previous available tools with 80% accuracy.
## Function Points

**Siemens Unadjusted Function Point Analysis**

Updated 2/15/06

<table>
<thead>
<tr>
<th>Use Cases</th>
<th>Transactions</th>
<th>Type</th>
<th>Complexity</th>
<th>UFP</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Tool 1:</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Input data file</td>
<td>1</td>
<td>I</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>User point modification</td>
<td>1</td>
<td>I</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>Load User Point Changes</td>
<td>1</td>
<td>I</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>Screenshot</td>
<td>1</td>
<td>O</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Save User Point Changes</td>
<td>1</td>
<td>O</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td><strong>Tool 2:</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Curve Fitting Algorithm</td>
<td>1</td>
<td>I</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>Find/Send points to tool 1</td>
<td>1</td>
<td>O</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td><strong>Point Selecting Algorithm</strong></td>
<td>7</td>
<td>N</td>
<td>15</td>
<td>105</td>
</tr>
<tr>
<td><strong>Tool 3:</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data from tool 1</td>
<td>1</td>
<td>I</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Rotating image (user control)</td>
<td>2</td>
<td>I</td>
<td>6</td>
<td>12</td>
</tr>
<tr>
<td>Snapshot</td>
<td>1</td>
<td>O</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Coloration of Image logic</td>
<td>1</td>
<td>N</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>3-D imaging/rotation logic</td>
<td>1</td>
<td>N</td>
<td>15</td>
<td>15</td>
</tr>
</tbody>
</table>

**Total Unadjusted Function Points** 189
Simplifications

• Narrowing of the requirements to only consider data from ‘healthy hearts.’
• Open source code: *NTGraph*.
• Before simplifications
  Unadjusted Function Points were 356
  now they are 189.
Function Points to LOC

- This conversion chart is shown below

<table>
<thead>
<tr>
<th>Language</th>
<th>SLOC per Function Point</th>
</tr>
</thead>
<tbody>
<tr>
<td>C++ Default</td>
<td>53</td>
</tr>
<tr>
<td>COBOL Default</td>
<td>107</td>
</tr>
<tr>
<td>Delphi 5</td>
<td>18</td>
</tr>
<tr>
<td>HTML 4</td>
<td>14</td>
</tr>
<tr>
<td>Java 2 Default</td>
<td>46</td>
</tr>
<tr>
<td>Visual Basic 6</td>
<td>24</td>
</tr>
<tr>
<td>SQL Default</td>
<td>13</td>
</tr>
</tbody>
</table>

- Thus for our system using the conversion factor of 53 LOC/FP since we will be programming in C++ we can find the estimated LOC for our system through the following formula:

\[
LOC = 53 \times UFP
\]

- Thus we can solve this equation to find the LOC estimated for our system.

\[
LOC = 53 \times 189 = 10,017 \text{ LOC}
\]

COCOMO

Effort/Staff Hours = A*(KNCSLOC)**B
Where KNCSLOC ≡ thousands of new and changed lines of code,
   A ≡ small project productivity,
   B≡ complexity factor

We use:
• Semidetached: A=3.0 B=1.12
• KNCSLOC = 10

Effort = 3.0*(10)**1.12 = \textbf{39.623} \approx 40 \text{ staff months}
ICED-T

<table>
<thead>
<tr>
<th>Metric</th>
<th>Build</th>
<th>Requirements</th>
<th>Architecture</th>
<th>Prototype</th>
<th>Development</th>
<th>Final</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intuitive</td>
<td></td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>Consistent</td>
<td></td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>Efficient</td>
<td></td>
<td>3</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Durable</td>
<td></td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Thoughtful</td>
<td></td>
<td>4</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>4</td>
</tr>
</tbody>
</table>
Reliability Requirement
Heisenbugs

Latent faults causing gradual deterioration a software process with respect to the use of some resource resulting in a failure.
Case Study: Pluto Express

- Duplicated computers for reliability.
- One computer runs at a time to minimize power drain.
- Hardware detects computer failure and switches to backup.
- Assume Prob. of unsuccessful switchover $= 10^{-8}$
Case Study: Pluto Express

Diagram:
- Boot
- Robust Operation
- Vulnerable Operation
- Failure
Case Study: Pluto Express

Let the rate of going from Robust State to Vulnerable State be: $10^{-3}$
Let the rate of going from the Vulnerable State to Failure be: $10^{-4}$
Then using Rejuvenation with a 6 week period increases system reliability by a factor of 10
Case Study: Pluto Express

If the failures double and the Rejuvenation interval is halved, system reliability with Rejuvenation is about 100 times more reliable than systems without Rejuvenation.
Parnas reliability checklist

Response to all failures in communication, secondary storage, memory, or any hardware that may interrupt a transaction:

- The SQL Server DBMS will not commit incomplete transactions. User will be notified of the error, and will have to redo the transaction.

• Operator errors:

  - Important operations are confirmed before they are completed to avoid large accidental errors.
Conditions That Cause Unreliability

- Poor Algorithms
- Missing Deadlines
- Roundoff Error Build Up
- Memory Leaks
- Broken Pointers
SEI Capability Model

**Key Process Areas**

- Process change management
- Technology change management
- Defect prevention

Software quality management
- Quantitative process management

People

Software Trustworthiness depends on people:

I propose that customers insist that software products identify a Software Architect and Software Project Manager in their contracts.
Software Architect:

- Affirms that the software product solves the customer’s problem
- Affirms that the software product is suitably reliable, easy-to-use, extendible, not harmful and robust. That it is trustworthy.
- Affirms that the requirements are valid.
Software Project Manager:

- Affirms that the software was successfully tested against the requirements.
- Affirms and identifies the good software engineering processes were used in the software development and integration.
- Affirms that the project is within budget, on-time and performs satisfactorily.
People

Software Trustworthiness depends on people:

I propose that customers insist that software products identify a Software Architect and Software Project Manager in their contracts.
Software Architect:

• Affirms that the software product solves the customer’s problem
• Affirms that the software product is suitably reliable, easy-to-use, extendible, not harmful and robust. That it is trustworthy.
• Affirms that the requirements are valid.
Software Project Manager:

- Affirms that the software was successfully tested against the requirements.
- Affirms and identifies the good software engineering processes were used in the software development and integration.
- Affirms that the project is within budget, on-time and performs satisfactorily.
Systems Engineering

“An interdisciplinary approach and means to enable the realization of successful systems.”

– INCOSE (The International Council on Systems Engineering)

System:

“A group of interacting, interrelated, or interdependent elements that together form a complex whole.”

– NGE Project (Next Generation Education Project)
QSE Lambda Protocol

- Prospectus
- Measurable Operational Value
- Prototyping or Modeling
- sQFD
- Schedule, Staffing, Quality Estimates
- ICED-T
- Trade-off Analysis
Requirements Engineer

Customer Needs

Understand Domain Knowledge

Customer Domain

Lambda Protocol for me!

Feature Packages

Communicate Development Strategy

Solution Domain