Module 1: Introduction to Computer System and Network Model Verification & Validation
What is Verification & Validation?

**Verification** – are we building the system correctly?

**Validation** – are we building the correct system?

Often intertwined and confused…. But by any pair of definitions the basic notions are

**Basic notions:**

- *Specification* - A description of what a system is supposed to do.
- *Realization* - A description of what a system is and does (a *model*)
- *Implementation* - A physical instance of the system.

We will often talk about ‘validating a model’ to mean aligning specification and realization, and realization with implementation
What’s a System?

• Many things, but in the context of this class, a collection of
  – hardware
  – networks
  – operating systems, and
  – application software
  that is intended to be dependable, secure, survivable or have predictable
  performance.

• Before learning how to analyze such systems we must review
  1) Basic performance and dependability concepts and measures
  2) Fault/Error types
  3) Fault avoidance and tolerance techniques
Property of Interest -- Dependability

- *Dependability* is the ability of a system to deliver a specified service.
- System service is classified as *proper* if it is delivered as specified; otherwise it is *improper*.
- System *failure* is a transition from proper to improper service.
- System *restoration* is a transition from improper to proper service.

⇒ The “properness” of service depends on the user’s viewpoint!

Examples of Specifications of Proper Service

- $k$ out of $N$ components are functioning.
- every working processor can communicate with every other working processor.
- every message is delivered within $t$ milliseconds from the time it is sent.
- all messages are delivered in the same order to all working processors.
- the system does not reach an unsafe state.
- 90% of all remote procedure calls return within $x$ seconds with a correct result.
- 99.999% of all telephone calls are correctly routed.

$\Rightarrow$ Notion of “proper service” provides a specification by which to evaluate a system’s dependability.
Dependability Concepts

• **Measures** - properties expected from a dependable system
  – Availability
  – Reliability
  – Safety
  – Confidentiality
  – Integrity
  – Maintainability
  – Coverage

• **Means** - methods to achieve dependability
  – Fault Avoidance
  – Fault Tolerance
  – Fault Removal
  – Dependability Assessment

• **Impairments** - causes of undependable operation
  – Faults
  – Errors
  – Failures
Faults, Errors, and Failures can Cause Improper Service

- **Failure** - transition from proper to improper service
- **Error** - that part of system state that is liable to lead to subsequent failure
- **Fault** - the hypothesized cause of error(s)
Dependability Measures: Availability

*Availability* - quantifies the alternation between deliveries of proper and improper service.

- \( A(t) \) is 1 if service is proper at time \( t \), 0 otherwise.
- \( E[A(t)] \) (Expected value of \( A(t) \)) is the probability that service is proper at time \( t \).
- \( A(0,t) \) is the fraction of time the system delivers proper service during \([0,t]\).
- \( E[A(0,t)] \) is the expected fraction of time service is proper during \([0,t]\).
- \( P[A(0,t) > t^*] \) \((0 \leq t^* \leq 1)\) is the probability that service is proper more than \(100t^*\)% of the time during \([0,t]\).
- \( A(0,t)_{t\rightarrow\infty} \) is the fraction of time that service is proper in steady state.
- \( E[A(0,t)_{t\rightarrow\infty}], P[A(0,t)_{t\rightarrow\infty} > t^*] \) as above.
Other Dependability Measures

- **Reliability** - a measure of the continuous delivery of service
  - $R(t)$ is the probability that a system delivers proper service throughout $[0,t]$.

- **Safety** - a measure of the time to catastrophic failure
  - $S(t)$ is the probability that no catastrophic failures occur during $[0,t]$.
  - Analogous to reliability, but concerned with catastrophic failures.

- **Time to Failure** - measure of the time to failure from last restoration. (Expected value of this measure is referred to as $MTTF - \text{Mean time to failure}$.)

- **Maintainability** - measure of the time to restoration from last experienced failure. (Expected value of this measure is referred to as $MTTR - \text{Mean time to repair}$.)

