Software Quality Assurance
Software Reliability Engineering

Objectives

- Define the relationship between reliability and availability
- Explain how one can quantify the relationship between failures
- List the tradeoffs in major software quality characteristics
- Explain the Software Reliability Engineering process
- Explain how one determines the appropriate reliability level
- Describe how one performs testing using an operational profile.

Software Reliability

- Reliability is usually defined in terms of a statistical measure for the operation of a software system without a failure occurring
- Software reliability is a measure for the probability of a software failure occurring during the time interval
- Two terms related to software reliability
  - **Fault**: a defect in the software, e.g. a bug in the code which may cause a failure
  - **Failure**: a derivation of the programs observed behavior from the required behavior

Software Availability

- Measure of how likely the system is available for use. Takes repair/restart time into account
- Availability of 0.998 means software is available for 998 out of 1000 time units
- Relevant for continuously running systems
  - telephone switching systems
  - Servers

Quantifying the relationship between failures

- Time of failure ⟵ when
- Time interval between failures — How far apart
- Cumulative failures experienced up to a given time — How many
- Failures experienced in a given time interval ⇒ How many per unit of time

Time Based Failure Specification

Failure Number	Failure Time (sec)	Failure Interval (sec)
1	10	10
2	19	9
3	32	13
4	43	11
5	58	15
6	70	12
7	88	18
8	103	15
9	125	22
10	150	25
11	169	19
12	199	30
13	231	32
14	256	25
15	296	40

Failure based Specification

Time (sec)	Cumulative Failures	Failures in Interval
30	2	2
60	5	3
90	7	2
120	8	1
150	10	2
180	11	1
210	12	1
240	13	1
270	14	1

Important Things to Consider

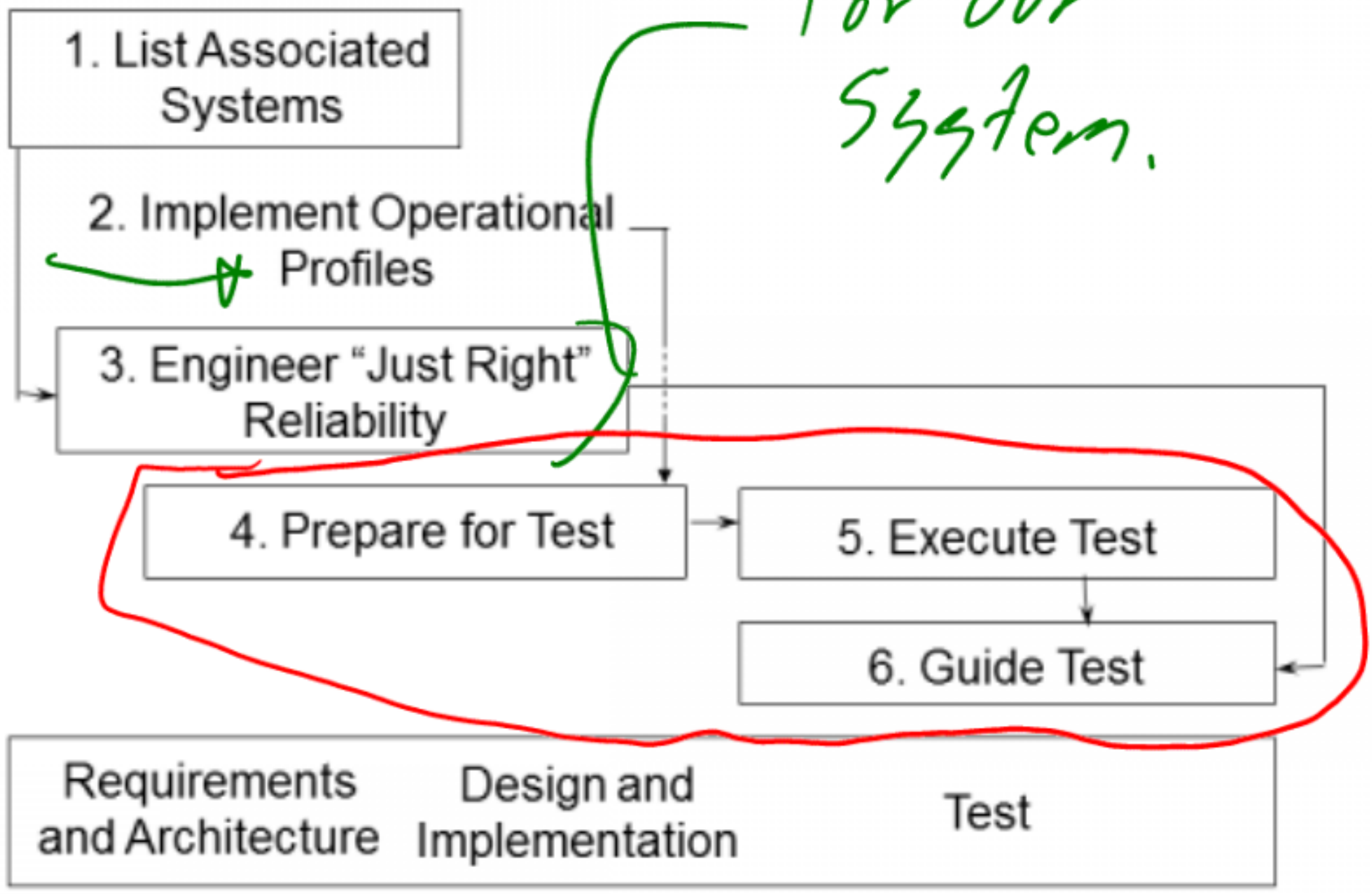
- We assume that software will fail in operation
- Our goal is to manage the failure and failure rates to be appropriate