- **Coverage** - the probability that, given a fault, the system can tolerate the fault and continue to deliver proper service.
Illustration of the Impact of Coverage on Dependability

- Consider two well-known architectures: simplex and duplex.

- The Markov model for both architectures is:

- The analytical expression of the MTTF can be calculated for each architecture using these Markov models.
Illustration of the Impact of Coverage, cont.

- The following plot shows the ratio of MTTF (duplex)/MTTF (simplex) for different values of coverage (all other parameter values being the same).
- The ratio shows the dependability gain by the duplex architecture.

- We observe that the coverage of the detection mechanism has a significant impact on the gain: a change of coverage of only $10^{-3}$ reduces the gain in dependability by the duplex system by a full order of magnitude.
Property of Interest -- Performance

- **Performance** is how well a system performs, provides proper service.
- Example (Generic) Performance Measures:
  - **throughput** -- the number of jobs processed per unit time.
  - **response time** -- the time to process a specific job.
  - **capacity** -- the maximum number of jobs that may be processed per unit time.

- Most practical performance measures are very application specific, and measure times to perform particular functions or, more generally, the probability distribution function of the time to perform a function.
A Combined Performance/Dependability Concept - Performability

- **Performability** quantifies how well a system performs, taking into account behavior due to the occurrence of faults.
- It generalizes the notion of dependability in two ways:
  - includes performance-related impairments to proper service.
  - considers multiple levels of service in specification, possibly an uncountable number.
- Performability measures are truly user-oriented, quantifying performance as perceived by users.

What is Validated? - Security (from MAFTIA presentation, David Powell & Yves Deswarte)

- Violation of a security property or intended security policy

Confidentiality

Integrity

Availability

Thou shalt not…

• ...

• ...

• ...

• ...

Thou shalt…

• ...

• ...

• ...

• ...

contradictory

mutually inconsistent
Fault Model for Security (from MAFTIA presentation, David Powell & Yves Deswarte)

- **attack** - malicious external activity aiming to intentionally violate one or more security properties; an *intrusion* attempt
- **vulnerability** - a malicious or non-malicious fault, in the requirements, the specification, the design or the configuration of the system, or in the way it is used, that could be exploited to create an *intrusion*
- **intrusion** - a malicious interaction fault resulting from an *attack* that has been successful in exploiting a *vulnerability*
Failure Sources and Frequencies

Non-Fault-Tolerant Systems
- Japan, 1383 organizations (Watanabe 1986, Siewiorek & Swarz 1992)
- USA, 450 companies (FIND/SVP 1993)

Mean time to failure: 6 to 12 weeks
Average outage duration after failure: 1 to 4 hours

Fault-Tolerant Systems
- Tandem Computers (Gray 1990)
- Bell Northern Research (Cramp et al. 1992)

Mean time to failure: 21 years (Tandem)

Failure Sources:
- 50% Hardware
- 25% Software
- 15% Communications Environment
- 10% Operations-Procedures

65%
How is Analysis Done?

- **Modeling**
  - **Simulation** (Fault Injection on Simulated System)
  - **Analysis/Numerical**
    - **Deterministic**
      - Continuous State
      - Discrete Event (state)
    - **Non-Deterministic**
      - Sequential
      - Parallel

- **Measurement**
  - **Passive** (no fault injection)
    - Without Contact
    - Hardware-Implemented
    - Stand-alone Systems
  - **Active** (Fault Injection on Prototype)
    - With Contact
    - Software-Implemented
    - Networks/Distributed Systems

- **Analysis**
  - **State-space-based**
  - **Non-State-space-based** (Combinatorial)
    - **Probabilistic**
      - **Non-Probabilistic**
When Does Analysis Take Place?

- **Specification** - Combinatorial modeling, Analytic/Numerical modeling
- **Design** - Analytic/Numerical modeling, Simulation modeling
- **Implementation** - Detailed Simulation modeling, Measurement, including fault injection
- **Operation** - Combinational modeling, Analytic/Numerical modeling, Detailed Simulation modeling, Measurement, including fault injection

Specification and Realization evolve throughout the lifecycle of a system.

Realization ultimately becomes an implementation.
Choosing Analysis Techniques

• There are several choices
  – Combinatorial modeling.
  – Analytic/numerical modeling.
  – Simulation (including fault injection on a simulated system).
  – Measurement (including performance benchmarking and fault injection on a prototype system).

each with differing advantages and disadvantages

• Choice of a validation method depends on
  – Stage of design (is it a proposed or existing system?).
  – Time (how long until results are required).
  – Tools available.
  – Accuracy.
  – Ability to compare alternatives.
  – Cost.
  – Scalability.
### Choosing Analysis Techniques cont.

<table>
<thead>
<tr>
<th>Criterion</th>
<th>Combinatorial</th>
<th>State-Space-Based</th>
<th>Simulation</th>
<th>Measurement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage</td>
<td>Any</td>
<td>Any</td>
<td>Any</td>
<td>Post-prototype</td>
</tr>
<tr>
<td>Time</td>
<td>Small</td>
<td>Medium</td>
<td>Medium</td>
<td>Varies</td>
</tr>
<tr>
<td>Tools</td>
<td>Formulae, spreadsheets</td>
<td>Languages &amp; tools</td>
<td>Languages &amp; tools</td>
<td>instrumentation</td>
</tr>
<tr>
<td>Accuracy</td>
<td>Low</td>
<td>Moderate</td>
<td>Moderate</td>
<td>high</td>
</tr>
<tr>
<td>Comparisons</td>
<td>Easy</td>
<td>Moderate</td>
<td>Moderate</td>
<td>Difficult</td>
</tr>
<tr>
<td>Cost</td>
<td>Low</td>
<td>Low/medium</td>
<td>Medium</td>
<td>High</td>
</tr>
<tr>
<td>Scalability</td>
<td>High</td>
<td>Low/medium</td>
<td>Medium</td>
<td>low</td>
</tr>
</tbody>
</table>
Some Rules of Thumb When Analyzing a System

- In general, always be suspicious of your results!... (e.g. don’t accredit to them more accuracy than is warranted)

- Guideline: cross-check
  - Compare simulations with analytic models and measured data
  - Compare analytic models with simulations and measured data
  - Compare measured data with analytic models and simulations

- And, in particular, always
  - Evaluate “boundary cases” to which you know the answers
  - Make sure trends are as you expect, or understand why they are not
The “Art” of Performance and Dependability Analysis

- Performance and Dependability analysis is an art because:
  - There is no recipe for producing a good analysis,
  - The key is knowing how to abstract away unimportant details, while retaining important components and relationships,
  - This intuition only comes from experience,
  - Experience comes from making mistakes.

- There are many ways to make mistakes.

- You learn a tremendous amount about the system just by the act of modeling it and studying the model predictions.
Doing it Right

- Understand the desired measure before you build the model or design a measurement or fault-injection experiment.
- The desired measure determines the type of model, performance benchmark, or fault-injection experiment and the level of detail required.
  - No model or measurement technique is universal.
- First step: choose the desired measures:
  - Choice of measures form a basis for comparison.
  - It’s easy to choose wrong measure and see patterns where none exist.
  - Measures should be refined during the design and validation process.
- Understand the meaning of the obtained measures:
  - Numbers are not insights.
  - Understand the accuracy of the obtained measures, e.g., confidence intervals for simulation.
More Doing it Right

- Include all critical aspects in a model of a system:
  - Once measures are chosen, you must choose what system aspects to include in the model.
  - It is almost never possible or practical to include all system aspects.

- Use representative input values:
  - The results of a model solution, performance benchmark, or fault injection experiment are only as good as the inputs.
  - Inputs will never be perfect.
  - Understand how uncertainty in inputs affects measures.
  - Do sensitivity analysis.