Based on domain.

Tradeoffs in major quality characteristics

	Reliability of Availability	Development Time	Price
Higher			
Reliability of availability		Increases	Increases
New functionality	Decrease	Increases	Increases
Lower			
Development Time	Decreases		Increases
Price	Decreases	Increases	

Software Reliability Engineering



List Associated Systems of the Product

1. Base product
2. Major variations of base product (for substantially different environments, platforms, or configurations)
3. Frequently used supersystems of base product or variations

Determining the "right" level of reliability

- Define what you mean by failure
- Choose a common reference unit for all failures
- Set system failure intensity for each associated system
- Measure the results

Failure Severity Classes

Failure Severity Class	System Capability	Example
1	Basic Service interruption	
2	Basic service degradation	
3	Inconvenience, correction can not be deferred	
4	Minor tolerable effects, correction deferrable	

Failure Severity Classes for Phone System

Failure Severity Class	System Capability	Example
1	Basic Service interruption	Calls misforwarded or not forwarded
2	Basic service degradation	Phone number entry inoperable
3	Inconvenience, correction can not be deferred	Graphical User interface for administrators inoperable
4	Minor tolerable effects, correction deferrable	Date missing from display

Another assessment of failure intensity

Avionics Systems

Failure Impact	Typical Failure intensity objective (failures / hour)	Time Between failures
Hundreds of deaths, more than 10^9 cost	10^{-9}	114,000 years
One death, around 10^6 cost	10^{-6}	114 years
Around $1000 cost	10^{-3}	6 weeks
Around $100 cost	10^{-2}	100 hours
Around $10 cost	10^{-1}	10 hours
Around $1 cost	1	1 hour

Relating failure intensity to availability

$$\lambda_f = \frac{(1-A)}{(At_m)}$$

00 Hr?

δ Availability

Downtime per failure

99% Availability

15 minutes to repair a failure

$$\lambda_f = \frac{(1-.99)}{(.99 * .25)} = \frac{.01}{.2425} * 100$$

→ 4 failures / 100hrs

Converting failure intensity to reliability

$$R = e^{(-\lambda t)}$$

if $\lambda t < 0.05$

$$R \approx 1\text{-}\lambda t$$

Converting reliability to failure intensity

$$\lambda = \frac{-\ln R}{t}$$

if $\lambda t < 0.05$

$$\lambda \approx \frac{(1-R)}{t}$$

Preparing our tests

- Determine the number of test cases for the system
  - Allocate them equally across the system in proportion to the occurrence rate of the system
  - Make certain each test has a minimum of one, adjusting total number as is necessary

or numbers as is necessary

Planning the number of tests

- Lets make the following assumptions:
  - It takes 1 hour to design and run a test
  - 5% of tests result in a failure
  - 4 hours are spent correcting each fault
  - Test budget is 1000 hours
- How many tests do we create?

$1T + (.05 * 4T) = 1000$

$T = 833$

Preparing our Tests

Operation	Occurrence Probability	Initial Test Case Count
Proc. Voice call, no pager, ans.	0.21	175
Process voice call, pager, ans.	0.19	158
Proc. Fax call	0.17	142
Proc Voice call, pager answer on page	0.13	109
Proc. Voice call, no pager, no answer	0.10	83
Proc voice call, pager, no ans. On page	0.10	83
Enter forwardees	0.09	75
Audit sect. phone number data base	0.009	8
Add subscriber	0.0005	1
Delete subscriber	0.0005	1
Recover from hardware failure	0.000001	1
Total	**1**	833