- Include important points in the design/parameter space:
  - Parameterize choices when design or input values are not fixed.
  - A complete parametric study is usually not possible.
  - Some parameters will have to be fixed at “nominal” values.
  - Make sure you vary the important ones.
More Doing it Right

• Make all your assumptions explicit:
  – Results from models, benchmarks, or fault-injection experiments are only as good as the assumptions that were made in obtaining them.
  – It’s easy to forget assumptions if they are not recorded explicitly.

• Use the appropriate model solution or measurement technique:
  – Just because you have a hammer doesn’t mean the world is a nail.
  – Fault injection and simulation, numerical/analytic, and combinatorial solutions all have their places.

• Keep social aspects in mind:
  – Dependability analysts almost always bring bad news.
  – Bearers of bad news are rarely welcomed.
  – In presentations, concentrate on results, not the process.
**Model Validation / Verification**

Models may be validated and verified using a number of methods:

- **Modular design:** test modules separately; interchange functionally similar modules,
- **N-version models:** high-level and detailed models should give similar results,
- **Run simplified cases:** e.g., one packet in the network,
- **Tracing:** examine one trajectory,
- **Understand trends:** understand the direction of the trends, and any discontinuities.
More Model Validation / Verification

• Models are frequently validated by three methods:
  – **Measurements**: measures on the model should match measures on the real system,
  – **Theoretical results**:
    • measure something to which you already know the answer,
    • e.g., throughput cannot exceed capacity,
  – **Insight of experts**:
    • Modeler expertise comes with experience,
    • Consult with people who understand the system.

• Validating a model is similar to validating software.
  – **Design review**: present the model design to a small committee that will critically review it.
  – **“Code”** review: someone else critically examines the model in detail.
  – **Assertions**: place assertions in the model to warn when things go wrong.
  – **Black box/White box testing.**
Course Overview

• Analytical/Numerical Methods
  – Review of Probability Theory
  – Combinatorial Methods
  – Fault Trees/Reliability Block Diagrams/Reliability Graphs
  – Review/Introduction to Stochastic Processes
  – Markov Modeling
  – Numerical/Tool Issues
  – High-Level Model Representations (Stochastic Petri Nets, Stochastic Activity Networks)
  – Numerical Issues in Markov Model Solution
  – Basic Queuing Theory
Course Overview, Cont

- Simulation
  - Simulation Basics
  - Random number generation
  - Experimental Design
  - Validation and Verification
  - Variance Reduction
  - Simulation-based Optimization

- The Art of Computer System Assessment
Course Emphasis

The course is structured to focus on you getting experience solving problems. Reading suggestions, lecture notes, and recorded lectures will be available before class meeting:

- I will expect that you have gone through a class’s prep material before meeting.
- Class time will be spent answering questions for clarification, and working problems:
  - Small teams, joint effort, staff available for guidance
  - Worked through problems will have strong correlation with ‘exam’ problems --- ‘continuously running exams’
  - A major chunk of your grade depends on your presence participating with your team(s)
- Two significant projects (approximately ½ way, and at the end)
  - Presentation day(s)
Course Emphasis

The course is structured to focus on you getting experience solving problems. Reading suggestions, lecture notes, and recorded lectures will be available before class meeting.

- I will expect that you have gone through a class’s prep material before meeting.
- Class time will be spent answering questions for clarification, and working problems
  - Small teams, joint effort, staff available for guidance
  - Worked through problems will have strong correlation with ‘exam’ problems — ‘continuously running exams’
  - A major chunk of your grade depends on your presence participating with your team(s)
- Two significant projects (approximately ½ way, and at the end)
  - Presentation day(s)
Course Emphasis

The course is structured to focus on you getting experience solving problems. Reading suggestions, lecture notes, and recorded lectures will be available before class meeting:

- I will expect that you have gone through a class’s prep material before meeting.
- Class time will be spent answering questions for clarification, and working problems:
  - Small teams, joint effort, staff available for guidance.
  - Worked through problems will have strong correlation with ‘exam’ problems --- ‘continuously running exams’
  - A major chunk of your grade depends on your presence participating with your team(s)
- Two significant projects (approximately ½ way, and at the end):
  - Presentation day(s)