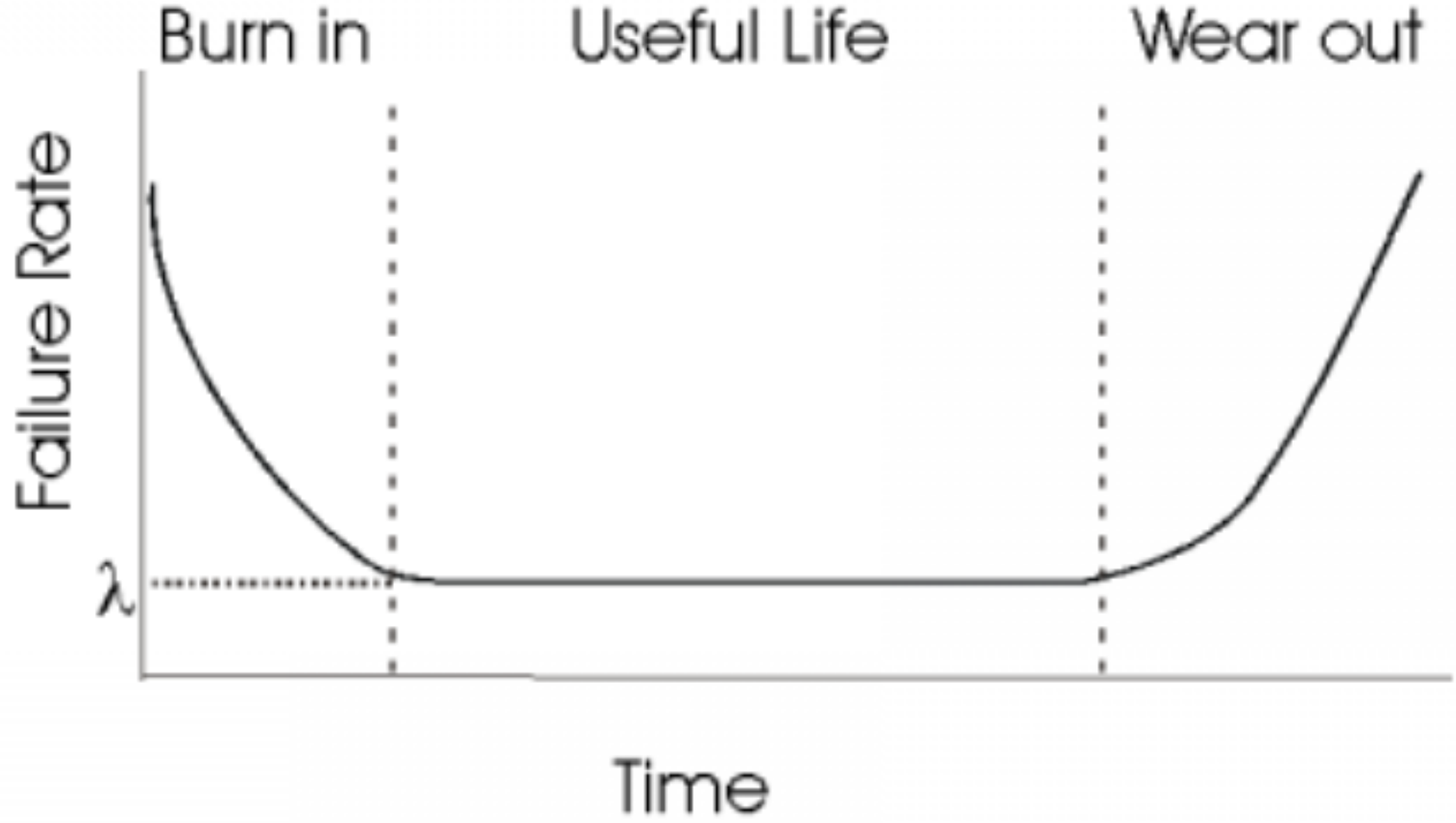
Preparing our Tests

Operation	Occurrence Probability	Initial Test Case Count
Proc. Voice call, no pager, ans.	0.21	
Process voice call, pager, ans.	0.19	
Proc. Fax call	0.17	
Proc Voice call, pager answer on page	0.13	
Proc. Voice call, no pager, no answer	0.10	
Proc voice call, pager, no ans. On page	0.10	
Enter forwardees	0.09	
Audit sect. phone number data base	0.009	
Add subscriber	0.0005	
Delete subscriber	0.0005	
Recover from hardware failure	0.000001	
Total	**1**	

Gforge Operations

Operation Description	Number of times performed	Number of test cases
Submit code into svn repository	24%	
Check out software version from svn repository	72%	
Submit a new bug into the bug tracking system	2%	
Run bug report	0.5%	
Update the status of a bug	1%	
Configure e-mail for submitted bugs	0.05%	
Add new document	0.0%	
Retrieve existing document	0.0%	
Add new project user	0.0%	
Remove project user	0.0005%	

Typical Reliability of Systems



Reliability Curves